Remote Operator Blended Intelligence System for Environmental Navigation and Discernment

Jonathan Elliot Gaines

Dissertation submitted to the faculty of the
Virginia Polytechnic Institute and State University
In partial fulfillment of the requirements for the degree of

Doctor of Philosophy
In
Mechanical Engineering

September 6th, 2011
Blacksburg, Virginia

Remote Operator Blended Intelligence System for Environmental Navigation and Discernment

ABSTRACT

Mini Rotorcraft Unmanned Aerial Vehicles (MRUAVs) flown at low altitude as a part of a human-robot team are potential sources of tactical information for local search missions. Traditionally, their effectiveness in this role has been limited by an inability to intelligently perceive unknown environments or integrate with human team members. Human-robot collaboration provides the theory for building cooperative relationships in this context. This theory, however, only addresses those human-robot teams that are either robot-centered or human-centered in their decision making processes or relationships. This work establishes a new branch of human-robot collaborative theory, Operator Blending, which creates codependent and cooperative relationships between a single robot and human team member for tactical missions. Joint Intension Theory is the basis of this approach, which allows both the human and robot to contribute what each does well in accomplishing the mission objectives. Information processing methods for shared visual information and object tracking take advantage of the human role in the perception process. In addition, coupling of translational commands and the search process establish navigation as the shared basis of communication between the MRUAV and human, for system integration purposes. Observation models relevant to both human and robotic collaborators are tracked through a boundary based approach deemed AIM-SHIFT. A system is developed to classify the semantic and functional relevance of an observation model to local search called the Code of Observational Genetics (COG). These COGs are used to qualitatively map the environment through Qualitative Unsupervised Intelligent Collaborative Keypoint (QUICK) mapping, created to support these methods.

Acknowledgements

Many individuals have impacted my life and deserve credit. If you know me, you realize that I can go on and on when it comes to things like this ~~so I have intentionally tried to keep it relatively short~~. I apologize in advance if I forget to include you. Please, charge it to my head and not my heart.

To God—I would first like to give God the glory for putting me in the position to finish this Thesis. Thank you to the St. Paul family for supporting me spiritually during my time at Virginia Tech.

To my Grandfather—Thank you for the phone calls to say you were thinking of me and to check and see if I was done. To this day, I have strived to be like you. Your kindness, dedication, and encouragement have been invaluable throughout this process. Most importantly, thank you for being my hero and giving me a man to look up to throughout my entire life.

To my Mother—I am the man I am today also largely because of you. The values you instilled in me led me down this path to get this degree. Thanks for being my biggest advocate and supporter. As a single mother early in my life, you provided everything I needed and I always knew we were a team. As I transitioned into adulthood, thanks for bringing quality individuals into my life. The love and support of the Martin family, particularly my step-father Ricky Martin has been invaluable. To the Martins, thank you for giving me another dimension to my concept of family. You all accepted me with open arms and because of that, I will forever be grateful.

To my Grandmother—One thing I have always been certain of is that I come from a long line of Superwomen. This came from coming home and watching you work in the garden, cook, move objects three times your size, lay bricks, and wake up early in the morning to teach 2nd graders. You have a tremendous heart and work ethic. During this process, thank you for having a fresh pound cake ready for me anytime I came home and for your overall level of awesomeness.

To the rest of my family—I am a 4th generation college graduate, which speaks volumes considering many individuals are first generation college students. My great-grandmother, Eunice Stephens, who just celebrated her 101st birthday, graduated from Benedict College in 1935. It is hard to believe her great-grandmother was a slave, but this reflects the perseverance and resourcefulness of my family. You all set a standard that I hope to continue to uphold through my personal character, impact on my community, and academic achievements. In particular, thank you Uncle Carl, Uncle John, the rest of the Gaines family, the Stephens family, the Miller family, and, again, thanks to the Martin family.

To my research family—Thank you Dr. Wicks for believing in me. There is no doubt that your approach to teaching, mentorship, and research has influenced many individual students. It has made a direct impact on me far beyond what you have done helping to shape this thesis. I believe you have been an outstanding example to the type of faculty member I would like to become. Also, thank you to those individuals at the Unmanned Systems Lab, Mechatronics Lab, and my committee for the guidance, comaraderie, and support I needed to complete this thesis.

To my NASA family—Thank you to the JPFP Fellowship Program for supporting my graduate education. Geoff Bland and Ted Miles, I really appreciate the research experience, networking, and general hospitality you extended towards me at Wallops Flight Facility.

To my Brothers—My Fraternity has taught me the value of personal relationships. A special thank you goes out to the Alpha Kappa Lambda Chapter and Theta Iota Chapter of Alpha Phi Alpha Fraternity, Inc. There are too many of you to mention individually, but many of you impacted my life on a very personal level. I hope that I left as big of an impression on you as you have on me. Onward and Upward.

To my boys—I don't know if I could have made it through this process if it wasn't for you. Jahi, Nick, Leemar, Hamilton, Osaro, and Trey. Chase, Jeff, Corey, Quintin, and Christian. Lamar and Jeremy. Brandon and Walter. You all have meant a lot to me over the years.

To my beautiful Fiance Kenya King—I love you and look forward to starting our life together. Finding you has made everything else worth it. Thanks for loving me.

Jonathan

Table of Contents

List of Figures

List of Tables

Nomenclature

u, v—Image coordinates

x, y—Global coorinate

i, j—Image row and column index

$I_R(i,j)$—Reference image of a stereo pair

$I_S(i,j)$—Search image of a stereo pair

Q—Reprojection matrix

C_x, C_y—Image center point location

B—Stereo camera baseline distance

d—Disparity value between pixel coordinates

e—Sub-pixel accuracy

p—Pixel size

r—Radial distance from imager focal point to real world location

s—scale for Gaussian Scale Space

o—octave for Gaussian Scale Space

σ—Standard deviation of a sample

μ--Mean value of a sample

$g_\sigma(x)$=Gaussian kernel

R—Candidate stability region for Maximally Stable Extremal Regions

$R_{+\Delta}$—Stability region with slightly higher threshold values

$R_{-\Delta}$—Stability region with slightly lower threshold values

m(x)—Average pixel value for a candidate region in Mean Shift

M(x)—Average pixel change for a centroid within a candidate region in Mean Shift

K(x)—Kernel function. For Mean Shift object tracking, this is the Kronecker delta function

h—Kernel function bandwidth. For Mean Shift object tracking this is the width of the candidate region

C—Normalization factor for the Kernel function to ensure kernel values sum to 1

p_u—Color histogram bin u of the candidate region in Mean Shift

q_u—Color histogram bin u of the object region in Mean Shift

$\rho(p_u, q_u)$— Bhattacharya distance between color histogram bins

δ—Distance measure between color histogram bins

$p(x|m)$—Bayesian probability that data points x are part of the map m

ρ—the minimum distance from the scan origin to the normal of a line model

α—the angle between the line model and the horizontal axis

D—the Mahalanobis distance between vectors of LIDAR measurements

I—Mass moment of inertia

α—Angular acceleration

τ—Torque

COV_x—Covariance along the dimension x

COV_y—Covariance along the dimension y

CORR—Correlation along the x and y dimensions

Acronyms and Abbreviations

MRUAV—Mini Rotorcraft Unmanned Aerial Vehicle

DoF—Degrees of Freedom

TTP—Tactics, Techniques, and Procedures

RobiSEND—Remote Operator Blended Intelligence System for Environmental Navigation and Discernment

VTOL—Vertical Take-off and Landing

UAS—Unmanned Aerial Systems

USAR—Urban Search and Rescue

UAV—Unmanned Aerial Vehicle

UGV—Unmanned Ground Vehicle

DASH—Drone Anti-Submarine Helicopter

DARPA—Defense Advanced Research Project Agency

ALV—Autonomous Land Vehicle

GSR—Ground Surveillance Robot

HRI—Human-Robot Interaction

AROD—Airborne Remotely Operated Device

AMGSSS—Air-Mobile Ground Security Surveillance System

OHOR—One Human One Robot

COG—Code of Observational Genetics

QUICK mapping—Qualitative Unsupervised Intelligent and Collaborative Keypoint mapping

AIM-Shift—Area Intersecting Mean Shift

UUV—Unmanned Underwater Vehicle

VOI theory—Value-Of-Information theory

SA—Situational Awareness

LIDAR—Scanning laser range finders

SAD—Sum of Absolute Difference method

MSER—Maximally Stable Extremal Region

DoG—Difference of Gaussian

RANSAC—Random Sample Consensus

SLAM—Simultaneous Localization and Mapping

EM—Expectation Maximization

HMM—Hidden Markov Models

RBPF—Rao-Blackwellized Particle Filter

SIFT—Scale Invariant Feature Transform

QSR—Qualitative Spatial Reasoning

POI—Passive Observation Interpretation

AOI—Active Observation Interpretation

E—Explicit trait

I—Indefinite (I)

N—Normal (N)

S—Special (S)

T—Temporary (T)

F—Full (F)

J—Juvenile (J)

P—Permanent (P)

GUI—Graphical User Interface (GUI)

CARMEN—Carnegie Mellon Robot Navigation Toolkit

GENDER BIAS DISCLAIMER: The use of ‘he’, ‘she’ and any similar gender reference are used for convenience or clarity and are not to reflect any gender bias of the author.

1. Introduction to Operator Blending

***"Environment is the mental feeding ground out of which the food that goes into our minds is extracted."* –Napoleon Hill**

In 1965, 20th Century Fox launched the TV series "Lost in Space". The plot involves the Robinsons, aided by an autonomous humanoid robot, and their journey in year 1997 to colonize an unknown planet in deep space. The robot is like an extended family member who is the source of an uncanny ability to provide resourcefulness and comic relief, aiding in the exploration of an unknown planet. Nearly 15 years after the fictional launch date of their futuristic journey, human beings in the real world have difficulty with complex forms of human-robot interaction. More realistic fantasies of jointly exploring unknown environments on Earth exist; however, in depth and collaborative relationships with robots still remain a major challenge.

To develop these capabilities, relationships between human and robotic team members are essential and based on perception and integration within surroundings. Challenges may be addressed through how humans and robots extract or "feed" on pertinent information in the environment. In a team, it is possible for the robot and human to optimize this by sharing perception and integration processes; however, this requires a rather substantial shift in approach. Effectiveness no longer is a question of the robot, human, or environment alone, but rather, how these pieces fit together.

If the mechanism is broken or unreliable that produces these relationships, then a robotic system, or a human for that matter, ultimately builds the wrong bonds. The environment is the link between the human and robot and is best understood through how these relationships are most effectively manufactured. Achievement of team *functionality* is the ultimate goal with the environment as the basis of common understanding.

1.1 Overview

The purpose of Section 1.1 is to outline the need for human-robot collaborative relationships with unmanned systems. Sub-section 1 summarizes the problem addressed in this research, sub-section 2 defines the concept of human-robot partnership, sub-section 3 presents collaborative sensing as a means to address the aforementioned problem, sub-section 4 introduces the basic robot system used to demonstrate these concepts, and sub-section 5 details the basic application of local search used in this research as an example for the novel theories developed. This section is important because it establishes the need for human-robot collaborative systems, particularly as they relate to the types of applications discussed in this research.

1.1.1 The problem

Unmanned systems are increasingly being implemented and viewed as additional team members under a wide range of circumstances. Particularly amongst small tactical teams, ground or aerial robots are even given names and personalities. The lack of reliability of current unmanned systems makes these relationships fleeting in dangerous situations within unknown environments. Comforting names like 'Ol Betsy' or 'Mr. Watson' are replaced by less favorable four letter words. The systems are quickly abandoned for more traditional means of solving problems.

This leaves two paths for development of unmanned system behavior in teaming situations. Researchers can concentrate on making behaviors more controlled and autonomous with the hope they can ultimately function at high enough levels to be trusted. Or, researchers can improve the quality of relationships that are built between unmanned systems and team members with the belief that this leads to more effective functionality. Both paths play a vital role moving unmanned systems technology forward.

The emphasis during development should largely depend on the nature of the environment and mission. For larger unmanned systems that operate at high altitude, precise autonomous control may be the primary means to optimize team effectiveness. For unmanned systems that operate in the vicinity of human life, there is quite likely a need to build stronger functional relationships. Because the human and robot share the environment, increased autonomy in the latter situation may lead to more distrust or lack of understanding. Nevertheless, development of unmanned systems technology has been skewed towards solving control theory problems and away from optimizing functional relationships of human-robot teams. This development, unfortunately, is usually done independently of the mission profile.

Many approaches require no understanding or teamwork between human and robot and are highly successful if these relationships are not needed. Outside of controlled circumstances, however, mission types require a deeper understanding of robotic team members and their relationship to the environment. When both human and robot contributions are necessary—a consequence of finite resources, safety and trust considerations, an insurmountable amount of uncertainty about the environment, or functional limitations, the human, knowingly or unknowingly, must build these relationships. The robot or human by itself does not have the ability to adequately accomplish mission objectives.

Teams currently either undervalue or overvalue the contribution of the human or robot teammates in these situations. In these missions, full autonomous or manual control does not lead to an effective solution. Recent evidence of this may be seen in the role of autonomy in commercial aviation. The pilot and the airplane have a shared task and a collaborative relationship. They both, with their various sensors and decision making abilities, have the goal of flying successfully to the final destination. Millions of human beings rely on this team of the pilot and the airplane to get them from point A to point B.

The two industry leaders, Boeing and Airbus have starkly different approaches. All of Boeing airplanes defer to the pilot. Although there are modes of operation where the airplane takes control, these must be authorized and initiated by the human in the loop. This is because of the human's perceived judgment and ability to introduce context to the situation to solve problems. Airbus, on the other hand, basically flies itself outside of a catastrophic event that triggers the manual mode. The Airbus system is assumed to be able to objectively assess situations through sensors more effectively than humans.

Both of these philosophies are usually very effective, yet each airline has had deadly crashes in the past 2 years. Boeing flight 812 crashed because the pilot fell asleep at the controls and Airbus flight 447 crashed as a result of a pitot tube sensor malfunction. In both cases, the human and robot did not remain an acceptable contributor to team success. Upon failure of the human or the robot, the secondary decision maker was not prepared or unable to intervene. As a result, the practicality of purely human-centered or robot-centered collaborative relationships is called into question. There must be a middle ground that keeps both engaged and leverages their very specific skill sets to accomplish the mission. When cooperation and codependence are needed, both agents must be relied upon to contribute to success.

1.1.2 Human-robot partnership for MRUAV based search

A collaborative theory is developed that acknowledges and takes advantage of codependent and cooperative relationships in human-robot teams based on their shared understanding within the environment. It is acknowledged that the human has the choice of abandoning the mission at any time by discontinuing this relationship; however, the mission itself cannot be acceptably completed unless these bonds are maintained. The specific application focused on is optimizing Mini Rotorcraft Unmanned Aerial Vehicle (MRUAV) functionality for local search missions.

Operators and unmanned systems at times derive representations of their surroundings that include boundary information and objects of interest. This process is made more difficult in settings with unreliable a priori information or the absence of precise global position; however, the absence of this information reflects more accurately the circumstance in real mission scenarios. Because of the dangers involved in exploring an unknown building in certain life threatening situations, MRUAVs will increasingly play a role in supplementing this capability. Sending a robot into a situation is advantageous when the human is incapable or the task is too dangerous to risk human life.

The existence of this technology in the absence of contributions by both robotic and human collaboration has proven to be difficult. What happens if an autonomous MRUAV has a sensor malfunction? What happens if a teleoperated MRUAV is crashed into a wall or takes multiple squad members to operate? Unfortunately, the answer quite possibly may be a loss of human life. These systems must be designed to leverage human and robotic resources simultaneously to be successful.

If the unmanned system acts intelligently, it has a spatial understanding of its surroundings that leads to reliable functionality. MRUAVs operational at low altitude may develop this for obstacle avoidance and navigation within cluttered, cramped spaces such as corridors, hallways, and small enclosures. This traditionally has involved the reconstruction of metric relationships through mapping—a task that is difficult to do in many real world environments.

In addition, when mapping an unknown area, a human-robot team has additional considerations. Collaboration requires the robot to share a semblance of human team member relationships to its surroundings. For both the robotic and human agents, oftentimes, this does not involve the accurate reconstruction of physical space. It involves a practical and functional understanding.

For example, if both human and robot are given the task of finding the optimal path home, the human usually has no idea how many total steps this might take or the width of an alleyway to the nearest centimeter—information that is easy to derive for the robot. The robot, on the other hand, does not have to know inconsequential information about the motivation of the mission. The fact that dinner will be cold if this is not done by 5:00pm is information grasped by the human that is inconsequential to the robot. At the same time, the robot might benefit from knowing to avoid Mrs. Wilson across the street, because she has a propensity to talk about her grandson and delay progress.

Information about the motivation of action that may impact mission effectiveness is important, because it may affect the path planning process. Likewise, the human is less likely to get lost if the robot is able to map the location of landmarks in a manner that gives a functional understanding about the structure of the environment. Only the most essential relationships shared between the robot, human and environment are necessary to achieve the result of walking through the front door.

1.1.3 Collaborative sensing

Before understanding how to develop these relationships, it must first be understood what these functional relationships are and the challenges for their development. If both humans and unmanned systems are to be localized partners, two primary issues interfere with shared understanding. Firstly, unmanned systems are robots and therefore have a difficulty being practical. They make decisions through the interpretation of data, computation to extract meaning, and memorization in the place of learning. Human operators more elegantly rely on contextual information. They are able to decide what is important and direct their attention accordingly.

The human ability to leverage contextual information is best shown in the illusion by Edward Adelson displayed in Figure 1.

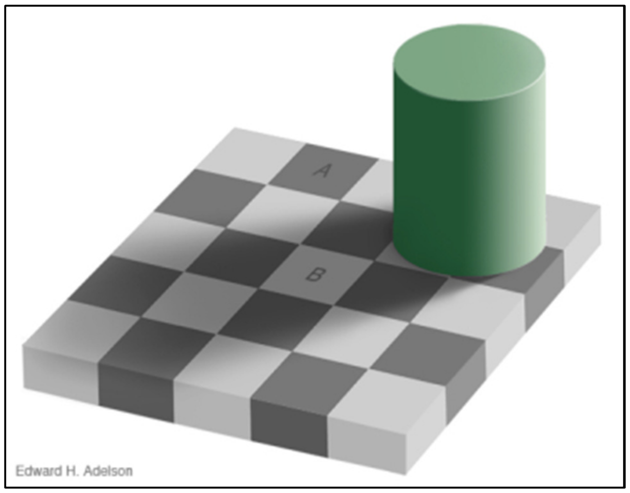


Figure 1: Optical Illusion by Edward Adelson that demonstrates how human beings use contextual information to make inferences about the environment. Reprinted from http://persci.mit.edu/gallery/checkershadow

Both square A and square B are the same gray scale value but are interpreted differently. Although the illusion is designed to trick human reasoning, the fact remains—if presented with context cues, the human perceptual system readily integrates this information. We process and keep what we believe to be important and build perceptual relationships between observations in the environment. This disconnect between how unmanned systems and humans process the environment must be overcome if a human operator expectation of practical behavior is to be realized. The unmanned system must either learn to process information more selectively or be provided with this ability in order to relate to human team members.

The team context itself is central to the second factor that interferes with the shared practical understanding of human-robot teams. When both systems are to be localized, it is not practical for them to develop completely independent functional understandings. Unfortunately, the majority of human-robot teams do not couple the goals of both agents. The blended goals of two agents within a team are the basis of Joint Intension Theory in collaborative robotics. If the mission requires teamwork, agents never set purely individual goals. Instead, the resources, capabilities, and needs of the team are considered and shape individual objectives that support team success.

If two human team members were exploring an unknown area, it would be unheard of to tell team member A to execute its mission as if he were completely isolated from team member B. Because both team members are accomplishing components of a common task, they would communicate and work together. The type of communication is not done in isolation of the mission profile, as mission specific goals may directly impact the means or frequency of communication. For example, soldiers on a covert mission may have to rely on hand signals and non-verbal cues for correspondence. Nevertheless, communication is a necessity. Team members should not function in isolation; otherwise, they are not being effective collaborators. The need to build these relationships dictates how each contributes individually.

For human-robot teams, if this practicality is not realized, the result, at times, is a marginally useful or dangerous system. The unmanned system does not meet the expectation of the human operator in its ability to perceive the environment or be integrated amongst team members. If the contribution cannot be trusted, the most natural human response is to disregard it as invalid. When the team member is an unmanned system, the robot literally and sometimes figuratively remains in the box—it goes unused, in a scrap heap, or is pigeon holed to those missions that do not require intelligent behavior.

Collaboration, however, may be essential for high level behavior if certain human-robot teaming missions are to become viable. The most basic of which is a collocated human-robot team with limited resources. The relationships that the robot and humans build between one another are relied upon to achieve global mission objectives. In addition, the practical functionality of each is enhanced by the contributions of the other. The development of human-robot collaborative systems is a need to address challenges and to create effective systems.

1.1.4: The platform

One platform that largely benefits from this type of development is the MRUAV. This research takes an applied approach to human-robot collaboration through the fabrication of a MRUAV payload for indoor, local search operations. Most current MRUAVs deployed outside of controlled circumstances require humans to be in the loop and designed for low altitude, close quartered operation. Because of this, MRUAV based missions are suited for the expansion of collaborative and cooperative theory. In addition, MRUAVs have the potential to be suited for search because of their ability to hover and navigate multiple Degrees of Freedom (DoF).

The long term plan for UAS involves the incorporation of MRUAVs for these types of low altitude missions. For the next 20 years, this capability is restricted to fly over reconnaissance behavior and does not involve a more comprehensive search capability as explored in this research. In the future, however, MRUAVs may give close quarter UAS the ability to contribute to local search—a task traditionally dominated by unmanned ground vehicles.

This work seeks to define an operational niche for MRUAV technology based on its contribution in human-robot collaborative local search. The argument is made that MRUAV limitations instead of their strengths are currently guiding their development. Human-robot collaboration is developed as the means to achieve greater access to MRUAV tactical capabilities. A payload is built to demonstrate the potential capabilities of MRUAVs. The concepts demonstrated by this payload take steps to make search in the vicinity of team members a possibility as hardware and control theory for these platforms continues to improve. Current MRUAVs lack the ability to excel in unknown environments, operate safely amongst human team members, or Tactics, Techniques, and Procedures (TTP) that take advantage of their unique characteristics.

The payload in this research deemed the Remote Operator Blended Intelligence System for Environmental Navigation and Discernment (RobiSEND) addresses these issues through expanding capabilities during teaming situations. This research also develops the theory for collaborative and cooperative relationships between humans and robots for short, tactical missions. The specific application of MRUAV based local search is expounded upon to convey these possibilities. The operator influences the perceptual capabilities and integration of the MRUAV through Operator Blending—a new branch of human-robot collaborative theory.

1.1.5: The Application

A search operation consists of the mapping of areas or objects of interest for the purpose of improved actionable intelligence. Vertical Take-off and Landing (VTOL) Unmanned Aerial Systems (UAS) are useful in search, because they are easy to deploy in combat situations, fly close to the ground, take off in cluttered environments, and provide useful information from hover. As a subcategory of VTOL UAS, MRUAVs are small enough to be deployed from the backpack of an individual. They are potentially a direct source of information that is otherwise impossible or too dangerous to attain through traditional means.

RobiSEND is used to create annotated maps from a search platform that is light and inexpensive enough to be added to the gear an individual might carry. For this purpose, it detects obstacles, documents surroundings, and provides the perceptual capabilities to potentially supplement user teleoperation. Operator blending also provides the means to decrease the number of system operators to one per MRUAV. In order to accomplish this, the human operator is viewed as a key source of contextual information.

Challenges are addressed considering the limited payload requirements of the system and the inert challenges of unstructured environments. The unique contributions of this work are in the areas of Qualitative Spatial Reasoning, Joint Intension Theory, Remote Sensing, unmanned systems, Urban Search and Rescue (USAR), and human-robot collaboration.

The following sections in Chapter 1 will outline a brief history and background involving unmanned systems in tactical situations and the motivation for this research. The three chapters that follow formulate the foundation for the Operator Blending theory. Chapter 2 will present literature encompassing factors in human-robot collaborative systems that most directly impact human team members. Chapter 3 will review approaches in the literature for sensors, information processing, and algorithmic approaches. These are considered the factors that most directly impact robotic systems. After the factors for human and robotic systems are presented, Chapter 4 introduces navigation and relationships built with the environment as the means of collaboration between team members. Chapter 5 details the methods of RobiSEND with an explanation of system components, approaches to processing, and function of individual system components. These building blocks are fit together through the Operator Blending theory of Chapter 6. Finally, Chapter 7 presents results, analysis, conclusions, and opportunities for future work.

1.2 Motivation

Section 1.2 outlines the relevance of MRUAVs used as local search systems. Sub-section 1 first gives a historic perspective to the introduction of MRUAVs for tactical purposes. Sub-section 2 and sub-section 3 look at this historical context with particular emphasis on the evolution of aerial and ground vehicles. Sub-section 4 then moves to discuss current MRUAV system strengths and weaknesses. Sub-section 4 expands upon this to discuss the future of MRUAV systems considering the state of technology. This section establishes the need to develop MRUAV technology from being an extension of other unmanned systems to coming into its own and being optimized for specific tactical purposes.

1.2.1 History: The root of MRUAV search systems

In many respects, MRUAVs have been developed as extensions to both Unmanned Aerial Vehicle (UAV) and Unmanned Ground Vehicle (UGV) operations. Although their relation to these technologies has historically cast these systems as experimental alternatives to UGVs and UAVs, this technology continues to evolve and come into its own. Its necessity lies in the inert deficiencies of both fixed-wing and ground vehicles to complete certain missions more suitable for MRUAVs. This brief historical overview of unmanned system development captures this evolution in light of where this technology must go in the future. These unique characteristics are leveraged for the theory of Operator Blending developed in this research and the other novel deliverables that further the field of Human-robot collaboration.

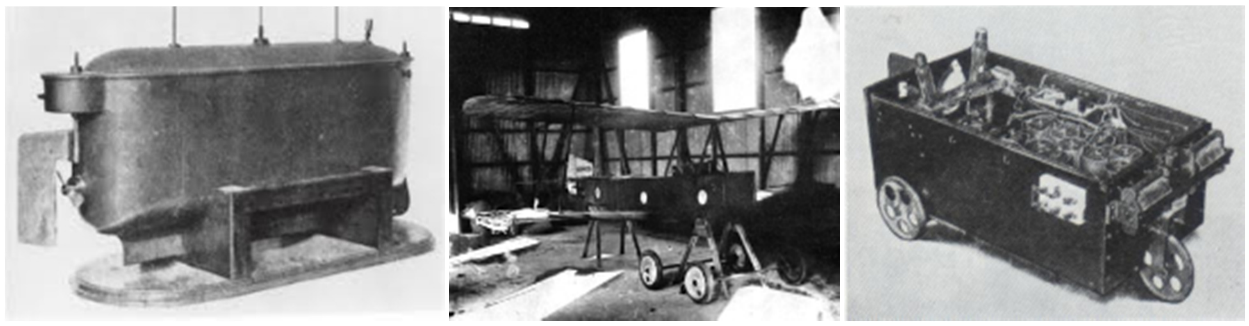


Figure 2: Early Systems. From left to right, Tesla's boat, Aerial Target, and Electric Dog. Public Domain

In 1898, Nicola Tesla introduced to the world a boat that he boasted as having a "borrowed mind" with the capability of being remotely operated through radio control. In the patent that he filed for this robot, he suggested it constitutes "the first of a race of robots, mechanical men which will do the laborious work of the human race" (Goodrich & Schultz, 2007). For all intents and purposes, this is the first introduction to both the reality of unmanned vehicles and the dream of what robots might potentially contribute to society. In the coming years, more complex remote teleoperated systems emerged that included the Sopwith "Aerial Torpedo" in 1917, Army's Kettering Bug in 1918, the Naval Research Laboratory's "Electric Dog" robot in 1923, and later, the British LARYNX in 1929 (Goodrich & Schultz, 2007). The state of the art in technology restricted these early robots and their ability to have the transformative effect that Tesla alluded to at the close of the 19th century.

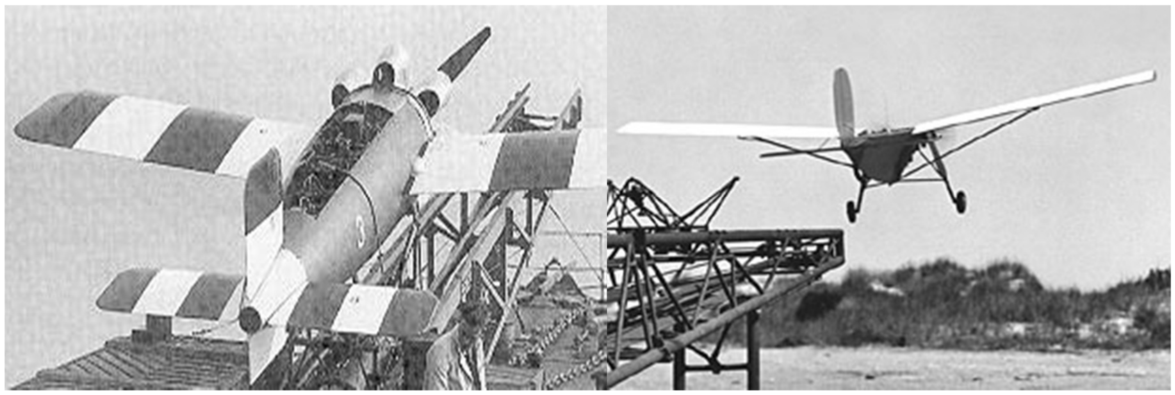


Figure 3: Early Unmanned Aerial Vehicles. From left to right are images of the British Larynx and RP-4. Public Domain

1.2.2 Unmanned Aerial Vehicle Genesis

As the success of teleoperated systems expanded, UAVs began to excel due to their unique characteristics, specifically for targeted military applications in tactical scenarios. World War II brought about the first mass produced UAV in the RP-4 which was based on state of the art RC planes and delivered to the warfighter at a number over 15,000 units. An early model of this system is depicted in Figure 3. In addition, well accepted technology was adapted for tactical missions. One example of this is the use of radio controlled B-17 and B-24 bombers in World War II. Towards the end of the war, radio planes began to be instrumented with RCA cameras resulting in the first successful remote torpedo attack in 1941 by the US Navy's TG-2.

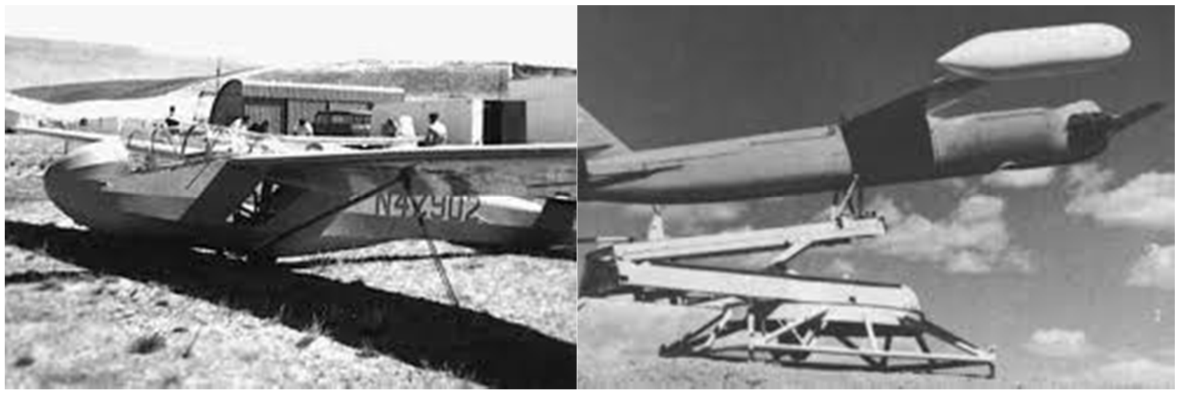


Figure 4: Early tactical UAS. From left to right, TG-2 and MQM-57 Falconer. Public Domain

By the end of the 1950s with the MQM-57 Falconer and toward the Vietnam War with the Ryan Model 147 Lightning Bug, UAVs began to come into their own as reconnaissance platforms. During the war, SAC's 100th Strategic Reconnaissance Wing flew over 3,000 tactical missions over North Vietnam, China, Laos, and other locations over Southeast Asia at lower cost than manned aircraft (Cook, 2007). The use of these UAVs for specific military objectives continued well into the Cold War.

As an extension of UAV based missions, the genesis of VTOL UAS development began in the early 1960s. The US Navy began incorporation of the Drone Anti-Submarine Helicopter (DASH), which was designed to carry nuclear depth charges to destroy enemy submarines. DASH was a dual coaxial rotor helicopter that later developed into the QH-50C for the US Navy in the late 1960s. The motivation of this design was through a deficiency in fixed-wing UAS to accurately track and destroy these covert underwater vessels (Cook, 2007). The abilities to take off in tight quarters, hover, and stare proved invaluable in this tactical situation. VTOL UAS were capable of fulfilling a specific role in the accomplishment of this mission.

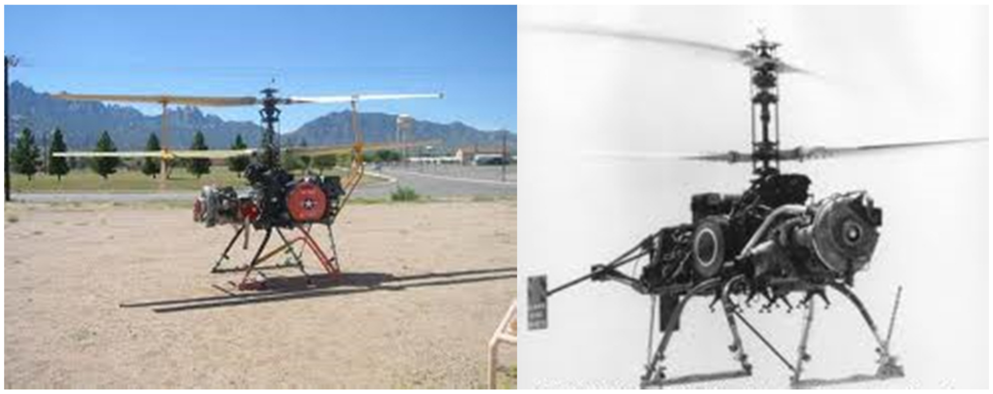


Figure 5: The US Navy Drone Anti-Submarine Helicopter (DASH). Public Domain

1.2.3 Ground Vehicles and Tactical Missions

If UAVs were seen as forays into unmanned systems for tactical purposes, ground vehicles were thought to eventually possess a more streamlined, autonomous approach. This is partly because, as a result of their larger payloads, early ground vehicles carried more diverse and largely more experimental sensor methods. From 1973 to 1981, Hans Moravec led the Stanford Cart project in one of the first explorations into the merits of stereo vision for navigation and obstacle avoidance. The Cart used a single TV camera and moved it to 9 different positions, processing the result at a remote ground station. For military applications, The Defense Advanced Research Project Agency (DARPA) led the charge with its Autonomous Land Vehicle (ALV) initiative (Gage, 1995). The ALV sensor suite included a color video camera and laser scanner capable of returning a 64x256 pixel range image at 1-2 second intervals (Everett, 1995), allowing it to autonomously follow a road at 10 km/h.

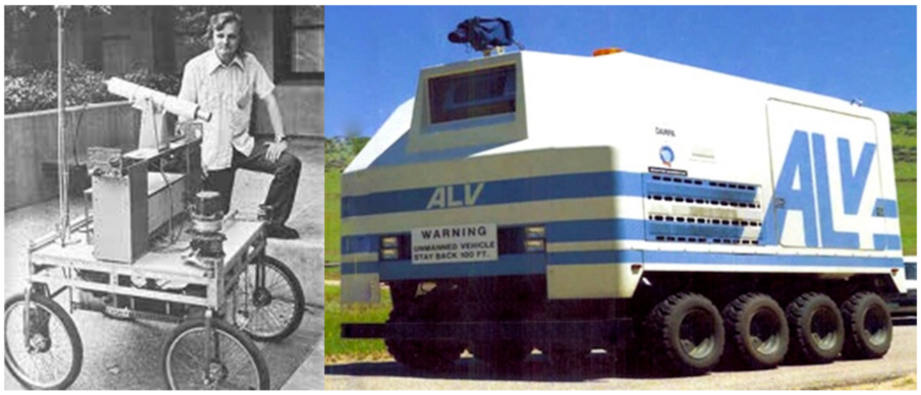


Figure 6: Early UGVs. From left to right, Stanford Cart Project and Darpa Autonomous Land Vehicle. Public Domain

Early successes later branched off into real world, military initiatives such as the US Marine Corps' Ground Surveillance Robot (GSR) and Advanced Teleoperator Technology's Dune Buggy. Both of these program administered contrasting approaches that expanded the use of ground vehicles for tactical missions. In 1986, the GSR was instrumented with an array of ultrasonic sensors and multiple PCs, giving it the capability of following a lead vehicle and walking human (Gage, 1995). Contrarily, the Dune Buggy concentrated on teleoperator control methodology and advancing spatially-correspondent multi-sensory human-robot interfaces. Both of these specified applications required an operator to be in the loop during missions.

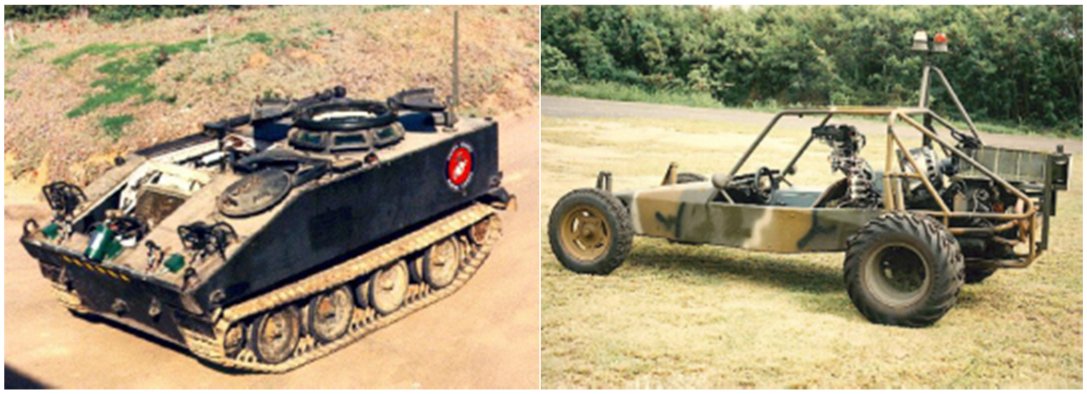


Figure 7: US Marine Corps Ground Surveillance Robot (GSR) and Advanced Teleoperator Technology's Dune Buggy. Public Domain

As a byproduct of these initiatives, the fields of Human-Robot Interaction (HRI) emerged during the mid-1980s as ways of studying how robots fit into the world of humans. HRI is a field that traditionally views autonomy as a means to maximize human interaction with robotic systems. Novel user interfaces with the US Military's GECKO and CARD programs immerged (Gage, 1995). This system allowed an operator to mark his desired driving path on a display with the ground vehicle autonomously following this path.

Through the GECKO and CARD programs in the 1980's, the first MRUAV system, the Airborne Remotely Operated Device (AROD), began its development. This device, along with its successor, the Air-Mobile Ground Security Surveillance System (AMGSSS), is a small ducted-fan UAV deployed for ground based surveillance and target acquisition. Although the incorporation of VTOL UAS was not new, the AROD and AMGSSS are distinguishable for their small size, transportability, and link to extending the capabilities of ground robots with added degrees of freedom.

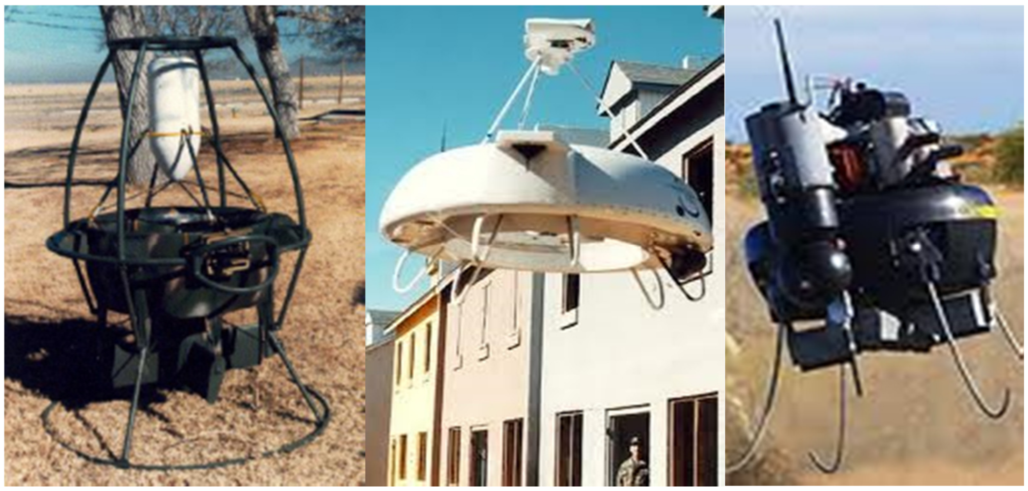


Figure 8: Early MRUAVs. From left to right, AROD and AMGSSS and T-Hawk. Public Domain

Although MRUAVs share common roots with the development of UGVs and UAS, their need is clear through how they have evolved over time. This technology continues to develop as developers begin to leverage more of the unique characteristics. Because of this history, current MRUAVs are seen as extensions of these other more established technologies. They must come into their own as unique platforms with novel, independently developed capabilities. In addition, their possession of some of the positive qualities of both ground and fixed-wing vehicles may make them more suitable for the support of tactical mission objectives.

1.2.4 State of Technology

The U.S. Air Force Scientific Advisory Board report explains "the roles of human operators are undergoing qualitative changes driven by technological changes…because of these transformational factors, the considerable base of human factors knowledge derived from cockpit experience may have limited applicability to future systems" (United States Air Force Scientific Advisory Board, 2004). As vessels for improved battlespace awareness, MRUAVs are an integral part of this evolution. By the year 2015, they are expected to play an even more vital role (Department of Defense, 2009). This maturation process includes not only a change in the nature of unmanned systems but a fundamental shift in the type of information they provide in support of the war fighter.

In order to reach their potential, MRUAVs have unique challenges amongst small tactical teams. To carve out their specific role in the future of unmanned systems, this research improves environmental perception and integration amongst those individuals they are designed to support. For example, war in the 21st century often consists of small, short term skirmishes from an enemy that capitalizes on a shortage of battlespace awareness. MRUAVs potentially provide information to attain an advantage quickly in these situations. The concepts developed in this research through RobiSEND may expand the types of missions MRUAVs conduct in these scenarios.

Streamlining the perceptual capabilities and implementation of MRUAVs requires a broadened conceptualization of what constitutes mission support. Current UAS employ an "over the next hill" approach to reconnaissance that pigeonhole's their usefulness to outdoor, line of sight control. Based on this specific application, technology without the capability of hover and stare is effective. It has the ability to provide macro-level reconnaissance over long ranges, extended periods of time, and in intense wind and weather conditions. This may not be what is necessary to achieve a quick assessment of a tactical team's immediate surroundings.

At low altitudes and in the close quarters of human team members; fixed-wing UAVs, larger VTOL UAS, and ground vehicles have limited capabilities. MRUAVs have the potential to add a dimension of functionality through maneuverability in six DoF. In order to optimize this, the role of MRUAVs must be re-evaluated and expanded beyond adding the capability of hover and stare to missions more suitable for other unmanned systems. Their contribution should take advantage of their unique abilities.

One unique characteristic MRUAVs possess is their potential to be controlled by a single operator. As an example, UAS currently implemented by the US Military require a team of trained specialists, essentially sacrificing the performance of two to three operators in favor of the MRUAV. In fly ahead situations this may be advantageous, however, this is to the detriment of the decision making capacity of soldiers with limited human resources. Because of this, the vision for future UAS in closer proximity with individuals on the ground must involve a minimized MRUAV to operator ratio. In addition, specific tasks and novel operator interfaces must be identified for systems to be a worthwhile addition to mission strategy.

The method and effectiveness of MRUAV perception potentially plays a part in integration but is a major challenge in its own right. Teams seldom have an accurate representation of their environment that reflects targets, goals, or threats. The environments themselves lack a structure that is known well enough to effectively communicate mission strategy. There is a need to re-evaluate the capabilities of current systems and either revamp, restructure, or recast them into solutions that build awareness and function in these types environments.

Local search is a capability that is useful in situations where information is needed of a small tactical team's immediate surroundings. Relationships of team members must be built and maintained that effectively accomplish the mission objectives. The medium of common understanding of spatial relationships must be how both human and robot relate within the environment. RobiSEND demonstrates concepts that extract the maximum potential out of MRUAV search platforms and to provide a level of synergy between MRUAV, operator, and environment that has yet to be realized in current research.

This research provides solutions to potentially minimize unmanned systems limitations in search, to improve system integration for tactical deployment, and to realize the capabilities of these platforms in close proximity of small tactical teams. The RobiSEND concept, when attached to a search platform, will be small enough to be considered part of the typical gear a single individual might operate and carry. The payload sensors are tested for their effectiveness in achieving these goals.

1.2.5 Future of MRUAVs

As stated in Section 1.2.3, MRUAV's have the potential to fit an important niche that is not possible from fixed-wing UAS or ground robots. Fixed-wing UAS are relied upon almost exclusively for GPS based waypoint navigation and a macro-level, exocentric view of the battlefield. They have less complicated dynamics and better endurance than MRUAVs; however, they usually do not have the ability to fly close to the ground, operate reliably in environments where GPS and magnetometer readings are not available, or the capability of hover and stare. These are all characteristics that are essential for search in indoor environments, alleyways, corridors, stairways, or other small enclosures.

On the other side of the unmanned systems spectrum, teleoperated ground vehicles are the unmanned platform of choice for local search. They are less complex to control than MRUAV's, currently more reliable, and have the potential to carry a heavier payload for longer distances. Ultimately, ground vehicles are limited in their contribution to search in that they are restricted to 2D, floor level motion that often impedes the search process. Hybrid systems exist that combine the behaviors of, ground, and fixed-wing unmanned systems; however, they require complex and expensive augmentations in hardware that make them unviable for cheap and reliable deployment.

Current VTOL UAS are limited in their effectiveness in search scenarios because of a lack of design and implementation strategies that minimize the effect of their weaknesses and take advantage of their strengths. Many of these systems are similar to the RQ-16A T-Hawk, which is currently deployed in Operation Iraqi Freedom and Operation Enduring Freedom to scan ahead of convoys for roadside bombs. Although this capability has proven to be useful, this system is for trained specialists in very limited circumstances. A characteristic of these systems that also limits their implementation is their size and weight. The RQ-16A T-Hawk is classified as backpack-able at a weight of 20 lbs. Many troops carry upwards to 150 pounds of equipment, making this weight a significant addition (Task Force Devil Combined Arms Assessment Team, 2003). This size is a necessity for the 45 minutes of flight time needed to support current reconnaissance missions. The system is underutilized in tactical situations beyond this specific application that may not be the best use of hover and stare capabilities.

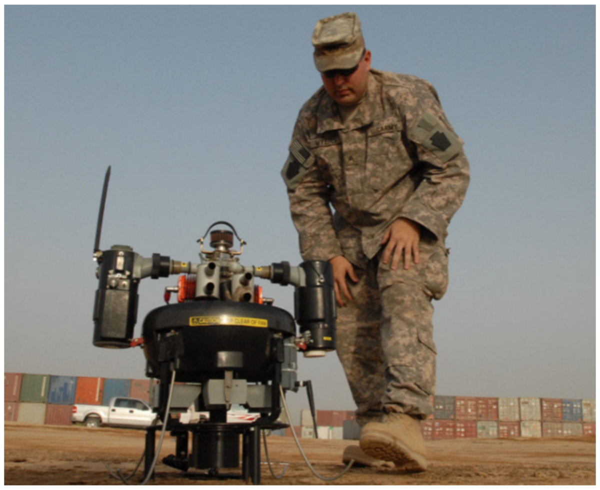


Figure 9: The RQ-16A T-hawk shown to scale next to military personnel. Public Domain

MRUAVs do not combine all of the positive characteristics of ground and fixed-wing UAS; however, their strengths make them suitable for local search operations. On a scale of 2kg, they have a limited flight time of 10-15 minutes and are more complex to control than other unmanned search systems. Although these limitations would make them impractical for macro level reconnaissance, local search would be a better application. This flight time is adequate considering they decrease search time and improve effectiveness by traversing 3 dimensions. In environments that are too dangerous or impossible for a human operator, they may provide the only reliable means to safely accomplish mission objectives. Fundamentally, this recasting of mission capabilities is impossible with current integration strategies and perceptual methods. These largely fail to build workable relationships to the environment or the small tactical teams they support.

1.3 The role of vision

Visual information is central to the approach to relationship building developed in this research. Section 1.3 introduces the concept of using visual information as a medium for information extracted from the environment that is relatable to both human and robotic agents. Sub-section 1 considers the challenges of MRUAVs and how visual sensors may potentially be used to address perception and integration based concerns. Sub-section 2 then gives an overview of the approach taken to build vision based models in this research considering these concerns. The definition of this particular capability is important to establish vision as a necessity for human-robot collaboration in the novel application of MRUAV technology.

1.3.1 Blending Vision for Perception

One way to establish workable relationships between robot, operator, and environment is to design a system with the operator as a source of information. Computer vision may be an ideal sensor both robots and humans can be made to share for understanding of previously unknown environments. For backpack-able MRUAVs with limited payload capacity, vision is a light weight, low cost, passive means of extracting spatial information for enhanced sensory system capabilities. In addition, it is also a necessary component of any system with a user in the loop outside of direct line of sight. Computer vision has challenges including reliability outside of controlled research environments and computational intensity. Nevertheless, its low cost and improvements in computational hardware make it a very attractive means for perception onboard payload limited systems.

Vision also provides an opportunity for expanded HRI by providing the user and MRUAV with a similar perception of the environment. Certain object's visible traits may establish perceptual meanings with shared significance to an MRUAV and operator. This commonality occurs when the object can be related to human and robotic function. For example, an object that is seen by both the MRUAV and operator as being too close for safe passage of the MRUAV results in their shared need to navigate away from the object. Vision is one of the most effective sensors in building these relationships.

This is particularly useful in the establishment of situational awareness during search that makes human-robot collaboration possible. Situational awareness is related to the operator understanding of the MRUAV search system capabilities, structure of the environment, and role in the search process. A visual medium may supply a more intuitive grasp of system capabilities and natural conceptualization of spatial relationships.

1.3.2 Sensor Contributions

To make the robot correspond more directly with the human, an approach is used to build observation models based on sensory information and other sources of intelligence that involve novel algorithms and inclusion of the operator in the loop. For applications where an operator is required to be equally invested in the search function of the unmanned system, it can be advantageous to leverage what both human and robot do well. The models from this approach have components generated from the operator and robot. Both play a unique and vital role to model building that plays to strengths and minimizes weaknesses.

As discussed in the previous section, the visual sensors extraction of spatial relationships establishes a common medium between human and robot. The user interface in this research gives the operator a vision based egocentric perspective with the intent of providing essential relationships to the searchable environment. Although egocentric systems have the drawback of being disorienting after extended periods of time, local search missions on the span of 10-15 minutes are one way to avoid this issue altogether. In addition, an immersive video interface allows the operator to maximize his efficiency navigating the system and searching the environment for objects of interest. When tasks are well defined, it is possible to decrease the number of MRUAV operators to one operator with multiple, yet separated roles. On top of the information provided by Electro-optical sensors, LIDAR provides global relationships that define the boundaries of the environment more precisely. It builds observation models that are relevant to the structure of boundary information.

The map built through this approach is relayed to the operator at the conclusion of the local search operation. Different observations from the operator and sensors onboard the MRUAV play distinctly different roles in the mapping function of the RobiSEND system. Considering the limited payload of MRUAVs, the level of perception and integration necessary is not possible without a comprehensive method fostering cooperation and codependence between robot and human realized through this approach.

1.4 Re-Evaluating the MRUAV in Research

Section 1.4 then establishes the need for the unique approach to human-robot collaboration in this dissertation defined as Operator Blending. The specific need for this novel branch of human-robot collaboration is first established. A more detailed look at the particular departures from the state of the art in unmanned systems for local search purposes is then given in sub-section 2. For each approach, a new objective is established for the MRUAV system of this work under Operator Blending. Examples are given to demonstrate each of these objectives so the need for Operator Blending is well established. Specific contributions are then given with the structure of this dissertation moving forward.

1.4.1 Research Overview

This work is a unique contribution in the development of MRUAV capabilities for real world, local search operations. The concept of UAS supported local search in unstructured environments has been observed sparingly within the literature. To the author's knowledge, it has not had a specific focus on sensor payload development with human-robot collaboration as a source of perception and integration problem solving. The need for research in this area lies in the limited ability of current approaches to perform search missions with meaningful results. MRUAVs provide the means for this because of their unique capabilities. The method of Operator Blending is developed considering sensor fusion, mission integration, hardware fabrication, and improved situational awareness. Although these areas have been studied separately, the entire picture is considered, methods are developed, and opportunities for effective solutions are proposed.

Moreover, the contribution of this work lies in bridging the gap between disciplines. The areas of human-robot collaboration, strategic unmanned systems implementation, system design, and computer vision, to name a few, are dealt with not only in separate studies within the literature, but in the completely different fields. The consolidation of knowledge within human-robot interaction, computer science, and mechatronics is essential for optimal implementation of any system onboard an MRUAV considering the payload constraints and logistics of the conceptualized local search. It is the theory of this work that a solution is not possible without this consolidation of knowledge and the development of an approach that couples many of these challenges. Recent advancements in hardware have expanded the in flight capabilities of these systems so that new and exciting research questions may be posed with ramifications towards perceptual advances and close quartered integration of MRUAVs.

1.4.2 Objectives

Table 1 captures the primary differences of approach between the RobiSEND system and current systems in the literature for localized search involving human-robot collaborative teams. The methods and tools established in this research are a departure from traditional approaches. Each identified departure could potentially be an independent area of research. This dissertation paints a comprehensive picture considering all of these areas and establishes a new research emphasis in human-robot collaboration called Operator Blending. In order to explain the specific role each of these departures has in the operator blending framework, specific examples are given.

Table 1: Departures taken by this research and the most popular current approach

	RobiSEND approach	Literature based approach
1. Communication and data acquisition	Local information	Global information
2. Translational motion	Rotorcraft based	Ground based
3. Information extraction and storage	Intelligent	Iterative
4. Teamming structure	1 Human 1 Robot	Multi Human 1 Robot
5. Primary Information Association	Recognition	Correspondence
6. Mapping structure	Qualitative	Quantitative
7. Principle Components	Functional	Perceptual
8. Human-robot interaction	Codependent	Robot-centered

1. ***Communication and data acquisition:*** The components that make up a map built by a search agent can be constructed from local or global components. Global components are relative to a universal reference, while local coordinates are relative to the body frame of the search agent. Deliverables may be constructed from a combination of either local or global pieces. Depending on their building blocks, they may have starkly different characteristics. The components of maps built from local components are more suitable for human-robot collaborative teams.

   The following example illustrates why. When asking for directions, oftentimes it is more efficient to interpret local as opposed to global information. If communicating global instructions to a teammate, the sub-tasks would sound much like the information given to find a hidden pirate's treasure. "From the Northeast entrance of Lane Stadium, go 1906 paces East, turn towards the North, go 1867 paces, then begin digging". It would be possible to use the pythagorian theory and reduce things to one global command. "From the Northeast entrance of Lane Stadium, go 2668 paces at a heading of 49.595^{o}, then begin digging". If you know the GPS location of the Northeast entrance of Lane Stadium, you know the GPS coordinate of the treasure.

   Although at times it may be simpler to communicate to a robot in these terms, this does not make this type of communication best for human-robot teams. Without help from outside agents like GPS, the quantitative process of counting each pace, its storage relative to the universal starting point, and ensuring the consistent fidelity of each pace length is inefficient and not the most effective manner to accomplish the task. Even when GPS satellites are at the treasure hunter's disposal, there is no guarantee that the triangulation of location information is reliable.

Local observations are more relatable when considering the fact that information must be shared between these agents with differing needs. Instructions based on local information would read "go forward until you reach the third stop light, turn right, and drive until you see the fountain on your left. Dig at the feet of the big Hokie Bird statue." Although this represents a different process to accomplish the same type of task, it is reasonable to say robots are capable of possessing sensors and algorithms that relate to this approach.

Likewise, humans naturally process information in this manner. Unless in the process of having an out of body experience, human beings are egocentric beings and therefore relate more effectively to information packaged locally. This manner of information sharing makes communication more salient for each type of team member. RobiSEND makes communication possible between each by building observation models based on local information.

2. ***Translational Motion:*** Search platforms not restricted to floor level 2D planar motion provide coverage of an unstructured environment even when the conditions are unmanageable by a ground vehicle. Ground based platforms can experience limited visibility, may be disorienting in the presence of unpredictable terrain, and at times may not be able to drive over or around certain obstacles. RobiSEND demonstrates concepts potentially used to augment unmanned systems not restricted to 2D motion.

   As humans, we see the world and negotiate space in three dimensions. Certain environments, however, may be more easily navigated if we were given the ability to fly or hover. In order to manage the disorientatioin that we might experience by walking over an obstacle, going around a sharp bend, or dealing with the inability to continue around a blocked path, humans have the ability to locally develop spatial and heading references that help maintain situational awareness.

This ability becomes more difficult when adding a robotic teammate to the equation. For example, human operators working with robots during rescue efforts have expressed difficulty understanding heading of the robot from a video feed. Even if a robot is able to drive over an object, a robot negotiating a bolder on a slight incline potentially can give a human operator in the loop the wrong impression of its relationship to other objects in the environment. If the camera is fixed to the body frame of the robot, the camera perspective is aimed downward, giving the operator a false perspective.

Considering ground robots also do not have the ability to fly in most cases, they can be considered to have the same types of limitations we do when traversing ground based obstacles. This problem is only compounded when considering the loss of awareness introduced with remote operation. If the local reference to location and heading is not communicated to the human teammate in these circumstances, the robot and human do not function adequately as a team. If the robot is not coupled to the ground then it operates independently to ground terrain.

The good news is that robots, unlike humans, do not have to be ground based. Giving a robot the ability to hover over obstacles or around sharp corners is not a prerequisite for human-robot collaboration. This, however, may make it possible for perception and integration to be observed from a robotic system not limited by traditionally debilitating ground based constraints.

3. ***Information extraction and storage:*** The effectiveness of much heavier and more data intense sensors may be achieved through intelligent use of smaller, simpler sensors if there is an ability to direct sensor attention to what is important in the environment. As a result, the performance of the system does not suffer and smaller, low-power sensors may be used to fit the computational, size, and weight constraints of a payload limited robot. RobiSEND will determine the adequate amount of information needed to solve perception and integration challenges through a novel approach to human-robot collaboration deemed Operator Blending.

Although human beings do not possess the best eyes—eagles see from further, keenest sense of smell—sharks may sense a drop of blood from a mile away, or most effective hearing—bats use sound to catch high speed prey, human sensors provide necessary information for the most intelligent behaviors in the animal kingdom. This intelligence is not a function of the ability to capture more data than other organisms. As discussed in the discussion about Figure 1, the human brain extracts what is perceived to be valuable and associates meaning. More data means more data processing and does not necessarily correlate to more effective performance. The human perceptual system goes to great lengths to actually reduce the amount of heavy processing done on sensory information if the processing is not needed. Attention is directed towards what is important in the environment.

Interpreting previously unknown environments requires multiple sensor modalities but not necessarily the most complex or high performance sensors. Information is only useful if it can be processed for meaning intelligently—a feat accomplished efficiently by humans with vision as our primary sensor. Because of this, the most effective means to direct the sensory system attention of a human-robot team is assumed to be input from the human team member. Operator Blending is introduced to the RobiSEND with this in mind system to solve integration and perception challenges.

4. ***Teamming structure:*** Search is executed most efficiently with a single individual operating the search platform, given that tasks are differentiated through the clear assignment of well-defined flight modes. RobiSEND uses novel concepts in human-systems collaboration that make it feasible to restrict the human-robot ratio to One Human One Robot (OHOR). This arrangement optimizes communication, sytem integration within small tactical teams, and a trust and understanding of a unified perspective amongst teaming agents.

   Even with human based teams, sharing information may become an inefficient and ambiguous process when involving more than two individuals. Oftentimes, the game 'telephone' is used to demonstrate this concept to middle school children. Children sit in a circle and a message is passed from person to person as accurately as possible. When the message gets back to the individual who initiated it, it almost always is completely altered during the information transfer process. There are simply too many variables involved for the message to remain intact.

Robots do what human beings program them to do; however, when the robot must give human beings information on its status or progress, this must be shared amongst those that are mission critical. In addition, human teammates must also understand those commands given to the robotic system by other teammates that impact his ability to perform. A tactical team that must keep multiple human teammates on the same page with the status of the robot has been shown to be difficult to manage.

If new methods are introduced that improve the functional relationships between human-robot collaborative teams, more effort may be spent towards mission performance as opposed to communication, trust, or awareness. A lack of performance in any three of these areas may result in the system being unmanageable. One way to reduce the complexity of team communication is to reduce the number of human teammates that are mission critical to the MRUAV's success. The smallest ratio between human and robot teammates is obviously OHOR.

For human-robot collaborative teams with limited resources, an additional consideration is how to most effectively use team resources. The objective is to spend as much time as possible on the local search mission without dedicating multiple team members to a particular task. If local search is determined to be a necessity, multiple operators must not spend more time gathering information about the state of the robot and environment then accomplishing mission objectives. These factors contribute to the mounting evidence that OHOR is necessary of the proper synergy is to be developed between collaborative human and robotic agents.

5. ***Primary information association:*** Emphasis must be put on the object recognition problem as opposed to the correspondence problem for human-robot collaborative systems. Vision is the sensor method that is most relatable between human and robotic agents for this purpose. For this process to be productive, the attention of the human agent must be devoted to building these relationships during the span of the tactical mission.

   It is widely believed that human beings use over 50% of our brains for vision related tasks. In addition, vision is compartmentalized for sub-tasks such as the general placement of shapes in the environment, lighting distribution, and object recognition. This is why some brain injury victims have the ability to identify shapes but are not able to recognize object types. The part of the brain responsible for visually identifying objects may have been damaged separately from the shape recognition portion.

The human visual system is efficient because these parallel processes are constantly being intelligently emphasized and de-emphasized during a task. For example, when important, we direct resources towards the object recognition problem, which takes away from our ability to do shape recognition. Although shape recognition may be done casually as the human conducts other tasks, object recognition over a consistent amount of time requires more undivided attention. Humans do not have the ability to concentrate on the object recognition problem for extended periods of time without consequences. The attention span of a human being has been studied to be effective no more than 15 minutes.

Vision is a useful tool for a robotic system to conduct various perception related tasks, although, humans have not programmed visual perception to emulate their own. Robots are typically programmed to take computer vision a step further to achieve point to point correspondence between pixels. Much like the robot ability to understand global information, this is at times even a more natural means for robots to interpret data. When given enough computational power, researchers have proven that after establishing the appropriate model, correspondence can drive the mapping process. Sensor data may be used to geometrically reconstruct an environment to the nearest measurement.

The difficulty is in finding the model and the computational power, which has in turn made the correspondence problem widely known as the most difficult problem in computer vision. As mentioned, Mother Nature submitted to the difficulty of this problem as human perceptual systems extrapolate meaning as opposed to correspondence between the smallest possible perceptual unit.

Therefore, for robots to be relatable to humans, emphasis must be given to the object recognition problem and attention direction as a substitute for the correspondence problem. Human limitations also require a limit to the duration of this process if it is to be dominated by the object recognition problem. RobiSEND will emmerse the operator in the local search process for 10-15 minute durations of time by using a direct video feed.

6. ***Mapping structure:*** Although the term 'qualitative mapping' at first glance may appear to be an oxymoron, it is akin to the manner in which biological systems produce mental models of their environments. It is mapping based on the object recognition problem as opposed to the correspondence problem. If the objective of a map is to communicate visually the environmental structure, qualitative maps achieve this by first recognizing the structure of the surroundings and then giving a visual representation that communicates this recognition. This is in contrast to quantitative mapping approaches that measure the dimensions of the environment and attempt to visually represent its spatial dimensions as precisely as possible.

   Qualitative mapping approaches capture the global characteristics of a previously unknown environment because they reflect the process of search as opposed to just communicating spatial relationships and structure. RobiSEND deliverables will contain topological annotations provided by the operator during the search process.

   If emphasis is to be given to the object recognition problem and local measurements are to be the building blocks for the local search process, these processes must be reflected in the produced deliverables. Quantitiative maps, which emphasize metric information, should not be the vessels for this new approach. Most quantitative maps are based on the correspondence problem as this relates to a mixture of local and global information and an iterative process for optimization of this correspondence over time. Scan matching is an approach discussed in this work as the use of purely local information and traditionally implemented without the need for temporal iteration. This method, however, still relies on the correspondence problem to establish the relationship between scans.

   Building a quantitative map based on the aforementioned components would, to a large degree, defeat the purpose, because quantitative maps rely on the correspondence problem. An example of a similar scenario would be using vision as a passive sensor but having to illuminate the test area with LED light. The objective in this case would not be to build an active sensor from a passive one. It would be to develop a passive sensing capability.

Qualitative mapping, as introduced in this research through novel matching approaches, has no tie to the correspondence problem and emphasizes the relationship between observations through the object recognition problem. During map building, this framework allows for easy integration of topological information that further defines these relationships.

7. ***Principle components:*** Oftentimes, it is most natural for human beings to think of principle components as the fundamental characteristics of an observation. Describing something based on its principle components involves identifying the basic descriptors necessary to recognize or characterize its meaning. Beauty, for example, has been found to be based on the identification of several principle components including the distance between the eyes and nose, mouth size, skin color, and eye shape, just to name a few. Eigenvectors and eigenvalues have been used in the past to numerically characterize beaty based on these purely perception based parameters. If the size of one's eyes or the size of one's mouth is known, this information may be used to determine someone's beauty without being given any additional information.

   The definition of principle components based on perceptual parameters, however, ignores an important aspect of characterization. Situational context and objectives play a role in shaping what needs to be known to characterize. A man or woman at the beginning of a night on the town may have drastically different ways of measuring beauty as compared to the end of a night on the town. Having a few beers can drastically change someone's characterization of beauty. There are also more subtle ways in which context and functional relevance impact the selection of principle components.

   For humans, the question of "what is this?" is often asked in tandem with the more sub-conscious question of "what does this mean to me?" Assuming someone identifies all of the perception based principle components to recognize a grizzly bear, this recognition has a very different significance depending on how far the grizzly bear is from that individual at the time of recognition. Instead of having principle components make up a model that was capable of determining the type of bear, it would be more useful to know that one was looking at a bear and how far the bear is from the observation point.

Robots are capable of reliable ways of conducting object recognition if principle components are identified that contribute to fine differentiation between models. For a human-robot collaborative team, shared principle components with a robotic teammate must have the ability to convey a functional understanding that involves both perceptual and contextual differentiations based in the function of the human-robot collaborative team.

8. ***Human-robot interaction:*** Human-robot interaction is currently thought of as robot centered or human centered. Agents act independently and querry a response depending on individual needs. This form of collaboration is useful in certain situations but not so useful in others.

   If a human-robot team have the responsibility of inforcing the rules for a professional baseball game, it may be useful to querry robotic sensors when a call is in question. Outside of these specific instances, play goes on in this case with no input from the robot at all. If a hard hit foul ball is able to be seen on video as hitting the foul pole then it can be ruled a homerun. Likewise, it also may be useful for a robot controlling a subway train to accept input from a human only in situations where there is a safety concern. Most subway trains currently operate in this manner with the human doing very little to impact the transportation process most of the time.

   For the baseball analogy, if the robot was given the responsibility of policing the strike zone and checking to see if balls are in play, then the human could stick to what he does well. In this scenario, the robot would not be accepted standing behind the plate and it would be impossible for it to argue with managers and emotionally throw them out of the game. It is difficult to accept at times what the robot may be capable of doing better. The argument that often is made is that it violates the tradition of the game. A similar argument is sometimes made for traditional approaches to human-robot interaction. There is an approach that emphasizes traditional methods that is difficult to change because it is simply how it has always been done. There is mounting evidence in several fields that an approach emphasizing what both agents do well could potentially be more effective.

   Human-robot collaborative theory may benefit from knowledge generated from control theory, human factors, autonomy, and sensor fusion. These fields, however, are explored with the understanding that solutions reached are optimized for how current systems fit into the world of humans and not necessarily for the future of human-robot collaboration. RobiSEND is developed with knowledge of basic approaches in each of these fields and implemented with concern towards developing a new branch of human-robot collaborative theory.

1.4.3 Specific Contribution

This application changes the approach to some well-established research questions. In addition, there are also new questions that may be posed that are unique to RobiSEND. More general questions that are documented in literature are followed by more specific research questions specific to this application. These questions include

1. What is appropriate human-robot collaboration for unmanned systems designed for the close quarters support of small tactical teams? In what form does this interaction take for MRUAV search systems involved in local search?
2. What approach to an unmanned systems interface is acceptable for human-robot collaborative teams? What is the most effective means to achieve a OHOR ratio when operating a MRUAV?
3. What is a typical mission that is achievable during a local search? How can an area of interest be mapped and annotated during a local search mission?
4. What algorithms, strategies, and sources of information are most appropriate for human-robot teams operating in unstructured environments? How can Operator Blending specifically contribute to the extraction of information and information sharing?
5. How can unmanned systems be made transportable to the war fight, cost efficient, and logistically feasable for local search? How are hardware components of RobiSEND combined and selected to achieve a payload that solves local search based integration concerns?
6. How is useful information, managed, extracted, and intelligently processed from data to produce deliverables that characterize an unknown environment? How is information best presented by map deliverables and function based principle components for Operator Blended human-robot collaborative teams?

All of these questions will be addressed in some capacity during this research. The first three sets of questions will not be studied for optimization. Instead, a workable solution will be found based on finding a more comprehensive solution to the latter 3 questions. Question areas 4-6 are the main focus of this dissertation and will be studied extensively to realize a solution to the challenges proposed. For question set 5, the limited resources available to the human-robot team are considered for system development. These include limited weight, size, and cost, which drive the baseline approach to coupling perception and integration challenges. It is believed that without coupling these issues, system performance may be inadequate. Question set 6 governs the approach to sensing and strategy for mapping. Lastly, question set 4 deals with the creation of the Operator Blending taxonomy to incorporate codependent and cooperative relationships between robots and humans to solve the aforementioned problems.

Some areas logically covered under these research questions are beyond the scope of this work and will be left to be addressed at a later date. These include

1. *Flight Dynamics and control systems engineering*: Integration of the developed sensor payload onboard an actual MRUAV system will not be explored beyond hardware-in-the-loop simulations. Although it is reasonable to assume that development must be done along this path for the behaviors of the RobiSEND system to be fully realized, this work is not concerned with the control laws or dynamics of autonomous systems.
2. *Procurement*: Packaging of the sensor payload or optimization of the communication requirements that would make it resistant to weather or the rigors of a typical mission. The system will not be tested for wear and tear or communication limitations or packaged in a manner that makes the system anything more than a means to demonstrate the relevant concepts of Operator Blending.
3. *Measurement of cognition*: Comparison of the cognitive load involving multiple users to one MRUAV search system or multiple search systems controlled by one user will not be done to the proposed OHOR ratio between MRUAV and operator. For mission integration reasons, a OHOR ratio is assumed to be the only viable option for the level of cooperation needed for MRUAV based local search systems.
4. *Map generation strategies*: Strategies of implementing maps will not be evaluated based on the optimal technique for mission planning. Instead, an effective means of map generation will be introduced through the developed RobiSEND deliverables that introduces topological and metric components to qualitative map generation.

All of these areas are significant to eventually developing a working system, but are not feasibly part of this research study. This leaves several areas of note for MRUAV search systems that this research will address including:

1. *Hardware in the loop design*: The fabrication of sensor payload to qualitatively map and place observations of an unknown, indoor environment. In addition, the fabricated payload will understand its functional relationship to observations and categorize them based on the Code of Observational Genetics (COG) concept created in this research.

2. *Backpack-ability*: Overall cost and weight of the system towards integration into the gear a typical individual might carry. The weight of the payload will be limited to 1kg. Form factor of the sensor payload to interchangeably fit on MRUAV search systems with limited adjustment of hardware or calibration of system components. Cost, weight, and size requirements frame the limited resources of the human-robot team and directly influence the approach to local search through Operator Blending.
3. *Mapping*: The mapping of an unknown area for local mapping operations will be conducted. The ease of merging topological information into the generated map based on the observation based approach will be introduced. Qualitative Unsupervised Intelligent and Collaborative Keypoint (QUICK) mapping is developed in this research to incorporate Operator Blending and COG concepts in a map that reflects the content of the area and the process of local search.
4. *Situational Awareness*: The situational awareness provided by the hardware-in-the-loop system will be considered. The ability of the system to interpret spatial relationships will be observed considering multiple scenarios involving varying degrees of situational awareness. Based on evidence in the literature, Situational Awareness will be improved based on the functional relationships RobiSEND establishes.
5. *Information representation, processing and storage*: An approach will be developed that efficiently saves, interprets, and processes information based on the needs of the human-robot collaborative team.

1.4.4 Deliverables

Ultimately, the following novel deliverables will result from this research and development of RobiSEND

1. Local observation based spatial propagation of a MRUAV in a teaming situation through navigation as the shared basis for communication.
2. Operator Blending theory as a branch of Human-robot interaction that couple perception and integration challenges of human-robot teams.
3. Code of Observational Genetics (COG) to capture the functional role of observations as principle components for the local search and mapping process.
4. User Interface design to solve integration challenges of MRUAVs at low altitude and specifically to reduce operator-MRUAV ratio to OHOR for small tactical teams.
5. QUICK mapping approach to capture metric and topological information from an unknown environment for local search based qualitative mapping.
6. RobiSEND payload concept development to augment a 2kg MRUAV for 15 minute search missions. This payload is designed to demonstrate concepts and support the Operator Blending, COG, and QUICK mapping approaches.

7. Computational approaches rooted in operator directed system attention as they contribute to Operator Blending. Area Intersecting Mean Shift (AIM-Shift) is given as one example.

The novel approaches of this research combine several areas addressed in the literature. There components are covered in a review of research that contributes to the theory behind this work. Because this work looks to fuse considerations of a range of fields, the following list reflects the substance of this literature review. This list includes

1. Human-systems integration and the impact of autonomy on situational awareness
2. User interface design for egocentric teleoperation in unknown environments
3. Perceptual sensors with emphasis on LIDAR and stereo vision systems
4. Object segmentation and categorization using visual data
5. Mapping techniques and considerations for unstructured environments

This research expands on these areas to support its methodology and also merges them in a novel approach to MRUAV search for small tactical teams. Chapter 2, Chapter 3, and Chapter 4 present a review of the relevant literature. Chapter 5 and Chapter 6 cover the novel approaches presented in this work. Chapter 7 presents results, discussion, conclusions, and opportunities for future work.

2. The Human Side: Codependence

"Knowledge comes, but wisdom lingers"—Alfred Lord Tennison

In order for unmanned systems to fulfill human needs in teaming situations, wisdom must be a central part of human-robot collaboration. Knowledge and wisdom, although related, differ in their level of investment in hard work, experience, and maturity. Both require the acquisition and processing of information towards the formulation of judgments. Knowledge has the luxury of coming as fast as one can learn it and may be learned fairly completely in a finite period. Wisdom takes time and its process is ongoing.

The process difference between wisdom and knowledge points to the difference in learning and understanding between human and robotic systems. Robots rely exclusively on finite snapshots of knowledge as opposed to wisdom. It is wisdom that introduces additional dimensions to a problem through unexpected sources of information. It is impossible to completely model wisdom; but, it is a necessity in order to comprehend a human understanding.

2.1 Overview

Section 2.1 establishes the need for a human to build relationships with a robotic agent. Because this chapter concentrates on human concerns, the establishment of this necessity from a human perspective is important. This need is then discussed in sub-section 2 to understand what can be done to align current goals for MRUAVs with the realities of these concerns.

2.1.1 Human role literature

Two agents contribute to human-robot collaboration, the human and the robot, with starkly different needs for learning, function, and interaction. Chapter 2 addresses human central factors that shape the development of human-robot teams. These concerns are half of the equation that leads to the foundation of Operator Blending developed in this thesis. Although the issues presented in this Chapter are pressing if local search based MRUAVs are to be designed to meet human needs, human-robot collaboration literature for these systems is minimal. The evidence for the creation of a new framework for MRUAV local search systems is covered in this Chapter.

Literature more thoroughly documents the concerns of human team members in these relationships through human factors, which answers the question "how do current systems fit within the world of humans?" This question is premature in certain application of human-robot interaction because current systems do not build and have yet to identify the proper relationships. The questions of "how must future systems be designed to build relationships more effectively with humans?" and "What are these relatinships?" are fundamentally different concepts. Progress towards mission effectiveness in human-robot collaboration involves addressing the latter two questions.

The necessity for the MRUAV to build relationships with the human is outlined through several facets of this literature review. The description of what local search means to the human agent, human crewing protocols, the role of autonomy and collaboration, and elements of situational awareness for the operator in teaming situations are outlined. This lays the foundation for the review of computational and algorithmic considerations in the literature addressed in Chapter 3.

2.1.2 Current Role of UAS

For human operators, development of MRUAVs in local search has been identified as a major strategic goal of the Department of Defense (Department of Defense, 2009). Due to the success of current systems in the sphere of combat, fixed-wing UAS are currently being used for fly over missions. Current DoD operational UAS have demonstrated their utility with UAS of 10 different types in military operations (Tvaryanas, Thompson, & Constable, 2005). By 2013, the money allocated for UAVs is projected to be an order of magnitude greater than UGVs and Unmanned Underwater Vehicles (UUVs) combined. In addition, the trend of funding is projected to shift away from research and development and towards procurement. With this logic, robust behaviors must be developed that encompass most of the needs in the sphere of combat. Capabilities must emerge that demonstrate behaviors beyond what is currently conceptualized.

Table 2: President's budget for Unmanned Systems in the Unmanned Systems Roadmap

	Funding Source	Fiscal Year 2009 ($)	Fiscal Year 2010 ($)	Fiscal Year 2011 ($)	Fiscal Year 2012 ($)	Fiscal Year 2013 ($)	Total ($)
Unmanned Ground Vehicles	RDT&E	1,291.2	747.5	136.2	108.7	68.9	2,353
	PROC	33.4	42.3	53.6	59.5	21.1	210
	O&M	2.9	3.9	3.0	12.8	10.1	33
Unmanned Aerial Vehicles	RDT&E	1,347.0	1,305.1	1,076.4	894.0	719.5	5,342
	PROC	1,875.5	2,006.1	1,704.7	1,734.3	1,576.2	8,897
	O&M	154.3	251.7	249.0	274.9	320.2	1,250
Unmanned Underwater Vehicles	RDT&E	57.3	73.8	63.2	70.1	76.9	341
	PROC	56.7	78.4	95.9	91.6	103.7	462
	O&M	5.0	4.5	11.3	13.5	13.9	48
		4,823	4,513	3,393	3,260	2,911	18,900

Where RDT&E stands for Research, Development, Test, and Evaluation, PROC is Procurement, and O&M means Operations and Maintenance.

Today, UAS continue to evolve and must be developed to provide sustainable functionality. As stated by the Office of the Secretary of Defense, “the reliability and sustainability of UASs is vitally important because it underlies their affordability, their mission availability, and their acceptance” (Department of Defense, 2004). In other words, the robust performance of UAS software and hardware is essential because it directly impacts system cost and usefulness. This is of particular importance because usefulness is not mentioned in terms of complex behaviors, but with emphasis on acceptance. On the most fundamental level, the tactical team must build a relationship to the UAS that results in the team embracing the system—an understanding that it will do what it is designed to do in practical circumstances.

Robust perception and integration are the limiting factors (Murphy, Pratt, & Burke, 2008; Lundberg, Christensen, & Hedstrom, 2005; De Visser, LeGoullon, Freedy, Freedy, Weltman, & Parasuraman, 2008). Unfortunately, the development of UAS is most often done without concern for complex environments or considering end user needs. A disconnect exists between researchers that develop UAS technology and those in the field that must rely on these systems (De Visser E. J., 2008). This results in a lack of implementation techniques, innovation, effectiveness, and trust. Researchers have developed higher levels of complexity with little regard to the usability or strategic role of their systems (Ferland, Pomerleau, Dinh, & Michaud, 2009). Operators, on the other hand, have relied on traditional means to solve problems as opposed to providing a clear, informed vision on how to push the capabilities of unmanned systems forward.

One result of this is the limited literature evaluating or detailing the effectiveness of MRUAVs. This is unfortunate, because MRUAVs hold very distinct advantages over fixed-wing, ground, and larger VTOL UAS. New behaviors can be developed that ultimately save human lives. Several factors, however, limit the contribution of MRUAVs to missions that require a more complex approach to situational awareness and utility (Murphy, Pratt, & Burke, 2008).

Because most real mission scenarios involve unstructured environments, some of the ambitious intentions of these systems are not being realized. This is reflected as a void in the literature and a lack of actual systems developed as collaborative team members. The concerns most often found in literature include modeling stable control systems, designing platforms for specific environmental conditions, negotiating inadequate wireless communication, and dealing with insufficient bandwidth for video-based feedback (Chen, Haas, Pillalamarri, & Jaconbson, 2006). These studies, although promising and active research fields in their own right, have not resulted in a system that is reliable for search related tasks (Department of Defense, 2009). Development must follow crewing and operational procedures, design considerations, and techniques utilized to achieve a cost effective, reliable, and trustworthy system.

These challenges are firmly rooted in to world of humans. Chapter 2 concentrates on human concerns in a human-robot collaborative system. This chapter includes the definition of search and human-robot collaborative systems, the role of autonomy, methods to improve situational awareness, crewing and protocols, and user interface design. Chapter 3 will review the technical approaches to human-robot collaboration with emphasis on perception. Techniques in previous research involving sensor fusion, image segmentation, and mapping are given that involve LIDAR, stereo cameras, or a combination of both. These aspects are then discussed considering the limited examples in the literature that also incorporate human-robot collaboration.

2.2 Elements of Search

Considering the human-centric concerns of Section 2.1, this section defines the specific application of local search as the example application used in this research. Local search is used as a simple application that must be cognizant of these factors. Sub-section 1 gives a definition for local search in unknown environments. Sub-section 2 then describes how humans traditionally relate to robotic systems that are part of human-robot teams during local search. Considering the perception and integration concerns that are specific to local search, the second half of this section then presents intelligent information gathering as a potential solution. The argument is made that if human and robotic agents are to relate in a new form of human-robot collaboration that allows for the contributions of both agents based on what each does well, intelligent information gathering must be present from both.

2.2.1 Search in Unknown Environments

As mission support systems, unmanned systems are currently most naturally used as reconnaissance platforms. From a robotic standpoint, reconnaissance involves mobility, camera control, and information extraction from sensors. Subtasks include passive perception through the interpretation of sensor data, quantitative judgments of observations, estimates of the velocity of ego motion, and the location of objects of interest within the environment (Fong, Thorpe, & Baur, 2002). These problems are traditionally addressed in the field of control systems engineering—modeling the system and environment to make intelligent inferences about performance.

Human operators conduct similar tasks but do not understand reconnaissance in the same way. Humans introduce context and significance to the reconnaissance process that makes it relevant to higher level team goals. This higher level information is what transforms a reconnaissance mission to a search mission. Search involves relationships built with the human operator and may be defined as the mapping of an object of interest for the purpose of improved actionable intelligence or functionality.

Operators must understand their relationship to the environment and role in the search process. In unstructured environments, Murphy et al. 2008 recommends crewing and operational protocols for tactical teams utilizing UAS. This theory may be generalized for MRUAVs. They define a localized tactical team as one where the operators of the system are the primary consumers of information and are physically co-located with the MRUAV.

A team conducting search localized with an unmanned system must be knowledgeable of many of the additional safety concerns. Local search and reconnaissance often assume the team is not localized to the unmanned system for this reason. The comfort level of the system operator with his general safety operating an MRUAV may have a major effect in his situational awareness. These systems operate under the following truths:

- The close urban environment is the MRUAVs primary workspace.
- The workspace is either near a complex structure, such as bridge, or in the cluttered region on the sides of an urban canyon rather than in the comparatively open area above streets
- The MRUAV may need to physically cover the entire workspace.
- The MRUAV remains below the altitude for manned aircraft.
- A high risk of collision is prevalent, as the MRUAV workspace is less than a few multiples of the vehicle's length from the structure or associated obstacles.
- The obstacles in the workspace cannot be accurately modeled a priori due to scale
- Human intrusion into MRUAV workspace is possible

This list leads to several challenges inherent in the operation of MRUAVs that potentially occupy these spaces. The system must be able to adequately perceive structures, obstacles, and human presence reliably with safety as a primary consideration. In addition, traditional approaches to human-robot interaction must be re-evaluated to ensure MRUAVs develop relationships that are effective considering these truths.

2.2.2 Crewing Protocols

One way relationships are built with unmanned systems is through crewing protocols. When an operator or group of operators are the primary decision makers, crewing protocols are developed to organize control decisions. It effectively allows multiple humans to build simple relationships to the MRUAV and environment. In order to ensure human operators understand their roles, they oversee subtasks of the system.

Three traditional roles are defined in a team of MRUAV operators (Murphy, Pratt, & Burke, 2008). The first notable role is that of the pilot, who is responsible for teleoperation of the vehicle, assessing general airworthiness of the MRUAV, and general maintenance. This role usually demands a high degree of training and concentration in the vicinity of structures and less than ideal weather conditions. The second role is fulfilled by the mission specialist, who collects data and directs the pilot where to go in order to frame a still image or gather the necessary video footage. He has the added burden of assessing mission objective completeness and must control the capabilities of the camera. Lastly, the flight director fills the role of safety officer for the wellbeing of team members, civilians, and the MRUAV. This individual possesses the ability to understand the progress of the team within the framework of overarching mission objectives.

Roles are also defined that do not correspond directly to the operation of the MRUAV (Scholtz, 2003). These are teammates not operating the system, mechanics designated to repair the system, and bystanders who are not affiliated with the tactical team. Therefore, a team of individuals can realistically assign 3-6 roles in the operation of an unmanned system. In reality, these jobs are sometimes seen as fluid (Barnes, Knap, Tillman, Walters, & Velicki, 2000; Williams, 2004) and many teaming situations do not have the resources to sacrifice multiple team members for the function of one unmanned system. The effectiveness of the team to accomplish tasks is ultimately the most important factor when determining future roles.

The reliability and efficiency in accomplishing these tasks are dependent on the system addressing training, procedural, focus, and motivation issues inherent in system integration. Opportunities are abound to leverage modes of autonomy, interface enhancements, and memory aids for more effective cognitive skills and knowledge base expansion (Tvaryanas, Thompson, & Constable, 2005). In addition, if human-robot teaming is the goal, these improvements must be studied as a means to build practical relationships. When these advances are considered, they potentially improve the quality of tasks systems are capable of achieving.

2.2.3 Intelligent Information Gathering Protocols

An additional approach for perceptual efficiency and trustworthy integration is to provide the system with intelligent information gather capabilities. Crewing protocols simplify complex behaviors in discretized, human controlled sub-tasks. Intelligent information gathering is also a means of simplification but has a different basis. This is the ability to direct the system towards what information is important, interpret this understanding efficiently, and to make informed actions for a functional advantage. The mission is not divided into sub-tasks that are the most repeatable, but rather organized based on efficiency and necessity. Biological systems thrive on their ability to accomplish this quickly with limited computational power.

For example, when a baker makes a cake, he or she doesn't measure one ingredient. It would be absolutely impossible for the baker to incorporate multiple individuals in the baking process. An understanding exists for what is needed, how much of it is needed, which components are essential to have a steady hand with, and which ones she can take more liberty with to change up the recipe. This is an example of using intelligent information gathering protocols. Although decisions have to be more centralized, they can be made more efficiently based on experience and the context of the situation. On the other hand, if a family were to make a similar cake from a recipe, their instructions would be specific and discretized. It would be advantageous to use crewing protocols to divide tasks amongst multiple family members. Which approach would produce the better cake? It depends on what is important. The family can most probably produce the cake faster, but the baker's cake would have better taste.

Intelligent information gathering is often necessary when tasks become too complex to be divided up in an efficient manner. If instead of baking a cake, the task was to play Blues on piano. There is no logical way to divide this task up amongst multiple team members even if there were enough to assign each to one note. It would be best to allow an individual who is trained to read key changes, play scales, and improvise. For MRUAVs it has been assumed that intelligent information gathering protocols are not as effective as crewing protocols. This notion is challenged in this research.

2.2.4 Intelligent Information Gathering Principles

Four main principles determine the theory for intelligent information-gathering systems: the ability for the system to be task-directed, probabilistic, computationally attentive, and to possess an efficient semantic formulation (Hager, 1990). Current unmanned systems currently have well defined probabilistic approaches rooted in Bayesian theories; however, they are deficient in the other three areas of intelligent information gathering. As a result, they favor systems that operate efficiently only in mostly controlled circumstances.

In fact, biological systems seldomly make decisions based on probability. Likelihood of success is jettisoned in favor of situational awareness to address an immediate need. Probabilistic calculations are reserved for programmable systems because it is thought that correspondence between inputs and outputs is always possible. The issue comes when the inputs and the outputs cannot be effectively isolated causing the model to be untrustworthy. The answer in control systems is to just look for a better model; however, much thought is not given to the idea that a basis solely in probability is incomplete. Although noise and uncertainty are worth modeling, determining a functional understanding through the three most neglected principles of intelligent information gathering can be more useful. If the environment is unpredictable, it can cause a de-emphasis of probabilistic approaches.

The neglected principles reflect the ability to reduce the amount of information for efficiency. Task-direction is the ability to make inferences about smaller groupings of features or tasks as a sum of their respective parts. Attention involves directing the system towards those objects or features in the environment of most importance. Efficient semantic formulation deals with simplification of information through conversion to a task relatable format.

The understanding of probabilistic principles involves calculating the likelihood of outcomes. The majority of current solutions to perception and integration challenges are firmly rooted in these theories. The approach of this research is to incorporate task-direction, directed attention, and semantic formulations to build models that are amenable to human-robot collaborative teams under Operator Blending. These models are more apt to solve perception and integration challenges.

2.2.5 Role of Full Autonomy

Increasing autonomy is often seen as the first method to implement crewing protocols or intelligent information gathering methods. The human has the mentality of letting the robot function independently or report back after the task is completed. The general logic of this approach is that the robotic system can and should be made the central agent in solving problems. If the robot is a part of a team, this assumes that it is capable of completing sub-tasks or operating independently and relating outcomes to the humans it is designed to support.

The effectiveness of these methods to build relationships with humans is often measured through the field of human factors. Outside of these studies that observe how existing technology fits in the world of humans, studies into the appropriate levels of autonomy are not found that discuss development of new technology. A byproduct of this is the lack of a strong basis in the literature for MRUAV centered research. In spite of this, conclusions may be reached about the integration of MRUAVs through the qualification of other unmanned systems based studies on autonomy.

The majority of autonomy analysis is based primarily on fixed-wing UAS, ground robots, and, to a lesser extent, large scale VTOL UAS. Fixed-wing UAS are tailored towards GPS based search of large outdoor environments as opposed to ground vehicles, which are generally designed for egocentric, in-planar motion. In many respects, ground vehicle search systems are applicable to the egocentric search conceptualized in this research. MRUAVs also utilize slow, controlled motions to reposition themselves as the robot comes to a complete stop or hover condition.

Ground vehicles, however, do not require the same operational concerns or capabilities of MRUAVs that may be more relevant to how autonomy is actually utilized. UAS fly and are therefore are reliant on many of the truths outlined by Murphy. Fixed-wing systems and larger VTOL UAS may be more closely representative of what has been done with respect to crewing protocols and operational concerns. Unmanned systems literature, however, does not often differentiate MRUAVs. Therefore, the best understanding of a MRUAV specific application comes from analysis of fixed-wing UAS and ground vehicles.

2.3 Autonomy

Particular emphasis is given to autonomy in Section 2.3, because it is such an integral part of how unmanned systems are thought about and developed as the technology moves forward. Sub-section 1 establishes autonomy as a tool that may satisfy several human needs for robotic systems. Sub-section 2 then establishes that human needs and requirements are not stagnant and therefore change based on mission and team circumstances. Adjustable autonomy is introduced in this sub-section as a potential solution to real world problems that demonstrate these characteristics. The reader will be shown in Chapter 5 that for an Operator Blended system, adjustable autonomy may be achieved through the inclusion of a flexible user interface that takes the concepts discussed in this section into account.

2.3.1 Modes of Autonomy

For UAS and ground vehicles, autonomy has been found to be useful under diverse circumstances and through its strategic use. The study of the nature of autonomy for unmanned systems continues to evolve as an open research question, particularly for systems designed to be in the close proximity of small tactical teams.

Several conventions introduce modes of control based on the autonomy level of the system (Parasuraman, Sheridan, & Wickens, 2000). For this research, the convention by (Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005) is used. Tele mode is the fully manual mode where the user controls all robot movement, Safe mode allows the user to control robot movement except in the case where the robot must take initiative to avoid obstacles, Shared mode gives the robot the ability to relieve the operator of direct control depending on its perception of the environment, and Autonomous mode establishes the robot as the primary decision maker, allowing input from the user if necessary at the tasking level. All of these modes require a user to have a basic understanding of the unmanned systems intentions during the mission.

Because this research addresses local search applications, Autonomous mode is not as useful as other modes (Parasuraman, Sheridan, & Wickens, 2000; Barnes, Knap, Tillman, Walters, & Velicki, 2000; Molloy & Parasuraman, 1996) and is not studied in favor of modes of autonomy that emphasize teamwork, where the robot is seen as part of the team (Murphy, Pratt, & Burke, 2008; Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005). For search, MRUAVs must develop relationships to human team members, targets, and civilians to be viable. This requires interaction with the system beyond the tasking level, primarily in Safe mode, Shared mode, or in some combination of both (Kaber & Endsley, 2004; De Visser, LeGoullon, Freedy, Freedy, Weltman, & Parasuraman, 2008).

Studies exist that make claims for and against autonomy in Shared mode and Safe mode. Increased autonomy for collision avoidance and take-off and landing for fixed-wing UAS may be beneficial to reduce demands on pilots (Murphy, Pratt, & Burke, 2008). In addition, mission specialists may locate targets more effectively from a ground robot with increased autonomy (Dixon & Wickens, 2003). However, the effectiveness of autonomy is difficult to assess because many studies do not test systems in truly multitasking environments (Chen, Haas, Pillalamarri, & Jaconbson, 2006). Negative effects of multitasking in unstructured environments with a system in Shared mode or Safe mode include reduced situation awareness, inappropriate trust, unbalanced workload, decision biases, overreliance, and complacency (Lee & See, 2004; Sarter, Woods, & Billings, 1997; Thomas & Wickens, 2000). These effects may be unavoidable; but, limited studies in the literature adequately deal with these concerns.

Autonomy may also deprive the operator of creating the necessary synergy with the system to understand his role as a respondent when the system malfunctions. Skill degradation occurs over time, which becomes problematic when the operator is forced to operate the system in crisis situations (Chen, Haas, Pillalamarri, & Jaconbson, 2006). If the human operator is not driving, the operator may lose situational awareness simply based on his lack of ongoing knowledge about the behavior of the robot. In the event where the operator must take back control from the robot after experiencing a lag in situation awareness, the result is often confusion or lack of mission effectiveness (Yanco, Drury, & Scholtz, 2004).

Most unmanned mobile robots are currently controlled by a form of Tele mode or Safe mode, such that the robot is part of the mission as a passive tool (Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005). Operation in Tele mode is a challenge due to difficulty judging distances and detecting obstacles (Fong, Thorpe, & Baur, 2002). This is also combined with a higher degree of hand/eye coordination to command the robot and the possibility of high cognitive load. Nevertheless, because of safety considerations, teleoperation is relied upon almost exclusively with a UAS in the vicinity of a small tactical team (Murphy, Pratt, & Burke, 2008).

2.3.2 Adjustable Autonomy

Most intelligent systems do not utilize autonomy in a rigid, inadaptable manner. Adjustable autonomy is autonomy in a system that is flexible and responsive to user needs, environmental demands, and context (De Visser, LeGoullon, Freedy, Freedy, Weltman, & Parasuraman, 2008). The balance of workload with situational awareness is the main objective of these systems (Kaber & Riley, 1999). It has been studied as a way to apply the most situational specific level of robot initiative to maximize the functionality of the human robot team.

Adjustable autonomy may further be split into adaptable autonomy and adaptive autonomy (Oppermann, 1994). Adaptable autonomy is autonomy that can be modified by the user whereas adaptive autonomy is autonomy invoked by the system upon the occurrence of a specific event. In terms of adaptive autonomy, three major categories of invocation exist: actions based on critical events, measurement of the operator's cognitive state, or the predefined system model, (De Visser, LeGoullon, Freedy, Freedy, Weltman, & Parasuraman, 2008; Inagaki, 2003). An operator's cognitive state is difficult to measure and therefore model accurately. Many systems, however, have used critical events and the system model to determine the initiation of autonomous behavior.

In observing its role in Shared mode and Safe mode, automation should be used for the purpose of sustained collaborative interaction (Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005) that often involves varying degrees of sub-system control by the robot (Parasuraman, Sheridan, & Wickens, 2000). When considered from this standpoint, the key question remains what parts of the system is automation most effective to support this collaboration. In spite of when automation is started or what role it plays in the system, it should ultimately consider the plans and intentions of those it is designed to support (Batkiewicz, et al., 2006).

2.4: Human-robot Collaboration

Human-robot collaboration is introduced in this section as a means to develop relationships between robotic and human team members when full autonomy is not capable of establishing them. Sub-section 1 gives traditional approaches to human-robot collaboration that involve human-centered and robot-centered approaches. The need to establish a new branch of human-robot collaborative theory is then presented in sub-section 2. This new branch is Operator Blending and is one of the major contributions of this research.

2.4.1: Traditional Approaches

In Shared mode and Safe mode, the most effective human-robot teams are found to be those that utilize a collaborative model of the environment and task (Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005; Hoffman & Breazeal, 2004). However, there have been limited efforts to formalize this collaboration. If it is incorporated in the system model, collaboration is usually observed through modeling when information should be shared and what information is necessary (Kaupp, Makarenko, & Durrant-Whyte, 2010). This is called collaborative human-robot perception and is said to have the capability to build complex world models, essential for both automated and human decision making. As opposed to being useful for one agent, the deliverables produced may be useful for both robot and human teammates. Additionally, they do not require shoulder to shoulder interaction between the two as full collaborators in task completion. The needs of the human or robot always take presidence.

In collaborative human-robot perception, the assumption is made that robots and humans must interact as peers (Fong, Thorpe, & Baur, 2002; Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005). Considering this, the amount of control given to agents contributes to what constitutes the peer to peer relationship. The two largest categorizations are robot-centered collaboration and operator-centered collaboration (Hoffman & Breazeal, 2004; Fong, Thorpe, & Baur, 2002). Robot-centered human-robot collaboration takes existing probabilistic robotics algorithms and the addition of the human as a tool for a more complete understanding of the environment (Fong, Thorpe, & Baur, 2002). What to communicate is determined by the probabilistic model and when to communicate is specified by the robot unless requests are given explicitly by the operator. The example given in Chapter 1 was that of Airbus's robot central emphasis to aviation. The model determines if the additional information provided by the human is viable and only requests communication on its own if the cost of communication is calculated to be beneficial (Kaupp, Makarenko, & Durrant-Whyte, 2010). This cost is calculated through Value-Of-Information theory (VOI) and may be considered a form of adaptive adjustable autonomy (Kaupp, Makarenko, & Durrant-Whyte, 2010). Issues with this approach are related to maintaining situational awareness without a clearly defined role for the human in the successes, failures, or mission goals of the robot.

User-centered human-robot collaboration models the robot as an extension of the humans it is designed to support (Gockley, et al., 2005). The operators actively participate in the robot's task with decision making control ultimately in the hands of humans (Gockley, et al., 2005). The robot does not truly function as an equal partner, because it has no means of making decisions for itself. In addition, tasks are not tackled cooperatively. Instead, the human allocates tasks in the event he deems the robot more efficient in solving them (Gockley, et al., 2005). And example of a human-centered approach is given in Chapter 1 as it relates to Boeing's approach to commercial aviation. The void in the literature exists when we consider systems that rely equally on human and robotic contributions.

2.4.2: New Approaches to Collaboration

Recent approaches stress the need for humans and robots to work towards a shared goal, establish salient communication, and to be in agreement for current and future plans (Hoffman & Breazeal, 2004). These approaches stem from Joint Intention Theory, which makes the case that for true collaboration to be realized, teammates must communicate to maintain shared beliefs and action coordination towards common goals (Cohen & Levesque, 1991; Bratman, 1992). This is because of their claim that collaborative plans do not reduce to the sum of individual plans. Instead, they exist and may only be interpreted through the context of the overarching objectives that impact both agents (Hoffman & Breazeal, 2004). These researchers aim to establish a sense of partnership that occurs when team members jointly work together as opposed to either being used as a tool for the other.

In order to establish this partnership, this unique form of human-robot collaboration makes the case that team members must have a shared basis (Clark, 1996). This is the understanding to communicate an action that is mutually accessible to all team members. Common ground is necessary with respect to the strata of objects involved in the action, the state of the current task, and the internal states of team members (Clark, 1996). Coordination devices provide the source of shared basis that is the foundation for this common ground. These are actions, initiated or completed by one agent, for the purpose of interpretation by another. Upon task completion, this work makes the argument that there must be a coordination device for a mutual understanding referred to as joint closure.

These approaches aim to establish true collaboration between human and robot team members by allocating jointly relevant tasks with respect to the needs of both agents. This results in a turn based decision making process that weighs the benefits of action by each agent and waits for coordination devices to indicate a common understanding. The robot has a means to determine which team member is better at achieving the current goal and allocates this responsibility accordingly. In addition, the robot may take into account the work done by a human on a task independent of the robot task and weigh its decisions based on his progress. Thus, this is a system that assumes a commitment from the human and robot to trust each may contribute to mission success. Depending on the nature of the task, the robot's future actions may be independent of the task result or reevaluated based on the consequences to overarching mission objective joint closure.

Human-robot interaction based on Joint Intention Theory is believed by this work to be a third branch of human-robot collaboration. A fourth branch involving codependent, rather than committed, agents under Joint Intention Theory should exist. This is operator blended collaborative human robot perception, referred to as Operator Blending from this point forward in this thesis. Like current collaboration theory, the task itself is understood through a shared basis. However, as opposed to a decision making process allocating tasks based on decisions from a human or robot, certain elements of the task are assumed to be best completed based on what each team member does more efficiently. Therefore, independent tasks are not allocated—robot and human complete tasks by both making their individual contributions. The robot's success is directly tied to the human's success and vice versa. In addition, the shared basis is assumed to be restricted to all commands that are semantically relevant, forcing communication to have a direct application to intelligent information gathering.

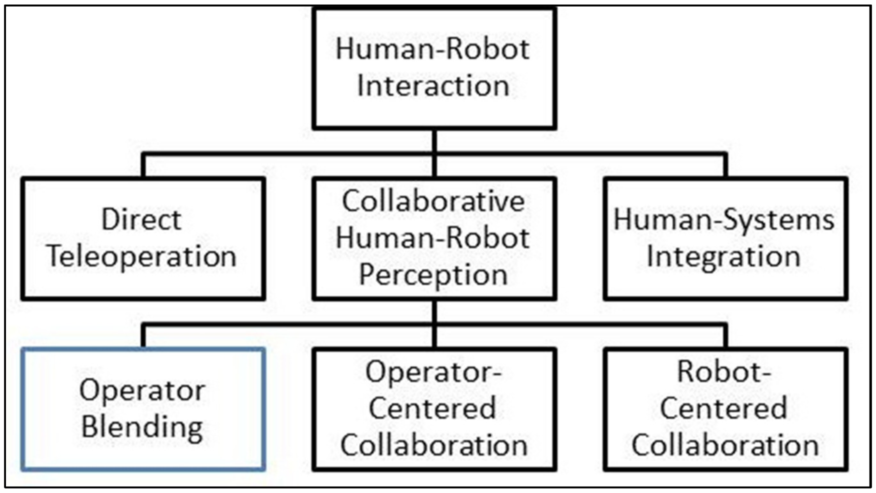

Human-Robot Interaction
Direct Teleoperation
Collaborative Human-Robot Perception
Human-Systems Integration
Operator Blending
Operator-Centered Collaboration
Robot-Centered Collaboration

Figure 10: Human-robot interaction branches of research

The proposed fourth branch draws upon aspects of the aforementioned previous three. Like user-centered human-robot collaboration models, the robot makes all decisions with the involvement of the operator. In addition, similar to robot-centered human-robot collaboration, the robot is actively accomplishing tasks through Shared mode autonomy. The nature of the operator role in the shared mode, however, is that of a codependent relationship. This forces the decision making process and function of the system to be cooperative as opposed to dominated by the needs of the operator or the robot. The operator thus has more than a commitment to contribute to the function of the robotic system. This approach will be discussed at length in Chapter 4.

2.5 Allocation of MRUAV Resources

Operator Blending in this research is established as a means to establish codependent and cooperative relationships between a single operator and single robotic agent. Other crewing protocol between the number of human and robotic agents are more prevalent in the literature, but OHOR is outlined as the best situation for the types of relationships necessary for Operator Blending considering the needs of human team members. This section outlines the pluses and minuses of the most popular teaming structures and presents the evidence for OHOR and the most practical for Operator Blending.

2.5.1 General Concerns

Increased autonomy, intelligent information gathering, or crewing protocols are means to generate an improved situational awareness (Hoffman & Breazeal, 2004). For human-robot collaborative teams in Shared mode or Safe mode, intelligent information gathering should take president over the other two approaches. The collaboration should be the source of intelligent behavior.

As discussed previously, without the relationships that can be built through human-robot collaboration, crewing protocols or increased autonomy are viewed as the areas of development. For current systems, a high human to robot ratio through crewing protocols has mitigated the lack of relationship between operator and human. For future systems, autonomy is seen as the solution. “If unmanned systems could only be self-sufficient, missions could allow them to contribute separately from concerns of human team members.”

Since full autonomy has its own issues as discussed previously, current systems are left with different ways of looking at crewing protocols to solve human-robot interaction problems. Yanco and Drury (2002) outline 8 team constructions including one human-one robot, one human-multiple robots, human team-one robot, and human team-multiple robots, all of which possess unique challenges and characteristics.

2.5.2 Single human, Multi-Robot

Giving a single human operator multiple assets is a major factor in system integration when the individual must micro-manage multiple unmanned systems (Batkiewicz, et al., 2006; Chen, Haas, Pillalamarri, & Jaconbson, 2006; Dixon & Wickens, 2003). In addition, handling of multiple robots or multitasking may result in a loss of situational awareness related to not understanding context of what the robot was doing before human intervention (Olsen & Goodrich, 2003) At the same time, the future role of operators is seen by some in the literature as more of a manager of independent autonomous systems than an actual operator (Batkiewicz, et al., 2006; Dixon & Wickens, 2003). An individual in the role of a safety officer or mission specialist may be able to coordinate the tasks of multiple robots if their relationships to other team members remove this individual from direct control (De Visser, LeGoullon, Freedy, Freedy, Weltman, & Parasuraman, 2008).

Furthering this claim, some research suggests a single operator may control more than one UAS under idealized conditions that include closely coordinated and correlated vehicle activities, a stable environment, and reliable automation (Cummings & Guerlain; Dixon & Wickens, 2003). Forms of human-robot collaboration may also be a means to achieve single operator control because of the operators ability to adapt a shared basis. Currently no research, however, supports this claim.

Other studies state the existence of a major reduction in performance when a heavy demand is imposed (Barnes, Knap, Tillman, Walters, & Velicki, 2000; Dixon & Wickens, 2003). In addition, some claim that any multitasking at all is detrimental to human performance independent of the modes of autonomy introduced to the system (Chen, Haas, Pillalamarri, & Jaconbson, 2006). Chen and Joyner 2009 test this hypothesis on a small tactical team in a true multitasking environment. They found that the gunner's target acquisition was best when monitoring and worst when teleoperating the UGV, leading the authors to conclude that multitasking should be limited to monitoring activities.

2.5.3 Multi-Team, One Robot

As discussed previously, one school of thought is that allocating multiple humans to one robot is the best way to eliminate multitasking. The human to robot ratio most commonly accepted and researched is human team-one robot with at least one operator on navigation and one on payload control (Van Erp & Padmos, 2003). Murphy 2004 states that the human team-one robot ratio is driven by logistics in transporting, maintaining, and operating the robot and therefore, as autonomy is increased, requirements for the size of the human team do not necessarily decrease. Instead, the role of the team members in operating the robot may become less strenuous and distracting from other tasks. Specifically studying fixed-wing UAV's Van Erp (1999) concludes that consolidating vehicle and payload control forces an operator to be too focused on targeting, losing situational awareness and vehicle control. This format is, to a large degree, standard operations for MRUAVs and is accompanied with allocating the traditional roles of crewing protocols amongst human team members.

At the same time, inherent issues with human team-one robot operation exist, particularly within small tactical teams. It is unclear how, or if, the roles of pilot, payload operator, and mission planner can be safely combined and therefore this should be considered an open research question with MRUAV systems (Murphy, Pratt, & Burke, 2008). Drury (2007) concludes that in a team scenario, the pilot must be aware of every facet of UAS operation, including the status of payload operation. Therefore, it may be to his detriment to not be knowledgeable of information excusive to the payload operator. Moving from the individual to multiple operators may be problematic, because an additional layer of cognitive requirements are introduced to get team members on the same page (Tvaryanas, Thompson, & Constable, 2005).

In addition, team members are not necessarily guaranteed to be collocated. Issues with physical dispersion on team workload and performance are issues that have not been well researched (Cooke, DeJoode, Pedersen, Gorman, O.Connor, & AnKiekel, 2004) with breakdowns in team performance, cooperation, communication, and trust contributing to UAS mishaps (Tvaryanas, Thompson, & Constable, 2005). Just as system performance may be degraded from a fight for control between human and robot in Shared mode, potential for similar issues between two human operators is also possible (Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005). These concerns influence the ability to function with multiple operators.

2.5.4 One Human, One Robot

Integration suffers in varying degrees from issues present in human team-one robot and one human-multiple robot scenarios. These issues may possess a multiplier effect when the complexities of more involved team architectures are introduced in real scenarios. An added level of organization must exist for changeovers or complexities in tasks not to result in decreased situational awareness.

The most effective approach within a small tactical team may be to allocate one robot resource to one operator (Murphy, Pratt, & Burke, 2008). This assumes that the traditional roles outlined in subsequent sections are able to be combined with acceptable levels of situational awareness. The benefit in this approach is the operator's ability to understand his role and the information gathered for tactical mission objectives and for him to be integrated into larger mission objectives. Although several theories persist, all findings echo that remote perception remains a fundamental challenge for unmanned systems (Murphy R. , 2004; Chen, Haas, Pillalamarri, & Jaconbson, 2006). These challenges must be uniquely addressed when considering MRUAV local search missions and strategies for collaboration towards intelligent information gathering.

2.6 Situational Awareness

Although situational awareness is not measured in this research and is left for a human factors based study, the theory of Operator Blending is understood to impact the situational awareness of operators for the specific application mention in this research. Section 2.6 discusses the meaning of situational awareness and what it means for the perception and integration potential of MRUAV local search systems. The purpose of the RobiSEND system is to improve the understanding of tactical teams in local search scenarios. Therefore, RobiSEND should be conceptualized to have a positive impact on situational awareness. Operator Blending addresses the perception and integration concerns of human-robot collaborative teams. In order to understand its potential impact on situational awareness, sub-section 3 categorizes and defines how situational awareness manifests itself for MRUAVs. This framework, taken and modified from previous research is used to guide development of Operator Blending

2.6.1 Defining Awareness

At the core of improving this for MRUAVs is the concept of situational awareness. Drury (2003) modifies an earlier definition of SA by Endsley (1988), defining it for robotic systems as "the perception of the robots' location, surroundings, and status; the comprehension of their meaning; and the projection of how the robot will behave in the near future." The primary issue with current approaches to this is their failure to expand object and obstacle location to a shared meaning with human operator. It is this human operator for which the robotic system is designed to support.

In order to realize the potential benefits of collaborative control, the human operator must be able to understand robot intentionality and predict behavior. Operators must have knowledge of the robot that goes beyond location, surroundings and status (Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005). In Shared mode and Safe mode, MRUAV initiatives and behavior must be understood to be rational as if the robot was an actual team member (Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005). With OHOR, the human and robot essentially serve as partners to expand SA for the small tactical team. This provides unique opportunities for human-robot collaboration.

2.6.2 Quantifying SA

Because of the subjective nature of measuring SA, it is inherently difficult to quantify. Strategies include physiological measurement (Byrne & Parasuraman, 1996), performance measurement (Kaber & Endsley, 2004), or a combination of both (Wilson & Russell, 2003). Yanco (2004) identifies some major shortcomings in current evaluations. Designers often enlist themselves as test users, provide informal and uncontrolled assessments, and make determinations on low numbers of participants.

Empirical methods, including the number of items found during a search, joystick usage, and human navigational error, are preferred (Yanco, Drury, & Scholtz, 2004). However, even with an expansive list of metrics, empirical measurement is difficult. All methods are subjective to some degree and fall into four categories: self-assessment of the subject, experimenter assessment of task performance, experimenter assessment of the subject's SA by asking questions during short suspensions of the task, subject assessment from verbal expression of their thoughts about system effectiveness during operation (Drury, Keyes, & Yanco, 2007). The latter method is effective in getting immediate feedback on what the operator is experiencing through his interaction with the system.

2.6.3 Situational Awareness for MRUAVs

Differences between SA acquired for ground vehicles and UAS are important to understand (Drury, Keyes, & Yanco, 2007). A UAS pilot must have an awareness of 3D spatial relationships that includes perception, integration, and system model understanding. The 10 items introduced by Drury (2007) are given as the following

1(a). Placement of the MRUAV with respect to terrain and targets
2(a). Knowledge of the MRUAV and points on the earth such as home base or landing zones
3(a). Resilience of the MRUAV to anything that could obscure sensors

1(b). Knowledge of the health or malfunction of the MRUAV
2(b). Understanding of MRUAV relationship to team members and bystanders
3(b). Degree to which the MRUAV can be trusted to respond as expected to user commands
4(b). MRUAV progress towards completing the mission

1(c). Predicted 3D spatial relationships of where the MRUAV will fly in the near future
2(c). Logic of the MRUAV response to preprogrammed conditions like communication failure
3(c). State of the control parameters for MRUAV subsystems such as camera control

The 10 items are grouped into perception challenges 1(a)-3(a), integration challenges 1(b)-4(b), and control systems challenges 1(c)-3(c) and ordered in terms of their importance to MRUAV local search. In addition, item 2(b) has been altered from "observing a fixed-wing UAS relationship to other UAS" to "observing an MRUAV's relationship to team members and bystanders" because of MRUAV operation at low altitude. As discussed previously, control systems challenges are beyond the scope of this work. This results in 7 principles rooted in perception and integration that are the basis of situational awareness for MRUAV to be applied to local search.

2.7 Situational Awareness Principles of Perception

Section 2.7 expands upon the aspects of situational awareness for tactical teams that is particularly relevant to human driven perception concerns. Human driven perception concerns pertain to how the human-robot team successfully fosters the ability for the human to relate to the environment and the presence of his robotic teammate. Because the human and robotic agents are codependent, the understanding of the operator's awareness in this context is of utmost importance.

2.7.1 Placement of the MRUAV with respect to terrain and targets

Situational awareness first involves an understanding of the placement of the MRUAV with respect to its surroundings. Effective video and mapping functions are a necessity for a remote operator capable of supporting a favorable human-robot ratio. Video-centric GUIs have video displays as the focus of attention. In contrast, map-centric systems have one or more maps as the central element of the display (Drury, Keyes, & Yanco, 2007).

Both sources of information are not weighted equally to system operators with users favoring real time video (Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005). Particularly in search scenarios, the user must be able to use landmarks for localization that would be useful in a video centric display, but he must also understand the path taken to get to that point (Drury, Scholtz, & Yanco, 2003) and the significance of the process that could be helped through map centric displays.

2.7.2 Knowledge of MRUAV points of interest

Several other techniques are developed around improving environmental understanding. An interface developed by Ferland (2009) demonstrates the capability of using a generated map to simulate both video-centric and map-centric displays in specific situations. They show an egocentric mode of control is necessary, but also the ability to tilt the viewpoint to view an exocentric view enhances the ability to navigate obstacles such as doorways. Drury (2003) shows the majority of mishaps do not happen on the front of the robot, requiring a simple method to measure proximity on all sides of the robot, which also may be helped by an exocentric display. In any accord, operator remote navigation of a MRUAV in an unstructured environment is challenging (Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005). Relationships to environmental objects can be built through the GUI and its ability to map terrain and targets (Darken & Peterson, 2001; Tittle, Roesler, & Woods, 2002).

Points of interest and object identification within the environment are also key to establishing situational awareness. The operator should have as much direct contact with the target environment as possible to reduce interfacing with the robot. These objects may also be used for navigation through mediums like touch screens (Chen, Haas, Pillalamarri, & Jaconbson, 2006). Cognitive load is reduced, because the operator needs only a mental model of the environment and not of the robot to successfully initiate commands. This is of particular interest to search operations, because the ultimate goal is to have as complete a model as possible of the environment that includes objects of interest.

Sketch interfaces have been developed for robot navigation and may be viewed as an extension of video based interfaces (Fong, Thorpe, & Baur, 2002; Skubic, Bailey, & Chronis, 2003). Skubic (2003) makes the claim that sketching is a natural and intuitive means to interface the human to an unmanned system with maximum situational awareness. Similarly to Fong (2003), it observes the usability of sketch interfaces on PDAs for controlling robotic movements. The teleoperator sketches a path for the robot to take relative to observable landmarks. Both studies tested for usability of the user interface and not on the functionality of the robotic system. Ferguson (2000) incorporates military strategic planning with touch screen based sketching. Situational awareness may be increased through sketch interfaces involving relationships to objects in the environment.

2.7.3 Resilience of MRUAV Sensors

The MRUAV must be resilient to all external factors that might obscure sensors or operation. UAS may be significantly affected by winds as low as 4 m/s (Orr, Rasmussen, Karni, & Blake, 2005). Studies documenting UAS mishaps demonstrate that a lack of air speed awareness can be catastrophic in a system designed for outdoor operation. In addition operators of UAS often encounter sensor failures that are due to the environment. Cameras that experience lighting, moisture, or mud may be problematic for systems that are reliant on visual cues for navigation and environmental analysis (Chen, Haas, Pillalamarri, & Jaconbson, 2006). In addition, occlusion is notoriously difficult to deal with and may be the source of sensor failure not accounted for in the user's mental model of the system. External factors that impact the visual system are addressed more thoroughly in Chapter 3.

In indoor or urban environments, GPS and other forms of wireless communication are not reliable enough for sustained autonomous flight and thus may impact SA. Some systems such as the RQ-16A T-Hawk will enter into Attitude Heading Reference System mode to stabilize system flight in environments where GPS is not available. Reduced signal strength from other forms of communication may also play a part. Factors such as electronic jamming, transmission loss in space, and signal occlusion by obstacles contribute to these communication issues (French, Ghirardelli, & Swoboda, 2003).

The effectiveness of sensors, the ability to establish the placement of the MRUAV with respect to terrain and targets, and the understanding of the placement of objects within the environment are not independent phenomenon. They often impact each other directly. For example, communication is minimized in a map-centric display, because it does not rely on real-time video transmission (Mohr, 2008) (Bruemmer, Few, Boring, Marble, Walton, & Nielsen, 2005). If video transmission is restricted, a map-centric display may be the only option, in spite of the video-centric displays higher effectiveness establishing an understanding of object placement.

2.8 Situational Awareness Principles of Integration

Situational awareness for the human operator in the Operator Blending context is also defined as having integration driven components. Section 2.8 specifically discusses these components and what they mean for MRUAV local search systems.

2.8.1 Knowledge or understanding of the health or malfunction of the MRUAV

User understanding of the health and operational status of the MRUAV is important to SA, because of safety as a primary concern for integration. Operational status is often measurable but can be affected by the conditions of operation. For example, when operating a UAV onboard a moving platform, visual cues often are not intuitive and can be cause for loss of situational awareness in determining of the UAS is truly operating as expected (Cowings, Toscano, DeRoshia, & Tauson, 1999).

Status information provided to the operator should be malleable, allowing the operator to see feedback and take corrective action. Information usually present in the operator short term memory should be externalized so that status can be continually monitored with minimal cognitive load. For MRUAVs this involves proximity readings of surrounding obstacles that might result in a collision. The operator's attention must be managed properly, directing him to critical information at the proper time.

2.8.2 Understanding MRUAV relationship to team members and bystanders

In addition, the system operator must be cognizant of those in the immediate vicinity including team members, pedestrians, or other UAVs. This is central to the utility of the system and safety considerations, both of which are paramount to MRUAV integration. Safety has become a pressing issue when attempting to approve systems for field use. If the relationship to bystanders cannot be established and demonstrated, the system may never be fielded.

For collaborative systems, this relationship may be expanded beyond spatial relationships to Joint Initiative Theory. In order to achieve situational awareness, this theory makes the case that relationships must encompass a shared basis with team members. Although this basis may not be in depth enough to understand every aspect of MRUAV operations, it must at least establish a common understanding for safety of team members and bystanders.

2.8.3 Degree to which the MRUAV can be trusted to respond to user commands

Trust of system performance considering user expectation is important as well. A lack of trust can be based on MRUAV hardware or software poor performance, a lack of training or understanding of the MRUAV operation, and ineffective relationships between team members, amongst other things.

Some sources of uncertainty are related to communication issues with remote video. Poor quality video feeds a teleoperator may rely upon for remote perception result in inadequate distance and size estimation to build trust with the system (Ferland, Pomerleau, Dinh, & Michaud, 2009; Van Erp & Padmos, 2003). Video degradation occurs in the form of frame rate less than 7-8Hz, reduced resolution of the video display, and a lower bit per frame (Chen, Haas, Pillalamarri, & Jaconbson, 2006; Rastogi, 1996). These factors must be balanced against each other to achieve optimized SA and operator buy in to the performance of the system. Individuals are able to detect 10-20ms of latency (Ellis, Mania, Adelstein, & Hill, 2004; Meehan, Razzaque, Whitton, P., & Brooks, 2003), but the upper limit for a high stress-inducing environment based on increased heart rate is 50ms of latency. Latencies of 170ms significantly degrade an operators control and SA, requiring the operator to decouple their commands from the robot's system response (MacKenzie & Ware, 1993). Constant lag, however, is better for trust and reliability of operation than variation in latency (Lane, Carignan, Sullivan, Akin, Hunt, & Cohen, 2992).

Training must be given to operators of unmanned systems that specializes egocentric Situational Awareness (SA) for individuals with limited flight experience for SA (Ferland, Pomerleau, Dinh, & Michaud, 2009). TTP are needed that are specific to unmanned assets. People tend to lose situational awareness through using TTP for unmanned vehicles that are meant for manned vehicles (Chen, Haas, Pillalamarri, & Jaconbson, 2006). Murphy (2008) limits search time during urban surveillance through the utilization of a MRUAV for short 10-15 minute search missions. Team dynamics and issues with migration of control may also be contributing factors. In ideal training situations, teams generally require 3-5 40 minute missions to reach asymptotic levels of performance and teamwork (Cooke, Pedersen, Connor, Gorman, & Andrews, 2006).

Proper user interface design may also be a source of trust or a lack thereof in MRUAV search systems (Cooper & Goodrich, 2008; Ferland, Pomerleau, Dinh, & Michaud, 2009). Effective, efficient, and usable interfaces require development in light of the user's perspective (Adams J. A., 2002). The lack of user inclusion in user interface design, according to Adams (2002), results in an interface that cannot complete the required task or unwillingness by the users to accept the technology. Some readily identified interface issues include displaying the perfect amount of information to the operator in a comprehensible manner, making clear the shift between modes, minimizing cognitive workload, and conveying transparency of the unmanned system's intentions (Kaber & Riley, 1999). Guidelines help to clarify how these issues may be addressed in the most efficient manner. The ultimate goal according to Raskin (2000) is to create HSI that is 'humane', meaning the user interface must be responsive to human needs and considerate of human frailties. This is particularly useful in developing human-robot collaborative systems under Joint Intention Theory.

2.8.4 MRUAV progress towards completing the mission

There must be a way to measure mission progress that takes into account realistic performance of agents involved. This is in light of the demands placed on human agents, particularly if the MRUAV is used to accomplish sub tasks in overarching mission objectives. For example, in an environment where soldiers may be under direct fire, their operation of a MRUAV is only of importance if what they are accomplishing demands their direct attention and does not detract from mission objectives. Tasks that require sustained attention of UAS operators are susceptible to degraded performance and increased risk for operator error (Mouloua, Gilson, Kring, & Hancock, 2001). This is in part because, although SA theoretically increased the longer individuals are in operation of a system (Tvaryanas, Thompson, & Constable, 2005), their attention span, boredom, and vigilance go down as early as 20-35 minutes after task initiation (Krueger, 1991; Hollands & Wickens, 1999). As discussed previously, missions for UAS that correspond to maximizing their capabilities must be identified. In addition, there must be a means and logical state for determining when these tasks are completed. MRUAV ability to contribute to these missions and produce deliverables corresponds to their ability to be integrated amongst small tactical teams with the proper situational awareness.

The human-robot team must also develop an understanding for when the mission is completed. This may involve the concept of joint closure as mentioned previously. If both agents have the ability to act independently, both must be notified when the mission has or has not been effectively finalized. This may be accomplished with and without communication between human and robotic agents if the human is given the ability to make this determination.

2.9 Chapter 2 Summary

For small tactical teams with MRUAV search systems as team members, considerations must be made towards the perceptual capabilities and integration of robotic systems in unstructured environments. Systems must be made viable for local search. Current literature does not support the development of MRUAVs along these lines using intelligent information gathering. Current approaches emphasize probabilistic approaches but do not reflect task directed, semantically relevant, or computationally attentive principles. Because of this, MRUAVs do not reach their full potential for local search applications.

Human-robot interaction, autonomy, and crewing protocols may all influence MRUAV effectiveness for local search; however, the structure of teams and the methods used for autonomy are still largely open research questions. Human-robot collaboration considering Joint Intension Theory is a relatively unexplored, yet promising research area as it relates to answering these questions. Room for growth in this field exists because the theory on human-robot collaborative systems is incomplete. It does not involve Joint Intension Theory applied to codependent and cooperative systems for tactical purposes. Because of the unique characteristics of MRUAVs, these ideas may hold the key for developing effective local search systems for real mission scenarios.

The effectiveness of these systems must be judged on their ability to provide useful deliverables that improve situational awareness. Situational awareness may be understood through seven principles that are both perception and integration based. Although situational awareness is difficult to quantify objectively, MRUAVs developed with these seven principles in mind are assumed to be effective in its establishment. They encompass MRUAV perception and integration challenges which are the limiting factors for improving situational awareness in teaming situations.

In light of these considerations, a MRUAV specific payload may be developed through Operator Blending. The payload must produce deliverables that effectively improve situational awareness, approaches that provide intelligent information gathering with actual sensors, and provide for operation amongst human team members through concrete approaches. The next chapter will outline the relevant literature that defines RobiSEND technical advances.

3. The Robotic Side: Collaboration

"Maturity of mind is the capacity to endure uncertainty."—John Finley

Robots must discretely measure and interpret the world around them. Although humans use sensors in much the same manner as robots to observe and store information, the process of interpretation is much more continuous. This allows for a more effective, robust, and comprehensive ways of handling uncertainty.

The ability to deal with uncertainty is an important aspect of maturity. Robots model this in the form of a model. Understanding uncertainty, for them, is as simple as developing the correct statistic or characterization that captures the discrete relationships they are programmed to consider. Aspects of maturity, however, are *deeply* rooted in embracing reality and the command of one's own destiny. It is about taking responsibility and ownership of the outcome even when uncertainty produces doubt.

3.1 Overview

Section 3.1 introduces sensors as the means for information extraction from the environment for robotic agents. For Operator Blending from the robotic perspective, information processing and approaches to achieve intelligent robotic perception must be quantified by robotic sensors. Sub-section 1 discusses the general contributions of robotic sensors and how they might fit within the Operator Blending framework. Sub-section 2 then specifically discusses the role of the primary sensors for RobiSEND: LIDAR and stereo vision. The information these sensors provide must be understood through how it complements the information provided from the human team member in Operator Blending.

3.1.1 Sensors and Robotics

The world can be understood quite effectively by robots if there is a possible model that is able to be generated. It is widely assumed that if the model is made inclusive enough, robots will be able to operate in complete autonomy, independent of human intervention. For an unmanned system, this is limited by the ability to measure the environment in the presence of uncertainty. Therefore, before any complex algorithms or approaches can be applied, robots must sense (Fox, Ko, Konolige, Limketkai, & Stewart, 2006). In addition, the effectiveness of sensing governs the role robots can play in the world of humans. Intelligent behavior does not exist without the processing of sensor information and what information is needed for intelligence is dependent on approach. Most intelligent beings have a process to deal with information that allows them to filter out what is important from the environment.

Sensors and algorithms are a major part of human-robot collaborative systems. If the robot is viewed as an additional team member, its sensors and algorithms may be viewed as how it views and thinks about its surroundings. As a result, the sensors and approaches to interpret sensor information lead to the perception and integration solutions that involve robotic systems. Therefore, if Chapter 2 summarizes literature relevant to human concerns in human-robot collaborative teams, Chapter 3 focuses on the literature that frames robotic team members, their sensors, and information processing that lead to robot based solutions.

The effectiveness of sensors is dependent on hardware limitations, integration through software, mission specific goals, and the ability to direct sensor attention intelligently towards meaningful information in the environment. Although data may be collected in an unintelligent manner and processed through software for meaning and, additionally, hardware iteration may provide researchers with sensors that can process more information faster, these approaches are often inefficient. In addition, no amount of massaging meaningless data can result in meaningful information.

A lack of synergy and cooperation between agents is also seldom resolved efficiently through the hand of the programmer or through hardware 'improvements'. As mentioned in Chapter 2, this may result in the marginal usefulness of the system even if information is processed efficiently. Research has been done observing how information processing can be simplified and presented to collaborating human beings to counteract loss of trust. Individuals that understand how information is processed are in a better position to interact with and understand the meaning of robot actions. According to these researchers, it may be more important than improving robots algorithmically for humans to relate to their robotic counterparts.

Aside from communicating the interpretation of every basic measurement the robot makes, reevaluation of the role of sensors towards intelligent information gather abilities may be the most effective means to optimize sensor performance. If sensors are made to extract more meaningful information from the environment then hardware may be simpler and software approaches more logical and effective. For the proof to this statement, one must look no further than human evolution. Although there are beings on this Earth that have better sensors for sight, touch, and hearing, humans evolved to process information more efficiently by intelligently directing sensor attention. This results in novel capabilities that, if made robust enough, expand system integration in human-robot collaborative systems. For local search, the sensor types and algorithms must be effective in building relationships to the operator and environment.

Human-robot collaborative systems provide an opportunity to rethink how humans relate to information provided by robotic systems and a chance to reevaluate the role of sensors towards intelligent information gathering. Chapter 3 expands upon these concepts through the lens of MRUAV based local search. This application frames the problem in order to better relate to solutions derived.

The literature for unmanned systems sensors and algorithmic approaches as they relate to human-robot collaborative systems is not specific to MRUAV search systems. This chapter seeks to explain the role of sensors relevant to human-robot collaborative local search, introduce feature extraction methods for these sensor modalities, and give an overview of basic approaches to processing this information. If particularly relevant to human-robot collaborative systems, the technical details of an approach are expanded upon to lay the foundation for Chapter 4 RobiSEND methodology.

3.1.2 Lidar and Stereo

Sensors come in a variety of forms with unique strengths and weaknesses. In the literature, the most viable sensors for cost effective relationships to the environment are scanning laser range finders (LIDAR) and electro-optical cameras for computer vision. Cameras have the ability to view objects passively while LIDAR produces more accurate and complete planar scans (Chang, Kuwabara, & and Yamamoto, 2008). Both types of sensors have been implemented on board MRUAVs.

The goal of perception systems must be to improve the quality of useful information. One approach to achieve this is to just gather more information in hopes of capturing a better profile of the environment. Many claim for the mapping of an unknown and enclosed environment, a new sensor must be developed that decreases sensing time without degrading resolution (Lee & See, 2004). In other words, the increased information is in the form of more scans over a given period of time. More data points per scan and more scans per second to optimize the performance of current sensors.

This is not the evolutionary path of biological systems. Human beings do not have the most effective vision in the animal kingdom. Instead, the ability to leverage sensor information with context allows for intelligence. The more important aim than optimization is innovation. For unmanned systems, this is the use of sensors to provide the necessary information in a reliable manner for intelligent behavior. As discussed in Chapter 2, this may involve a reevaluation of the role of the operator in the sensing process. To achieve this, the strengths and weaknesses of sensors must be understood. These sensors and their approaches to capturing information as they relate to human-robot collaboration are reviewed in this Chapter.

3.2 Primary Sensors

As mentioned in Section 3.1, the primary sensors for RobiSEND are a stereo vision camera and a LIDAR. Section 3.2 gives technical details, specific strengths, and areas of weaknesses for each type of sensor. Sub-section 3 presents an in depth explaination of how stereo processing is conducted to extract distance information from a stereo vision system. This process is important because it gives a relatable example for information extraction as it is conducted by a robot. In addition, the distance information extracted from a stereo vision camera is an example of assigning functional relevance to data gathered by a robot. Operator Blending relies on the ability to assign functional meaning to the synthesis of observations from both human and robotic agents.

3.2.1 Scanning laser range finders

LIDAR is the most prominent sensor for mapping onboard mobile robots (Poppinga, Birk, & Pathak, 2008). It is robust, provides high frequency data, and ease of feature extraction (Kanade, 1989). These sensors have been applied to object tracking (Kogut, 2007), obstacle avoidance (Chang, Kuwabara, & and Yamamoto, 2008; Choi, and Hong, & Park, 2007; Zeng & Weng, 2007), feature extraction (Premebida & Nunes, 2005; Nguyen, Martinelli, Tomatis, & Siegwart, 2005), map building (Ye & Borenstein, 2003; Ueda & al., 2006; Zhang & Ghosh, 2000), and self-localization (Sohn & Kim, A robust localization algorithm for mobile robots with laser range finders, 2005; Lingemann & al., 2005). Off the shelf, the LMS 200 manufactured by the German company Sick, Inc. was the leading sensor of this type for UAS until the mid-2000s. The Japanese company Hokuyo, Ltd. has since released a series of small 2D LRF that are viewed as more suitable for small robots due to their lower weight, form factor, and power consumption (Okubo, Ye, & Borenstein, 2009).

These sensors are becoming more common on relatively inexpensive robotics applications. Among the different types of LIDAR sensors, coherent detector based micropulse systems use phase sensitive measurements with eye safe lasers. These are the laser types found in most mobile robot applications. They give data at high data rates with roughly millimeter resolution. Figure 11 displays an overhead view of the type of scan produced by LIDAR. Time of flight of a radiated laser beam is measured and returned. A mirror directs this laser beam 360˚ around its spinning axis with a portion of this range not visible because of the hardware enclosure that blocks laser transmission. This is known as the non-radiated area. Most LIDAR use this portion of time each resolution to transmit and receive information.

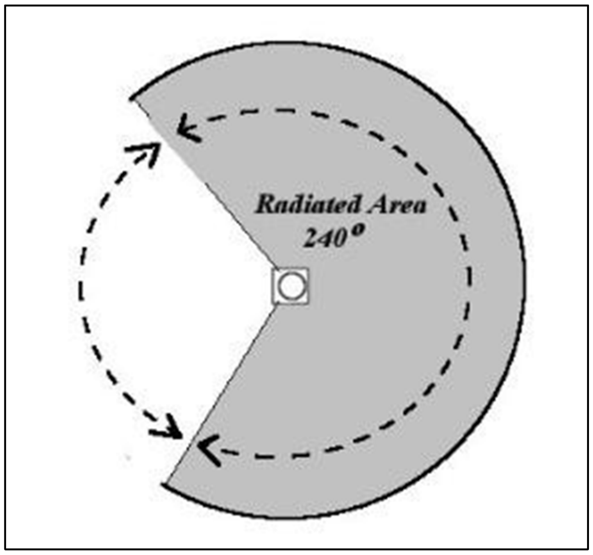


Figure 11: Typical LIDAR scan from sensor with a 240^{o} Field of View

LIDAR proposes distinct perception challenges onboard MRUAVs. In spite of the Hokuyo sensor comparable form factor and weight to electro-optical sensors, it is still more costly and high power in comparison (Lee & Cho, 2007). More importantly, the actual indoor environments MRUAVs are deployed in are not conducive to planar LRF scans because they are not formed by only vertical planes (Cacciola, 2007; Ikeda & Miura, 2006). This means that objects that are in the scanning plane of the LIDAR are not necessarily indicative of the actual structure of the environment. As seen in Figure 12, the scan of a table may only yield the sensing of the table legs as opposed to the table top that poses an obstacle to the MRUAV.

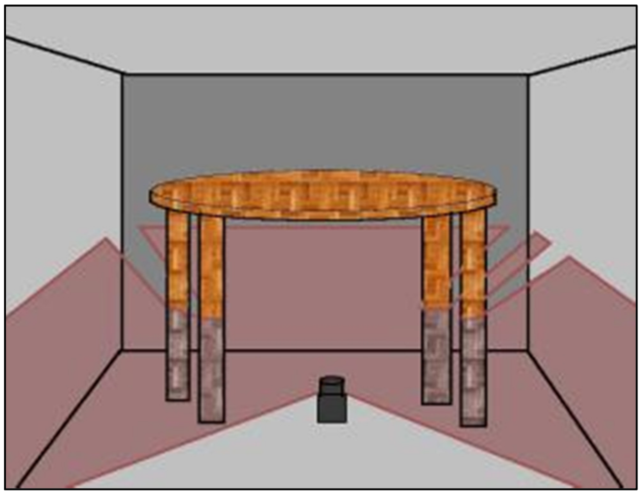


Figure 12: Scan Plane of a LIDAR sensing the legs of a table

Many newer LIDAR systems such as the Velodyne HDL-32E involve multiple scan planes at slightly different angles to establish a vertical field of view. The size, cost, required bandwidth, and complexity of these sensors, however, are usually not suitable for MRUAV missions.

There is a need to have height resolution for perception in unstructured environments (Surmann, Nuchter, & Hertzberg, 2003; Wulf, Brenneke, & Wagner, 2004; Nuechter, Wulf, Lingemann, Hertzberg, Wagner, & Surmann, 2006). 3D data allows response robots to better estimate small enclosures (Nuechter, Wulf, Lingemann, Hertzberg, Wagner, & Surmann, 2006), construct more realistic maps (Howard, Wolf, & Sukhatme, 2004; Hahnel, Fox, Burgard, & Thrun, 2003; Liu, Emery, Chakrabarti, Burgard, & Thrun, 2001), and classify objects in an environment including humans from movement and shape (Nuechter, Wulf, Lingemann, Hertzberg, Wagner, & Surmann, 2006). There are alternatives to achieve 3D scans such as actuating a 2D scanning laser range finder to register multiple 2D planes in a spherical coordinate system (Poppinga, Birk, & Pathak, 2008), or mounting 2 scanners perpendicularly and exploit the movement of the actual robot (Thrun, et al., 2004; Ye & Borenstein, 2003). Although this may yield very accurate results, the time per scan or computation is too high for most mobile robot applications (Okubo, Ye, & Borenstein, 2009; Iocchi & Pellegrini, 2007).

3.2.2 Stereo Vision

Images when initially captured reduce complex 3D environments to 2D projections. Robots at times desire to extract 3D information back from these 2D projections. The issue is that, because of the initial perspective transformation to capture the image, a single image alone does not have enough information to reconstruct 3D relationships. Additional sources of information through time or space is necessary. One way to accomplish this is through stereo vision.

When done well, stereo vision addresses the primary issue with LIDAR systems. It is a relatively quick way to capture 3D characteristics of the environment. The computational time of stereo vision approaches may be problematic for some applications; however, vision has garnered more attention recently in the research community because of the improved capabilities of Systems on Chip or other embedded devices to perform these calculations.

Onboard MRUAVs, these sensors usually take the form of monocular or binocular stereo vision systems due to weight considerations and simpler calibration methods. Monocular stereo vision is cheaper, has a smaller form factor, and has fewer decalibration issues when compared to stereo vision (Klappstein, Vaudrey, Rabe, Wedel, & Klette, 2009). On the other hand, stereo vision sensors provide direct range measurements and resistance to lighting and reflectance changes over time. Essentially, both operate similarly. The difference being that stereo vision attempts to develop correspondences over space, while monocular vision finds correspondences over time and space.

Both monocular and stereo vision systems are usually based on the pinhole camera assumption. This approach involves modeling the origin of the camera coordinate system as a single point in space. It is located the focal length *f* from the center of the image plane. Image points on the image plane are re-projected into the real world by going through this pinhole location. For real systems, lenses distort the pinhole relationship, causing distortion.

As a result, lens selection plays a major role in the performance of the system and therefore must be considered when vision based data collection is to occur. The camera optic is just as important as the camera hardware itself. Visual information intended to be shared amongst human and robotic team members must not only produce correspondences for features the robot can understand, but the output must be also viewable by the human in the loop. Drastic Vignetting or distortion in the image can result in difficulty for either to view the environment effectively. In addition, these distortion effects can cause disorientation for the human viewing the information. Examples of possible effects are shown in Figure 13 and Figure 14.

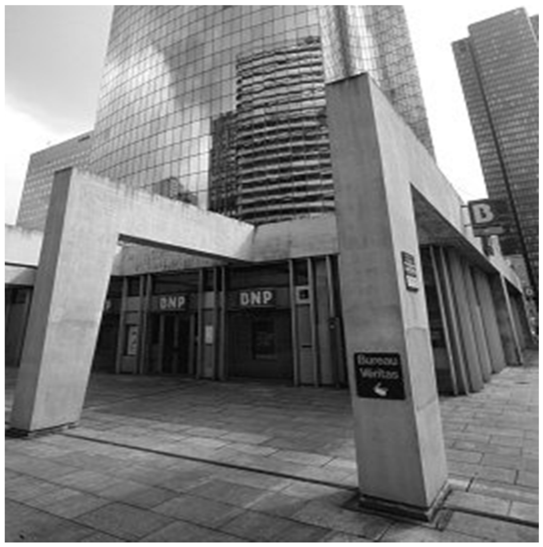


Figure 13: Image demonstrating lens distortion of cameras. Reprinted from Paul van Walree. http://toothwalker.org/optics/distortion.html

Pixels near the edge of the image appear curved inward resulting in the curved appearance of straight lines in the actual world. In addition, images may appear darker towards their edges due to vignetting, which is more exaggerated when lenses are used with large apertures due to the varying direction of incident light. CCD arrays are rectangular but lenses are round, restricting light from reaching the CCD corners.

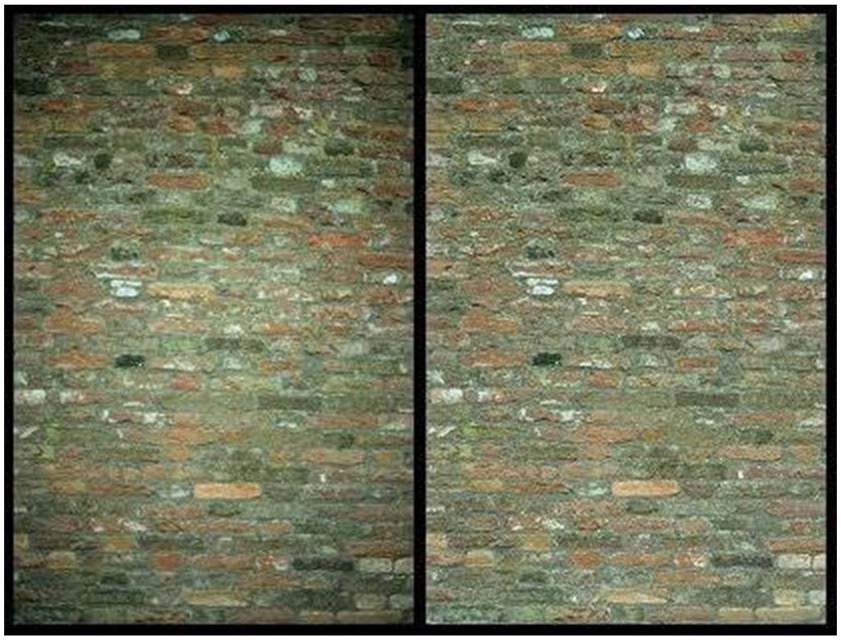


Figure 14: Image demonstrating Vignetting. Reprinted from Paul van Walree http://toothwalker.org/optics/vignetting.html

Both vignetting and lens distortion can detract from the quality of stereo vision information. Camera fish-eye effect may be compensated for through image rectification which straightens the edges of the image based on a calibration procedure. Vignetting, however, is a loss of data but is only an issue with improper lens selection. Objects observed at close range may possess some unavoidable fish-eye effect dealt with through software. Once images are captured they are ready to be processed for range information.

3.2.3 Stereo processing

The primary applications for stereo vision in robotics are object recognition and navigation (Grimson, 1993). To accomplish this, stereo vision methodology is usually divided into 4 steps for range detection: registering and recognizing the location of features, developing a correspondence of features between views, determining camera parameters, and using this information to calculate distance (Grimson, 1993; Lucas & Kanade, 1981). The primary advantage of stereo vision systems when compared to LIDAR is image intensity data is acquired passively, rapidly, and yet still at adequate resolution. It is also the least expensive range sensing method with very limited power dissipation over range (Okubo, Ye, & Borenstein, 2009).

The simplest form of binocular stereo vision involves two cameras with parallel optical axis and identical focal lengths a set baseline apart. The basic structure of this system is displayed in Figure 15 with essential parameters for stereo processing labeled.

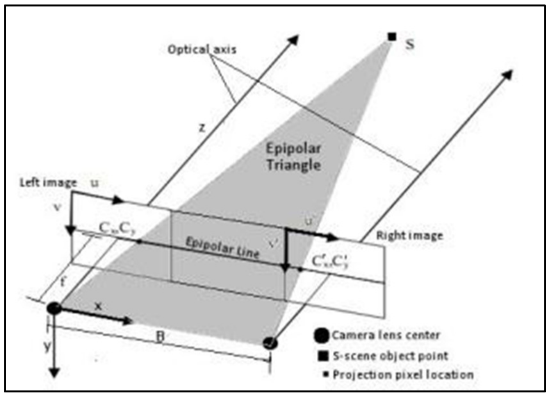


Figure 15: A basic stereo vision description with relevant parameters

The variables *x*,*y*, and *z* are the global world coordinates, *u* and *v* are image coordinates, *B* is the baseline distance between the parallel optical axis of the cameras, f is the focal length, C_x, C_y, $C_{x'}$, and $C_{y'}$ are the *u* and *v* center pixel locations of the left and right image, respectively, and *s* is the scene object point of the physical object in the real world. Figure 15 conveys that because the left camera lens center, right camera lens center, *u* and *v* in the left and right images, and the scene object point all are members of an epipolar triangle in the same plane. If other relevant values such as the pixel size *p* and sub-pixel accuracy *e* are known, this triangle may be used to solve for unknown values. Pixel size is a constant value found in the data sheet and sub-pixel accuracy is set in software and is either to ½ or ¼ of a pixel.

For this stereo vision system, the relationship between image coordinates and global world coordinates is shown in the matrices of Equation (1).

$$\boldsymbol{Q} \equiv \begin{bmatrix} 1 & 0 & 0 & -C_x \\ 0 & 1 & 0 & -C_y \\ 0 & 0 & 0 & f \\ 0 & 0 & \frac{-e^2 p}{B} & 0 \end{bmatrix} \qquad \begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} = \boldsymbol{Q} \begin{bmatrix} u \\ v \\ d \\ 1 \end{bmatrix} \qquad \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} X/W \\ Y/W \\ Z/W \end{bmatrix} \tag{1}$$

X,*Y*, and *Z* are intermediate values for the global world coordinates before the weighting value *W* is applied to transfer their values from pixel coordinates. The value of *d* is calculated from Equation (2) and is the difference between the *u* values of projection pixel locations along the same epipolar triangle that portray the scene object point.

$$d = abs(u - u') \tag{2}$$

This matrix Q is known as the reprojection matrix and may be used to find *x*, *y*, and *z* values when *u*, *v*, and *d* are found from the stereo camera.

The process of finding *u*, *v*, and *d* from the stereo camera involves finding values for *u* and *u'* along epipolar lines in the two images. This is done through a match search process. This process is shown in Figure 16. If the match search process is ineffective due to misalignment of the two cameras, invalid calibration, or failure to properly rectify image distortion, range information is registered in the disparity image as noise. This noise is filtered out through several approaches to removing invalid measurements. Weak stereo images with poor correspondence oftentimes result in disparity images with few range measurements. In these situations, unreliable disparities dominate the image.

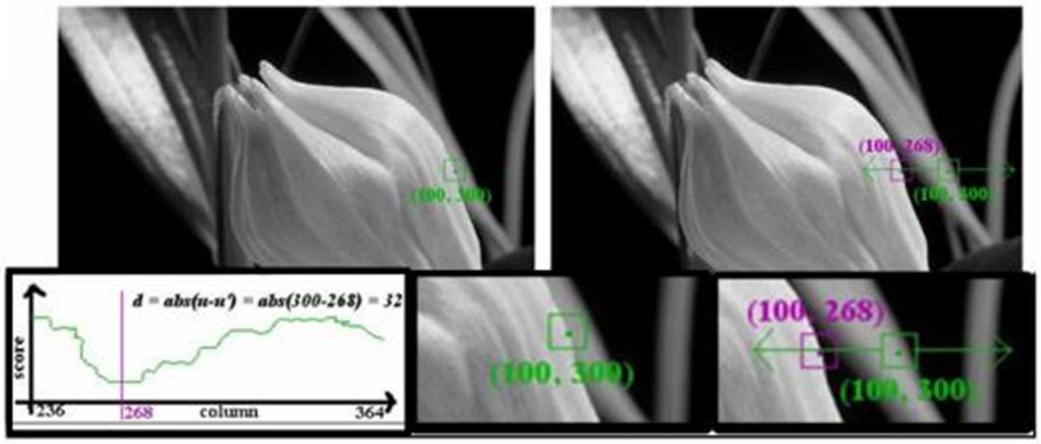


Figure 16: Disparity search and score process. Base figure reprinted from Steven Pinker. http://www.stevepinker.com/Other/Stereo-flower-pairs/10733475_3muVi#747905624_5STMb-A-LB

The left image is the reference image and is captured sequentially with the right search image. In order to calculate a unique signature for each pixel, every pixel in the image plus a region of interest that includes its local surrounding area is indexed as a pixel window. Each pixel window in the reference image is scored for similarity by comparing pixel windows of the search image. The comparison is made between pixel windows with center pixels of the same epipolar line and a certain threshold within a similar column index. This threshold is known as the number of disparities searched for and is a parameter set in software. There are several methods used to compute a pixel window score. The most popular of which is the Sum of Absolute Difference method (SAD) because of its low computational speed and fairly reliable result. If the pixel window area is denoted by the sliding region *M* in the image and index *i* and *j* are the horizontal and vertical indexes within this region, respectively, the Sum of Absolute Difference method is given in Equation (3)

$$\sum_{(i,j)\in M} |I_R(i,j) - I_S(x+i, y+j)| \tag{3}$$

where I_R and I_S are the reference and search image pixel indexed by the row index *i* and the column index *j*. A related approach is the Sum of Squared Difference method, which is similar but utilizes a square of the difference between pixel windows as opposed to the absolute value.

Depending on how many pixels one varies the column index to find a minimum score, the pixel in the reference image is given a disparity value that is proportional to distance from the camera. Low disparity values correspond to far field distances while high disparity values correspond to near field distances. Once each pixel in an image is given a disparity value, these values may be converted to 3D coordinates through reprojection as discussed previously. When *x, y,* or *z* must be solved for individually, Equation (2) may be converted outside of matrix form

$$x = \frac{(C_x - u)B}{de^2p} \tag{4}$$

$$y = \frac{(C_y - v)B}{de^2p} \tag{5}$$

$$z = \frac{-Bf}{de^2p} \tag{6}$$

Mapping may also be made to favor local as opposed to global information and reprojection may not be necessary. When done, two reasons are the primary motivation: global coordinate assignment may be impossible and global coordinates may not be useful. These approaches only measure the metric properties in terms of object distance to a camera and never convert this information into real world coordinates. Parallels may be drawn to the spatial memory system of biological systems as studied in neurobiology (Thau, 1997). As long as the system has a spatial reference point, local information can suffice.

Which disparities searched for defines the spatial range of the system and is indicated by the variable δ. If the system searches for δ from 0-16, then satisfactory matches are only found for objects in the far field. Likewise, if the system searches for δ from 32-48, only near field objects will return satisfactory matches. This range of detectable disparity planes is known as the stereo system horoptor and is displayed in Figure 17. In addition, this spatial resolution in the *z*-axis direction is not linear. Equation (7) shows the resolution possible given camera parameters.

$$\Delta r = \left(\frac{e^2 p}{Bf}\right) r^2 \tag{7}$$

As shown, the resolution is some constant derived from the camera parameters multiplied by the distance from the camera squared. The range resolution is better for objects closer to the camera.

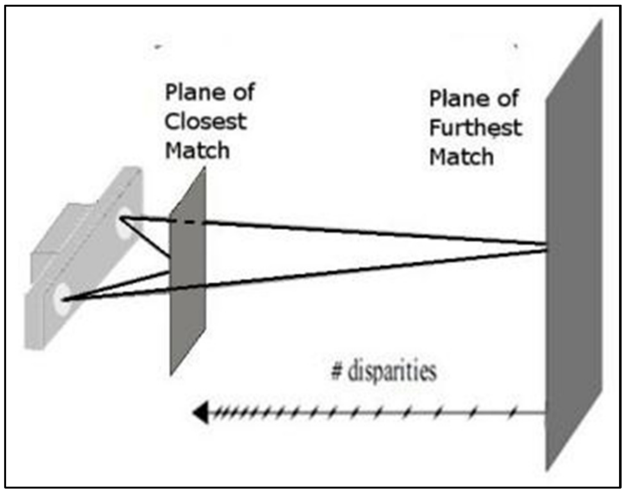


Figure 17: Horoptor sets the disparity range of the system

Because stereo vision does an exhaustive search over every pixel value in the image, pixels that do not produce accurate scores fluctuate values in the image disparity. Therefore, image filters are needed to get rid of the inaccurate disparity information. Confidence, uniqueness, and speckle filters eliminate inaccurate disparity measurements. The confidence filter detects if the disparity was reached with a indistinct minimum score. The uniqueness filter compares the disparity value to surrounding disparity values. If the value does not seem logical, then it is eliminated. The speckle filter eliminates disparity values when they are not near other disparity values and thus appear as isolated dots on the disparity map. In addition, large pixel window sizes for the SAD calculation may be used to improve the rate of correspondence in the image at the detriment of object resolution in the *u* and *v* image coordinates and computational time. Figure 18 displays the differences between using a 5x5 pixel window versus a 23x23 pixel window.

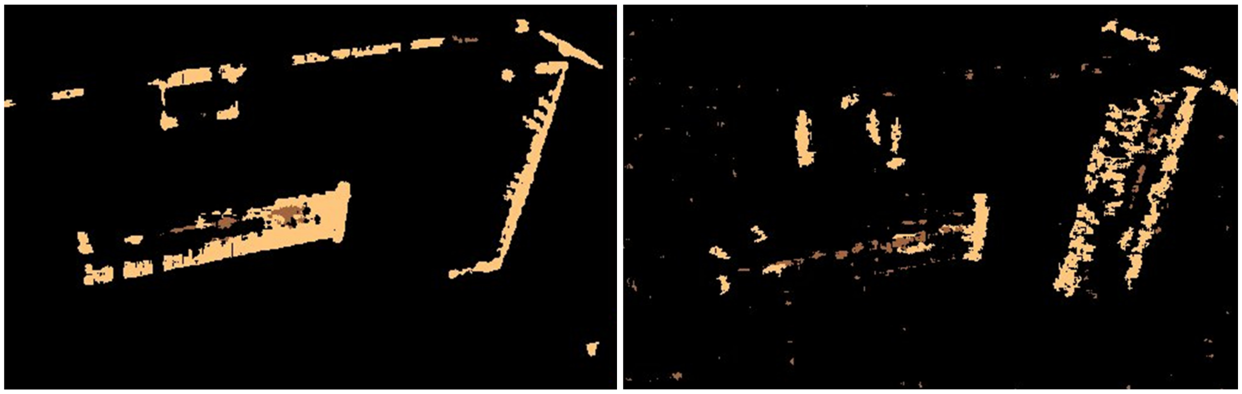


Figure 18: Large size pixel window disparity (left) and low size pixel window disparity (right)

3.3 Alternative Stereo Vision Correspondence

Because stereo vision plays such a vital role in the approach to human-robot collaboration presented in this research, further details on its significance to the feature extraction process are presented in this section. Sub-section 1 emphasizes the importance of the correspondence problem for stereo vision systems and mentiones alternative approaches to the traditional brute force approach. In addition, nuissances, or environmental effects that impact feature extraction or feature correspondence, are discussed in sub-section 2. It is then established that the intended purpose for the stereo vision system within overarching mission objectives plays a role in the nature of the feature extraction problem. For RobiSEND, feature extraction is done primarily for the object recognition problem as opposed to the correspondence problem which requires point to point correspondence and 3D reconstruction of the observed environment.

3.3.1 Correspondence Options

Stereo vision systems suffer from inability to build correspondences, sudden changes in intensity due to illumination noise, and insufficient feature information (Lee & Cho, 2007). The limitation of stereo imagery over range is a direct result of the correspondence problem (Tate & Smith, 1995), which is controlled algorithmically and through stereo parameters. The nonlinearity of stereo measurement, as shown in Equation (7) adds to the ineffectiveness of the system at long range.

Methods to solve the correspondence problem range in computational complexity. The most basic technique for the correspondence of two images is to calculate and measure at all possible potential values of the disparity vector. Iterative approaches to disparity calculation such as the hill-climbing technique may also be conducted. This approach does not search every disparity in the search threshold, but intelligently does the search by selecting distinct pixels in the search image and disparity number values. Additionally, some correspondence algorithms are based around the course-fine search strategy where a primary correspondence is found by analyzing the images at low resolution and as needed uses higher resolution images. Although the many approaches to disparity calculation are outside of the scope of this work, they are means to speed up this calculation at the detriment of accuracy. The use of FPGAs has made SAD practical approaches for real time performance.

3.3.2 Nuisance Effect on Correspondence

The correspondence problem is also widely influenced by the number of detectable distinctions within the scene. This leads to indiscernible matches between cameras on surfaces with limited contrast (Okubo, Ye, & Borenstein, 2009; Lee & Cho, 2007). Figure 19 shows walls not returning disparity values within regions of limited context.

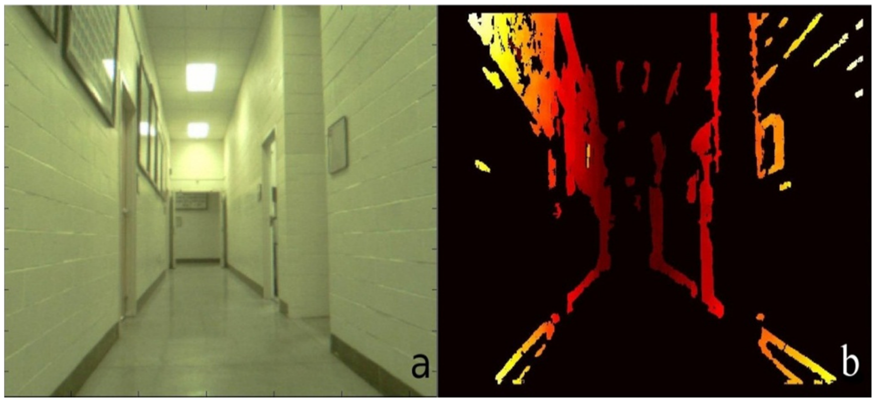


Figure 19: disparity values in a hallway using Videre STH-DCSG stereo camera. 19a shows the color image of the disparity information. 19b shown the filtered disparity image

As with other forms of intensity based sensing, it is subject to lighting and surface reflectance effects that may lead to a lack of detection across shadows, low contrast, bright, or dull spots. Stereo imagery deals with these issues better than monocular imagery, although the issue is still problematic (Tate & Smith, 1995). Factors related to camera parameters such as motion blur, noise, and resolution can also be major issues that lead to invalid stereo measurements (Elinas, 2007).

The number of matches the stereo system is able to generate is also dependent on the relative orientation of the cameras, for example, whether the optical axes are fixed parallel to each other. The ability for the system to produce reliable results is dependent on the calibration method maintaining robustness over time. The common assumption in the literature is, because of these considerations and their respective limitations, the correspondence problem is the most difficult problem in stereo vision (Grimson, 1993).

3.3.3 Correspondence and 3D reconstruction

Systems use correspondences of features between images primarily for 3D reconstruction (Thrun, et al., 2004; Iocchi & Pellegrini, 2007; Stamos & Allen, 2002; Nevado, Garcia-Bermejo, & Casanova, 2004), where complete knowledge of the location of objects in the environment is the goal. The ability for stereo vision to quickly and inexpensively acquire 3D data allows for the fabrication of the spatial relationships to potentially achieve this. When this is done on a pixel basis, the spatial relationships are displayed in the form of a point cloud. These maps are usually built from post processed data because of the computational intensity involved with registering multi-dimensional information. For MRUAV search systems that operate outside of direct line of sight, map building must be done as close to real-time as possible to contribute to battle space awareness.

Stereo vision, however, is not always viewed as a means for 3D reconstruction. Grimson (1993) suggests that stereo algorithms should be developed in light of the needs of the task at hand as opposed to virtually reproducing the environment. They argue a shift towards local map correspondence of 3D data and registration of objects for recognition. Faugeras (1992) also make this case with the argument that one can represent the scene structure around a moving robot without precise calibration and no metrical reconstruction of the scene through the use of relative coordinate systems and object recognition. The hard part of solving this problem is the selection of what to process to extract depth relationships. If stereo vision is intended for selection then 3D correspondence is not necessary beyond what is needed to identify data feature subsets that make up the components of the object models. This concept is explored further in Chapter 5 on the RobiSEND methodology.

3.4 Vision based Feature Extraction

Other feature extraction methods are used in this research to establish functional relevance to the robot contribution of the human-robot team. An overview of feature matching in general is first discussed in sub-section 1. Because stereo vision is already established as an example, feature extraction methods are contrasted with this approach. Specific features are then given in the remainder of this section. Their significance of color histogram and SIFT features to object tracking is then discussed as similar concepts are used to develop the AIM-Shift approach for the RobiSEND system.

3.4.1 Overview

Although most stereo vision systems have a fixed geometry that ensures feature matches share the same epipolar lines, other approaches to feature extraction are more generalized. In addition, the greedy search that is performed in the SSD or SAD methods also cannot be reproduced for these systems. Feature matches may occur anywhere in the image within a reasonable scope. One of the basic building blocks for vision based applications is feature extraction. There are several approaches to finding correspondence between features with various benefits considering their process. The most prominent units of attention for correspondence are intensity based, however, some approaches use higher level features.

There are usually three phases to feature matching in images. These are the detection of favorable feature point candidates, the derivation of feature descriptors for those candidate points, and the matching of descriptors between two sets of features. Because every pixel is evaluated as a feature point candidate in stereo vision systems, the process of feature detection is unnecessary. Said another way, every pixel in the reference image of a stereo vision pair that can be matched up with a feature in the search image is considered a feature. Most feature extraction methods treat this as a unique stage as only points in the image well suited for correspondence are evaluated. Unique signatures are built for these detected feature points with descriptors that consist of surrounding information. The proportion of feature detectors and descriptors to pixel values in the image varies by approach. They are chosen to give a representative sample that is resilient to nuisances such as scale, lighting, and rotational changes.

3.4.2 SIFT

The root of many modern approaches to local feature extraction and correspondence is the SIFT detector and descriptor (Lowe, 1999). Many other approaches are adaptations of this general technique. The SIFT detector extracts keypoints in an image region with four parameters: the keypoint center coordinates x and y, its scale, and its orientation.

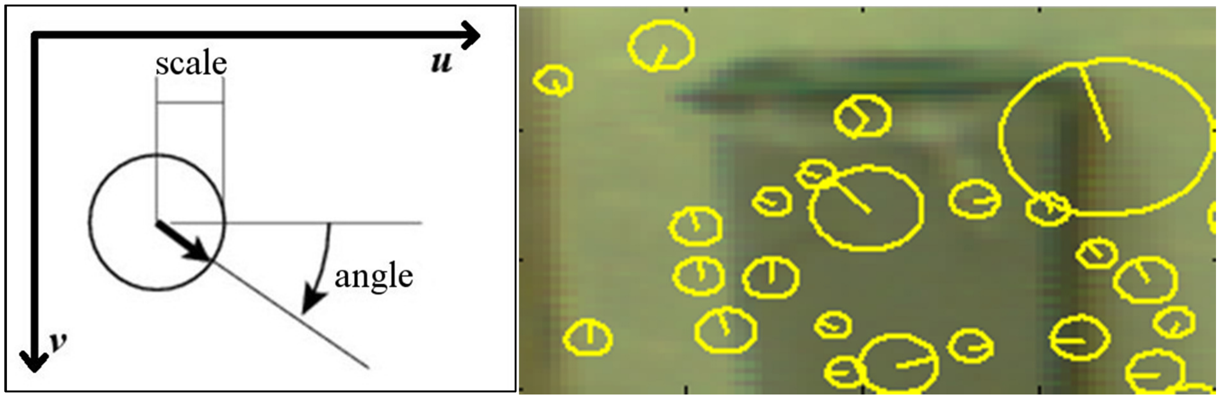


Figure 20: Detector parameters of x, y location, scale and orientation with example. Leftmost image a reprint from Dr. A. Vidaldi and B. Fulkerson. http://www.vlfeat.org/api/sift_8h.html

These keypoints are identified by searching for blobs in the image at multiple image scales and positions. If keypoints are present at multiple scales, they display the characteristics of being invariant to translation, rotation, and rescaling of the image during the correspondence phase.

The first step for finding these keypoints is building the image Difference of Gaussian (DoG) scale space. Convolution of a Gaussian kernel with the input image is done at multiple scales. The Guassian kernel is defined by Equation (8)

$$g_\sigma(\mathbf{x}) = \frac{1}{2\pi\sigma^2} e^{\frac{-\mathbf{x}^T\mathbf{x}}{2\sigma^2}}, \tag{8}$$

The value of σ is determined by the number of scales per octave S and the number of octaves O. For Gaussian scale space, the value of σ is found by Equation (9)

$$\sigma_0 = 1.6 \cdot 2^{1/S}, \quad \sigma(o,s) = \sigma_0 2^{o+s/S}, \quad s = 0, \dots, S-1, \quad o = 1, \dots, O-1 \tag{9}$$

where O is the number of octaves observed, o is the current octave, S is the number of scales, s is the current scale, and σ_0 is the initial convolution base value shown above. Once the number of scales per octave is reached, the value of *o* is increased and the image is sub sampled by half, decreasing its resolution. This is done because an image blurred the value of an octave has a quarter as much information. This information can be just as effectively represented in an image with half resolution. Successive scales are then subtracted from each other to get the Difference of Guassian scale space as shown in Figure 21.

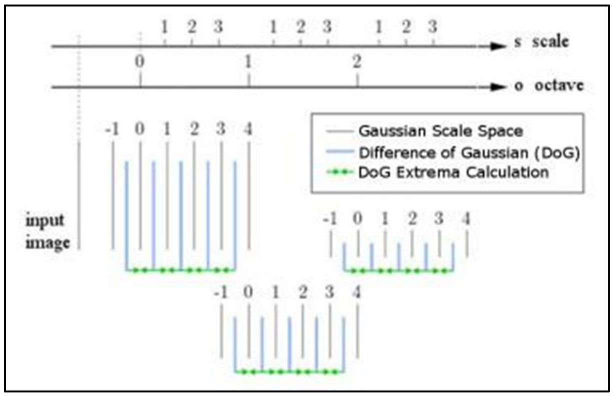


Figure 21: Difference of Gaussian Scale Space calculation. Reprint from Dr. A. Vidaldi and B. Fulkerson. http://www.vlfeat.org/api/sift_8h.html

The minimums and maximums of this scale space across scale are the candidate points for keypoint locations. This concept is portrayed in Figure 22. Candidate points from the DoG scale space is compared to its 26 neighbors. If this point is a minimum or a maximum, it is identified as a keypoint.

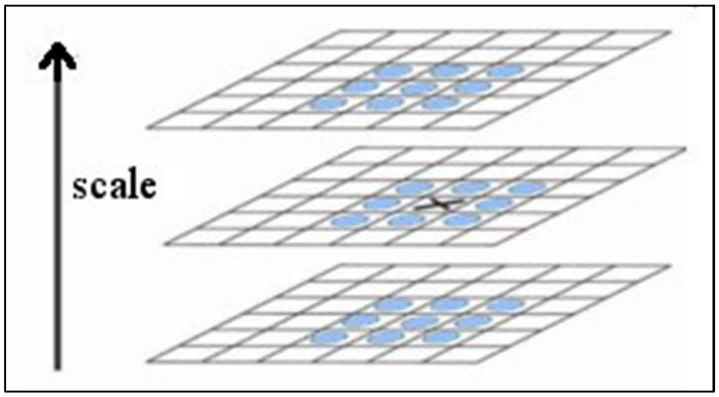


Figure 22: Keypoint identification process. Reprint from Dr. A. Vidaldi and B. Fulkerson. http://www.vlfeat.org/api/sift_8h.html

The SIFT descriptor is a 3D spatial histogram that gives each keypoint from the SIFT detector a unique signature. This signature is resistant to the aforementioned nuisances. A Guassian weighting function is first applied to the keypoint region to give more emphasis to image data closer to the center pixel. A histogram with 8 bins is found for the local image information of small sub regions surrounding each keypoint. Figure 23 gives this histogram when only 1 sub region is used around the center pixel. Each arrow represents the magnitude and direction of the image gradient in the spatial region occupied in its area. The sub region descriptor is the sum of image gradients for 8 bins. Each bin corresponds to a direction in increments of 45˚. The magnitude of the bin is determined by summing up the 16 gradient values of the sub-regions and normalizing this by the total number of pixels in the sub region.

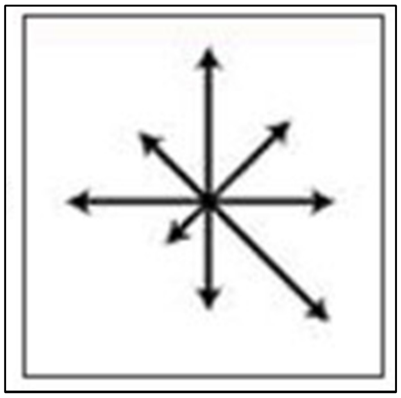


Figure 23: Histogram of 8 bins created for a keypoint descriptor with 1 sub region. Reprint from Dr. A. Vidaldi and B. Fulkerson. http://www.vlfeat.org/api/sift_8h.html

The size and orientation of these regions is determined by the values for scale and orientation found by the keypoint detector as shown in Figure 20. As opposed to 1 sub region, SIFT is usually made up of 16 sub regions with 8 bins each.

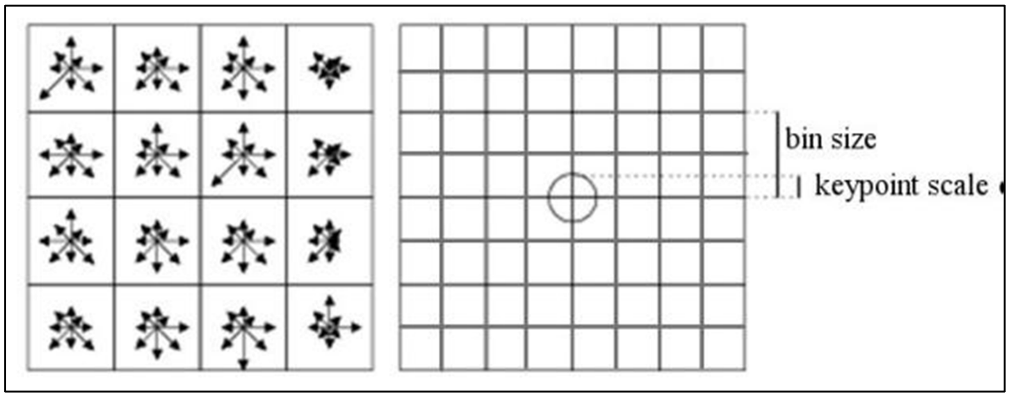


Figure 24: 16 sub region SIFT descriptor with scale information. Reprint from Dr. A. Vidaldi and B. Fulkerson. http://www.vlfeat.org/api/sift_8h.html

Therefore, each keypoint descriptor consists of 128 total values corresponding to the 16 sub regions and 8 histogram bins for each sub region.

SIFT is favorable because of the detector resistance to scale and rotation changes and the relatively large number of feature points it finds relative to other approaches. Other approaches based on the same general approach such as SURF, RIFT, DSIFT, PCA-SIFT and GLOH do not provide the same volume of points and therefore result in fewer correspondences. The DSIFT approach by Dalai et al. for example provides a means to reduce computational time by eliminating rotational invariance of the feature detector. This is useful for applications that involve a stationary or stable camera observing static objects and has been found to be considerably faster than SIFT. As another example, USIFT reduces the feature descriptor size from 128 elements to 64 for computational efficiency at the detriment of accuracy.

Implementations of SIFT are popular and, if they provide the computational speed for the application, can produce effective feature detectors and descriptors. For a MRUAV, resistance to scale changes, rotation, and affine transformations is a must. These characteristics are trademarks of SIFT.

3.4.3 Maximally Stable Extremal Regions

In addition to pixel based local features, region based local features are effective when there is a relatively dynamic range of image intensity values. Maximally Stable Extremal Regions (MSERs) are region based features that are the building blocks for this type of approach. These features satisfy a stability criterion and merge with other MSERs when they can be considered a member of the same region, or branch in the image. This approach was initially proposed to find correspondences between images with resistance to large rotational changes but now may be related to object recognition and segmentation applications.

The approach to find initial candidates for MSERs is similar to thresholding the image a different intensity values. The resulting image contains patches of similar intensities above and below the threshold. In order to find the maximally stable patches, the candidates are tested against a stability criteria as shown in Equation (10)

$$v(R) = \frac{(\mathrm{R}+)-(\mathrm{R}-)}{R} \tag{10}$$

Where *R*+ is the image thresholded at a slightly higher threshold value and *R*- is the image thresholded at a slightly lower value as displayed in Figure 25 for a region of high stability. If the region is stable, the value of $v(R)$ is small. The ability to establish effective threshold values for the selection of R+ and R- can be difficult. If these values are not selected appropriately, MSERs that are local maximas and minimas are formulated. Although some local maximas or minimas are tolerable, these cannot dominate the feature point selection process.

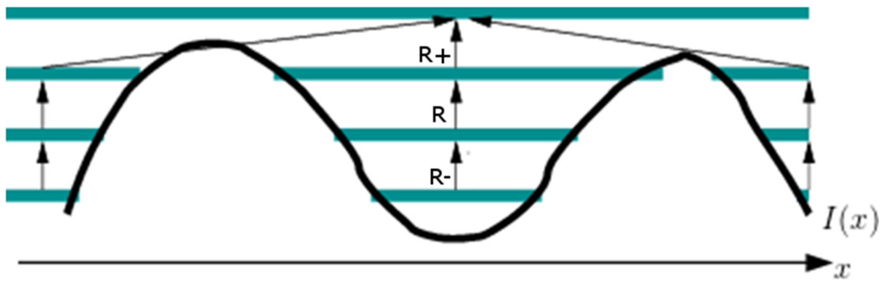


Figure 25: The 2D case of an extremal regions. Reprint from Dr. A. Vidaldi and B. Fulkerson. http://www.vlfeat.org/api/mser_8h.html

MSERs have distinct advantages over other local features. In comparison to non-region based features, they are resistant to affine transformations, stable under monotonic lighting changes, and, unlike SIFT features, are inherently multiscale without any image smoothing or image pyramids. Also, in comparison to other region based feature detectors including Harris-affine, Hessian-affine, and edge-based regions, MSERS are better at detecting small regions, handle repeated textures better, and are the most rebust under lighting changes. The feature detector comes in second to the Hessian-affine detector in handling motion blur and has difficulty handling blur. Possessing drawbacks are characteristic of all region based features as there is not one that completely dominates in performance over others. MSERs have adequate performance under most conditions.

3.4.4 Color histogram and shape

Color histograms and shape features are both region based features global features. As opposed to selected features that stand out as quality individual points, these features globally characterize groups of pixels in the image. Color histograms are used as the basis for object tracking methods such as mean shift object tracking and its derivatives. Shape is a feature that is widely used by humans to characterize information generally before further processing.

For color histograms, image pixel values are divided into color bins based on the parameter *binSize*. A pixel is a member of a bin if its intensity values when rounded to the nearest multiple of *binSize* match the bin pixel values. If *binSize*=64, 4 possible intensity ranges are possible for each of the three channels of the color image. These ranges are 0-64, 65-128, 128-192, and 193-256. Therefore, for the number of possible ranges equal to 4, there are up to 4^3 = 64 possible bin pixel value combinations. An example color histogram is shown in Figure 26

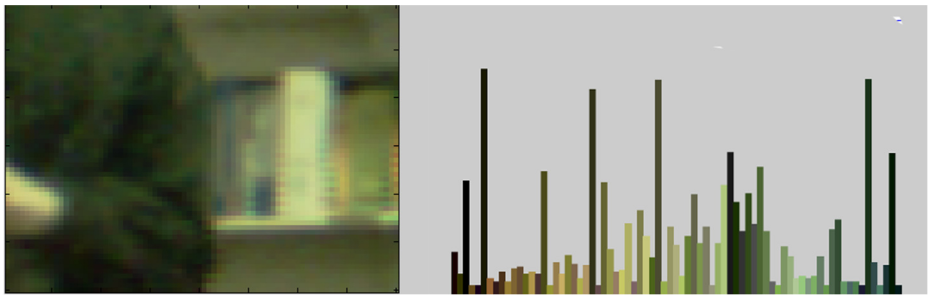


Figure 26: Example of possible colors for a color histogram and source image

Shape is also a prominent feature representation. Although shape representation takes a variety of forms in the literature, one of the most fundamental and far reaching approaches is the measurement of differences between finite point sets (Veltkamp & Hagendoorn, 2000). It is possible to create a feature vector capable of identifying the general shape characteristics of a foreground object. An example of two feature vectors describing the general shape of a foreground object is shown in Figure 27

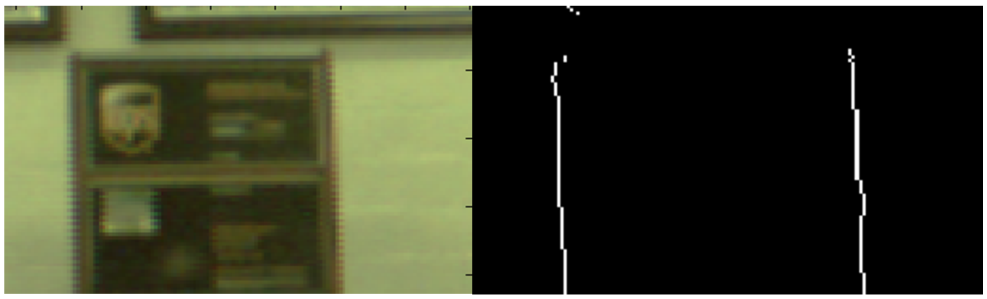


Figure 27: Feature vectors describing the shape of a foreground object

This approach first involves quality segmentation methods to separate foreground objects from background object. Once a boundary has been established, the edge of the boundary is used in shape characterization. Each of the line segments in Figure 27 is represented by a vector of values that characterize their constituent point locations.

3.4.5 Object tracking using color histogram information

Segmentation may also be viewed as the first step in object tracking. One approach in particular takes advantage of color histogram information. Mean shift is a non-parametric, iterative algorithm primarily used for finding modes in data through a kernel density estimation and gradient ascent until convergence. Applied to object tracking, it is a very powerful and relatively simple technique. The mean shift algorithm has a time complexity of O(Tn^2) where T is the number of iterations and n is the number of data points (Cheng, 1998). The following explanation is derived from Comaniciu (2000). The reader is referred to this source for further information.

The technique requires an initial segmentation. Once made, the algorithm tracks the location of the object by measuring the propagation of a region *R* as it moves within the camera field of view. The objective is to find the sub-region color histogram from n possible candidates in the current frame that is most similar to the color histogram from the matched region of the previous image. The value of *n* is usually the total number of pixels in the image.

If a kernel function *K* is used that simply decides whether a pixel in an input image with $\mathbf{x}=x_1 \ldots x_n$ input pixels is a member of the region R, the average pixel value of that region may be given by Equation (11). *K* outputs a value of 1 if the pixel is a member of the region and 0 if the pixel is outside of the region. This type of function is known as the Kronecker delta function.

$$m(\mathbf{x}) = \frac{\sum_{i=1}^{n} K(x-x_i)x_i}{\sum_{i=1}^{n} K(x-x_i)} \tag{11}$$

From frame to frame, this mean value changes with the kernel function *K*, assuming that the pixel membership of input pixels changes (Comaniciu, Ramesh, & Meer, 2000). If, instead of using a kernel that decides pixel membership, the kernel *K* is differentiated to indicate the change in pixel membership, the function *g(x)* may be used.

$$g(x) = -k'(x) \tag{12}$$

Using *g(x)* as opposed to *K(x)*, the new mean value *M(x)* is used to estimate the centroid of the region in concurrent images. From kernel density estimation, the probability density function of a region *R* from its kernel function *g*(x) is given by Equation (13)

$$f(x) = \frac{C}{nh^2} \sum_{i=1}^{n} g\left(\left\|\frac{x-x_i}{h}\right\|^2\right) \tag{13}$$

Where *C* is a normalization constant and *h* is the bandwidth determined by the size of the search region. Therefore, the mean shift value from the kernel function *f*(x) and Equation (11) is found to be

$$M(x) = \frac{\sum_{i=1}^{n} g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)x_i}{\sum_{i=1}^{n} g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)} - m(x) \tag{14}$$

This equation predicts that region *R* has moved to the location *M*(x) in the next image (Comaniciu, Ramesh, & Meer, 2000). In order to confirm this prediction, the color histogram *p* of the candidate region *R* extracted from the region surrounding *M*(x) and the color histogram *q* from a previous region where the object is confirmed to be present are compared using the Bhattacharya distance equation

$$\rho(p,q) = \sum_{u=1}^{m} \sqrt{p_u q_u} \tag{15}$$

Where *u* is the index of like color bins. To convert this result for minimization, Equation (16) is used

$$\delta = \sqrt{1 - \rho(p,q)} \tag{16}$$

With small values of *δ* corresponding to similar distributions. Given the color distribution *q* of an object and centroid of the region y_t from a previous frame, the algorithm usually follows the pseudo code given below presented by Comaniciu (2000)

```
    1. Initialize location of target in the current frame as $y_t$
    2. Compute color histogram pt of region R surrounding centroid yt in current image and find the
       distance measure δt between q of the previous image and pt of the current image
    3. Apply Mean shift of the region Rt in the previous image resulting in candidate regions Rt+1 and
       candidate centroid yt+1
    4. Find the color histogram pt+1 of Rt+1 with centroid yt+1 and find the distance measure δt+1
       between q of the previous image and pt+1 of the current image.
    5. While δt+1 is greater than δt
           a. substitute centroid yt+1 with a value closer to centroid yt
           b. Go to step 4
end
```

Figure 28: Pseudo code for mean shift based object tracking

The problem of tracking moving objects in a scene with a stationary camera is almost identical to the problem of tracking stationary objects on board a moving sensor platform. The difference being that all changes in the object appearance in the latter case are due to the movement of the sensor platform. Although RobiSEND may track dynamic objects, the crux of its function is in the latter case. Therefore, movement and measurement over time are seen as additional sources of information as the observation of interest does not change its appearance in the scene without it. This is especially important when considering color histogram information, because unsolicited orientation changes of objects may be problematic.

3.5 Vision for Segmentation

It is established in Section 3.3 that features derived from stereo vision information do not always have to have the role of 3D reconstruction of space centered around the correspondence problem. Section 3.5 presents the idea of using stereo vision information to achieve object segmentation from background information. Segmentation is first defined in sub-sections 1 and 2. Segmentation as it particularly has been implemented with disparity information is then addressed in sub-section 3. The potential impact of an operator in the loop to this segmentation process in real time is addressed considering this approach is used for RobiSEND and Operator Blending.

3.5.1 Segmentation Using Intensity

The identification of data feature subsets that can be extended to the tracking problem is a form of segmentation. Features are extracted to ultimately produce these subsets. Related information is associated and extracted through correspondences. For mobile robotics, vision may be considered the means for useful correspondences that result in segmentations (Taylor, 2009). There are four main approaches to segmentation that include thresholding, boundary-based, region-based, and hybrid techniques which are a combination of boundary and region methods (Adams & Bischof, 1994; Zhang, Fritts, & Goldman, 2008). These approaches have strengths and weaknesses that are mainly defined by their units of attention.

Thresholding, also known as pixel segmentation, is susceptible to noise and blurring that occurs at the boundary of objects. They group pixels with similar features while ignoring the spatial relationships of pixel groups. This technique may be used to determine global characteristics about an image or object. Background segmentation from foreground may also take advantage of thresholding techniques. Examples include k-means clustering, basic thresholding, and histogram thresholding techniques (Zhang, Fritts, & Goldman, 2008).

Boundary-based approaches look for changes at the edges of objects in the region by observing the image gradient. Once these gradients are identified, objects are defined by the pixels contained within these closed boundaries. There is no potential to grow these regions as in region based approaches. Boundary-based approaches have the advantage of grouping pixels with high variability in characteristics. This is not possible for thresholding or region based approaches, both of which rely on pixel similarity. The downside, however, is clear identification of what constitutes the edge of a region may be difficult. Because of this, pure edge based methods are not utilized without ground truth. Examples of these algorithms include color snakes and edge-flow methods (Zhang, Fritts, & Goldman, 2008).

Region based approaches tackle this problem from the inside out. Pixels assumed to be part of the body of an object are compared to their neighbors. The region grows if pixels are decided to possess their neighbor's like characteristics. Thus, pixels are grouped based on their similarity and spatial connectedness. Split-and-merge methods and region-growing methods are the primary examples (Zhang, Fritts, & Goldman, 2008). Hybrid approaches exist that are primarily based on region approaches, such as the watershed algorithm (Meyer & Beucher, 1990), seeded region growing (Adams & Bischof, 1994), or variable order surface fitting (Besl & Jain, 1988). They identify similar pixels and grow regions until they meet with potential boundaries. If these boundaries are similar then the regions may be merged until objects are identifiable. In Seeded region growing, seed points are identified and grown until a pixel meets the criteria to be labeled a boundary pixel.

For systems with static cameras, the segmentation of moving objects in dynamic environments often consists of a region based approach utilizing optical flow algorithms. The issue with this approach onboard a moving system is the relative movement of background pixels and a requirement to process the entire frame of visual data. 2D optical flow information alone is not sufficient for this operation in the presence of the unknown egomotion of the sensor platform in the presence of vibration or immeasurable amounts of motion. Research has been done to estimate the expected image motion with IMU data or a motion model; however, these approaches often oversimplify the problem or require impractical amounts of computational power.

3.5.2 Unsupervised Segmentation

When there is no ground truth to measure the effectiveness of the segmentation, it is said to be unsupervised (Zhang, Fritts, & Goldman, 2008). In unmanned systems, this is usually coupled with a lack of complete information about the motion of the camera. Although it is a challenge to understand the effectiveness of these approaches, there are characteristics of a good segmentation that may be determined.

The four criteria for a good segmentation are: regions should be uniform and homogeneous with respect to some characteristic, adjacent regions should have significant differences with respect to the characteristic on which they are uniform, region interiors should be simple and without holes, and boundaries should be simple, not ragged, and be spatially accurate (Zhang, Fritts, & Goldman, 2008). Zhang (2008) characterizes the first two as Characteristic criteria, because they are related to the characteristics of the object, and the latter two as Semantic criteria, because they are related to how the object appears to humans. Most segmentation criteria are based soundly in characteristic criteria, because semantic criteria are very application dependent. Semantic criteria, however, may be more important than Characteristic criteria in applications where the operator directs the attention of the system. There must be aspects of the target object that attract the attention of the operator. This may have little to do with the actual homogeneity or content of adjacent regions.

In order to measure these criteria, unsupervised segmentation metrics are based on their ability to achieve intra-region uniformity to address criteria 1, inter-region uniformity to address criteria 2, and semantic cues, such as shape, for criteria 3 and 4 (Zhang, Fritts, & Goldman, 2008). These metrics are not usually observed alone. They are combined for a composite effectiveness measure. For color images, intra-region uniformity is mainly addressed through a color error metric that evaluates the logical consistency of pixel values within the segmented region. There are also alternatives to color error metrics that measure texture and entropy. Inter-region segmentation can be difficult to achieve with color information. The most popular method for this is finding the average color of regions or boundaries of potential objects. This may obviously be problematic for real world objects that do not have color uniformity.

3.5.3 Segmentation with disparity

Regardless of approach, the major challenge with object segmentation from stereo vision data is that segmentation within itself is an ill-defined problem (Hebert, Unnikrishnan, & Pantofaru, 2007). There is no clear manner to verify the effectiveness of segmentation or to deal with feature detection changes of objects over time. Stereo vision provides an opportunity to link the visual information of a disparity image to real world distance values. When this distance is reliable or may be verified through other sources of sensory information, this information may be used to develop unsupervised segmentations or to verify the quality of segmentation methods.

Approaches provide a means of segmentation that involves multiple sensor modalities to deal with this issue. In many instances, this involves the incorporation of LIDAR for geometric information and stereo vision data for segmentation. This may have little impact on the computational intensity of the system. Segmentation is considered a secondary vision operation, meaning that it relies on primary methods of feature extraction that use up the majority of available computational power. It may, however, provide the context necessary to make meaningful conclusions from electro optical information.

The aforementioned approaches may all be applied to stereo vision data by replacing the intensity criteria with disparity (Thakoor, Gao, & Jung, 2010). Doing this is advantageous because it serves as a means to filter out non distinctive features and also gives the resulting objects spatial significance. There are two main ways to use segmentation information from a disparity map. The first finds relationships directly from the disparity map and does not require 3D reconstruction of the environment. The second finds these relationships through the transformation of disparity information into 3D point clouds for the extraction of objects through reprojection equations as explained in Section 3.2.3. For stereo systems not designed for 3D reconstruction, extracting this information directly from the disparity information is the best approach. This bears resemblance to the human visual system in its approach to matching disparities over narrow ranges.

Real time disparity calculation is a reality for off the shelf systems such as Videre design's Stereo-on-a-chip, SRI International's Small Vision System, and TYZX's Deep Sea software (Thakoor, Gao, & Jung, 2010). Segmentation approaches are based on shape or boundary criteria. Simplifications are usually made, as with most vision processing research, that often restricts the implementation of these systems to environments with distinct geometries. Although over simplification is to the detriment of the application of this technology, some simplification at times reflects the abundance of simple shapes and surfaces in real environments (Thakoor, Gao, & Jung, 2010). By incorporating techniques that segment based on disparity, making key assumptions about the environment, and utilizing approaches that only process portions of an image, computational intensity theoretically may be drastically reduced.

3.5.4 Other segmentation techniques

One way to view obstacle detection is the process of segmenting objects from the image ground plane (Taylor, 2009). This involves making the ground plane appear as a single feature in the image. The most popular algorithm in the vision processing community for segmentation of the ground plane in dynamic environments is the Random Sample Consensus (RANSAC) algorithm. This method estimates the ground plane robustly by taking image features in a region of interest, fits the ground plane features, and discards the obstacle features as outliers. This, however, assumes that the ground play can be modeled. This method does not require knowing the motion of the vehicle.

Once the ground plane is properly identified, it may lead to additional information about objects. As mentioned in Section 3.2.3, constructed 3D information from 2D image projections requires additional information or assumptions. If it is assumed that objects meet the floor at their points closest to the sensor system, this information may be exploited to find distance information through Inverse Perspective Mapping (Taylor, 2009). Pixels along the ground plane may be counted to get a rough estimate of distance from a camera. In approaches to ground plane segmentation that involve cameras, the camera is often angled towards the ground, as any information above the horizontal plane cannot be a part of the floor (Taylor, 2009).

New approaches may need to be developed, particularly for stereo imagers that produce relatively weak correspondence (Trucco, Isgro, & Bracchi, 2003). An operator may be used to identify candidate points. These can develop segmentations through thresholding or region based methods. The inability to negotiate object boundaries limits the application of boundary based approaches. An iterative Segmentation-Estimation approach for planar surfaces through connected component analysis may address this problem (Thakoor, Gao, & Jung, 2010). This approach uses simple geometries to create components and combines them based on a similarity criteria. An example algorithm conducts the following steps:

```
1. Subsample and rectify the left and right camera images
2. Calculate the disparity along with occlusion
3. Initialize the plane labels, label the occluded areas as zero
4. Estimate the plane parameters for the labels 1 to N
5. Label the pixels to minimize the residues
6. Carry out the connected component analysis
7. Re-label the largest spatially continuous region for each plane with the plane number
   and the other components as zero
Repeat steps 4-7 until convergence
```

Figure 29: Example of a simple segmentation algorithm by Thakoor (2010)

This is similar to the feature model building phase of the split-and-merge algorithm for feature based mapping discussed in Section 4.3.2. The key aspect of this approach is the intelligent selection of plane labels. Semantic metrics for plane identification through human-robot collaboration may provide a solution. Weak correspondences may be unavoidable in stereo imagers, particularly in the presence of low lighting or loss of quality calibration.

3.5.5 Operator in the loop impact

In the face of many approaches to feature association, the most difficult problem persists, that of providing semantic meaning to segmentations (Zhang, Fritts, & Goldman, 2008). Methods rely on bottom-up feature extraction that emphasizes criteria 1 and criteria 2 of effective segmentation: regions should be uniform and homogeneous with member features and adjacent regions should have significant differences with non-member features. Semantic segmentations have proved to be more difficult.

As a result, the best segmentation as measured with current techniques may not be the best within the context of a human mission. This may be addressed by having an operator in the loop impact the segmentation process. This is not supervised validation; this is a direct role for the human to determine semantic criteria for the segmentation algorithm.

If segmentation is aided by an operator in the loop, there are some basic impacts on boundary, region, and thresholding approaches. Humans have a much higher variability in the size of objects they are able to segment from back ground information (Zhang, Fritts, & Goldman, 2008). This corresponds well with the realities of real environments, which consist of high variability in size. In addition, this leads to grouping smaller objects into larger contiguous regions for segmentation. One large object is detected as opposed to several smaller sub objects. Lastly, human operators have a tendency of allowing few objects to dominate their interpretation of information. Therefore, the most important objects are segmented from the background, even though others might be more suitable for the segmentation process. This is one example of the benefits of operator inclusion in the robot processing of environmental information.

3.6 Role of the Operator

This section establishes the potential role of the operator as a source of perception and integration information usable by a robotic system. Sub-section 1 first gives an introduction to this potential. No clear solution currently exists in the literature; although, the merits of human-robot collaboration for sensory performance of robotics is addressed in the following sections. Operator Blending provides the ability to provide the robot in a codependent and cooperative human-robot relationship with the ability to leverage an operator in the loop for information that helps its perception and integration capability.

3.6.1 Introduction to Operator Involvement

Although human-robot collaboration may improve sensor capabilities and algorithmic approaches to segmentation and object identification, the opposite school of thought exists. Many researchers seek object recognition as a means to decrease user interaction (Iocchi & Pellegrini, 2007). Autonomy is seen as a source for better situational awareness for human team members because they do not have to concentrate on the activities competed by the robotic system. It is debatable as to whether this is true in actual systems. Talukder (2003) claims that object recognition and autonomy play a role in decreasing cognitive workload but not necessarily towards a decreased involvement of the user in the operation of the system.

The most insightful source of information available to a teleoperated MRUAV search systems is perhaps the operator. As mentioned in Chapter 1 and Chapter 2, most systems in the literature are developed to reflect the largely unrealistic goal of complete autonomy. During search involving systems at close quarters of a small tactical team; however, someone is always in the loop for safety and mission support reasons. Issues with occlusion from vision, correspondence, and categorization of scanned objects may all be minimized through the logic of human interaction. This adds another dimension to the sensor fusion problem—how is the information a human operator is able to provide integrated into the perception model?

An object level approach to sensor fusion allows for the user to have more direct interaction at the fusion stage. This interaction has the potential to avoid relying on assumptions about objects that are necessities for autonomous systems. For example, Cacciola (2007) detects the presence of cars with stereo vision and LRF readings. Although the height of the detected object is not measured, it is estimated based on the measured length of the car. In systems with user feedback, the operator may be able to supply a more accurate estimate. It may be difficult of an operator to introduce information to the system, depending on his involvement in completing other parallel tasks.

3.6.3 LIDAR, Vision, and Operator Input

Considering the aforementioned strengths and weaknesses of both stereo vision and scanning laser range finders as they contribute to 3D perception onboard MRUAVs, the ideal sensor would be a combination of the positive qualities of both (Poppinga, Birk, & Pathak, 2008; Bajracharya, Moghaddam, Howard, Brennan, & Matthies, 2009). For search systems, a user is in the loop to provide additional information to the sensory system that may further enhance its performance. These systems typically look at how LIDAR and vision work together but new approaches may consider human-robot collaboration. They may have drastically different formulations in a human-robot collaborative framework.

Early robotics research has been done specifically on how to combine the qualities of scanning LRFs and stereo vision systems (Tate & Smith, 1995). Fusion of sensor data can happen at the data, the feature, or the decision level (Cacciola, 2007). Data fusion occurs when raw data from sensors are combined without significant post processing. Feature fusion is when features are extracted at the sensor level before combination. Sensor fusion at the decision level involves the accumulation of data before fusing. Lastly, decision fusion involves making choices based on weighing contributions from several sources of raw sensor data. All of these require accurate calibration between sensors. Human-robot collaboration usually involves some form of feature or decision based fusion.

One challenge in sensor fusion is that it is difficult to discern which sensor reading is the most accurate (Cacciola, 2007). Specifically, there would have to be added logic associated with the perception algorithm to differentiate when to trust the laser readings over the stereo vision readings. The most common way to deal with this uncertainty is through the development of weighting functions that influence the input of each sensor modalities depending on situation.

Sensors may also be used in situations where one method supplements the deficiencies of another. A 2D approach is also employed to sketch the rough boundaries of an unknown area. In this approach, stereo data is used as a supplementary source of depth information to provide smooth depth values in the presence of laser reflectance problems, to register finer details of objects, and to provide depth information for portions of the surface the LIDAR is unable to view due to reflectance or absorption (Chang, Kuwabara, & and Yamamoto, 2008). Color and texture information from a camera system may also be vital to object classification as a result of the inability of the LRF to make this distinction (Chang, Kuwabara, & and Yamamoto, 2008).

Another approach for spatial sensor fusion involving LRF and stereo vision is to process their measurements separately and combine them without any major decision process (Bajracharya, Moghaddam, Howard, Brennan, & Matthies, 2009; Chang, Kuwabara, & and Yamamoto, 2008). Similar approaches use omni-directional vision to fuse with scanning laser range finders with 360 degree FOV (Bajracharya, Moghaddam, Howard, Brennan, & Matthies, 2009). Matched pixels between the stereo system and LIDAR are both plotted in global coordinates. Data from one sensor source may be averaged with the other or information weighted based on how well it is in agreement.

This is related to fusion on the decision level. Fusion on the decision level eliminates concern with measuring the accuracy of the data. Instead, the information that results in the desired behavior must be extracted from multiple sources of raw data. Observation models are created that are made up of information from multiple sensor modalities. This information is observed holistically. Although certain sources of raw data may be relied upon more heavily considering the situation, the synthesis is achieved with the resulting action. Raw information is used to make an algorithmic determination. This information is only fused on the task level in top-down behaviors.

The types of deliverables sensors must contribute towards in human-robot collaborative systems are sources for collaboration, cooperation, and codependence. The latter term is exclusive for operator blended systems and implies a partnership and reliance on both agents to accomplish overarching mission goals. For local search systems, sensors contribute to object models and maps of the surroundings. In addition, sensors should give the operator status of the robotic system reflecting the progress of the mission and the relative health of the system. Most importantly, sensors should add to situational awareness in human-robot collaborative situations.

3.7 Chapter 3 Summary

The building blocks of perception for the RobiSEND payload are local and global features and the approaches presented in this Chapter. These features, found by exploring the environment, are used to build observation models used by both the human and MRUAV. The operator interface developed in this research presents the human with this information for his decision making. Likewise, the algorithms developed for higher level decision making by the MRUAV leverage this information.

In an operator blended system, each contributing agent must have the necessary information to contribute to the overarching mission objectives. All bottom up information is attained through sensors. The processing of this information for the MRUAV must be written down and discretely expressed through programming and code. Humans, on the other hand, process information through the remarkable asset of the human mind.

At the conclusion of initial processing of bottom up information, the result is hopefully the essential information from the environment for both the human operator and MRUAV. Under operator blending, this essential information is not just what the individual agent needs to complete the task but it also includes information the team needs for overarching mission objectives. This information is then combined through the RobiSEND system operator blended approach.

4. Navigation, Mapping, and Intelligence

"The ability to move is fundamental to intelligence"—Stefano Soatto, UCLA VisionLab Director

Intelligent beings can move. Sight is the sense that is usually linked to movement for many of these organisms. If forced to function without sight, they may learn to use other sensors; however, Sight most readily allows them to identify those characteristics and observations that are desirable for movement. As a result, spatial relationships may be altered to attain additional information.

4.1 Overview

Section 4.1 establishes the role of navigation and subsequently mapping in the shared understanding of human-robot collaborative teams that operate under Operator Blending and conduct local search operations. Sub-section 1 explains the role of navigation in establishing these relationships. Sub-section 2 then expands this understanding and extends it to the process of developing a map. An understanding of the different types of maps and how they might be potentially impacted by an Operator Blended approach is necessary, because the mapping process used in this research demonstrates the benefits of the new approach. Ultimately, it is shown that qualitative maps synthesize the relationships needed by both human and robotic agents under Operator Blending. In addition, the navigation process used to build these maps provides for human and robotic participation in the object recognition problem and the establishment of functionally relevant observation models.

4.1.1 Role of Navigation

Motion oftentimes facilitates knowledge of the environment. If we expect robots to perform intelligently in teaming situations, movement may be of utmost importance. To relate robot abilities to human team members as in Chapter 2 or to build effective models of observations as in Chapter 3, it is a source of information that diversifies the approach. Particularly in unknown and unstructured environments that have lighting variation and occlusion, it may be the only means to acquire the necessary information over time.

Because of the need to actively seek out additional information, movement is central to the local search and navigation processes. Local search is the identification of what observations are relevant to improve situational awareness while navigation is how the movement within the environment is controlled. When navigation is semantically tied to the local search process, it is a shared basis for human-robot collaboration. Movement is then an integral part of the manufacture of team deliverables.

This chapter defines the navigation process as the shared basis for human-robot collaboration under operator blending. The schools of thought are covered that first introduce a general concept of navigation and the value of movement. This lays the foundation for improved perception of observation models and integration amongst team members in unknown environments. Because navigation in local search generates maps, an overview of modern mapping techniques in the literature is then presented. This covers quantitative and then qualitative approaches that have metric and topological constructions. In each approach to mapping, navigation is shown to contribute to the mapping process. This contribution is first introduced in the most fundamental manner as the mechanism for exploration. It is then broadened to include navigation as a component of qualitative mapping approaches.

4.1.2 Navigation

Navigation is the method of control used to move within an environment and gather information. It must be executed based on the specific application, capabilities of the system, and nature of the data, three conditions discussed in Chapter 2 and Chapter 3 for human-robot collaborative systems. Particularly for the RobiSEND system, the application is local search, the system capabilities are those of a theoretical MRUAV, and the data is limited by low cost and low weight range sensors.

The navigation process may thus be dictated by the system requirements. For example, the specific application covered in this research is local search at low altitude. Navigation approaches are therefore restricted considering the elements of local search and the mission profile given in Chapter 1. In addition, the modes of autonomy of Chapter 2 give the basic responsibilities of human-robot teams. Human-robot collaborative systems consist of human and robot participation in the control process. As a result, navigation must involve either Shared mode or Safe mode, eliminating options in Tele mode or Autonomous mode. Lastly, the manner in which the data is presented to the operator or gathered by the robot plays a role in eliminating certain navigational approaches. For human-robot collaboration, these restrictions are discussed at length in Chapter 3.

If navigation is used as a shared basis for interaction in a human-robot collaborative system then this additionally shapes the approach. An additional assumption is introduced—navigation is important to both team members and its goals may be used to communicate shared objectives and facilitate cooperation. The process of dictating what to do is replaced by a mutual understanding of what is important to navigate towards within the group context. Both robot and human share this mutual understanding and can interpret the others efforts to contribute to mission success through navigation. This makes the navigation process the means of communication, drastically changes the nature of goals during the local search process, and forces navigation to be effective within this context.

It is possible to contrast this with the majority of approaches to navigation in the literature that rely on the robot or the operator dictating what is important. Thereby, the shared basis is the ability of one agent to be in control of the other. Teleop mode and Autonomous mode have this in common. In these systems, the navigation process is in place to carry out these commands and does not serve as a method for any type of communication beyond this purpose.

This work assumes that communication is a mission requirement to solve perception and integration challenges because the resources of individual agents are insufficient to be successful. In addition, this shared communication must not only exist, but must build collaborative and codependent relationships. If the shared basis is navigation, the relationships that are built within the local search process will be more relevant to solving perception and integration challenges for the human-robot team. Maps are built under these conditions through the operator blending framework. In the remainder of this chapter, the various approaches to producing map deliverables at the conclusion of a search in the literature are presented. These techniques are needed to contrast the approach to human-robot collaboration discussed in this work with maps built without navigation as a shared basis.

4.1.3 Mapping

Traditionally, four questions are applicable to navigation: 'Where am I going?', 'What's the best way to get there?', 'Where have I been?', and 'Where am I?' (Murphy R. , Introduction to AI Robotics, 2000). The first 2 are path planning questions that require some form of a map, the third requires some type of memory, and the last localization. All of these are dependent on being able to sense the environment to determine spatial relationships (Taylor, 2009). The nature of these spatial relationships, mode of autonomy, and the emphasis given to each question is largely dependent on approach. These questions are revisited when discussing the approach to mapping of the RobiSEND system in Chapter 6.

Path planning requires some form of a map. The use of sensors for map generation has been a source of extensive research (Chang, Kuwabara, & and Yamamoto, 2008). These approaches are based on two major delineations in their simplest forms: metric maps versus topological maps and quantitative maps versus qualitative maps. Quantitative maps and qualitative maps are lumped into the class of hybrid maps because of their tendency to exhibit both metric and topological characteristics. Although quantitative maps are usually dominated by their metric components and qualitative maps are most often majority topological, this is not necessarily the case.

The manner in which spatial information is relayed delineates metric and topological maps. Metric maps relay information for the purpose of reconstructing spatial relationships as precisely as possible. Topological maps communicate functional relationships. The similarity between metric and topological maps is that they are both usually built to be navigation oriented. This is a logical limitation for metric maps but not so much for topological maps, which realistically could communicate relationships that go beyond spatial significance. As a result, although topological maps describe relationships in the environment more so than the structure of the environment, these maps fail to be relevant beyond robot navigation applications (Nguyen, Martinelli, Tomatis, & Siegwart, 2005).

Metric maps dominate the majority of techniques for the purpose of more complete 3D reconstruction of precise geometric relationships (Thrun, 2002). Topological maps are often related to metric maps in that the basis of most relationships is in the spatial domain. The objective of a topological map is, rather than represent the environment with as much spatial accuracy as possible, the relationships between points or features is the primary concern. One of the most straight forward examples is a map of the United States shown in metric and topological form as shown in Figure 30

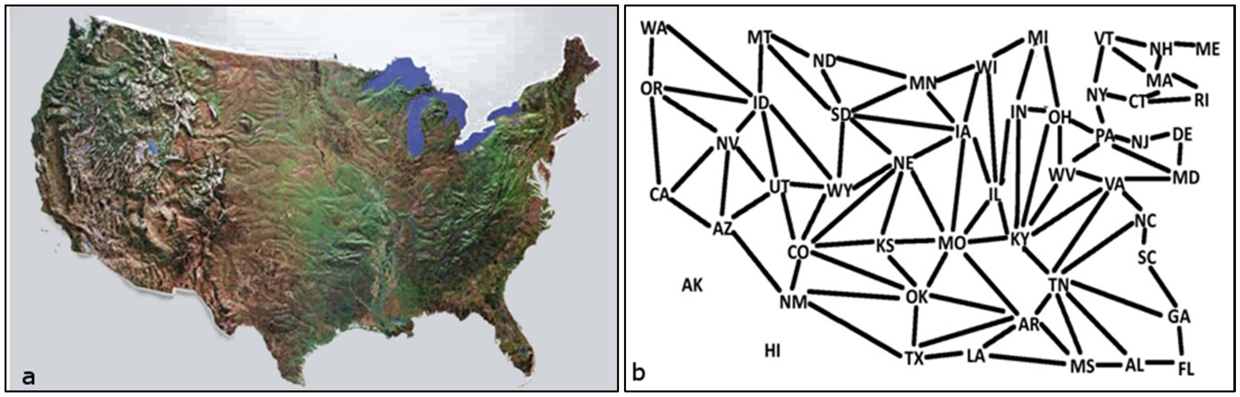


Figure 30: Metric and topological maps of the US. Figure 30(a) displays a metric map reprinted from http://www.geomart.com/products/raisedrelief/unitedstates.htm. Figure 30(b) is a topological representation showing the relationships between states. Connected states share a border

The difference between quantitative maps and qualitative maps is based in the manner in which the map relationships are built rather than their intended purpose. Quantitative maps use measurements to discretize continuous environmental space. Qualitative maps leverage relative changes and produce maps based on integration of relationships over time or space. Figure 30 gives examples of qualitative and quantitative maps of the United States.

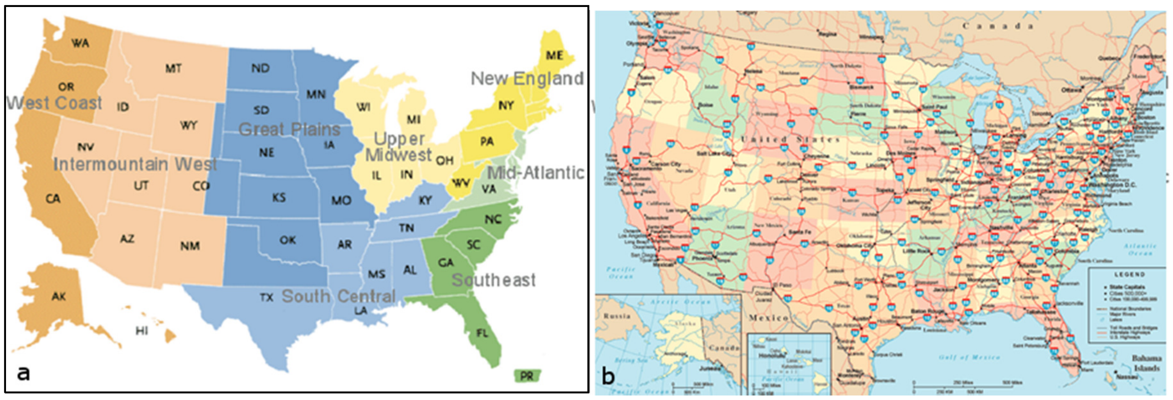


Figure 31: Qualitative and quantitative maps of the US. Figure 31(a) gives a qualitative representation that associates state boundaries with regional segmentations. Reprinted from www.cyberleadership.org/nli/Steve_overview/index. Figure 31(b) gives a quantitative portrayal showing road information and highways obtained from http://www.onlineatlas.us/interstate-highways.htm

As shown, both maps have both topological and metric components. The qualitative nature of the map displayed in Figure 31(a) corresponds to the grouping of states into sub-regions. Figure 31(b) displays a different process of associating information. The precise location of interstate information reflects measurement of road location.

Although it is somewhat counterintuitive, metric maps such as those based on some scan matching techniques can exhibit qualitative characteristics if they are built based on local information, and topological maps may possibly be highly quantitative if the functional relationships they convey are based on precise spatial measurements such as those covered under qualitative spatial reasoning. These types of maps are covered in more detail in Section 4.4. In addition, a hierarchal representation of space is useful when combining characteristics from metric and topological maps for qualitative or quantitative map constructions (Kuipers, 2000; Nguyen, Martinelli, Tomatis, & Siegwart, 2005).

A handful of these approaches may be run in real time depending on the sensor, approach, and if assumptions about the environment are made. Once a measurement is taken, most approaches are metric and based on quantitative methods (Amigoni, Gasparini, & Gini, 2006).

4.2 Quantitative mapping and Particle Filters

Quantitative methods to map building involving point to point correspondence methods are presented in Section 4.2. Sub-sections 1, 2, and 3 gives a brief overview of probabilistic approaches to mapping. Sub-section 4 then gives a popular approach to displaying probabilistic mapping approaches called occupancy grid mapping. An understanding of basic quantitative methods is necessary in order to establish the background for comparison with the qualitative mappin approach created in this work.

4.2.1 Probabilistic Approaches

Quantitative maps are the most popular maps and may be found infused in the daily activities of human beings. If a new student is spotted on Virginia Tech campus, the map he or she holds is purely quantitative. It reflects the metric spatial relationships in the environment and has several of the important points labeled with topological information. As mentioned, this map is quantitative because the way in which these relationships are manufactured. A surveyor, at some time before or directly after the buildings on campus were built, created the campus map to reflect the spatial placement of buildings to the nearest measurement.

For quantitative maps buildt by robots, point based matching and feature based matching are the two most common techniques (Nguyen, Martinelli, Tomatis, & Siegwart, 2005). Point based techniques interpret raw sensor readings, while feature based approaches transform this information into geometric shapes first. These geometric features are then used for the mapping stage. Therefore, topological and metric maps both rely on correspondences of points or features. Many metric mapping approaches lean towards point based correspondence, while topological maps focus on more feature based matching. This is a logical trend and not a requirement as discussed in the subsequent sections.

Most recent quantitative techniques are based probabilistically (Thrun, 2002). Measurement noise is statistically dependent and therefore previous errors cause a growth in uncertainty over time. Because uncertainty exists in these measurements, probabilistic mapping is dominant in the literature to correct for errors. These methods recursively compute posterior probabilities for map building considering previous sensor measurements and the motion of the system.

Smith (1990) began the terminology of Simultaneous Localization and Mapping (SLAM) in 1990 that is widely used for this purpose. Their probabilistic approach was a necessity for four major reasons. It deals with measurement noise, high dimensionality of the problem, feature recognition during redundancy, and the dynamic nature of environments (Thrun, 2002). The mapping system must converge in a reasonable amount of time and amidst seemingly redundant and dynamic environments. If an algorithm was capable of accomplishing all four problems addressed by SLAM perfectly then it would be able to map *any* environment accurately and robustly. Unfortunately, most approaches assume a static environment where the robot is the only active agent.

Statistical frameworks for mapping are not the only solution, however, they are key in understanding the state of the art technology. At the heart of statistical approaches is Bayes filter, which is the grandfather of Kalman filters (Kalman, 1960). A Kalman filter is a recursive state estimator. A complete discussion of the Kalman Filter may be found in Choset (2005). It is dependent on 3 basic assumptions: the motion model is linear with added Gaussian noise, the perceptual model is linear with added Gaussian noise, and the initial uncertainty must also be Gaussian. For Kalman filter approaches, the system achieves strong convergence and may be implemented in dynamic environments; however, it does not close the loop in feature correspondence and suffers from high computational intensity (Thrun, 2002). Slight alterations of the Kalman filter algorithm such as FastSLAM may reduce computational intensity, but sacrifice accuracy. For the Kalman filter approach to be practical for robot mapping, the number of features must lie between a few dozen and about 100. Because the Kalman filter approach has no means to differentiate feature points in the map based on individual sensor measurements, features must be sparsely distributed.

The primary advantage of the Kalman filter approach is its ability to estimate the state posterior potentially in real-time. The Rao-Blackwellized particle filter and the mixture of Gaussian methods are extensions of the Kalman method and therefore are also capable of full posterior estimation. The assumption of Gaussian noise may be invalid in real world systems; nevertheless, the Kalman Filter approach produces the strongest convergence in existing robotic mapping systems.

The simplicity and accuracy of the Extended Kalman Filter SLAM method make it the most established SLAM algorithm. An EKF, like the Kalman filter, assumes the probability distributions over robot poses and maps can be represented by a Guassian distribution. As mentioned previously, this is an assumption that does not usually hold true. Computationally, the EKF scales quadratically with the number of landmarks and handles incorrect data associations poorly (Thrun, 2002), making them poor for medium to large environments.

4.2.2 Partical Filters

Multi Particle Filters are direct descendants of the EKF and address some of the issues with EKF SLAM (Thrun, 2002). It is based on Monte Carlo Localization which makes the claim that the pose of the robot may be represented as a set of particles. A particle is pose information accompanied by the necessary state information required to update the pose. As the robot moves in the environment, particles trace out its trajectory (Taylor, 2009). MPF SLAM differs from Kalman based SLAM in that it uses low dimensional particle filters to estimate landmark positions with different probability distributions.

Performance varies over the various approaches. Overall, MPF SLAM is faster than EKF SLAM when the number of detectable landmarks is greater than 25 (Thrun, 2002). In addition, EKF SLAM also no longer functions in real time when the number of objects is greater than 75-100 (Yuen & MacDonald, 2004). The empirical complexity of EKF with over 100 landmarks is quadratic, while a MPF has a complexity that has been tested to be logarithmic (Thrun, 2002). One drawback of MPF approaches is, because each particle develops its own local map, a particle that is inaccurate may drastically influence the final result. Particles are usually weighted based on how well they agree with other particles to deal with this issue (Taylor, 2009).

There is also difficulty understanding the distribution of particles. There is no clear way of identifying how many particles are necessary to have enough to close the loop. Too many particles results in a system that is too computationally demanding. Too few particles results in an unstable map.

4.2.3 EM Algorithm and Hybrid Approaches

An alternative probabilistic approach is the Expectation Maximization (EM) algorithm. It is based on the maximum likelihood estimation with latent variables (Dempster, Laird, & Rubin, 1977). The EM algorithm strength is the feature recognition correspondence problem (Burgard, Fox, Jans, Matenar, & Thrun, 1999). In particular, EM algorithms generate maps of large-scale cyclic environments even if the features detected are similar in appearance. The major departure from the Kalman filter approach is the absence of a full estimate of uncertainty. Hill climbing is instead used to estimate the most likely map. Because of this, EM algorithms must process data multiple times, making it incapable of real-time operation.

There are several examples in the literature of hybrid approaches that combine near complete estimates of the posteriors of states with computationally more efficient maximum likelihood estimates. One example is the incremental maximum likelihood method (Moravec, 1988; Yamauchi & Beer, Spatial learning for navigation in dynamic environments, 1996). A map is built as sensor data arrives through this approach without keeping track of residual uncertainty. This is essentially a more simplistic approach to the EM algorithm with emphasis on maximization of point correspondence as opposed to prediction. The major advantage of this alteration is that maps are now able to be built in real time, but without tracking uncertainty. Data revision is therefore impossible based on future data.

Hybrid approaches deal with this limitation through using the incremental maximum likelihood approach to initially build maps and keeping track of uncertainty in robot pose to correct them (Thrun, 2002). The basic logic behind this is that the only data needed to correct errors is associated with pose estimation. The major drawback of the hybrid approach is that there is no completely reliable algorithmic method to determine when or how to apply the corrective action based on the pose to the map. The errors present in mapping may also grow too large for the corrective steps, particularly when traversing the same loop multiple times. Depending on how often corrective action is taken, the mapping function may not function in real-time.

Hidden Markov Models (HMM) have also been used. The first-order Markov assumption is a basic assumption during map building (Taylor, 2009). New information is independent of the previous robot motion except for the immediate prior pose. This pose may be a direct measurement or contained in information cells of the map. This makes past and future information independent statistically (Taylor, 2009).

Taylor (2009) makes the case that this is an assumption that is not often valid. The environment has a pattern that may be exploited for map building in certain circumstances. Because of this, the author argues that the map building approach may predict future map information based on prior knowledge. The example given is that if a robot is following a wall inside of a building, there is a high probability that the wall will continue or there will be another wall found perpendicular to it. There is a low probability that this wall will lead to empty space.

In spite of progress, there is not a standardized approach to off the shelf mapping that accommodates real environments with a variety of sensors. Approaches that employ multiple sensors are the most reliable choice. For systems that employ multiple sources of sensor data, the MPF algorithms are useful with the appropriate feature extraction methods. Probabilistic approaches to the SLAM problem has been extended to the 3D case; however, this approach is hard to tune properly, is computationally demanding, and not very robust (Nuechter, Wulf, Lingemann, Hertzberg, Wagner, & Surmann, 2006). Most commercially available SLAM packages post-process saved data. Some adaptations do exist for online mapping packages; however, they are often cost prohibitive and do not meet all requirements specific to the application or system they are implemented on. This calls for independent development of mapping based on the specific application that often takes away from system development time.

There are also probabilistic algorithms that are capable of dealing with dynamic environments. The Kalman filter approach has perturbations that allow for slowly moving objects with easy to predict movements. This is accomplished through the inclusion of a Gaussian motion variable into landmark location. Essentially, this alteration adds uncertainty to the location of landmarks, ultimately dealt with through sensor contributions in the filter (Thrun, 2002). Occupancy grid maps may also deal with certain types of motion through decaying occupancy over time (Yamauchi & Langley, 1997). Yamauchi and Langley, through the Exploration and Learning in Dynamic Environments Project, gives the first tested algorithm for object detection in a dynamic environment, uses hill climbing for correspondence, and develops occupancy grid maps. These are options when considering the best option for dynamic mapping.

There are systems that employ SLAM completely on stereo vision data. This is known as visual SLAM and usually based on the theory of the Rao-Blackwellized Particle Filter (RBPF) and a Bayesian framework for probability estimation over time (Elinas, 2007). It is necessary to derive an estimate of the robots motion from disparity measurements and multi-view geometry techniques known as visual odometry (Bajcsy, Kamberova, & Nocera, 2000; Bar-Shalom & Fortmann, 1998). The most useful feature extraction method for this algorithm is the Scale Invariant Feature Transform (SIFT) which identifies pixels in a stereo pair that are resistant to scaling, translation, and rotation (Elinas, 2007).

4.2.4 Occupancy Grid Maps

Occupancy grid maps are largely used in combination with one of the aforementioned probabilistic approaches and point based matching. They involve knowing the relative size of the mapped area and dividing this space into voxels. Voxels are non-overlapping shapes that follow specific rules. The most straight forward voxel is a square of identical size known as a voxel in most approaches. Relatively noisy and uncertain information may then be used over time to contribute to the value of these voxels.

There are 3 design issues for the registration problem: the choice of which parts between 2 maps that are used for registration, the method to find the rotation angle and translation that achieve overlap for a good match, and the cost function that determines the degree of alignment (Chang, Kuwabara, & and Yamamoto, 2008). Occupancy grid maps have been utilized for LIDAR based methods (Castellanos & Tardios, 2000)and stereo vision based methods (Chatila & Laumond, 1985) to achieve the cost function.

Voxels hold the probability that an obstacle exists in its representative location in space. Each voxel may have the value of occupied, free, and unknown (Lee & Cho, 2007; Bajracharya, Moghaddam, Howard, Brennan, & Matthies, 2009). Most often, the value of a voxel is indicated by an 8-bit grayscale value. White usually corresponds to definitely free and black to a high probability of occupation. The maps that result are usually viewed as 2D but may cover all three spatial dimensions (Moravec & Martin, 1994).

This technique is often linked to Bayes filters through the method used to determine voxel status. As discussed in Section 2.2.4, Bayes rule relates conditional probabilities. For voxels, this models the current value of the voxel based on previous probability of its value. The previous estimate of the voxel value is given by *p(m)*, where m corresponds to a particular voxel location. When a new scan *z* is taken, the new cell occupancy probability $p(m|z)$ is given by Equation (17)

$$p(m|z) = \frac{p(z|m)p(m)}{p(z)} \tag{17}$$

Where *p(z|m)* is the probability of getting a specific sensor reading given the occupancy associated with the *m*th voxel and *p(z)*, because it does not depend on *m*, is a normalization constant that is the same for all values of *m* (Taylor, 2009). Occupancy grid maps are known to be robust and easy to implement.

There are several disadvantages to occupancy grid map based approaches. They have an inability to accommodate pose uncertainty and assume independent noise much like the Kalman filter. In addition, they are inefficient representations of the environment in terms of memory usage and do not scale well for large areas (Sohn & Kim, 2008). There are occupancy grid map approaches that attempt to deal with these issues. Hahnel (2004) uses a cumulative approach as opposed to a Bayesian approach to defining voxel occupancy. Basically, voxels accumulate hits over time as it is found in space with a range scan. Voxels that are hit multiple times were determined to be occupied. In addition, this approach only recorded obstacles found through these hits. The robot is intended to be driven manually; therefore, keeping track of unexplored areas and free space in the map was viewed as unimportant. Assumed teleoperation during SLAM map generation directly affected requirements.

The quantitative maps discussed in Section 4.2 are in contrast to the feature based quantitative maps of Section 4.3 and the qualitative spatial reasoning of Section 4.4. Feature based quantitative maps take the first step towards qualitative map building. As opposed to individual point correspondences, correspondences are built between a collection of features observed together. Quantitative feature based maps, however, like the quantitative maps discussed in this section, both are based on the correspondence problem. As the reader will see in Section 4.4, there is a framework in the literature for developing qualitative relationships as opposed to quantitative relationships for human-robot collaborative teams called Qualitative Spatial Reasoning. To the author's knowledge, the extension has yet to be made to fully develop qualitative mapping techniques that take advantage of Qualitative Spatial Reasoning. In this research, this involves the leveraging of observations over time as the contribute to the object recognition problem for mapping as opposed to the reliance on the correspondence problem as seen in quantitative mapping approaches.

4.3 Feature Based Mapping

Feature based mapping is presented in Section 4.3 as somewhat of a bridge between quantitative approaches in the literature and the qualitative approach to mapping developed in this dissertation. Although point to point correspondence is not used for feature based matching, the correspondence of feature associated points is used. Sub-section 1 gives the general structure of basic feature mapping techniques. Line segment feature matching is then concentrated on to give tangeable examples of this approach. Data association is then further expanded upon through traditional approaches presented in sub-section 4 and hybrid approaches discussed in sub-section 5.

4.3.1 General Structure

Quantitative metric maps may also take the form of feature based maps. As opposed to points being used as the building block for metric reconstruction of the space, their relationships to each other are exploited to build intermediate feature primitives. These features are more efficient in memory requirements and data association (Sohn & Kim, 2008). Feature based mapping was first proposed by Chatila and Laumond (1985), using sets of polyhedra to describe the geometry of environments. Because the features themselves capture the structure of the environment, the need for representations such as occupancy grid maps is unnecessary.

Five basic advantages exist for feature based maps over grid maps: they are more compact in structured environments, they are potentially more accurate depending on the accuracy of object models, they are necessary for dynamic environments where objects might change location, they may be constructed so as not to be reliant on pose information, and they are closer to operators natural perception of the environment. In addition, depending on the task, they may be more useful. The major drawback to feature based maps is that they are in general restricted to simple geometries. They do, however, scale with the complexity of the environment and not the environmental scale.

Algorithms based on geometric features are more efficient compared to point-based matching approaches (Sohn & Kim, 2008; Nguyen, Martinelli, Tomatis, & Siegwart, 2005). This advantage, however, is only realized when the number of features is kept as small as possible to avoid data redundancy. Feature based approaches usually follow this three step process: feature model extraction from sensor data, feature correspondence using a metric similar to the Mahalanobis distance, and pose estimation most commonly with a least squares method (Sohn & Kim, 2008). Most approaches assume all sensors perceive a constant and identical height and that relevant objects are at the height of sensor perception (Amigoni, Gasparini, & Gini, 2006). They usually work in three phases: line model acquisition, scan alignment, and feature fusion. Line model acquisition is in the form of data clustering. It involves determining where the lines are in the data, what data corresponds to which lines, and estimating the line parameters to describe these lines (Nguyen, Martinelli, Tomatis, & Siegwart, 2005).

Line segments are the simplest form of geometric feature primitives and are particularly useful when mapping office-like environments (Sohn & Kim, 2008). Robot localization, dynamic map building, 2D scan segmentation, and pose estimation reflect early efforts to apply this technology to robotic systems (Nguyen, Martinelli, Tomatis, & Siegwart, 2005; Castellanos & Tardios, 2000). Equation (18) displays the most common representation of line models

$$x\,cos\,\alpha + y\,sin\,\alpha = \rho \tag{18}$$

Where ρ is the minimum distance from the scan origin to the normal of the line and α is the angle between the line and the x axis. Other additional parameters may be involved in determining a line model (Amigoni, Gasparini, & Gini, 2006) as shown in Equation (19).

$$s = f\left(\alpha, \rho, \{x_1, y_1\}, \{x_2, y_2\}, \sigma_\alpha, \sigma_\rho\right) \tag{19}$$

Where α and ρ are the line parameters, $\{x_1, y_1\}, \{x_2, y_2\}$ are the coordinates of extreme points in a scan, and values of σ represent uncertainties of the aforementioned parameters. For LIDAR data, σ_ρ is a function of σ_α in accordance with Equation (20)

$$\sigma_\alpha = \sin^{-1}\left(\frac{\sigma_\rho}{L/2}\right) \tag{20}$$

Where, σ_ρ is the distance between the line segment and the scan point of maximum distance away from it and *L* is the line segment length found by subtracting the extreme points. Figure 32 displays these parameters and their relationships to one another.

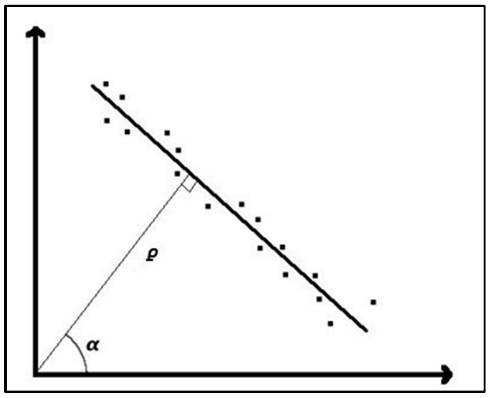


Figure 32: General structure of line parameters for feature based approach to mapping involving line segments

After line parameters are determined, their correspondence is found in three main steps: finding possible transformations between scans, evaluating these options to find the best transformation, and applying the transformation to achieve new merged line parameters (Amigoni, Gasparini, & Gini, 2006). After line clustering is optimized, partial scans must then also be combined with other partial scans relative to the order information was received.

4.3.2 Model Building

These questions are addressed while making several assumptions about data collection. Scan points, although collected quickly, are not simultaneously acquired. Techniques assume batch acquisition, which means that points of a common scan are asynchronously measured. This is a valid assumption as long as the scan rate of the LIDAR is significantly high and the robot translational and rotational speeds are small (Nguyen, Martinelli, Tomatis, & Siegwart, 2005).

In building line feature models, the split-and merge technique is one of the oldest, yet most effective methods (Nguyen, Martinelli, Tomatis, & Siegwart, 2005). In this approach, scan points are split into groups based on how well they correspond to line models fit to the data. This is repeated until all points are members of cluster groups. Then, collinear groups are merged to find lines in the data. The pseudo code for this approach is shown in Figure 33.

Oftentimes, feature association is done by building clusters of data as opposed to the example of line segment features. K-means clustering is a popular approach that is tied to the Expectation-Maximization process. The rules that govern split-and-merge on the other hand depend on whether features fit certain geometric characteristics.

Algorithm 1: Split-and-Merge

```
1  Initial: set s1 consists of N points. Put s1 in a list L
2  Fit a line to the next set si in L
3  Detect point P with maximum distance dP to the line
4  If dP is less than a threshold, continue (go to 2)
5  Otherwise, split si at P into si1  and si2, replace si in L
by si1 and si2, continue (go to 2)
6  When all sets (segments) in L have been checked, merge
collinear segments.
```

Figure 33: Pseudo code for Split and Merge technique

The line data for step 2 is generated differently considering the application. Points are usually initially grouped based on proximity. An alternative approach is to take the first and last points in a scan for the line fit. This is referred to as iterative-end-point-fit (Nguyen, Martinelli, Tomatis, & Siegwart, 2005).

Sohn and Kim (2008) proposes a line segment feature model extraction approach they claim is more efficient than split-and-merge. This approach they call the sequential segmentation algorithm does not assign every point to a line segment. Instead, it searches for a minimum of three points in a scan that are geometrically close, consecutive, and have a low relative variance between each other. The line model parameters are found through least squares fitting. Resulting vectors consist of vector lengths, standard deviation of points that make up the vectors, and end points. As the number of extracted features increases, the computational time for sequential segmentation becomes increasingly more advantageous over the split and merge technique.

Other approaches, such as the incremental algorithm, Hough transform algorithm, line regression algorithm, RANSAC algorithm, and Expectation-Maximization do not perform as well as the split and merge technique when metrics such as speed and correctness are considered (Liu, Emery, Chakrabarti, Burgard, & Thrun, 2001; Nguyen, Martinelli, Tomatis, & Siegwart, 2005). Split-and-merge is fast because it leverages raw scan points and has high correctness because of very few false positives. The benefit of RANSAC, Hough Transform, and EM for line modeling of LIDAR data is in their ability to produce precise lines after several iterations. Although they estimate high false positives, they can resolve these mistakes and ultimately provide precise line locations that correspond geometrically with actual surfaces (Nguyen, Martinelli, Tomatis, & Siegwart, 2005). Therefore, split-merge techniques are best used for real time applications that are not as concerned with precise geometric reconstruction.

In order to merge co-linear clusters, a chi-square test is used to compute the Mahalanobis distance between each line segment based on the covariance matrix of line parameters (Nguyen, Martinelli, Tomatis, & Siegwart, 2005). If *x* is an array of values where rows correspond to measurements and columns observations, μ is the mean vector of measurements in *x*, and *S* is the covariance matrix of *x*, the Mahalanobis distance *D* is governed by Equation (21)

$$D = \sqrt{(x-\mu)S^{-1}(x-\mu)} \tag{21}$$

If this distance is small within a threshold, the two line segments are merged to constitute one line. After merge, r and α are recomputed for the raw scan points that make up the set of merged data.

Once points are classified in their respective line segments, a correspondence between features must be established. In general, this process is referred to as scan matching; however, the information matched is purely based on the features chosen in the environment. Maps integrate new information from scans with the information already integrated into the map. This opens the door for multi-robot based mapping because of the potential to decrease map complexity. Scan matching has been found to be more accurate than Markov localization but less robust as noise levels increase (Taylor, 2009).

4.3.3 Pose estimation

Most techniques that involve merging sub-maps assume that the pose information is known at all times (Amigoni, Gasparini, & Gini, 2006). Others view pose as a global feature. One of the early approaches inputs the orientation of each individual line segment into a histogram where global pose is computed by cross correlation (Weiss, Wetzler, & Puttkamer, 1994). This method is not very robust under dynamic environments. A few approaches treat pose information as a local feature. These approaches observe pose relationships between individual line segments in a scan (Amigoni, Gasparini, & Gini, 2006). This relationship is first exploited in the line model building phase as lines are merged if their pose relative to other lines is below a certain threshold. Pose information is then viewed as a more global feature as a scan is compared to the information already compiled within the global map.

Because angles are viewed as global features, the correspondence step of these approaches involves aligning angles between segments as opposed to the segments themselves (Amigoni, Gasparini, & Gini, 2006). It is usually not practical to compare every line segment in a scan; therefore, there are several rules that help to eliminate comparisons. First, if extremal points are part of the segment model, line segments are understood to be consecutive when they have these points in common. This point being shared is like a pivot point between two linkages. Lines are assumed to be tied together. With LIDAR data, this consecutiveness may be assumed in data that is ordered within the same scan. Line length is also a parameter that limits the number of comparisons. If lines are not long enough then they aren't part of the scan used for correspondence. Angles between line segments are also weighted based on the length of those segments that make the angle. It is assumed that longer segments give more accurate information about the environment.

4.3.4 Make the transformation

All possible transformations must first be evaluated in order to find the best one for the given data. This is done by comparing the common parts of matched lines as they are projected on one another. There are a variety of approaches that achieve this according to the literature. Once the best transformation has been identified, consecutive lines are usually represented as a polyline and fit into the larger mapped space (Amigoni, Gasparini, & Gini, 2006).

After polylines are generated, there are three methods for merging partial maps: sequential method, tree method, and pivot method (Amigoni, Gasparini, & Gini, 2006). The sequential method requires *n-1* potential integration steps. The map is grown by encompassing the next scan at each iteration. The main issue with the sequential method is the map continues to grow and therefore becomes more difficult to match to scans because the possibilities increase. The tree method looks to avoid adding a small scan to a much larger partial map. In this technique, each partial map is merged with its similar sized peers of similar makeup. The intermediate integrated maps are then merged in a secondary merge step. The tree approach calls for (n-1)n/2 map merging integration steps, making it difficult to implement in online algorithms. There are modifications to this approach that can be made to speed it up for real time use. One method involves determining if an intermediate step is necessary if adequate performance can be achieved by merging smaller maps directly. Another issue with the tree method involves merging similar parts of a map repeatedly in the presence of inaccuracies. The pivot method combines features of both the sequence and tree methods. Intermediate maps are first constructed similar to that of the tree method. These intermediate maps are then merged either with other intermediate maps or individual scans, depending on their number of features. This reduces the number of integration steps to be less than the tree method and slightly more than the sequence method (Amigoni, Gasparini, & Gini, 2006).

4.3.5 Hybrid Data association Techniques

There are hybrid data associations that combine grid map style representation with the object based approach and to assume a simple office-like environment (Thrun, 2002), but there are also iterative approaches that correct errors in each respective geometric surface over time (Liu, Emery, Chakrabarti, Burgard, & Thrun, 2001). Although the terminology is the same, the hybrid approaches discussed here are different from hybrid maps that involve both metric and topological components. These approaches use a combination of point based and feature based metric representations of observations to build the map.

Some approaches involve more point based and probabilistic principles that involve leverage of features. An alternative approach to occupancy grid maps for mapping in dynamic environments is known as Dogma's method and is a derivative of the EM algorithm. This approach constructs a sequence of object models which are collections of local occupancy grid maps. The algorithm then extracts object identifiers using image differencing techniques (Thrun, 2002). These footprints are then transcribed to the global model with the constraint that the same object may not appear twice to prevent registration of identical objects separately. Assuming that a feature based correspondence may be maintained over time, objects are tracked through their unique characteristics in the global environment. The drawbacks of this algorithm are that it has difficulty with fast moving objects, it is limited to those objects that are easily segmented in preprocessing, and it cannot deal with non-contiguous objects (Thrun, 2002).

For feature based maps, some make the argument that approaches will never be robust that rely on reproducing geometric relationships. There are feature based approaches that involve point based probabilistic principles. IPSLAM creates a map of geometric primitives in real time from features derived directly from sensor data. In order to build these features, it uses Principle Component Analysis with FastSLAM techniques that leverage line segments. Similar to this approach are those similar to VecSLAM (Sohn & Kim, 2008), which considers sensor characteristics in mapping and loop closure.

4.4 Qualitative Reasoning

Section 4.4 establishes the background and potential for a qualitative mapper. Qualitative feature extraction is not a new theory; however, the development of a qualitative mapper is novel. An introduction to the backbone of a qualitative mapping approach is first given through the presentation of qualitative spatial reasoning literature. Then, the imact of qualitative reasoning is discussed through the possibilities of establishing a qualitative mapper. This discussion is done with the particular technical challenges in mind and the potential opportunities that emerge from Operator Blending.

4.4.1 Introduction to Qualitative Reasoning

To the author's knowledge, the reader will be presented with the first fully qualitative approach to mapping in Chapter 5 of this dissertation. Qualitative techniques have contributed to information extraction in support of quantitative techniques, however, maps independent of direct spatial correspondence have yet to be developed. The literature for current qualitative information extraction techniques are presented in this section.

If feature based maps move closer to representing environments more along the lines of biological systems, qualitative maps attempt to emulate the typical biological representation. These approaches revolve around reproducing a semantic formulation of the environment. For human-robot teams, the first step to qualitative mapping is often seen as producing a compatibility between the robot and human perception of the environment (Kuipers, 2000; Nguyen, Martinelli, Tomatis, & Siegwart, 2005). Once a semantic spatial relevance is extracted, the maps produced have a higher human-robot compatibility.

Traditional quantitative models for mapping may not adequately abstract underlying spatial information needed for mission planning (Kohler, Ottlik, Nagel, & Nebel, 2004). Bayesian Belief Networks, Hidden Markov Models, or Neural Networks are probabilistic and may be used to incorporate a limited degree of learning into the system. These methods, however, are not intelligent, because they do not possess a qualitative component that takes advantage of contextual information. Intelligent systems use spatial reasoning based on local information (Thau, 1997). The reasoning tasks that qualitative systems perform are the derivation of new knowledge form given information, verification of consistency of information, update of knowledge, and the achievement of a minimal representation (Kohler, Ottlik, Nagel, & Nebel, 2004).

Although the poverty conjecture given by Forbus (1995) states that there is no purely qualitative spatial reasoning mechanism, a branch of literature concentrates on making decisions and representing data qualitatively. This has led to qualitative approaches to mapping that represent information, make decisions based on qualitative inferences, and represent data based on relative distance measurements for a more localized interpretation.

Qualitative Spatial Reasoning (QSR) is most popular in the fields of artificial intelligence and motion analysis but branches out into the realm of qualitative mapping (Kohler, Ottlik, Nagel, & Nebel, 2004). It is the theory that allows robots to represent and reason with spatial entities without resorting to traditional quantitative techniques. The majority of the systems make metric and topological distinctions to varying degrees. What they have in common is the exact measurements of system components are not important compared to the meaning of the relationships between components. The rational for its establishment as an alternative to quantitative approaches is that traditional methods do not support intuitive or common-sense human-computer integration (Kohler, Ottlik, Nagel, & Nebel, 2004).

It follows 3 guidelines: no precise representations of geometric constraints, qualitative spatial descriptions are used to serve as a guide to the exact motion analysis, the results of which can be explained intuitively, and computational tractability and efficiency. QSR relies on representing the position of a robot relative to its environment in the position parameter space of the robot, known as the configuration space, or C-space. For mapping, the most important issue to be addressed is how to map continuous spatial quantities into qualitative representations within the C-space. Consider the C-space the eigenspace of qualitative observations the robot may use for decision making. This approach is considered to be similar to the manner in which humans conceptualize spatial information (Renz & Mitra, 2004).

Emphasis is on path planning in the origins of QSR. This, however, may be easily extended to mapping if the spatial analysis goes from the question of planning a future path to representing prior information. Approaches to QSR that involve local information are particularly applicable to mapping methods. Qualitative information from data scans may be related to each other qualitatively. Faltings (1987) introduced the concept of place vocabulary, which makes the case that the connectivity relationships between objects or information that makes up the C-space should be used as the main components of qualitative states. In other words, the content of a scans at time step t should be analyzed for its significance qualitatively to a scan taken at time step *t*+1. Their precise geometric relationship is not as important as their significance.

Local information in QSR may also be used for route planning which differs slightly from path planning. Route planning considers current information and makes decisions in real-time for the translational decisions of a robot. Path planning, on the other hand figures out the best path to get from point a to point b and then executes. Therefore, route planning may benefit widely by the implementation of local information based QSR approach. Kuipers (2000) creates approaches to route planning based on incident relations. This approach creates maps by inferring from commands issued by the robot to turn or go forward based on decisions from route planning algorithms. Fuzzy maps make these qualitative decisions by representing ranges of spatial relationships to have the ability to represent the metric characteristics of an environment.

The extension of route planning to place vocabulary is found in qualitative spatial inference techniques (Liu & Daneshmend, 2003). Simplified and intelligently chosen trigonometric rules are used in this approach for geometric scene description. Once a scene has a description, a primitive representing the characteristics of the scene, may be used as a building block for mapping. Kuipers (2000) qualitative simulation theory is the most well-known method for deriving these types of rules from demonstrated robotic behaviors.

What makes this process qualitative is the use of qualitative spatial inference or the use of available data to come to a conclusion at a higher level of abstraction. A simplistic example is, if given three points a, b and c in space where the Euclidean distance between a and b is much less than the distance between a and c, and the angle between ac is slightly acute, deduce the relative orientation of bc and ac and the distance between b and c (Liu & Daneshmend, 2003).

Forbus (1995) makes the case that no purely QSR method exists to adequately produce non-ambiguous results. He claims that QSR is still useful as a method to guide and improve the efficiency of quantitative approaches. A popular method for taking advantage of this system is to use qualitative spatial inferencing to provide an initial estimate of the movement behavior and then search for exact configuration positions based on numerical ranges.

<u>4.4.2 Modeling Qualitative Space</u>

Heading and distance relationships are considered two of the most effective ways to model space qualitatively (Renz & Mitra, 2004). Place vocabularies must have two qualities: computationally efficient spatial inference and precise spatial descriptions with minimal ambiguity (Liu & Daneshmend, 2003). Inference is generalization based on incomplete, relative information and context.

Distance in QSR is always expressed relative to another reference distance *dconst*. The description "the distance x is much shorter than *dconst*" or "the distance x is comparatively similar to dconst" are typical examples. Qualitative distance is usually denoted by [x|*dconst*]. The qualitative relationship may also be understood as the displacement from point p to point q, which would be denoted similarly as [pq|*dconst*]. Normalized distance values x/*dconst* or pq/*dconst* may then be used to define set spatial intervals based on set numerical interval values. These values that define intervals are often based on experimental results or quantitative observations about the environment. For instance, if the interval values are {2/3, 1, 3/2}, it defines 5 possible inferences for the spatial domain.

This framework is useful because it allows for the space to be measured based on local information and the semantic relevance of measurements. As opposed to quantifying information to the nearest unit length, the length of an object is compared to a known length with significance to an objective. These types of determinations are useful for the establishment of a functional categorization of observations. If the observed distance is determined to not be comparable to *dconst*, it also does not possess the functional characteristics associated with a length comparable to *dconst*. Likewise, if the distance is comparable, it may be assumed to have the known functional characteristics associated with the repective measurement.

Ambiguity may exist when these inferences are combined. Therefore, there must be rules in place to ensure consistency. The basic approach is to assume a combination of inferences has more than one possible value in these circumstances. For instance, if five inference measurements are conducted as a robot travels down a hallway and four inference measurements are made on its way back to the same location, the qualitative framework must allow the robot to end up at the start point. How this is dealt with is discussed further in Section 4.4.3.

$$\text{Less} = \{x | x \in R, x/dconst \in (0, 2/3)\}$$
$$\text{SlightlyLess} = \{x | x \in R, x/dconst \in (2/3 , 1)\}$$
$$\text{Equal} = \{x | x \in R, x/dconst = 1\}$$
$$\text{SlightlyGreater} = \{x | x \in R, x/dconst \in (1, 3/2)\}$$
$$\text{Greater} = \{x | x \in R, x/dconst \in (3/2 , \infty)\}$$

Figure 34: Distance determination example in Qualitative Spatial Reasoning

Angles are described qualitatively in a similar manner. If a ray is projected from the heading of a robot before rotation, the angle would be the angular displacement of the robot from projected ray to final position. Angle interval values are set in much the same manner as those set for distance. For example, if the angle is assumed to range from 0 to π, the interval values may be set to (π/3 , π/2 , 2π/3) which correspond to 5 angular inferences.

$$\text{Acute} = \{\theta | \theta \in [0, \pi/3)\}$$
$$\text{SlightlyAcute} = \{\theta | \theta \in [\pi/3 , \pi/2)\}$$
$$\text{RightAngle} = \{x | x \in R, x/dconst = \pi/2\}$$
$$\text{SlightlyObtuse} = \{x | x \in R, x/dconst \in (\pi/2 , 2\pi/3]\}$$
$$\text{Obtuse} = \{x | x \in R, x/dconst \in (2\pi/3 , \pi)\}$$

Figure 35: Angle determination example in Qualitative Spatial Reasoning

The basis of inferences in neutral references is similar to the manner in which humans use contextual information to make similar judgments (Liu & Daneshmend, 2003).

4.4.3 Components of Qualitative Theory

For mapping, there are widely considered four components: ontology, spatial relations, mereology, and mereotopology (Kohler, Ottlik, Nagel, & Nebel, 2004). Ontology is also known as the basic space of measurement. An example of this is making a decision on a 3D or 2D representation of the space. The ontology for a 3D representation would be building blocks or measurements that lead to that goal. Spatial relations are metric in nature and may be represented as exact measurements or relative measurements depending on the approach. Mereology is the theory of parthood. It makes the case that more complex object models may be represented by their constituent parts. Mereotopology is how mereology and topology may be integrated together. This indirectly makes the case that topology must be viewed as a compliment to metric approaches. Much like the poverty conjecture, mereotopology makes the point that QSR involves both metric and topological models. Topological information may be the driving force behind a QSR approach but there are no purely topological models in the theory.

Inferences are used for mapping through qualitative trigonometry or qualitative arithmetic. Qualitative Trigonometry deals with relating angular inferences and translational inferences together based on their geometric relationships. Qualitative arithmetic involves relating successive translational inferences or angular inferences when they are added to each other.

Qualitative Trigonometry involves deducing a spatial relationship or perspective range of spatial relationships from qualitative inferences in triangles. The simplest example is given that A and B are qualitatively observed to be Less than a *dconst* and the angle between them is Acute, it may be inferred that C is also acute and its adjacent angle is Obtuse or SlightlyObtuse. The status of the angle is found by observing the bounds set by the interval values in an inference and solving for the value of the angle. After each angle value is solved for considering the bounds for the given conditions in the problem statement, its value puts it within the bounds of Obtuse or SlightlyObtuse. A complete qualitative trigonometric framework calculates this for all of the possible inferences of the length A, length B and angle between the two segments.

Qualitative addition observes the result when two spatial inferences are added together. This is more relevant to qualitative mapping because the final state may be inferred by observing the addition of two previous states. Table and Table show examples for qualitative addition for translational inferences and angular inferences respectively. In these tables, ‘l’=Less, ‘sl’=SlightlyLess, ‘eq’=Equal, ‘sg’=SlightlyGreater, ‘g’=Greater, ‘a’= Acute, ‘sa’=SlightlyAcute, ‘r’=RightAngle, ‘so’=SlightlyObtuse, and ‘o’=Obtuse.

Table 3: Qualitative arithmetic for addition of two translational inferences

ADD		<L$_2$>				
		<l>	<sl>	<eq>	<sg>	<g>
<L$_1$>	<l>	<sg>	<sg—g>	<g>	<g>	<g>
	<sl>	<sg—g>	<sg—g>	<g>	<g>	<g>
	<eq>	<g>	<g>	<g>	<g>	<g>
	<sg>	<g>	<g>	<g>	<g>	<g>
	<g>	<g>	<g>	<g>	<g>	<g>

Table 4: Qualitative arithmetic for addition of two rotational inferences

ADD		<θ2>				
		<a>	<sa>	<r>	<so>	<o>
<θ1>	<a>	<a—so>	<sa—o>	<so—o>	<so—o>	<o>
	<sa>	<sa—o>	<o>	<o>	<o>	<nil>
	<r>	<so—o>	<o>	<o>	<nil>	<nil>
	<so>	<so—o>	<o>	<nil>	<nil>	<nil>
	<o>	<o>	<nil>	<nil>	<nil>	<nil>

These tables assume that both inferences are in the same direction. The value of nil means that when added, the value of the resultant angle is greater than the specified range of 0 to π and therefore is invalid. Also, multiple possibilities are represented by the '–' notation in the table. This means that all values from the smallest value indicated in the table to the largest value indicated in the table are possibilities. To resolve these ambiguities, an additional source of information is needed. For the simplest case involving 2D information, the translational inferences and angular inferences may construct the place vocabulary for the building of qualitative maps.

4.4.4 Mapping Qualitatively

In control theory and some metric mapping approaches, qualitative inferences are sometimes called 'fuzzy' if it may be used as partial evidence to support a range of results. Control rules are defined for each of the inferences specified. The accuracy of sensors must be taken into account when building maps through combining inferences. Therefore, two sources of ambiguity exist, that from the actual sensor and that from the 'fuzzy' observation. Fuzzy information is used to produce more precise maps as a means to resolve ambiguities or low confidence for quantitative maps. It has the potential to reduce computational intensity by shifting away from the point or feature correspondence problem, although this role for fuzzy information is not typical.

The main drawback of QSR is that, because of the poverty conjecture, it is not seen as a solution that completely stands alone without a quantitative component. This component may be a very small part of the overall approach that reflects a specific type of measurement such as forward translation. Or QSR may take more of a supportive role as the fuzzy logic component of a particle filter. In addition, if the robot uses QSR for navigation in Autonomous mode or Shared mode, it must translate top-down decision making to the base sensorimotor commands to achieve the desired actions. Understanding sensorimotor of individual actuators based on sensor values is known as the Piano Mover's problem and often involves a more quantitative approach. This work assumes that sensorimotor information is not known precisely, which establishes one of the main departures from traditional QSR theory.

4.4.5 Movement and Qualitative approaches

As discussed, Qualitative and quantitative models may be metric, topological, or a combination of both. If raw sensor values are used to represent qualitative or quantitative relationships, the map is considered metric. Likewise, if the relationships are presented to represent sensor values, this map is topological. Although many approaches are presented as purely metric, they often involve probabilistic methods that take qualitative information into account. The poverty injecture for the qualitative mapping research community makes the statement outright that there is no such thing as a purely qualitative map. Therefore, topological maps include some form of mereotopological information that makes aspects of the map quantitative.

Because there are no purely quantitative or qualitative approaches, there are pluses and minuses to building models over time. Qualitative approaches involve making judgments based on the sum total of sensor information. Because of this, it is easy to incorporate a system that integrates information over time and measurement over time may be advantageous in some respects. The U_x and U_h error accumulates for qualitative approaches; however, the function of the resultant models relies on matching relationships rather than the correspondence of metric measurements. For quantitative approaches, even in systems that take advantage of measurement over time, the benefits are balanced with the effects of sensor error on correspondence. The inaccuracies accumulate over time for quantitative approaches as error accumulates. Statistical methods are introduced to give some qualitative way of dealing with measurement uncertainty mathematically, however, the effects of measurement over time dominate the performance of the system. For search systems designed to translate within the environment, movement is a function of time, because sensor information over time is integrated into the map.

The advantageous use of movement for more accurate mapping may therefore be more meaningful for qualitative mapping techniques. To make inferences and understand the relationships of objects, more complete spatial information is beneficial. In quantitative approaches, movement should be limited to what can be modeled by the quantitative system based on the task at hand. Qualitative models, however, may benefit from movement for the sake of more complete models.

4.5 Chapter 4 Summary

This chapter presents the navigation and subsequent mapping process as core components for intelligent information gathering. The majority of biological systems use navigation as the means in which necessary information is gathered about their environment. The approach to mapping varies based on the type of information that makes up the map, which delineates metric and topological representations, and the methods used to generate the map, which separates qualitative and quantitative methods.

The approach to mapping and the structure of the map should largely be based on the goals of the mission. Quantitative mapping techniques tend to favor metric information, because these maps are based on probabilistic theories that require a majority metric component. Qualitative maps, on the other hand, tend to consist of a majority of topological information, because their intent is to communicate the relationships that dominate spatial representations. QSR is a research field that has emerged to formulate qualitative relationships and how they might be represented in space. These theories may provide the basis for qualitative map construction.

For human-robot collaboration, the process of acquiring information and how this information is represented spatially forces navigation and mapping to impact team communication. Both agents must have a functional understanding of the others view of the environment and relate to this representation. The common understanding of spatial relationships between robots and humans must be established. This chapter presents the dominant approaches to building these relationships in the literature, with QSR as a potential means to extract the needed relationships. This approach is used in Chapter 5 and Chapter 6 to solve perception and integration challenges of human-robot teams.

5. RobiSEND Methodology

"Creativity is the defeat of habit by originality." **–Arthur Koestler**

This research shows the value in presenting an original approach to human-robot relationship building. Developers of technology must avoid only advancing technology through iteration or the mentality of 'give me a better version of the same'. Robots may be programmed to jump higher and move faster with few major attempts at rethinking their approach.

Innovation requires an embracement of being different and willingness to be creative. Researchers must take a step back and re-evaluate their role in the process of development. Although iteration may even be encouraged, it is the responsibility of researchers to lead the charge for innovation. Only then will we truly make a truly novel contribution.

Human-robot teaming as addressed in this research draws upon the breadth of knowledge in several areas to innovate MRUAV based search missions. Out of this synthesis comes a new theory that creatively blends the fields of human-robot interaction and mechatronics to develop a system that originally solves new and exciting research problems.

5.1 Overview

Section 5.1 provides an overview of how the mereology of previous chapters contributes to the RobiSEND system and Operator Blending. A brief synopsis is given in sub-section 1. Particular emphasis is given to the novel ways in which RobiSEND establishes relationships between the robotic and human agent is given in sub-section 2. This is to establish the foundation for the novel contributions of this work in Chapter 5 and Chapter 6.

5.1.1 Methodological Synopsis

This chapter presents how the literature of Chapter 2, Chapter3, and Chapter 4 contribute to the perception and integration methods of the RobiSEND system. The technical approach to object tracking, feature extraction, and observation modeling are covered along with the model for the operator role in their development. The functions and algorithms presented in this chapter are the building blocks for Operator Blending, Code of Observational Genetics, and Qualitative Unsupervised Intelligent and Collaborative Keypoint (QUICK) mapping outlined in Chapter 6.

Human-robot collaborative systems are impacted by human aspects of Chapter 2, robotic algorithms and approaches of Chapter 3, and navigation, movement, and mapping of Chapter 4. Although Operator Blending is a new branch of this theory, the aforementioned literature defines its components. All aspects are critical for localized human-robot teaming situations when collaboration and cooperation are necessary. In order for operator blending to be tangible, the RobiSEND payload is fabricated to present these ideas. The application of MRUAV based local search is the muse for operator blending and its development.

Although these theories may be applied to any unmanned system in a teaming situation, the RobiSEND payload is designed for free exploration only possible by the conceptualized MRUAV capable of stable hover and 15 minutes of flight time. This may be a rather ambitious assumption; however, more traditional assumptions to simplify the problem are not made in this research that include perception and integration challenges.

5.1.2 Importance of Relationships

The nature of perception and integration is not trivial when the operator and MRUAV must build relationships to the environment and each other. Control theorists and the many researchers that advance the field believe that measurement of the robot state is both possible and necessary with little concern to how the robot relates to its surroundings or those it is designed to support. They, however, do not address the challenges of perception and integration that are now one of the major limiting factors in the implementation of this technology. These challenges may only be solvable through human-robot collaborative theory. Operator blending as presented in Chapter 6 is the branch of this theory that leads to collaborative and cooperative relationships.

The specific application addressed in this research is local search conducted within a MRUAV OHOR search scenario. In this scenario, the odometry data of the robot is not known. In addition, global feature extraction methods such as heading acquired from a magnetometer or position acquired from GPS are not known. Therefore, it is difficult to build and update control theory related parameters of the MRUAV model. This work operates under the new theory that, as opposed to modeling the robot, it is necessary to model the operator and robot relationships. Simply because of the lack of global information from the aforementioned sensors and complete knowledge of the MRUAV model, these relationships are seen as a necessity. In addition, if the relationships the robot must generate to its environment and teammates are modeled, more useful deliverables also may result. If these relationships involve collaboration and cooperation from robotic and human team members, the process of modeling these functional relationships is Operator Blending. Both human and robot contribute what they do well to the success of the team.

An overview of the research approach is given, followed by a description of the typical mission. Individual hardware components of the system are then presented with justification for their selection.

The user interface, navigation methods, and approach to operator feedback are introduced next along with an understanding of operator attention essential to the Operator Blending framework. Next, the technical aspects of feature extraction methods are presented that cover object tracking and object modeling of operator selections. Similarly, the technical aspects of modeling for the mapping approach are provided. Chapter 6 then integrates these components into the Operator Blending methodology.

5.2 Research Summary

The constraints of the RobiSEND design are given in Section 5.2. Integration factors are first addressed in sub-section 1 followed by perception factors. These factors are summarized in light of the specific approach to solving these problems given in this Chapter.

5.2.1 Application Introduce Constraints

The development of the RobiSEND payload is largely cost, weight, size, and mission driven. These constraints are derived from the robot and operator being reliant of the performance of each other due to limited resources. Table 4 breaks these limited resources down and displays how they lead to the specified constraints.

Table 5: Application based constraints for the RobiSEND operator and MRUAV

Factor	Human Constraints	MRUAV Constraints	Result
Cost	Human team members have high human capital and high percentage of team resources outside of tactical situation	MRUAV has low human capital so must have low percentage of team resources to be viable	MRUAV may be expendable with little impact on team ability to operate outside of the tactical situation
Weight & Size	Human team members have finite maneuverability due to environmental and endurance based limitations	MRUAV is highly maneuverable in the environment but only when operational	Human team members may have difficulty transporting the MRUAV before operation
Mission	Human team members do not have necessary information about the environment to perform local search efficiently	MRUAV does not have all necessary information from sensors to perform local search efficiently	Operator Blending framework necessary for perception and integration capability of team

The following assumptions about the developed payload were made considering these constraints. The theoretical MRUAV is assumed to be capable of a steady hover, a stable speed of 1 m/s, movement with 6 DoF, and a flight time of 15 minutes and is tested in a hardware-in-the-loop scenario. All system components are off the shelf and total one order of magnitude more expensive than the platform itself. Because of this, the total cost of the system under $2,500. This difference between sensor suite and MRUAV cost may be nearly eliminated in a mass production cycle.

Considering the relatively low cost and requirements, the level of performance needed is possible because of Operator Blending. Operator Blending provides novel ways of achieving MRUAV perception and integration for local search tasks in unknown mission environments. Chapter 6 will present the full theory of Operator Blending. The purpose of Chapter 5 is to flesh out the details for specific building blocks of this theory considering the explored example application of MRUAV based local search.

5.2.2 Function of Human, Robot, and Navigation

The primary sources of information are a binocular stereo camera system, LIDAR, and operator input. The LIDAR provides information in the near field and a global map representation through feature based observation modeling. RobiSEND updates the accuracy of the map and annotates it with additional models developed to convey topological information. Towards topology and through the video centric GUI, the operator identifies objects, observations of merit, and points of the departure for the MRUAV by providing intelligent information gathering capabilities. Sensor observations with particularly definitive characteristics are classified based on their functional relevance to the search process. The novel contribution of this work is the approach, which leverages human input for positive influence on algorithmic processes and more useful deliverables for human team members.

Perception and integration challenges shape the necessity for codependent and cooperative relationships between the operator and MRUAV. It is desirable for the MRUAV and operator to have a 1 to 1 human to robot ratio, the ability to engage cluttered environments more effectively than traditional approaches, and the capability to map unknown areas effectively for the specified search mission profile. The need for these approaches is outlined in Chapters 1-3.

The operation of RobiSEND, the navigation process, and the nature of deliverables for codependent and cooperative relationships is framed in Chapter 4. System navigation is coupled to the search process to achieve a 1 to 1 operator to MRUAV ratio, reduce operational complexity, and increase mission cohesion. The mapping of realistic environments that include clutter is made more useful through the advent of operator directed attention in map construction. This improves the effectiveness of the map and team considering limited resources. At the conclusion of the flight, a fully annotated map specifies portals, boundaries, and objects of interest.

5.3 Research Approach

Section 5.3 establishes the approach to research and system development taken in this dissertation. Sub-section 1 gives a general synopsis of the main components of the final system based on MRUAV local search constraints. Sub-section 2 then gives an example of a typical mission that a RobiSEND system might impact if implemented.

5.3.1 Synopsis of Approach

As mentioned in Chapter 2, autonomy has been the traditional focus for advancement of unmanned systems technology. Mission goals have been structured accordingly. For example, soldiers request UAS flight times on the order of 45 minutes to an hour, which is a requirement for fly ahead or macro-level surveillance but not reflective of the amount of time a soldier should direct his attention in a search or mapping operation. It is through local search, however, that MRUAVs have the opportunity to realize their potential. In close quarters, environments are three dimensional, requiring MRUAVs to provide pertinent information as quickly and usefully as possible.

The sole motivation of the payload for this MRUAV search system is to improve situational awareness for a tactical team through remote exploration of an unknown environment. This incorporates additional considerations that are tied to how the search platform interfaces most effectively with the operator. RobiSEND is built considering human-robot interaction, sensor fusion, perception, and mission protocols and procedures and based on the operator blending framework outlined in this Chapter.

In light of human involvement, some system limitations become opportunities for solutions. For example, although MRUAV search systems have limited payloads, their low total weight makes them backpack-able and easy to deploy. Their limited flight time may not allow them to be suitable for long range missions; however, short search durations are more suitable for maintaining operator situational awareness. Complete metric reconstruction of an unknown environment may not be plausible; however, semantic topological maps are more operator relevant deliverables in teaming situations. Therefore, the MRUAV function, role of the operator, and mission goals are recast to improve the value of autonomy in teaming situations.

The building blocks of feature based mapping conducted by the RobiSEND system are object and mapper models. All models are built from building blocks based on local information. The nature of this local information and how each of these two types of observation models are included differs depending on this distinciton. Object models leverage local information and are themselves local representations, but they may be used to annotate global deliverables. Mapper models are also built upon local information, but they are combined to become global deliverables. This is an important distinction and must both be included in the RobiSEND framework. Said in another way, mapper models are similar to Legoes. Their purpose is to be combined with other mapper models to make a global structure. Object models are like accessories. They stand alone and can be appreciated as their own entity or added to a global entity to improve its appearance or performance.

For RobiSEND, as mapper models are created, they ultimately make the structure of the map. Object models are used to annotate the final map and drive the navigation process. LIDAR is used to develop mapper models and the binocular stereo camera is primarily the source of object model information, with the LIDAR used when disparity is not available. Both sensors are used relative to how they contribute to specific aspects of observation model building and local search. Object modeling involves target identification, tracking, and categorization that is best dealt with through an electro optical approach that is supported by laser data. Likewise, during the construction of the global map deliverable through mapper model generation, laser data does this more precisely. The operator directs the attention of the system for the observation model generation process.

5.3.2 Typical Mission

The system is designed to support the mapping of buildings and the identification of objects of interest. Local measurements are relative to the egocentric perspective of the MRUAV and the localized starting location of the human and robot system. To a large extent, the system is intended to be expendable and light enough to marginally affect the total weight of gear the operator might carry.

The proposed platform is designed for the purpose of environmental awareness beyond direct line of sight. This is accomplished through perception that actively captures information through a user driven search process. The user pursues relevant information in environments where the search process too dangerous or impossible to be carried out on foot. Agents of autonomy are introduced to the system to simplify the search process; however, the individual is intended to devote as much attention as possible to the MRUAV until it runs out of power or it becomes infeasible to do so.

An example of a typical mission consists of a soldier in an uncertain environment with the desire to enter adjacent buildings. The squad has secured the current building and there is a need to scope out a strategy for the systematic search of nearby points of interest. They do not have a map that reflects the current structure of the dynamic agents within the environment. These include people, vehicles, or possible damage to buildings or structures not indicated by the current visual aids in his possession. A map produced from an MRUAV search system is needed to develop spatial relationships to dynamic objects, static boundaries or perimeters, and also to landmarks that may be used in the development of mission strategy.

The MRUAV is deployed in forward flight mode to navigate into position. The operator then switches to RobiSEND mode and proceeds to search for or label relevant landmarks, structural elements, and points of departure for the MRUAV. The MRUAV adjusts altitude and flies from object to object. As it navigates, objects of interest are stored and annotated within the generated 2D SLAM map. The RobiSEND system finds a portal and adds its location to its generated map as a high priority object. Once power is significantly low or the search process complete, the MRUAV is left for future recovery. As the MRUAV lands, the annotated SLAM map is transmitted to the operator. A map is relayed to all of the soldiers in the squad complete with topological information and metric relationships. Components of the map may be defined based on the Operator Blended approach developed in this research.

5.4 Hardware

The hardware components of the RobiSEND system are detailed in this section. Sub-section 1 gives a synopsis of system components and the motivations for their selection. Each individual RobiSEND component is then discussed in detail with rationale for their selection. How these individual pieces fit together considering the goals of Operator Blending is then discussed. An understanding of RobiSEND components is needed to understand how RobiSEND demonstrates The Operator Blending concepts defined in Chapter 6.

5.4.1 Hardware Synopsis

The primary components of the payload system are LIDAR for precise 2D measurements, binocular stereo camera for visual information, and servo controlled tilt LIDAR mount used to simulate altitude changes of the MRUAV system. The binocular stereo camera transmits 8-bit, 3-channel color images and 16-bit single channel disparity images. All information is sent to a computer running MATLAB 2010b, which processes information from the operator and combines this with the information from RobiSEND sensors.

Because the payload developed in this research is intended for a MRUAV, the weight and size goals for RobiSEND construction from off the shelf components is 1 kg weight and total cost under $2,500. Approximately a third of the weight and two thirds of the cost corresponds to the stereo vision, servo motor, and LIDAR sensors. Sensor mounting and packaging, on board communication devices, on board embedded systems, and miscellaneous wiring are not designed in this work. The weight that these components potentially contribute is assumed to be no more than the weight of the primary sensors an actuators. Therefore, 0.5kg was allocated for this purpose. Therefore, the primary sensors and actuators must weigh less than 0.5kg as well. This parameter is realistically based in the payload carrying capabilities of MRUAVs on the scale of the RQ-16A T-Hawk.

Table 5 gives an overview of individual system components. The included parameters in the table capture the weight, size, power, and cost parameters. These components are not assumed to be the best components for an operator blended system. Rather, they satisfy the specified constraints of Chapter 4.

Table 6: Primary Sensors and Actuators of the RobiSEND system

	Component	Dimensions (cm)	Power (W)	Cost	Weight (g)
	Videre STH-DCSG Camera	4.3x13.2x4.1	1.6	$1,000	204
	Hokuyo Scanning LRF	5.0x5.0x7.1	2.5	**$1,100**	160
	Hitec High Torque Servo	4.1x2.0x3.8	1.4	$40	55

As shown, the total cost of the payload is less than $3,000 per unit. The total system weight is 1kg, which is a reasonable payload for state of the art MRUAVs with a minimum of 15 minute flight time. Further explanation on the selection of these components is given in the following sections

5.4.2 Videre STH-DCSG

As suggested in Chapter 2, applications that require real time disparity and color images for Shared mode autonomy need minimum frame rates of 7fps with a lag constant less than 150ms. This performance is in the presence of losses that may occur during operation in an unknown environment. The Videre product line was selected for its ability to perform the disparity calculation up to 20 fps, rigid packaging, and off the shelf cost of $1,000. This makes the disparity calculation reasonable for RobiSEND which is designed to operate at 10 fps.

The Videre STH-DCSG with 9cm baseline and 3.5mm lenses was chosen (Videre-design, 2010). The 3.5mm lens focal length was necessary to make the horizontal field of view over 90^{o} and the depth of field a minimum of 0.5m at infinity focus. The actual field of view of the left camera is 94x69˚. This camera possesses CMOS imagers, making image formulation faster with the disadvantage of increased signal to noise ratio. At the same time, because CMOS imagers convert charge from photon energy to voltage at each individual sensor, they can even out local overexposures and have more flexible shutter options.

The Videre STH-DCSG possesses a global shutter which eliminates the effects of vibration not fixable in post processing (Liang & Chen, 2005). MRUAVs, which experience high amounts of vibration, therefore, require a global shutter camera system with simultaneous exposure to eliminate vibrational effects. Camera specifications were also chosen so that stereo calculations produced acceptable results at distances at or shorter than 10m. Figure 36 displays how *Δr* changes as a function of distance from the camera and is calculated from Equation (7) with *e*=0.25, p=7µm, *B*=90mm, and *f*=3.5mm

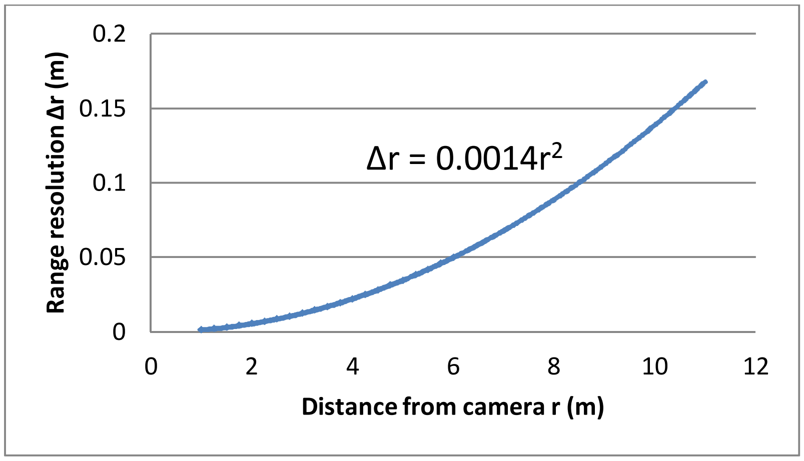


Figure 36: Videre stereo camera range versus distance from the camera

At about a meter out, the camera has a *Δr* of less than 1 cm. At 10m, this value is above 15cm. Although data from this camera is too inconsistent over distance for 3D reconstruction (Chang, Kuwabara, & and Yamamoto, 2008), it is useful as an instrument for RobiSEND object model creation.

The disparity parameters of the camera are held constant as images are captured. The camera uses the SAD method with sub-pixel accuracy. Filter values are set as follows: 100 for the speckle filter, 10 for the correspondence filter, and 10 for the uniqueness filter. Images are saved as 640x480 16-bit images. SAD is used to maximize the disparity calculation time with adequate correspondence.

The camera searches over 32 disparities for high resolution at close distances and horopter starting 1m in front of the camera. As a result of sub-pixel accuracy, possible disparity values δ_f go from zero for far away objects to 496 for close objects. This follows from Equation (22)

$$\delta_f = \frac{(\delta - 1)}{e^2} = \frac{(32\text{ pixels} - 1\text{ pixel})}{0.25^2} = 496\text{ pixels} \qquad (22)$$

where *e,* as covered in Chapter 3, is the camera sub-pixel accuracy.

Lastly, a large pixel window size of 23x23 was used for the SAD calculation. Although this dropped the camera frame rate to 14 fps, the higher correspondence generated over more pixels allows algorithms to associate disparity values to more objects in the camera FOV.

5.4.3 Hokuyo URG-04LX-G01

In order to limit the total cost of the system to under \$2,500 from off the shelf components, the Hokuyo URG-04LX-G01 was chosen (Hokuyo, 2011). It costs \$1,000 and has a 240^o field of view with an angular resolution of 0.36^o. Its maximum range is 5.7m with a depth resolution of 0.03m at distances less than 1m and 3% of the observed distance at distances over 1m. This makes it reliable for near field obstacle detection for obstacle avoidance and mapping at low speeds. It is capable of measurement scans at 10Hz and has been used as a sensor for SLAM and object detection.

The scan speed matches the needs of the stereo camera with more precise measurements in the 2D scan plane. Although the range is limited to 5m, the types of features needed for mapping are most prevalent in that range for low speed operation. In addition, the sensor is light weight and low power as shown in Table 5. Figure 37 compares the range resolution of the stereo camera and LIDAR.

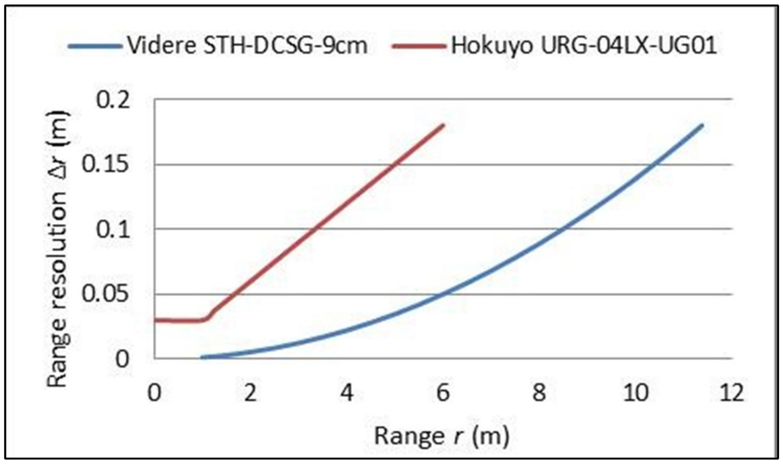


Figure 37: Range resolution of Videre stereo camera and Hokuyo LIDAR as a function of distance from the camera. This information is interpolated through sub-pixel accuracy for the stereo camera

The Videre STH-DCSG-9cm and Hokuyo URG-04LX-UG01 complement each other over the desired range of 20cm to 12m for object recognition. Figure 37 conveys that the camera may use subpixel accuracy to interpolate range measurements at about twice as long a range as the LIDAR sensor. The precision of these values is non-linear over range and has an unpredictable return rate that is dependent on lighting and contrast of object features. The calibration of the camera may also range in quality, further reducing the possibility of interpolating range information. These strengths and weaknesses make it a useful sensor for object model distance measures but not as much for the more global mapper models.

Although the LIDAR does not produce as high a precision, its resolution has a linear relationship with distance beginning at 1m. It is also capable of measuring object distances inside of the 1m minimum range of the stereo camera. As mentioned in Chapter 3, although LIDAR has a slight susceptibility to surface effects diminishing the rate of return of the sensor, it has a much more reliable return rate over its range than that of the stereo camera. The linear nature of the sensor, much more encompassing FoV, and reliability over its range make it a better candidate for mapper models. The LIDAR actuation in the RobiSEND system also gives it the ability to measure environmental characteristics outside of its rest position in the horizontal plane.

Compared to other LIDAR sensors of its class, the Hokuyo URG-04LX-UG01 is the least precise, lightest, shortest range, and low cost option available. The assumed low speed of propagation of the MRUAV allows for this sensor to be used for mapping in spite of its limited precision and range. The cost, weight, and low power of the system are fitting for the constraints of the RobiSEND payload.

5.4.4 Servo and Tilt platform

Ideally, RobiSEND would be capable of translational motion in the vertical direction. Because the payload is tested in a hardware in the loop scenario, the ability to change the altitude of LIDAR scans is simulated by actuating the scanner through a servo and tilt platform. Although the angle between the servo and the horizontal scan plane slightly skews the range information, it is assumed adequate for the qualitative approach to mapping. As discussed in Chapter 3, because there is no direct correspondence between scans, the portion of the LIDAR scan that is considered for object models only consists of a partial scan. As long as the actuation angle remains relatively small, this rotation of the LIDAR system is an acceptable means simulate the change in height of the MRUAV.

Actuation of the LIDAR sensor allows for object model generation when the LIDAR data is needed. Many tilt platforms are available. One platform in particular manufactured by Hitec fits the needs of the system appropriately. This tilt platform with servo installed weighs approximately 0.2kg. It possesses flat surfaces that fit the form factor of both a stereo vision system to be held stationary and 125˚ of rotation for the LIDAR system. Tying the yaw of the MRUAV to this platform gives it a pan capability of the full 360˚. The mass of the rotating platform plus the weight of the LIDAR system *m* was measured to be 0.30kg with the majority of the mass near the end of the moment arm.

The servo selection for this tilt platform must be done with respect to the torque requirements. If servo arm of rotation is modeled as a point mass, the mass moment of inertia about the tilt axis may be conservatively estimated by Equation (23) as

$$I = mL^2 = (0.30\text{kg})(0.1143\text{m}^2) = 0.0039\text{kg-m}^2 \tag{23}$$

The servo is assumed to travel the entire range of motion of the system of and back in one second or 125˚ in 0.5s. In addition, the servo also must ramp up to top speed in 0.1s. Therefore, the angular velocity must be at least $\omega = 8.726\,{}^{\mathrm{rad}}\!/_{s}$ and the angular acceleration calculated by Equation (24) as

$$\alpha = \frac{\omega}{s_{acc}} = \frac{8.726\,{}^{\mathrm{rad}}\!/_{\mathrm{s}}}{0.1\mathrm{s}} = 87.26\,{}^{rad}\!/_{s^2} \tag{24}$$

As a result, the torque of the selected servo must be capable of 0.342N-m, as displayed in Equation (25)

$$\tau = I\alpha = \left(0.0039\mathrm{kg\text{-}m}^2\right)\left(87.26\,{}^{rad}\!/_{s^2}\right) = 0.342\mathrm{N\text{-}m} \tag{25}$$

A standard torque for typical servos is 3kgf-cm or approximately 0.294N-m. Therefore, a high torque servo must be selected for the RobiSEND application. The servo used in this research is the Hitec HS-645MG high torque servo (Hitec, 2011). This servo has an operating speed of 4ms/1˚ and maximum output torque of 0.755 N-m, giving it the capability to move the LIDAR sensor its full range of motion and back in one second.

5.4.5 Approach to Use of Hardware

The RobiSEND system uses a fixed forward stereo camera and actuated LIDAR to adjust the angle of the 2D scan plane for simulated MRUAV height adjustment. As information is received from the system's sensors and operator input, the LIDAR may be directed towards specific objects or points of interest within the environment that may be overlaid within the camera FOV. This movement is to simulate the actual vertical translation of the MRUAV. If the theories of this thesis were to be tested on an actual MRUAV system, there would be no need for the tilt platform, because the MRUAV would just use its ability to translate vertically. The concepts of RobiSEND would also be useful for a system without the ability for vertical translation. In this case, the servo and tilt would still be implemented to achieve information gathering with the LIDAR sensor outside of the unmanned system's horizontal plane.

Only about 90$^{\mathrm{o}}$ out of the 240$^{\mathrm{o}}$ horizontal scan plane of the LIDAR overlaps with the camera FOV. Because this is a small percentage of the LIDAR total points and because of the Operator Blending emphasis on the object recognition problem over the correspondence problem, the geometric distortion caused from angling the LIDAR upwards is not assumed to impact the distribution of points from the LIDAR system. Therefore, the object models that are generated and the placement of these observations within generated maps do not require precise measurements to the LIDAR, but rather, relative measurements used to establish the qualitative inferences. This is discussed further in Chapter 6 on the Operator Blending framework.

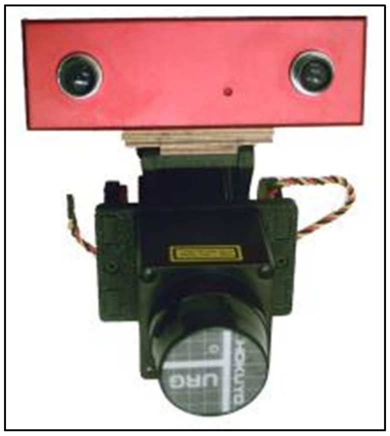


Figure 38: Hardware of RobiSEND system payload assembled

Figure 38 gives the overall structure of the RobiSEND payload that collects data at 10 Hz. Through IEEE 1394, the camera relays 640x480x3 raw color and 640x480 disparity images to a computer for processing. Although only one computer was used for processing both the LIDAR and the vision information, one computer could be dedicated to processing the mapper models and global map deliverable and the other could be used exclusively for vision processing. All information is combined on the decision level due to the Operator Blending framework of Chapter 6.

A Phidgets 1-motor controller pictured in Figure 39 is used to precisely set the angle of the LIDAR system (Phidgets, 2011).

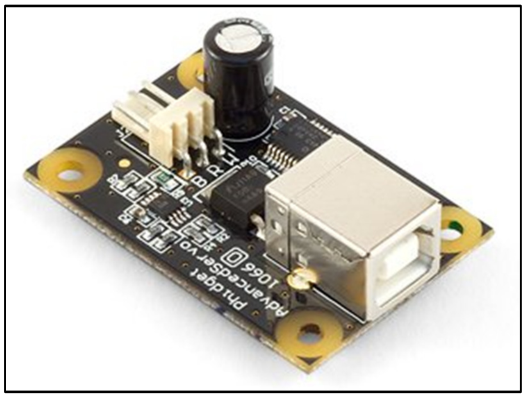


Figure 39: Phidgets 1-motor servo controller

When the operator is not interacting with the system, the LIDAR is fixed at 90˚ for horizontal scans. The theoretical RobiSEND MRUAV hovers and is in heading hold.

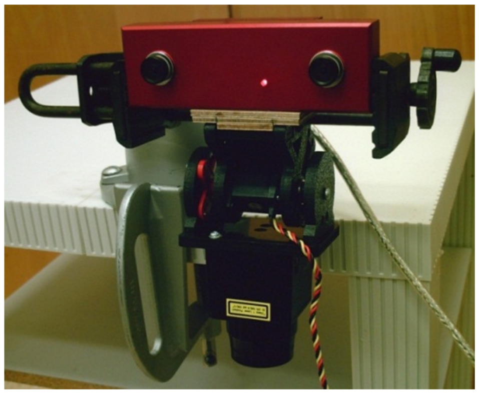


Figure 40: RobiSEND system attached to test cart

Once objects are selected, the position of RobiSEND acts according to their functional profile as generated through the Operator Blending framework. For certain qualified objects, this results in the translational motion of the theoretical MRUAV.

Both the LIDAR, stereo camera, and operator provide diverse sources of information to the system. The stereo vision system has no range measurement capability at distances under 1m. It is limited in its resolution at ranges over 15m as shown in Figure 37 and only returns data from high contrast information and areas of the image with appropriate lighting. Areas along the walls are particularly difficult to measure because of the low contrast from the lack of identifiable features. It captures the 3D characteristics of the scene and is able to give depth information for objects. The LIDAR system gives more precise measurements, particularly of the low contrast surfaces, however, only captures information in 2D planes, and is limited to 5.7m maximum distance.

As opposed to capturing 3D information with both sensors, the LIDAR is actuated for specific scan planes above and below the 90˚ position. The tilt degree of freedom introduced is to simulate the ability of the MRUAV to adjust its height based on its sensor measurements and not the natural pitching motion present in most real world MRUAV systems. If the RobiSEND payload were to be flown on board an actual MRUAV, the dynamic model of this MRUAV must be approximated, aided by a control system for MRUAV stabilization to minimize unwanted pitch and roll during navigation. These are identified as control challenges in this research and are outside of the scope of this work as outlined in Chapter 1. Therefore, the contribution of the tilt platform is representative of simulated height adjustment assuming an ideal model of a MRUAV search system. The MRUAV is assumed to be capable of a perfect hover capability under translational motion which includes no unwanted rotational or translational motion under user commands.

The operator directs the attention of the system. Therefore, when certain objects are selected from the video feed, the LIDAR tilts to capture points in its scan plane. As shown in Figure 41, this is particularly useful in understanding the boundary information of a room with clutter. The horizontal position of the MRUAV is at the same height as the desk and airplane. By actuating the LIDAR only 5˚, a much better representation of the size of the room and the room structure is attainable. It is beneficial to attain this additional information without relying on stereo vision data or having to adjust the height of the RobiSEND payload. As opposed to LIDAR actuation, the RobiSEND system implemented on an actual MRUAV would fly higher or lower depending on operator directed attention.

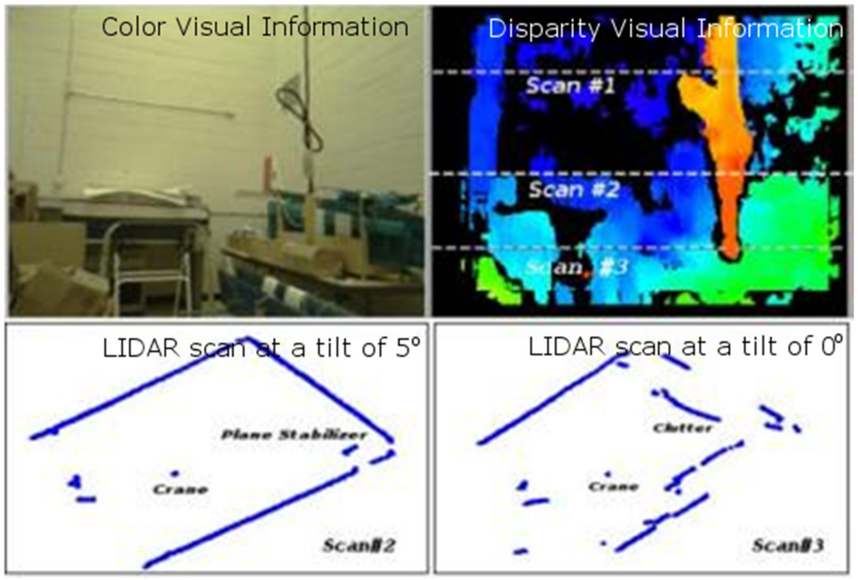


Figure 41: Contributions of Stereo Vision and LIDAR. Scan #3 is at the true height of the RobiSEND payload in hardware in the loop testing. Scan #2 is with the LIDAR tilted 5^{o}, which simulates the MRUAV flying slightly higher in the environment. Scan #3 is actuated at 10^{o}, which is an even higher simulated change in MRUAV altitude.

5.4.6 Operator Attention

Central to information processing from visual information is the concept of operator directed attention. The operator uses a mouse or touch screen to select image objects. This sets the boundary of a region of interest for feature extraction, object tracking, and object identification operations. Figure 42 displays an overlay of these regions upon object selection.

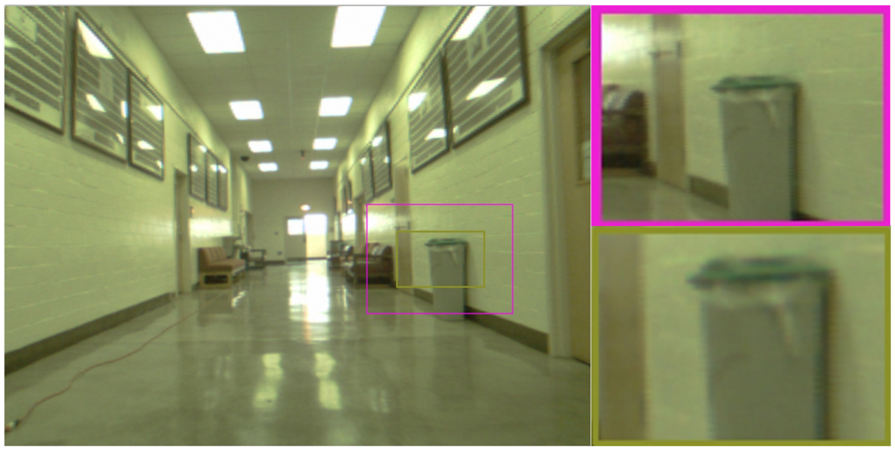


Figure 42: Image displaying boundary dimension during object selection

The region of interest is 160x120x3 and is referred to as the object background and extracted from the surrounding area of the selection location. At the center of the object background, a second boundary based segmentation is conducted to extract an 80x60x3 region referred to as the object template. An object background disparity image of size 160x120 and an object template disparity image are extracted from the disparity information. Local and global features are not found for the entire color or the entire disparity image. Instead, features are found from the object template image and disparity object template image and compared to the object background image and disparity background. The location of the object background image and disparity background image in subsequent video frames is found by computing how the template images have propagated within the background image. All of the image processing approaches are therefore localized to these regions as they move in the image.

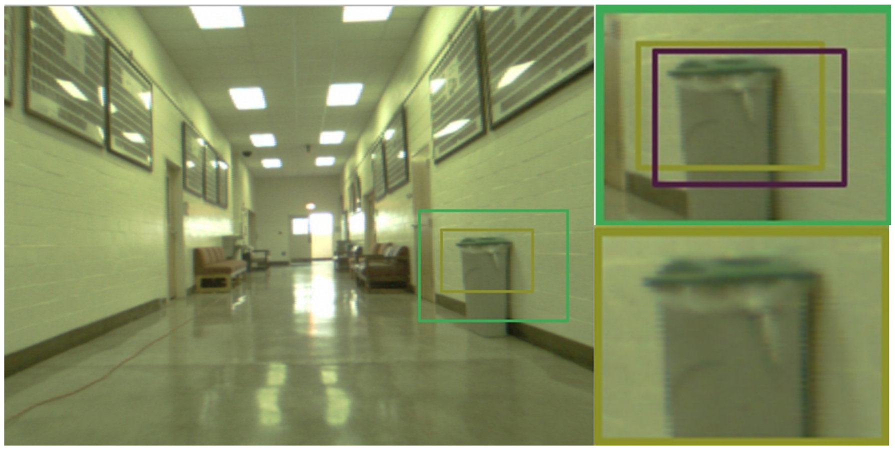


Figure 43: Object tracking over time. The green region is the initial boundary. The yellow region is where the template image is found

5.5 Object Tracking

Object tracking is the prerequisite for observation model building as discussed in the remainder of this chapter. The approach taken by the RobiSEND system for object tracking is central to feature extraction. In addition, the novel AIM-Shift method developed in this work is detailed in Section 5.5.

5.5.1 General Approach

As mentioned in the previous section, the operator introduces boundary based segmentations to contribute to object tracking. The human operator selects objects through the GUI direct video feed that direct the attention of the system towards what is important. The direction of system attention reduces the computational requirement because the boundary segmentation reduces processing to the region segmented from the full frame image.

In order to reduce the computational requirement for processing, the object background and disparity background images are stored from frame to frame. Therefore, as opposed to finding local and global features for the entire color or disparity image, features are found in 1/16th of the color image that is central to the tracked location of the object within the image FoV. The new location of the object background image from frame to frame is dependent on where the object template image is found within its bounds. Objects are tracked as long as the object background image remains in the camera FoV. The disparity image is aligned with the left image of the stereo camera; therefore, the disparity background image is known to shift an identical amount as the object background image.

In other words, background images are the search region for matches to template images. Although the entire color and entire disparity images could be used instead of the background images, it is known that the object background will contain feature matches under reasonably small translational and rotational motion. The tracking process follows the pseudo code of Figure 44

```
1. Full Color (640x480x3) and Full Disparity images captured by stereo camera
2. Operator makes observation initiating boundary based segmentation or Go to 1.
3. Sub images a-b extracted from Full Disparity and Full Color with center pixel at click location
    a. Object Template (80x60x3) from Full Intensity
    b. Disparity Template (80x60) from Full Disparity
4. Grayscale Object Template (80x60) stored for SIFT feature extraction
5. Features from Grayscale Object Template (80x60) & Disparity Template (80x60)
6. Until tracking interrupted, run subroutine a-g
    a. Full Color (640x480x3) and Full Disparity images captured by stereo camera
    b. AIM-SHIFT applied to correct for large translations and rotations between frames
        i. Color shift magnitude found
        ii. New object location found by adding color shift to previous object location
    c. Sub images i-ii from Full Disparity and Full Color with center pixel at object location
        i. Object Background (160x120x3) from Full Intensity
        ii. Disparity Background (160x120) from Full Disparity
    d. Grayscale Object Background (160x120) stored for SIFT feature extraction
    e. Features from Grascale Object Background (160x120) & Disparity Background (160x120)
    f. Matches found between Template features of 5 and Background features of 6e
        i. Feature compare: Object Template (160x120) to Object Background (80x60)
        ii. local shift magnitude found from average Euclidean Distance between matches
    g. New object location found by adding local shift magnitude to previous object location
7. Go to 1
```

Figure 44: Pseudo code for object tracking

The procedure for AIM-SHIFT is outlined in the following section. As the object is tracked, the object model is also built. The boundary based segmentation allows for pixels that do not exhibit similar characteristics to be grouped together based on their proximity to the operator selected center pixel. These pixel neighbors are analyzed in this research through local and global feature extraction methods.

5.5.2 Area Intersection Mean-Shift algorithm (AIM-SHIFT)

As mentioned in the previous section, the manner in which objects are tracked is based on a combination of local SIFT feature matching and the newly developed Area Intersection Mean-Shift (AIM-SHIFT) algorithm. AIM-SHIFT is related to classical mean-shift algorithms but differs in its feature extraction methods and objectives. The intent of AIM-SHIFT is to complement local feature based tracking methods for single channel images through low computational cost, color based enhancement. As a reminder, tracking is needed because the initial observation location moves as the MRUAV undergoes translational and rotational motion. Although turbulence is assumed to be a control theory problem and therefore not addressed in this research, the object tracking approach should be able to deal with small translational motion due to its influence as well as motion introduced by the user. AIM-Shift introduces an added degree of functionality because it leverages color information. Although the performance of AIM-Shift is not evaluated, it is presented as an example for the types of techniques possible under an Operator Blending framework.

Color is independent of motion blur and is useful even if the object of interest possesses a low number of local features. Local feature based tracking methods can be susceptible to reduced feature matches under large translational motion and large rotations. Motion blur may occur that distorts local feature information but the global color distribution of the image remains intact. In additions, algorithms may have difficulty finding feature matches under these circumstances but may be able to track objects that have definitive color characteristics. In addition, SIFT is traditionally a method that uses single channel images and so therefore does not use color information.

Traditional Mean Shift algorithms, as presented in Section 3.5, also may not converge to the target object when the candidate region estimate that results from Equation (14) is not similar enough to the matched region from the previous image. The algorithm, however, is capable of handling blur affine transformation, and occlusion in situations where the estimate of candidate region location is in the vicinity of the target object. Mean Shift also is an iterative process and therefore may require significant computational time. AIM-SHIFT provides a means to use the additional color information to improve tracking effectiveness at little computational cost.

Computational cost is reduced by AIM-SHIFT by taking advantage of the boundary based segmentation introduced by the system operator to extract Object Background images. The general concept involves the computation of intersecting sub-images within the Object Background image called Area images. Area images have the same resolution of Object Template images and their boundaries are placed strategically within the Object Background image in accordance with Figure 45.

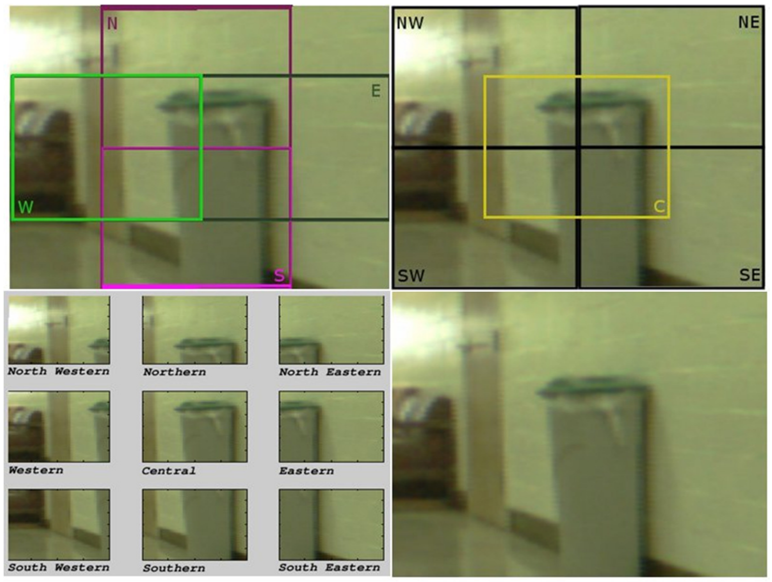


Figure 45: Area image extraction for the octal image structure

They are referred to collectively as the Octal image, and capture the north western, northern, north eastern, eastern, south eastern, southern, south western, and western areas. Color histograms are found for the Octal image and compared to the color histogram of the Object Template image. An octal image and the resulting color histograms are shown in Figure 46.

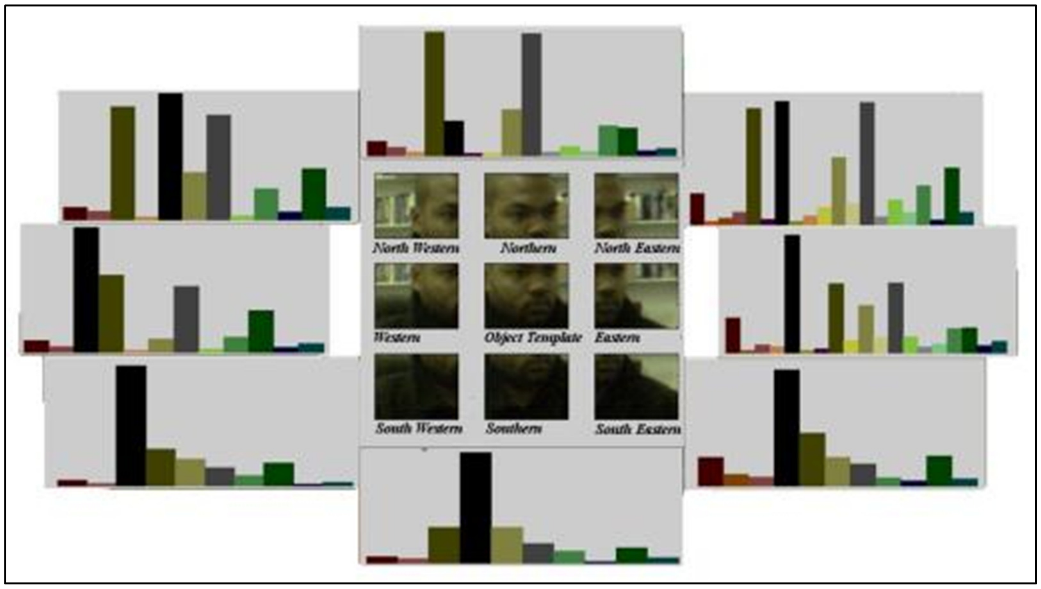


Figure 46: Octal Image representation with color histograms for each of the match images

As opposed to using mean shift to find the location of a single candidate region in the current image and iterating the result until convergence, AIM-SHIFT finds the color histograms of the Area Images surrounding the object region from the previous frame. If one of the Area images has a low enough distance measure found from Equation (16), the object is said to have undergone extreme translation.

The Area image identifies the general direction of shift towards R_{t+1} in the current frame. The magnitude of this shift is determined by the size of the distance measure. Three possible magnitudes of shift are possible depending on how high or low the distance measure is. Low match scores correspond to the low shift magnitude value. Therefore, there are 9 possible output directions and 3 output magnitudes. The magnitude and direction of shift are collectively referred to as the color shift in the tracking algorithm.

For AIM-SHIFT, several simplifications are used to reduce computation. First, because rotations and translations are small when compared between frames, potential candidate regions are reduced to those represented by the Area images. This is realistic if we consider the rotational motion of the

MRUAV to be reduced to 90^o per second or 9^o per frame. A 90^o rotation over a second under these conditions would correspond to a maximum image translation of less than 60 pixels. The sub image is 80 pixels in diameter. Therefore, it can be assumed that the observation be contained to the object background image.

The entire image is not processed in favor of the area immediately surrounding the operator initiated boundary segmentation and its resulting tracked locations. Secondly, as opposed to finding an initial estimate of candidate region location from Equation (14), the representation of the Area images is assumed to encapsulate one of 9 possibilities for the direction of object translation. Although these 9 choices may not result in the exact direction of translation, it provides a means to compensate for extreme translation in these general directions. Lastly, the magnitude of correction is not iterated over time. Rather, the distance measure score from Equation (16) is used to determine the color shift magnitude from frame to frame.

It is believed that the SIFT shift magnitude gives adequate initial estimates of the R_{t+1} location to ensure that AIM-SHIFT converges, although this has not been verified in this current research. Also, in cases of extreme translation or rotation where local feature based approaches fail, AIM-SHIFT provides an ability to leverage additional color based information. As mentioned in Section, AIM-SHIFT is capable of assisting object tracking under conditions where local feature based methods are susceptible to failure. After the color shift is applied if necessary, SIFT based object tracking is used in this research for fine adjustments in tracking the object region.

5.5.4 SIFT feature object tracking

As discussed in Chapter 3, local SIFT features are found that also contribute to the object model. These features are resistant to occlusion and affine transformations as well and have some additional resistance to scale changes over local histogram information. In addition, because these features are local in nature, they may be used to provide unique object signatures based on their distribution and descriptors. Although SIFT features are not the most computationally efficient features for this task, they provide resistance to scaling and rotation in three degrees of freedom and are densely distributed throughout the object background image. This allows for several feature correspondences to be made for object tracking and object recognition.

It is understood that SIFT has been expanded upon in many ways to optimize local feature performance. Some of these alternatives are presented in Chapter 3. Because almost all other approaches use SIFT as a baseline for comparison, it is the local feature detector used for this work. Since other approaches claim better performance than SIFT, it is assumed that the computational approaches given in this research demonstrates a baseline capability that may only be expanded upon if future researchers desire to use an alternative local feature detector. These opportunities for future work are outlined in Chapter 7.

The SIFT detector and descriptor are initially run on the object template upon operator selection. Key point locations and their descriptors are saved in the object structure. This is for object recognition and tracking. Matches between descriptors are found between frames for feature correspondence as discussed in Chapter 3. As the object background region of interest propagates, the local SIFT detector features are calculated for the object background image. The sets of SIFT features from the object template and the object background are compared through a distance method given in Chapter 3. The feature matching is done through finding the square of the Euclidian distance between matches. This technique returns a correspondence between features if their descriptor likeness score is within a threshold. Once matched, the *u* and *v* distance in pixels between matched features is returned. All of the feature components implemented from SIFT and MSER are extracted through the VLFEAT library of functions (Vedaldi & Fulkerson, 2008).

5.5.5 Object Tracking with SIFT features

After the color shift magnitude and direction is applied towards tracking, the SIFT shift magnitude and direction is determined based on local SIFT feature matches. As covered in Chapter 3, local features have the advantage of robustness under rotation and affine transformations and more drastic lighting and occlusion than color histogram based features.

After the color shift magnitude and direction is applied, SIFT features are found in each new object background image. These SIFT features are compared to the SIFT features of the object template images. For each match, a shift is found by computing their shift with respect to each image frame.

In order to compensate for outliers in the matching process that do not return an accurate pixel shift, the shift distances are not averaged for each feature. Instead, similar pixel shift distances are clustered together through k-means clustering. These shift values are grouped into 3 clusters. If two clusters are significantly close together, their centroid values are averaged to achieve the pixel shift. If they are not found to be significantly close together, the lowest centroid value as measured from the pevious observation model location is taken as the pixel shift.

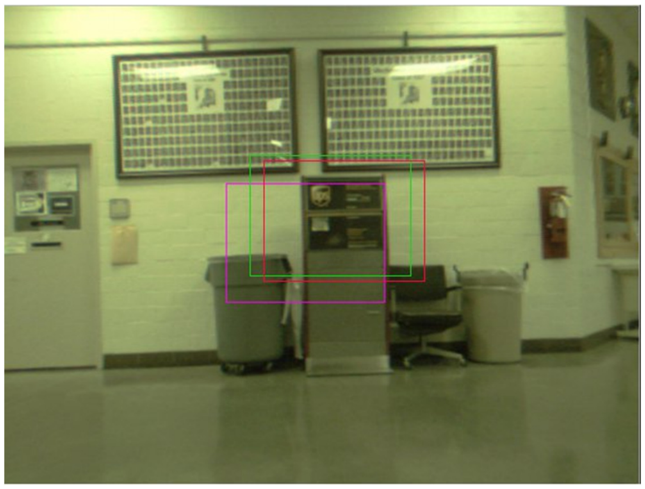


Figure 47: Image showing object tracking approach. The purple box shows the location of the object region before color shift or SIFT shift is applied. The green box shows the object region after color shift in the North Eastern direction. The red box shows the final object region location after SIFT shift

The total pixel shift of an object is known in control theory as the local motion vector and in this work is the color histogram shift added to the SIFT pixel shift. Each matched pixel should contain a similar pixel shift. The local motion vector for an object is always under estimated because background pixels are part of the boundary based segmentation. As discussed in Section 3.2.3 about stereo vision, pixel shifts at greater distances are smaller than pixel shifts in the foreground due to parallax. The color shift presented in Section 5.2.2 compensates for this under estimation. Therefore, when tracking the object from frame to frame, the color shift magnitude and direction is first applied to move the object background ROI. Then the SIFT shift is applied to provide the finer corrections. The combination of both shift methods to get the local motion vector allows the observation to be tracked as feature models are generated over time

5.6 Features for Object Model

Each type of feature extracted for object model generation is discussed in this section. The intensity based features from the left color image of the stereo camera are first discussed. The disparity based features from the disparity image are then presented. The concept of observation model improvement over building the model over time and space is then discussed. These methods are the basis for object models that contribute to the RobiSEND Operator Blending framework.

5.6.1 Classes of Features

Five object model features are found from a single object template image. These features are the SIFT descriptor, MSERs, shape, color histogram, and the object template image itself. As mentioned in Chapter 4, a single boundary based operator initiated segmentation may result in multiple object template images if the object tracking process has weak feature matching for the current object template image. When a new object template image is stored, it inherits its name and location information from its parent object template image. Local and global features are stored separately for both and are considered parallel components of the object model.

There are two features derived from the intensity and two features derived from the disparity. As shown in Table 6, a local and a global feature are found for both.

Table 7: Features for object models

	Local Feature	Global Feature
Disparity Information	Maximally Stable Extremal Regions	Shape Metric
Color Information	Scale Invariant Feature Transform	Color Histogram

Local features are point based features determined directly from the pixel values of the object template and disparity template images. Global features are region based features derived from the relationships of pixels to each other within their respective images. All pixels are inter-region relative to each other. At the same time, as discussed in Chapter 3, object template and disparity template sub-images are intra-region in respect to the object background and disparity background images. Feature change over time may be observed by measuring how the template images move within the larger background images

When an object is initially selected, its distance relative to the MRUAV is calculated from a combination of disparity information and MSERs and placed in the global map relative to other observations. MSERs are used to decide which disparity values are foreground and which are background from the disparity template image. This may be seen in Figure 50. If this results in a distance to MRUAV of the selected object, a shape metric is calculated. The object template image may be output for future object identification by the operator. Figure 48 shows examples of each type of feature stored from the initial boundary based segmentation. This is considered a “bag of words” approach to object characterization.

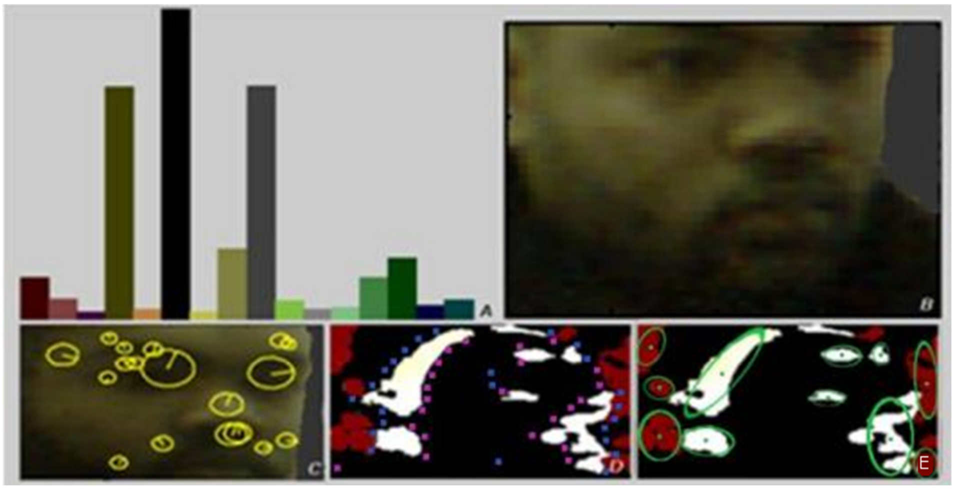


Figure 48: Example of local and global features. Figure 48(A) is the color histogram, Figure 48(B) is the object background image, Figure 48(C) shows the SIFT features overlaid on the object template image, Figure 48(D) shows the shape metric locations, and Figure 48(E) displays the MSER features

All features play a role in defining the object functional role outlined in the chapter on Operator Blending. This is in contrast to traditional approaches that use a “bag of words” technique and an object identification tree for comparison against a large database of potential object identities. The particular name, class, or appearance of the object is largely irrelevant outside of its functional significance to the RobiSEND search process. In other words, classes of objects are based on their functional roles as opposed to their actual class. Unless labeled differently by the operator, a telephone pole would be characterized in much the same way as a flag pole, because both have a similar role in the Operator Blending approach. This functional identity is the object genetic code as outlined in Chapter 6.

5.6.2 Intensity Based Features

After the initial boundary based segmentation initiated by the operator, the object template image must be tracked over time to build to object model. Object template based features found from the three channel color image are used for this purpose in addition to their role as components of the object model. For tracking, the object background image is the search region that propagates within the FOV of the camera. The object background position is updated at the camera frame rate. The features of this region are compared to the intensity based features stored from object template images. These features will be contained within the object background image as long as the object motion and the translational and rotational motion of the MRUAV are not large.

Color histogram information for the object template provides a strictly color based metric that is independent of object translation as long as the color characteristics of the selected object remain constant. Because the initial segmentation is a boundary based approach, the color histogram strays from traditional color error approaches. There is no requirement that pixels have any relationship to one another outside of the fact that they are within the object template boundary or sub-image windows.

5.6.3 Disparity Based Features MSER

Disparity based features are indirectly derived from the intensity information. The disparity is calculated from the red color plane of the left and right intensity images to derive object features with spatial significance. Because of this, the number of disparity pixels found through SAD is related to the quality and consistency of intensity information.

Disparity relationship to real world distance, however, is an opportunity to solve some traditional problems. Because the initial segmentation is a boundary based approach from the operator selection of the object, the disparity may provide an inter-region relationship between pixels. In addition, temporary nuisances may be dealt with more logically. If an object distance from the stereo camera is measurable at a distance of two meters, it does not matter if the majority of the object is occluded or the lighting changes—it will still be located two meters from the camera. Both the global shape metric and local MSER features are included in the object model because of this. In addition, target object relationship to occluding objects and background objects is important for the object model. Also because of their distance relationship to the MRUAV, these features are used to place object locations within the mapping of the environment.

Local MSER features are used to identify large and uninterrupted extremal regions in the disparity information. Figure 49 displays the pseudo code for the MSER search process. The extraction of MSER features is important because the center pixel or the average of the pixels in a disparity image may not reflect the disparity of the foreground object in the image. MSER features find continuous regions of disparity in the disparity image. The larger continuous regions of disparity are known to correspond to object information. This allows for segmentation to occur between foreground and background objects.

```
1. Full intensity (640x480x3) and disparity (640x480) images are captured by the stereo camera.
   Pose estimation done
2. Operator selects object in the camera FOV initiating a boundary based segmentation or Go to 1
3. Sub images are extracted from disparity and intensity images with center pixel at operator click
   location: object template (80x60x3), object background (160x120x3), disparity template
   (80x60), disparity background (160x120)
4. MSER feature locations found and ordered based on size
5. Feature regions are compared for similarity and grouped into foreground and background based
   on their disparity values.
6. Similar feature regions characterized as foreground are averaged for the object disparity
   distance
7. Go to 1
```

Figure 49: Pseudo code for MSER search process

The disparity template and disparity background images are searched for MSER features. MSER features are ordered based on the number of disparity pixels that make up the feature. Larger blobs reflect objects with a distinct and highly confident distance measure. Within the boundary based segmentation provided by the operator, definitive disparity also has significance to the object function for operator blending as discussed in Chapter 6.

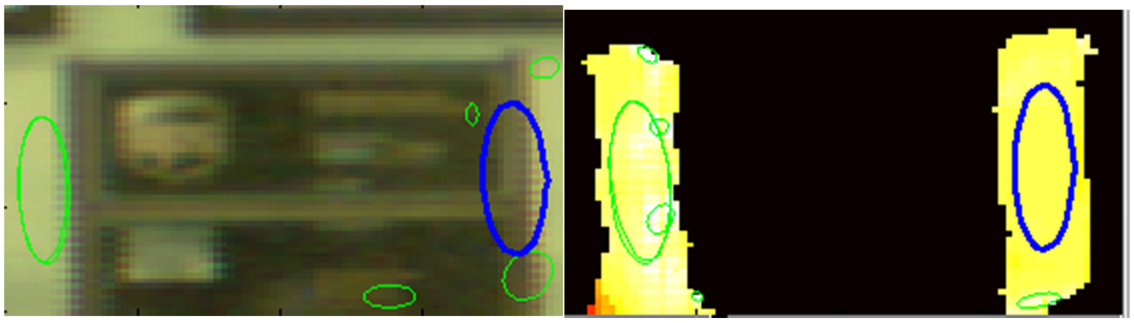


Figure 50: Object template image and disparity image with MSER features overlaid

All of the disparity values that belong to a specific feature are averaged for the object distance to the stereo camera. Once an object is selected, the average distance of the largest disparity blobs are returned as the object distance from the stereo camera. Therefore, even in the presence of significant clutter, the predominant distance measure is returned. This is a much more effective technique than returning a specifically indexed disparity value or an average of all disparity values in the disparity template.

Once these features are found, they are characterized as members of either the foreground, background, or indeterminate based on their value and relationship to other MSER features. Objects found to be members of the foreground are most impactful for operator blending and are central to the RobiSEND navigation process as discussed in Chapter 4.

5.6.4 Disparity Based Features Shape

The global shape metric is adapted from feature vector comparison. It provides a semantic segmentation to the disparity template and disparity background images. For the information provided by the Videre stereo camera, positive and negative shapes are searched for. Positive regions are regions where lower values of disparity are object background, higher values of disparity are object foreground, and the outside edges of the object foreground provide a clear segmentation from background information.

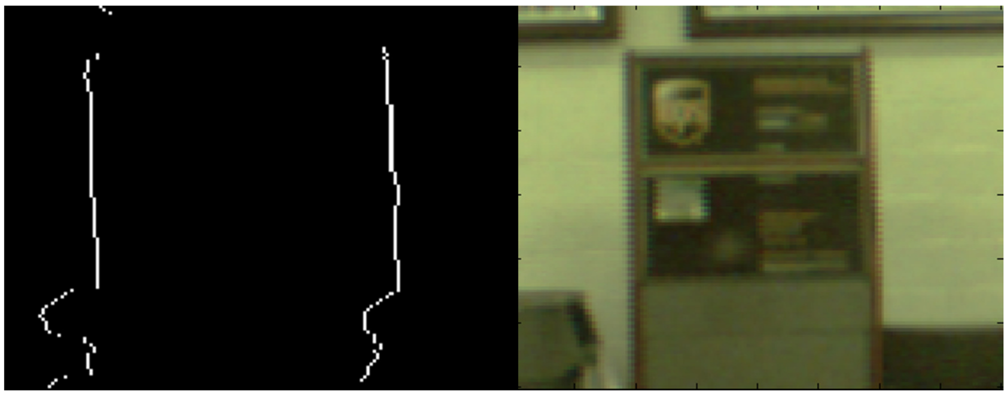


Figure 51: Positive shape feature vectors from foreground objects

Objects that are clearly part of the foreground with high contrast features are good positive shapes. Examples of these shapes include poles, trees, or the crane displayed in Figure 41.

Similarly to positive regions, negative regions include high disparity values for foreground and low disparity values for background. The difference is that for negative regions, clear shapes are defined from the inside out of the object. Negative shape is used in the same manner as positive shape for Operator Blending characterization of the functional relevance of objects. A negative shape example is given in Figure 52.

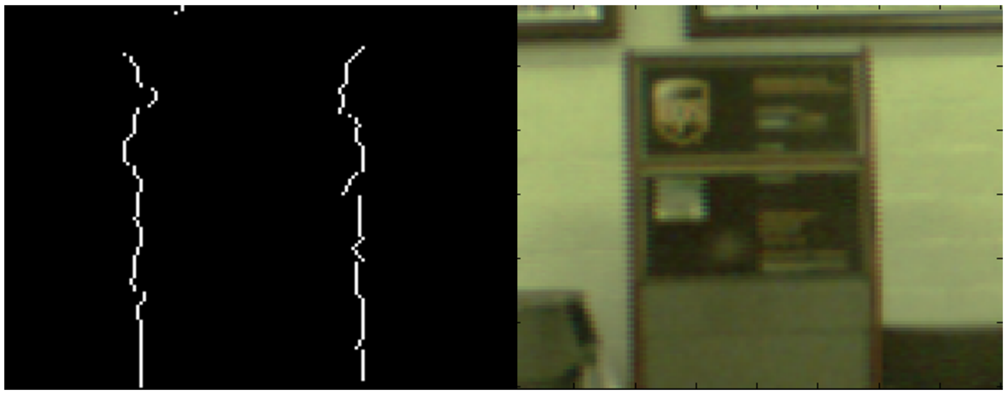


Figure 52: Negative shape feature vectors from foreground object

This is particularly useful when observing objects that are part of the background, have clearly defined features along edges, and low contrast towards the object interior. Doors and windows are primary examples of objects that would be well defined negative shapes.

After an object is identified, a positive and negative shape metric is generated. For the positive shape metric, a positive change in disparity is searched for by propagating along rows starting at the left and right edges of the disparity background image. The negative shape metric is found by starting at the middle of the disparity background image and propagating towards the left and the right for a positive change in disparity.

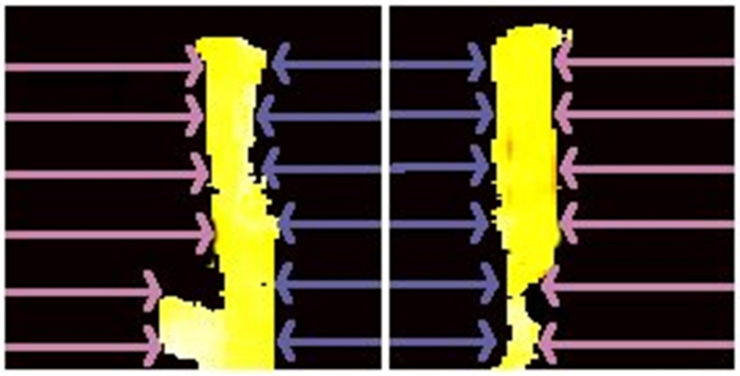


Figure 53: Shape metric extraction process

Both the negative and positive shape metrics are 2x120 feature vectors that correspond to the number of rows of propagation before the set change in disparity is experienced. Therefore, a total of 480 bytes of data are needed for this feature method. This adds a hybrid of the thresholding segmentation and region based segmentation methods to the object model.

This metric is used as a measurement of consistency in establishing a segmentation from background information. Like the global color histogram features, the shape features are resistant to translation and movement within the local object region of influence.

5.6.5 Feature Maturity over Time

Object model components are stored consistent with the Operator Blending framework presented in Chapter 6. This methodology allows for the object model to mature over time as the object is tracked. Chapter 4 covers the merits of object maturity, time, and movement to the classification process. Because of this, the object models are able to become more complex if successfully tracked in the camera Field of View.

Once feature correspondence drops below 5 for an object template image during the matching process, a new object template image is taken from the center pixels of the object background image and a full object model is extracted for it. Therefore, object models may have multiple object template images and features. At the conclusion of this process, object features are stored with a unique label and are the sum of all models derived from their object template images. The object model is the basic building block for functional relationships of Operator Blending as discussed in Chapter 6.

5.7 Mapping Considerations

Section 5.6 gives the components for object models used in RobiSEND's Operator Blending implementation. Section 5.7 presents mapper model components. The general procedure for developing mapper models is presented along with the process for combining these models into a qualitative map.

5.7.1 Mapping Overview

A qualitative approach to mapping with metric and topological components is developed for the RobiSEND system. Complete 3D reconstruction of a search area is not the most computationally efficient or effective method to communicate the structure of an unknown environment from local search. Most biological systems capture visual cues and create a semantically relevant map tied to propagation within the environment. This work assumes that the same types of maps may be generated by human-robot collaborative teams in local search. This novel approach is QUICK mapping. This name was chosen because the approach is akin to the natural intelligent means in which biological systems perform mapping and the quick manner computations for this approach are executed.

Local information in this approach is used exclusively during real time mapping. Information is extracted from the raw LIDAR data as scans are considered over time. Mapper models are the building blocks for this approach, triggered through observed conditions within the data. This process is outlined in the following section.

Global considerations are made to update the accuracy of the map based on what this work deems passive and active observation interpretation. Passive Observation Interpretation (POI) occurs during the search process and corresponds to recognition of individual mapper models or combinations of mapper models. The rules that govern POI are given in Chapter 6. Active Observation Interpretation (AOI) is map correction based on the placement of object models. It is the optimization stage of the mapping process that occurs at the conclusion of local search. Operator observations with matching features during the search process are merged together.

Object models are then placed with topological annotation within the final map considering when observed, their corrected locations, and individual model parameters. The annotated and corrected map deliverable communicates the qualitative structure of the environment and an emphasis on topological annotation if operator blending occurs by user participation in the mapping process. Even if the robot is interacted with in Safe mode, the qualitative map reflects the navigation process.

The local search mission is to produce deliverables that are effective tools for team planning and execution. Even though the final map may not be a precise metrical representation of the unknown environment, it is an accurate depiction of the human-robot team's experience. The metric structure adequate for autonomous path planning is not enough. The approach must give meaning to measurement that may only be captured qualitatively. Qualitative maps do this well, although there are not many actual systems outside of simulation that take advantage of their characteristics.

Furthermore, the qualitative approach reflects the perception and integration necessities of the RobiSEND system as given in Chapters 1-3. Because the navigation process is the shared basis for human-robot collaboration and the operator actively introduces topological information in an Operator Blending framework, the mapping process must reflect the meaning that these methods introduce. In addition, RobiSEND reduces computational requirements by processing local information and emphasizing the object recognition problem over feature correspondence for metric reconstruction. A qualitative approach is a necessity, because of its ability to communicate the significance of spatial information considering minimized computational load.

As a result, RobiSEND is the first system to the author's knowledge that uses a purely qualitative method for mapping with actual hardware in real time. Other systems consider qualitative information only as a means to correct quantitative maps. A qualitative map is possible because of operator involvement in perception processes. The approach to QUICK mapping is briefly described here alongside the individual mapper model construction procedures. Mapper model observation model usage is explained in the mapper model section of Chapter 6 on Operator Blending. Both object models and mapper models are needed for the full construction of qualitative maps.

5.7.2 Mapping Procedure

The mapping process follows the procedures displayed in Figure 54 and Figure 55. As a reminder for LIDAR point plot generation, the horizontal axis is always the forward facing direction of the MRUAV. The vertical axis, therefore, is the lateral motion of the MRUAV. Each scan, therefore, depicts LIDAR data from a top down view considering how the sensor is mounted for RobiSEND.

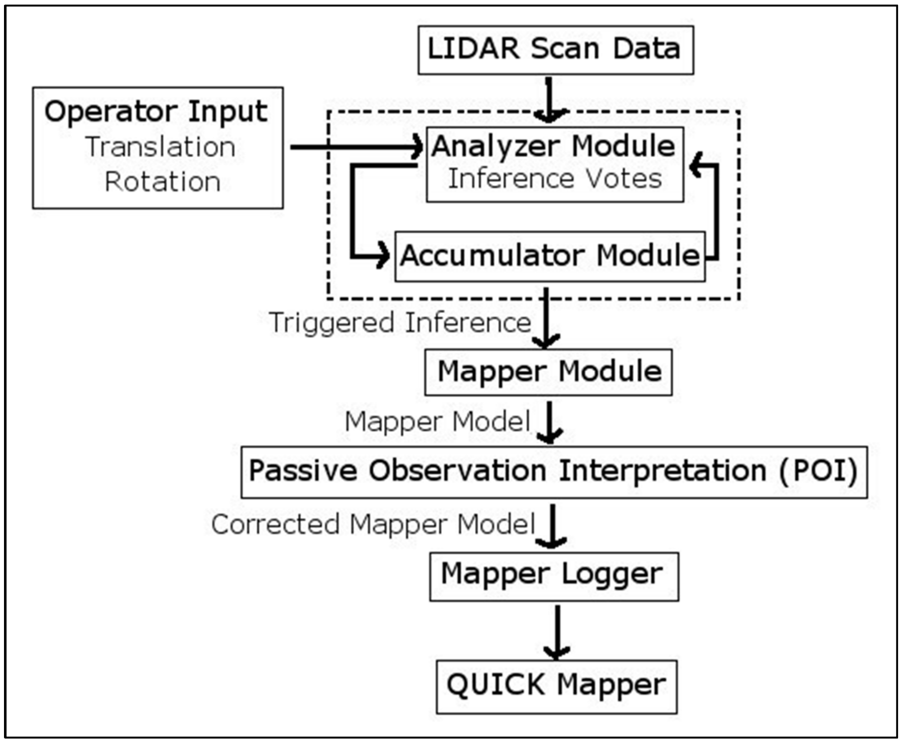


Figure 54: The flow of the mapping process during search

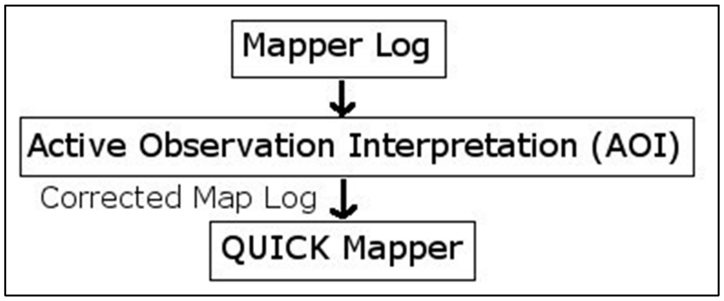


Figure 55: Mapping process at the conclusion of search

As shown, the RobiSEND system first is given its translational commands, moves to a new location, and captures a LIDAR scan. The scan analyzer module then uses principal component analysis and k-means clustering techniques to determine the scan inference information. This stage converts raw information into qualitative observations. Inference information over time is input into the accumulator module that outputs the mapper model from the place vocabulary. The mapper module is then triggered if certain qualitative conditions accumulate over time.

Although more diverse place vocabularies are possible, RobiSEND features five general place vocabulary classes: angles, hallways, junctions, content space, and open space. These classes are multiple mapper model formulations possible from each place vocabulary class. A hallway may be long or short. A junction may be full or partial. Space may be large or small. These characteristics are determined by the accumulator module and discussed in more detail in later sections.

As each mapper model is triggered and the operator makes observations impacting the local search and navigation process, the qualitative map of the environment is built with POI. Object locations are ultimately plotted on the qualitative map with respect to their local relationship to the RobiSEND payload.

The map cursor is initially placed at the center of the developed map with global x position signified by the value of *x_pos* and global y position by *y_pos* as shown in Figure 56. As information is plotted by the qualitative mapper, the resulting plot is a global deliverable. Therefore, the horizontal axis is the initial direction of the MRUAV local search system within the global environment. All qualitative representations are then plotted relative to that initial direction. This is in contrast to the local information of the object and mapper models. The horizontal axis of all LIDAR scans, for example, is always the direction of propogation of the MRUAV. Additionally, all visual information is of the information directly in front of the MRUAV along the direction of propogation. For qualitative maps, mapper models and object models, which are both based on local information, are built over time into a global deliverable. Therefore, the path the MRUAV takes is reflected in the path indicated by the qualitative mapper referenced to the initial MRUAV angle.

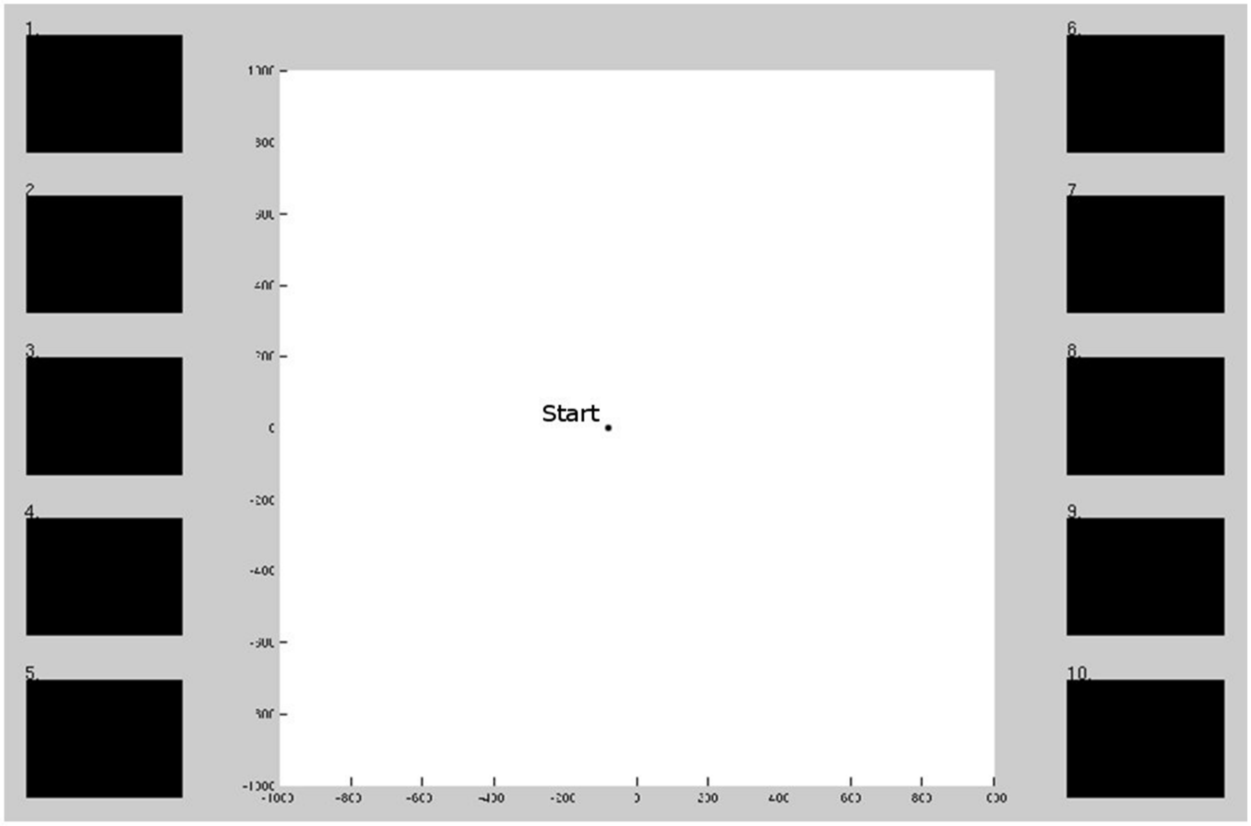


Figure 56: Initial conditions for the qualitative mapper with start point labeled for demonstration purposes. The horizontal axis is the initial direction of the MRUAV when the local search process begins.

Global heading *head_pos* is also stored and updated as user commands are issued. Place vocabulary information is stored along with object models for AOI map optimization at the conclusion of the search process. Although the map optimization is useful for ensuring that loop closure occurs and object model locations match each other upon object recognition, this is not a necessity. POI may be sufficient in portraying the environment effectively. The operator is given the option of relying on this representation.

5.8 Qualitative Features of Maps

Each module established in Section 5.7 is discussed in detail with their respective role in the QUICK mapping process. Sub-section 1 concentrates on the analyzer module and discusses the technical details for feature extraction from a LIDAR scan. The accumulator and mapper module are explained in sub-section 2. The visual components used for the created mapper in MATLAB are presented here. These details are necessary in order to understand how a LIDAR scan is processed to fit within the Operator Blending framework of Chapter 6.

5.8.1 Analyzer Module

The first phase of qualitative mapping approaches is modeling inferences from quantitative data. This is done through the analyzer module of the RobiSEND system. Two types of qualitative features are extracted from individual LIDAR scans. The first is based on the covariance of LIDAR kmeans clustering of similar groupings of data points. The second is based on the Principle Component Analysis of the same kmeans clustering based groupings.

Initially, kmeans clustering groups LIDAR data points based on their vicinity spatially to one another. The process inputs a LIDAR scan and outputs these points as members of up to 20 groupings determined by the squared Euclidean distance grouped points share between one another. Although 20 groupings are possible, based on the number of clusters set as a maximum in software, the actual number of groupings per each scan varies on the trend of the data. Once groupings are finalized, their average Euclidean distance within their cluster is found and a centroid location determined that signifies the center of the grouping data. The covariance between grouping members is also calculated. This is shown in Figure 57.

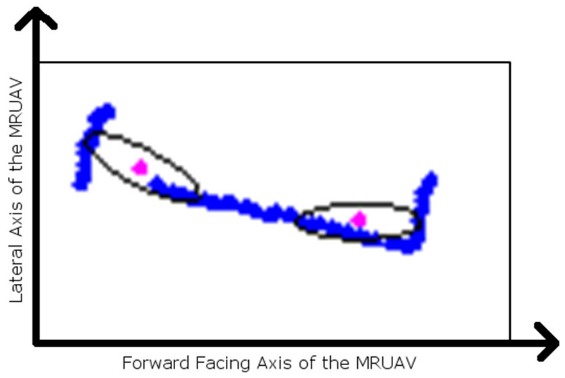


Figure 57: Covariance values displayed with centroid locations in magenta of clusters of points from a LIDAR scan segmented by kmeans clustering. The blue data points are LIDAR data. The magenta points are the centroids of the kmeans clusters. Ovals are representations of the variance of the cluster.

Centroid locations of cluster groupings are shown in magenta with the covariance modeled as a black oval around the centroid point for each data cluster.

By definition, the covariance of each grouping consists of the variance of the data along the local x direction COV_x, the variance of the information along the local y direction COV_y, and the correlation between the x and y data CORR.

$$\begin{bmatrix} \mathrm{COV}_x & \mathrm{CORR} \\ \mathrm{CORR} & \mathrm{COV}_y \end{bmatrix} \tag{20}$$

The covariance consists of 2D information and is a measure of LIDAR point distribution in the horizontal and vertical directions. Points that have a high COV$_x$ and low COV$_y$ are considered high in structure along the x direction because they constitute a grouping along a locally horizontal flat surface. At the same time, those points with low COV$_x$ and high COV$_y$ are members of a grouping with high structure along the local y direction.

The parameter used to judge the structure of local scans is called the F parameter. The reason for this naming convention is explained further in Chapter 6. It is defined by Equation (26)

$$\left(F_{param}\right)_i = \frac{COV_{xi}}{COV_{yi}} \tag{26}$$

Where *i* corresponds to the *i*th grouping of the current LIDAR scan data points. When F_{param} is high, the grouping of LIDAR points is considered to have a high structure in the x direction. Likewise, if F_{param} is much less than 1, then the grouping of points has a high structure in the y direction. An F_{param} from 1 to 2 orders of magnitude signifies a lack of general structural trend in either the x or y direction. If RobiSEND is surrounded by walls, kmean clusters will reflect high amounts of structure. Likewise, if an abundance of curved or cluttered surfaces are present, the structure parameter is reduced.

The second parameter used by the analyzer module is the U parameter U_{param} defined as the amount of linearity of a kmeans cluster grouping. This parameter observes the principal components of the same identified groupings used to find the F_{param} of a scan. The eigenvalues of a kmeans cluster grouping are found that estimate the parameters of variation along the primary and secondary axis of the data point distribution independent of member point orientation. Two Principle Components are found for each grouping of LIDAR points. The larger principle component corresponds to data point variation along the major axis, while the smaller principle component gives the variance along the minor axis.

Because the principle components are independent of cluster point grouping local orientation, they are used to make a determination of wide or narrow distributions of points. A large major axis and small minor axis principle component corresponds to a linear distribution. Conversely, a similar major axis and minor axis principle component defines a grouping of points with no general linear trend. Equation (27) defines the U_{param} values of a single scan kmeans cluster grouping of points.

$$\left(U_{param}\right)_i = \frac{(EIG_{max})_i}{(EIG_{min})_i} \tag{27}$$

Where EIG_{max} is the Principle Component along the major axis of the grouping data and EIG_{min} is the Principle Component along the minor axis of the grouping data within the individual scan. Therefore, points along a diagonal wall have high linearity and U_{param} value because they would have a high EIG_{max} and low EIG_{min}. Points with no linear trend would have a low U_{param} value closer to 1.

As a result, the kmeans clusters of each scan, which can possess, but do not have to possess, as many as 20 groupings, all have characteristic F_{param} and U_{param} values. If F_{param} of the scan grouping is higher than a threshold, it is said to 'vote' towards defining the scan as high in structure. If the U_{param} is higher than a threshold, it contributes linearity to the scan, otherwise it votes for non-linearity. Each grouping also has a 'vote' in determining the structure of the overall scan. If the F_{param} value is higher than a threshold, it votes that the overall scan is structured. To get the total amount of linearity or structure of an individual scan, the number of cluster groupings that vote for a parameter are divided by the total number of grouping kmeans clusters in the scan.

Therefore, the analyzer module defines each LIDAR scan by whether it is high/low in structure or high/low in linearity. Once this determination is made, this information is sent to the accumulator module that makes a qualitative determination of the structure or linearity of LIDAR information over time. The accumulator sums up the evidence and once specific conditions are met, it decides which place vocabulary accurately represents the data.

5.8.2 Accumulator Module and Mapper Module

The accumulator module takes input from the analysis of individual scans and triggers one of the qualitative mappers defined in Section 5.7.2 from the place vocabulary. Therefore, the accumulator is responsible for the mapper module building process over time. Simply put, the accumulator module makes qualitative judgments from the output of the analyzer module. Over time, it determines which place vocabulary is triggered and what characteristics the resulting mapper module should have taking into account the sum total of qualitative judgments, user input during this time period, and the translational motion of the RobiSEND system. These inferences do not reflect individual scans but rather the conditions encountered by the MRUAV during the local search process.

The accumulator does not just take into account LIDAR data for this judgment. It involves interpretation of what commands were issued to the theoretical MRUAV in Shared mode or Safe mode, the distance traveled by the MRUAV, what other observations are made, or whether the area of the map is recognized as being visited previously. All of these factors contribute to the mapper model COG as discussed in Chapter 6.

Once a threshold of qualitative observations is made, the accumulator module triggers the mapper module which stores all relevant parameters and may output a graphical depiction of the mapper model component. A depiction of what each mapper model component is shown in Figure 58.

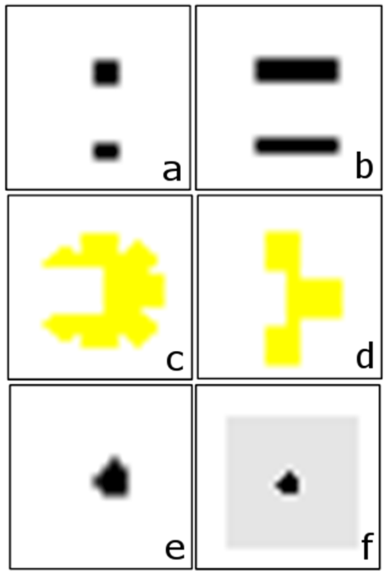


Figure 58: Qualitative mapper model representations. 58(a) shows the short hallway representation. 58(b) shows thelong hallway representation, Figure 58(c) displays a full junction, 58(d) shows a partial junction, 58(e) gives the indicator for content space, 58(f) shows the open space indicator

The accumulator then resets and gathers more evidence from the analyzer module for the next mapper module output. As more and more mapper models are built and parameters stored, the qualitative map is built based on the guidelines of QUICK mapping as discussed in Chapter 6. The subsequent sections outline the components of the place vocabularies in more detail and define the qualitative parameters that the accumulator module uses to define their characteristics. All components of QUICK mapping are implemented and created in MATLAB 2010b in order to create a qualitative mapper for the Operator Blending framework.

5.9 Place Vocabularies

Qualitative Spatial Reasoning is used to establish the place vocabularies for the mapper module in this section. The inference conditions that trigger each place vocabulary are presented with examples of each. In addition, the mathematica framework used for the qualitative mapping used in this dissertation is presented.

5.9.1 Angle place vocabulary

AngleMap place vocabulary is the means to express rotational commands given by the operator. This is the simplest place vocabulary and is used in consort with the other three place vocabularies to update heading information. It is the only mapper model that is not triggered from Fparam and Uparam votes from the analyzer module or graphically depicted on the qualitative map. It is triggered when the MRUAV is sent a rotational command from the operator in Safe mode that results in a measurable change in SIFT feature point matches between frames. It may be used in combination with the other place vocabularies to reflect a heading change.

As discussed in Chapter 4, when presenting the RobiSEND user interface, the operator may locally control the heading of the MRUAV in 45^o intervals up to 90^o each direction. This value was chosen because of the field of view of the camera being slightly over 90^o. The MRUAV itself is assumed to have sensorimotor control that makes small angular adjustments to center it along viewed corridors or alleyways. Although this is an assumption that relies on technologies that may be currently unavailable or inefficiently implemented, engineers have slated its development firmly in control systems theory. As stated in Chapter 1, control challenges are outside the scope of this work. Therefore, in an effort to isolate perception and integration challenges, an assumption that may be viewed as an oversimplification in control systems theory is necessary in order to address previously ignored challenges within the scope of this work. The operator controlled angles are not the means for heading control but rather exploration that couples perception and integration processes facilitated by Operator Blending.

When angular rotation is initiated, it is assumed to result in small angle or large angle rotation. Small angle rotation is defined as follows. If an ObjectBackground Image is taken from the middle of the previous frame before rotation and compared to an ObjectBackground image taken at the conclusion of rotation, there is significant similarity between the two frames. This similarity is measured through qualitatively comparing the number of SIFT features in each of the aforementioned ObjectBackground images. After the command for rotation is issued, proper angular rotation is measured by tracking SIFT features in the image. As mentioned in Chapter 3, the feature match comparison is done by finding a match between SIFT points by measuring the Squared Euclidian Distance between their descriptors. If after the 45^o rotation, the ratio of SIFT matches before and after is below 10%, the actual angular rotation is inferred to be closer to 90^o than 45^o. In addition, if this ratio is above 90%, it is inferred that no significant angular rotation takes place. This is the root of the five classes of angular rotation. Figure 59 displays the intervals set for the Angle place vocabulary.

Near45Left $= \{\theta | \theta \in R, \mathrm{SIFT}_t/\mathrm{SIFT}_{t+1} \in (0,0.1)\}$ and rotation command issued Left

Near45Right $= \{x | x \in R, \mathrm{SIFT}_t/\mathrm{SIFT}_{t+1} \in (0,0.1)\}$ and rotation command issued Left

Forward $= \{x | x \in R, \mathrm{SIFT}_t/\mathrm{SIFT}_{t+1} \in (0.9,1)\}$ and rotation command issued Left or Right

Near90Left $= \{x | x \in R, \mathrm{SIFT}_t/\mathrm{SIFT}_{t+1} \in (0.1,0.9)\}$ and rotation command issued Right

Near90Right $= \{x | x \in R, \mathrm{SIFT}_t/\mathrm{SIFT}_{t+1} \in (0.1,0.9)\}$ and rotation command issued Right

Figure 59: Angle place vocabulary intervals

Where SIFT_t and SIFT_{t+1} are the number of SIFT matches before and after rotation, respectively and theta is the angle rotation of inference. The short hand for these inferences is as follows '45L'=Near45Left, '45R'=Near45Right, '90L'=Near90Left, '90R'=Near90Right, and 'FW'=Forward. As a result, the MRUAV may only display 8 total heading values on the qualitative map that are intervals of 45^o.

AngleMap is also the only place vocabulary that does not have a permanent visible effect on the map. The only change is the heading of the map cursor by the inferred difference in heading. Table 7 shows how AngleMap place vocabularies are stored in the map history table. The current x and y global coordinates, initial global heading, and final global heading values are stored in addition to any object model information identified before or after rotation.

Table 8: Place vocabulary storage table of parameters needed for mappers

Mapper model #	Heading (degrees)	Global x position (m)	Global y position (m)	Place width	Place type	Selection time (s)
1	0	0.34	0	W	FW	3
2	0	0.82	0	N	FW	5
3	45	1.1	0	W	45L	7
4	45	2.6	1.5	W	FW	23
5	45	2.8	1.7	N	FW	25
6	45	2.9	1.8	N	FW	27
7	90	3.0	1.9	N	45L	50
8	90	3.0	6.8	W	FW	139
9	180	4.2	6.8	N	90L	155
10	180	5.0	7.2	N	FW	170
11	180	5.2	7.2	N	FW	175
12	90	5.8	7.2	N	90R	203
13	90	5.8	7.5	W	FW	210
14	90	5.9	7.8	N	FW	219

5.9.2 Hallway place vocabulary

HallwayMap is the place vocabulary within the mapper that specifically plots hallways, corridors, or narrow passageways. When inferences are made that correspond to movement over time in spaces with elongated flat walls, the HallwayMap place vocabulary is invoked that plots this corridor at the angle specified by the current heading value and relative direction of motion. It is triggered when the accumulator module observes inferences that, over time, display a trend towards high structure and linearity.

There are 3 classes of the HallwayMap place vocabulary. These are based on qualitative inferences relative to the distance *x_const*=6m. As RobiSEND moves from object model to object model as specified in Chapter 4, its local displacements are tracked over time. If it is inferred that RobiSEND is in a long and structured space resembling a hallway, forward displacements accumulate, being added together in the variable *x_hallway* as long as the robot follows the same local heading. Figure 60 gives an example of the type of data that results in a Hallway place vocabulary inference

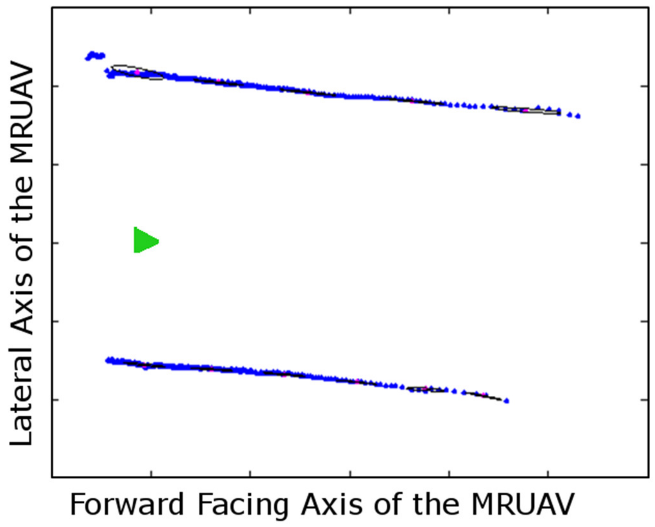


Figure 60: Example data that results in a triggered Hallway place vocabulary

Once *x_hallway* reaches intervals to trigger the HallwayMap mapper or the operator invokes this manually through the operator blending framework, it is plotted on the global map and its place vocabulary parameters are stored in the mapping table. There are 3 classes of the HallwayMap place vocabulary. These values include GreaterThanX when the ratio of x_hallway and x_const is within the interval 3/2 to infinity, EqualX when x_const is around the same 6m value as x_hallway with the ratio in the interval form 2/3 to 3/2, and LessThanX when x_hallway divided by x_const is less than 2/3. Figure 61 gives the formal definition of these qualitative methods

$$\boxed{\begin{gathered}\text{LessThanX} = \{x|x \in R, x\_hallway/x\_const \in (0, 2/3)\} \\ \text{EqualX} = \{x|x \in R, x\_hallway/x\_const \in (2/3\,, 3/2)\} \\ \text{GreaterThanX} = \{x|x \in R, x\_hallway/x\_const \in (3/2\,, \infty)\}\end{gathered}}$$

Figure 61: Hallway place vocabulary intervals

Where x is the inferred distance value. The shorthand for these qualitative classes is ‘g’=GreaterThanX, ‘l’=LessThanX, and ‘eq’=EqualX.

In addition to these classes, HallwayMap place vocabularies may also inherit or calculate width from sensor data. Width is a property shared by the JunctionMap and SpaceMap place vocabularies as well to express how wide or thin a plotted place vocabulary is inferred to be. Inferred map component width may either be Wide, designated by a ‘w’, normal designated by a ‘n’, or thin, given by a ‘t’. If LIDAR data is found to indicate a wide hallway, its corresponding place vocabulary will be plotted as wide in the global map. Therefore, there are 9 possible iterations of the HallwayMap place vocabulary.

5.9.3 Junction place vocabulary

Junctions are used to communicate when there are several route options at a given point in space. They are triggered when the accumulator observes high amounts of structure but high amounts of linearity in the x direction. Figure 62 gives an example of a sensor scan where a junction is triggered.

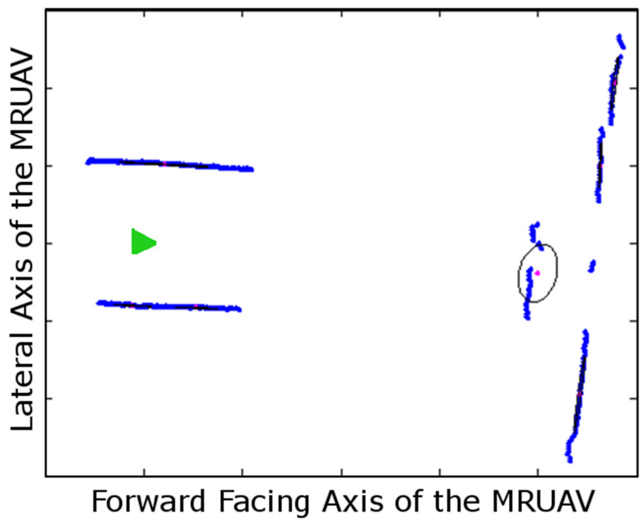


Figure 62: Image where a junction place vocabulary is triggered

Two different junction types exist: major junctions and minor junctions. Minor junctions are plotted by the MinorJunction mapper and are represented by the symbol in Figure 58. As shown, MajorJunction plots a place vocabulary that has 8 possible paths emerging from it that correspond to the possible angle values for paths from a major junction. Minor junctions only have four paths as shown in Figure. This is because, these are junctions that represent choices observed in passing over a short span of time. An example of a situation that triggers a MinorJunction is shown in Figure 63. On a QUICK map, MinorJunction is used to display options for exploration that are seen quickly, when RobiSEND is moving. In essence, each MajorJunction begins as a MinorJunction.

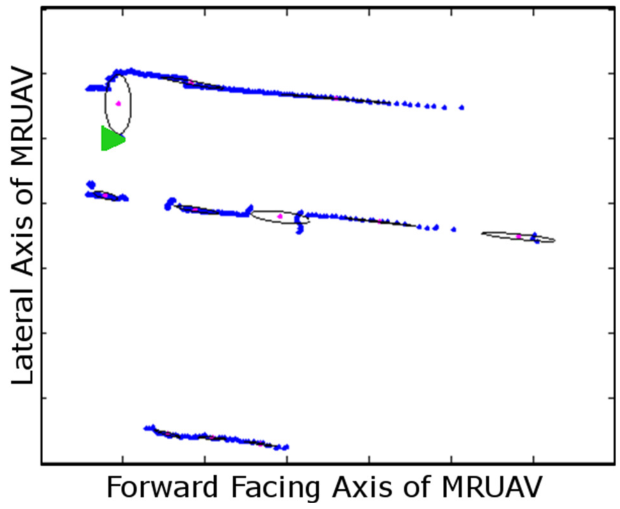


Figure 63: example situation that triggers a Minor Junction

Junction type is inferred from the amount of time observing a particular junction inference condition and whether or not an angular rotation is input during this time. A value of *t_const*=2s is selected that sets the threshold of time spent observing a MinorJunction before it becomes a MajorJunction. At the same time, if the robot stops at a location where a junction condition may be observed over an extended amount of time, the MajorJunction mapper is triggered.

Corner junctions are a special case of MajorJunctions. This junction occurs when a MajorJunction is detected and it is followed by two 90^0 rotations. The output is the raw sensor data at the moment of rotation printed in front of the robot path. This is to show that potential moving along the depicted path was an option and the basic structure of the environment for future reference. Basic corner junction structure is displayed in Figure 64

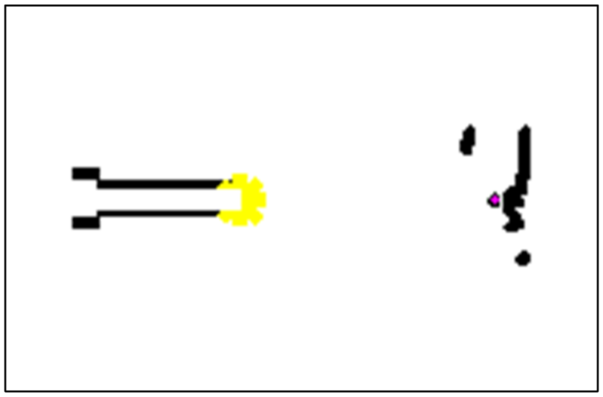


Figure 64: Example of a Corner Junction triggered

The formal definition of all inferences is as follows:

MinorJunction $= \{t | t \in R, t/t\_const \in (0,1)\}$

MajorJunction $= \{t | t \in R, t/t\_const \in (1, \infty)\}$

CornerJunction $= \{t | t \in R, t/t\_const \in (1, 2)\}$*

*180^0 Change in heading or no forward path

Figure 65: Different Junction inferences

Much like Hallway place vocabulary and mappers, junction may also have width. They always inherit width from proceeding objects. As a result, there are 9 possible junctions dependent on the mapper that is chosen and the width value assigned from previous information.

5.9.4 Space place vocabulary

Space place vocabulary is used to map areas that do not trigger the Hallway or Junction place vocabularies. There are two general categories of space that are treated as separate mapper models: open space and content space. Both are differentiated from hallways and junctions from their low amount of linearity. Content space, however, has a moderate amount of structure, whereas, open space has considerably less. In other words, Open space is a sensed area with sparse sensor measurements. Content space has a moderate to large amount of sensor measurements result in structured data amongst kmeans clusters. Figure 66 shows the difference between the different classes of Space.

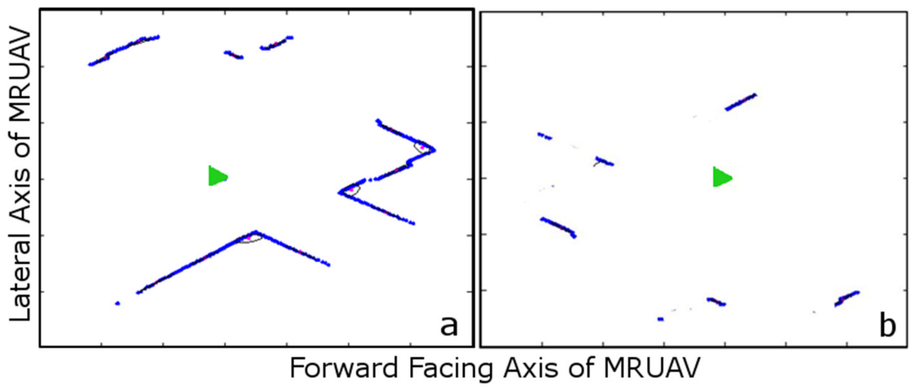


Figure 66: Image of the different types of Space and what triggers each. 66(a) gives a situation that triggers content space. 66(b) gives an example situation that triggers open space.

Both open space and content space mapper model classes are determined by two parameters: where the robot is perceived to exit the space locally and the size of the space. ContentSpace has the added ability to integrate raw sensor information into the qualitative map. Once ContentSpace is determined and output from the mapper, the LIDAR scan taken upon entry into the space is plotted on the map. Each time content space is re-entered after this initial determination, a new LIDAR scan is taken and also shown. Therefore, a functional difference between ContentSpace and OpenSpace is this functionality and the manner in which they are depicted in the final map. In addition, OpenSpace is depicted in larger overall regions.

The width of Space mapper model is determined by the amount of displacement during inference accumulation. The actual path of the robot inside of space is not plotted but the heading and relative position of the robot is stored as the values x_space, y_space and heading_space. Space is divided into 8 sub-areas that signify the 8 exit points of space after propagation within the space during the accumulator. The sub region divisions of the area of interest are similar to the manner in which the object background images are divided for the color histogram based tracking of Section 5.2.2. The region in Space in which the robot is inferred to reside is compared to constant displacement values x_constant=4m and y_constant=8m. The ratio of x_space to x_constant determines the region in the local x direction with the same being true for y_space and y_constant. This is defined by Figure 67.

$$\text{North} = \{x, y | x, y \in R^2, x/x_{const} \in (-1/4, 1/4), y/y_{const} \in (3/4, \infty)\}$$
$$\text{NorthWest} = \{x, y | x, y \in R^2, x/x_{const} \in (3/4, \infty), y/y_{const} \in (3/4, \infty)\}$$
$$\text{West} = \{x, y | x, y \in R^2, x/x_{const} \in (3/4, \infty), y/y_{const} \in (1/4, 3/4)\}$$
$$\text{SouthWest} = \{x, y | x, y \in R^2, x/x_{const} \in (3/4, \infty), y/y_{const} \in (0, 1/4)\}$$
$$\text{South} = \{x, y | x, y \in R^2, x/x_{const} \in (-1/4, 1/4), y/y_{const} \in (-1/4, 1/4)\}$$
$$\text{SouthEast} = \{x, y | x, y \in R^2, x/x_{const} \in (-3/4, -\infty), y/y_{const} \in (0, 1/4)\}$$
$$\text{East} = \{x, y | x, y \in R^2, x/x_{const} \in (-3/4, -\infty), y/y_{const} \in (1/4, 3/4)\}$$
$$\text{NorthEast} = \{x, y | x, y \in R^2, x/x_{const} \in (-3/4, -\infty), y/y_{const} \in (3/4, \infty)\}$$

Figure 67: Types of inferences of space based on exit location

5.10 Chapter 5 Summary and Triggering the Mapper

Chapter 5 presents the individual building blocks for object and mapper models. They all fit within the qualitative mapping framework and Operator Blending. How the different observational models are combined and used to make a qualitative depiction of the environment is discussed. The Chapter introduces Operator Blending in detail and covers more specifically how it is used to set the characteristics of a mapper upon selection and uses object models to introduce topological information. More detail is given on how, once one of the four place vocabulary categories is selected, how the inference measurements over time set the characteristics of the mapper.

The method of selecting the appropriate mapper for a given circumstance is done through operator blending and discussed in Chapter 6. Operator Blending ensures that the place vocabulary selection process involves input from the robot and the human and is done in a manner that solves perception and integration challenges. The map is also passively and actively corrected based on the contributions of individual object models. This specific form of qualitative mapping through operator blending is called QUICK mapping and discussed further in the following chapter.

6. Operator Blending

"We are like violins. We can be used for doorstops, or we can make music." **—Barbara Sher**

Most effective team relationships are shaped by an individual member's sense of purpose and the ability to communicate a plan to collaborators. Team members are then able to contribute what each does well to accomplish mutual objectives. It is easy to grasp this concept in human only teams because it is central to the everyday process of solving problems. A team leader motivates teammates because of a clear plan and effectiveness in conveying who is affected, why the task is important, and how it is accomplished.

In human-robot teams, it is possible for the human to provide understanding, context, and maintain clear, manageable relationships with robotic teammates that bring to the team unique capabilities. The key, as in any team relationship, is the establishment of motivation and effective communication—a responsibility that realistically can only be executed by the human. The most difficult first step may be for human teammates to reevaluate their role and how to effectively lend meaning to a robotic counterpart. This involves abandoning more traditional, passive, and ineffective purposes for those that lead to the extraction of the team potential.

6.1 Overview

Human-robot team relationships are re-emphasized with particular emphasis on what has been done in this research. Sub-section 1 establishes the approach to human-robot teaming seen with Operator Blending. Sub-section 2 re-emphasizes the role of the human in this process and how it is the source of novel contributions in this research. Lastly, as Operator Blending is formally established, sub-section 3 recalls the contributions already discussed in previous chapters.

6.1.1 Human-robot teams

Human-robot collaborative systems in this research undergo a recasting of purpose. As discussed in Chapter 1, Collaboration and cooperation are prerequisites for mission types where perception and integration must be shared. Local tactical missions conducted by a MRUAV are examples of one of the simplest cases. Both human and robot are localized to a similar environment and have clear strengths and weaknesses that make individual achievement of the mission undesirable or even impossible. They are reliant on each other to *effectively* accomplish group tasks.

The human's strength is in his ability to provide contextual information, ethical judgment, and the direction of attention of team resources. In a small tactical team, weaknesses are the high cost of human capital to invest in dangerous situations, a lack of maneuverability during search, and the limited capability of communicating the structure of environments in high stress situations. Robots may be used to provide the human with the capability to search his immediate surroundings much more effectively than what is currently done with a human only team.

The MRUAV's strengths, on the other hand, are its low cost to team resources, maneuverability, and ability to structure observations. It is limited by its inability to understand complex environments well enough to be trusted and produce useful reconstructions of the environment. If given the proper sensors, approach, and direction, the robot may produce team deliverables that would be impossible to attain through a robot only approach.

The amount of interaction between the two types of agents ideally would lend to the strengths of one reinforcing the weaknesses of the other. This takes trust, functional relationships, and a level of common understanding. In these circumstances, prosperous team synergy would be automatic. But, like any partnership, achieving these goals takes time. The amount of time and the nature of these relationships can be influenced by the approach.

Because both unmanned system and human operator provide unique skills and abilities to this task and neither may function independently towards completion, traditional concepts related to human-robot collaboration must be re-evaluated. Currently, human-robot collaborative systems are either human or robot centered. Operator Blending introduces a new branch of this research that requires operator and robot to be invested in making joint contributions to a shared task. They share the load and provide what each does well to accomplish the mission. Decisions are made on high levels of abstraction but require the full attention of the human operator. The result is codependence and cooperation within the framework of a shared dependent relationship that maximizes mission effectiveness. These concepts are presented in the following Chapter.

To the author's knowledge, this is a novel approach to human-robot collaboration. Operator blending contains the following stages:

1. Operator observation to direct system attention and introduce topological information
2. Object tracking and/or building of object or mapper models
3. MRUAV navigation, and determination of observation model functional relationships
4. Integration of object and/or mapper models within the global map

Each stage may be carried out independently because their combination is on the task level, although all together are necessary for complete functionality of the system as Operator Blended. Chapter 5 introduces stage 1 and 2 as it discusses the operator boundary segmentation process and the components of object and mapper models. Chapter 6 covers the definition of functional relationships with emphasis on stage 3 and 4. How all of these stages fit together is then presented at the conclusion of this chapter.

6.1.2 Operator Blending from the human perspective

Operator blending, much like Human Systems Integration, observes how robots and humans might interact to improve mission effectiveness. HSI incorporates abstraction at the highest level, considering systems that coexist meaningfully with humans as separate entities or team members. Operator blending encompasses the systems level integration of HSI, but does so with an assumption of cooperation and codependence at lower levels of abstraction. For the robot, this relationship influences how data is interpreted, the method used to solve problems, and the types of tasks the system is capable of performing. For the RobiSEND system, this impacts MRUAV perception and integration—the operator provides the robot with the means for intelligent information gathering capabilities.

Operator Blending thus results in the robot and operator being codependent as opposed to coexistent. On one hand, codependence improves perception and integration through directing human attention towards team goals and fostering a level of trust and cohesion between both the human and robotic system. On the other hand, codependence eliminates the possibility of trusting the system to act independently and drastically reduces the ability for the human operator to perform a supervisory role with the system. Thus, the multiplier effect of multi-robot to one human is not realizable through this approach.

However, there are those missions where a human operator in the loop is assumed to be an essential component of the system. In these missions, the operator must not only have an operational understanding of his environment and the robotic system cohabitating it but also a rigorous knowledge of their relationship to each other and to him. This requires a level of codependence because, as discussed in Chapter 2, success is not possible through the passive inclusion of the operator within the system or limited, untrustworthy robotic capabilities, both symptoms of human independent automation. Local search in unstructured, tactical situations is identified in this research as the basic mission type that benefits from codependence and cooperation. This is because a user in the loop is already assumed to be a requirement for safety reasons and the operator's relationship to the search system directly influences system deliverables.

In terms of cooperation, operator blending considers the human as a component in the MRUAV sensory system model, but not in the traditional sense of human-robot collaboration. These methods take a robot centered approach that the MRUAV is only concerned with input from the operator when it reaches a sensory conflict or the human centered approach which says communication is only necessary when the human needs more information. In an Operator Blending framework, the communicational requirement between the robot and the operator is constant and open for the duration of the mission. It is also reduced to the space of mutually beneficial actions or objectives. For RobiSEND, this is the navigation and exploration process, core mechanisms for local search, that build the functional relationship of observations to the environment.

6.1.3 Contributions of stage1 and stage 2

Observation models are the building blocks of operator blending. As mentioned in Chapter 5, these observations have spatial significance but are *not* metric reconstructions. They are represented by a “bag-of-words” approach that includes topological information and features initially represented relative to the RobiSEND coordinate frame. The model serves as a means for a functional definition of the observation. The manner in which these features are extracted is tied to Operator Blending.

For object models, operator blending initially comes into play by introducing boundary based segmentation to user specified regions. These segmentations result in particular sensory observations that are important to the local search process. These features are resistant to lighting, partial occlusion, and affine transformations. Object model recognition and tracking must be conducted to understand the identity and behavior of the object over time. As they are tracked, their functional relationships are established. Their characteristics may be used to improve the usefulness of the map and develop rules for human and operator movement in the environment.

The object model role in the overall local search mission is to improve map accuracy and to drive the navigation process for the unknown environment. They are first tracked over time, then their models are built, and finally their functional relationships extracted. The manner in which object models are stored, incorporated in the local search process, and utilized for mapping is dependent on the nature of these characteristics. Object models improve map precision by being a part of the map optimization stage and improve accuracy by annotating the map to reflect the human-operator team search experience.

Mapper models have a slightly different role. As mention in Chapter 5, these models are the building blocks for the qualitative mapping approach. Operator blending defines the functional role of LIDAR information over time, defining what each model contributes to the global mapping process. It differs from an object model because, as opposed to producing local deliverables to place within the map, mapper models are combined to produce the deliverable which is the global structure of the actual map. Therefore, it is a necessity to present both object models and mapper models in this research. The combination of global representations of an unknown area and local representations must be realized for qualitative mapping based on object recognition from local observations.

6.2 User Interface

The User Interface presented for the RobiSEND system differs from traditional approaches to user interfaces because its purpose is to foster codependent and cooperative contributions from the system operator. Section 6.2 presents the RobiSEND GUI which is based on operator observations through a direct video feed as the only means of translational motion for the MRUAV. As a result, the map building process, observation model building process, and navigation process are all coupled to the local search process. The process of observation model selection is first given. This is followed by what these observation models mean to the object recognition and navigation processes.

6.2.1 Object model Observation Selection

The intensity information from the three channel color images from the left camera of the binocular stereo vision system are used to find local SIFT features and global color histogram information. These two features are leveraged for object tracking, the main prerequisite for object model building over time. The following sections describe the tracking approach used for the RobiSEND system along with the method for feature correspondence and recognition. These features are derived in real time without the utilization of training data.

Two types of observation models built by the RobiSEND system rely on tracking to build relationships over time. Both are initiated when the operator makes selections through a GUI. Unlike traditional GUIs that present the operator with exocentric information and may rely on global data such as GPS, this GUI strictly operates off of local information given to the operator in an endocentric format. The GUI is also video centric, as video relayed from the left camera of the stereo system, is the primary means in which the operator contributes to the control of the system. This interaction results in observation models used for the operator blending framework. Although not presented to the operator through the GUI, mapper models build the global map with LIDAR information and qualitative mapping components that make up the RobiSEND place vocabulary.

The GUI created in this work is to demonstrate GUI concepts that would be useful for an Operator Blended system. The example continually used in this work is that of a MRUAV conducting a local search mission. Although the MRUAV is developed with the concepts in the literature presented in Chapter 2, the interface itself has not undergone a Human Factors based expert user evaluation. The area of Human Factors fits under the umbrella of Human-Systems Integration but is not contained within the field of Human-Robot Collaboration. As such, it is not within the scope of this work as explained in Chapter 1. The work presented in this dissertation does not study how to optimize the medium of communication through the GUI. Rather, the methods of communication are studied here. The GUI shown presents the ideas behind the novel field of Operator Blending by giving an example that takes advantage of its contributions.

6.2.2 GUI for object model formulation

The manner in which heading and translation are controlled is a major component of this type of GUI. Translation and rotation must be done relative to observations made in the environment. For the RobiSEND GUI, these observations are the object models. Heading change must be repeatable and relayed to the operator relative to the FOV of the camera which contains the generated object models. Translation is also done relative to the spatial placement of modeled observations. Two similar video-centric panels were developed for heading control and object interaction with the MRUAV. Object interaction is most directly related to the translational motion of the MRUAV through the Operator Blending framework. Rotational motion is most directly related to the heading control panel.

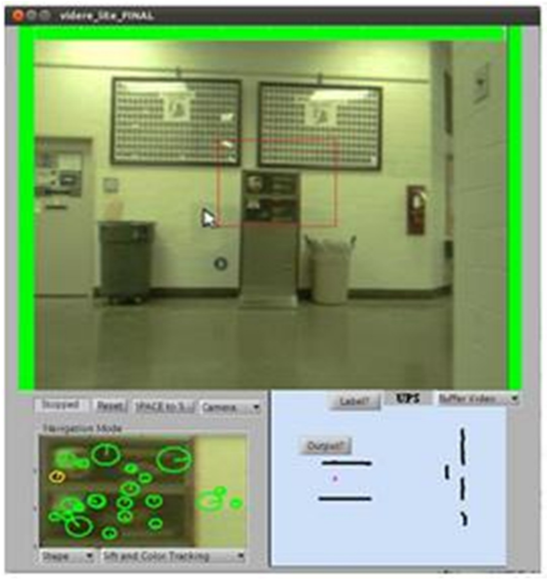


Figure 68: Figure displaying GUI of the RobiSEND system after object selection

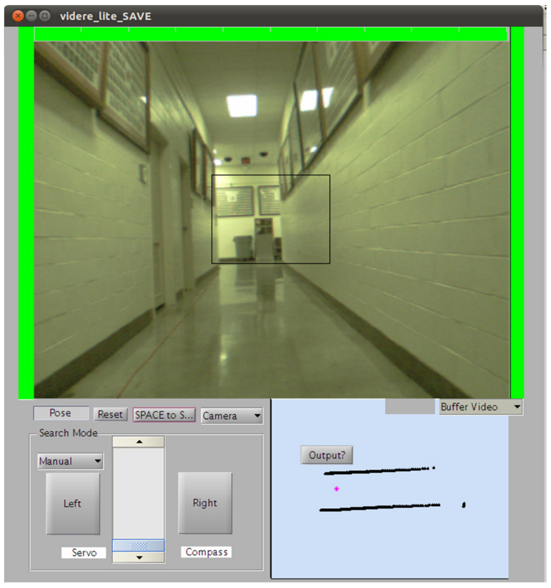


Figure 69: Image of RobiSEND GUI during Safe mode with no object selected

Figure 68 and Figure 69 show the operator interface for the RobiSEND system. When the operator is controlling the heading of the MRUAV, he is presented with Figure 68, which is activated upon mouse click outside of the video feed. When interacting with the video feed, he controls the MRUAV via Figure 69, activated via mouse click inside the video feed.

The operator is presented with three options on the heading control panel. He may rotate the FOV of the MRUAV left and right in 45^{o} increments or move forward. No altitude control exists because the MRUAV adjusts this altitude autonomously based on the functional relationship of observed objects. In addition, because the FOV of the camera is greater than 90^{o}, 45^{o} increments for heading adjustments reduces the possible headings to 8 while still allowing the operator to have control over MRUAV yaw in 360^{o} and not losing an object upon a single rotation. If the operator commands the MRUAV to move forward, it moves until it attains a 2m distance from the tracked object model. When an object has been selected before the heading control panel is activated, the operator also has the option of labeling the active object. The object label is done with keyboard input and stored in the object model.

The video feed panel replaces the options of heading control with a view of the object template image overlaid with SIFT features. When the operator interacts with the video feed, the MRUAV in in heading hold and its translational motion is over 3 degrees of freedom set through object functional relationships. The operator has two options: make an observation denoted as normal or make an observation that is designated important. Normal observations are by left clicking a point of interest in the FOV of the camera, while important observations are by right clicking. When interacting with this panel, the operator also may give the object a specific label through keyboard input.

As the MRUAV and operator team explore the environment, they both contribute to building the map. As mentioned previously, both agents accomplish what they do best under the Operator Blending framework. The operator makes observations by using a mouse to interact with the video feed. Once an object is identified, it is tracked as long as it stays in the visual field of view of the stereo camera. This tracking is for the building of the object model over time. The operator also has the option of taking more of a Safe mode approach to MRUAV control to change the heading of the MRUAV. Translational motion in this mode is also done relative to observation models in the environment.

6.2.3 Object Recognition

During object model tracking, a boundary segmentation is done by left clicking or right clicking in the video feed provided from the left camera of the stereo system. As discussed in the previous section, the initial supply of contextual information is supplied because the operator has the option of left clicking or right clicking based on observed importance. This influences the object functional role and the manner in which the object model is built. He also has the ability to give the object a specific label through keyboard input.

During tracking, the robot builds the observation model as outlined in Chapter 4. SIFT features and color histogram features, in addition to being components of object model, are also used for observation tracking and recognition. All observations designated by a right click or labeled by the user are characterized as important. The object model is tracked as long as it remains in the field of view of the camera, giving the operator time to add or change information. A green box the size of the object background designates a normal object and a red box the object background for important objects. Objects may be made important during tracking simply by supplying them with an object name.

Observations are saved up to 3 additional times as alternate views. During tracking, if the object has a correspondence of fewer than 10 but more than 5 SIFT features or if the color histogram score drops below 200 but is above 100, features are built again as alternate views. The initial object model is also saved for correspondence and the object model name is duplicated. In the event the number of object matches drops below 2, the initial object model template is loaded as the default template.

In addition to knowing how the object propagates within the scene, the robot must know whether an object has been visited before and its functional classification. When a new object is selected, its template features are first compared to the database of SIFT features and operator supplied object labels for a match. Even in situations where the operator chooses not to participate in this, objects are stored relative to other objects. Therefore, even if a match is not possible, object model labels are used to generate object recognition.

6.2.4 GUI role for mapper model creation

The GUI is an important aspect of mapper model creation as well because it controls the map update stage. As discussed in Chapter 4, a major challenge for qualitative mapping systems for autonomous system is the ability to control the update stage intelligently. Because the RobiSEND system operates in Shared mode or Safe mode in a human-robot collaborative environment, the GUI may be the source of this information.

For RobiSEND, any operator given command results in the output of a mapper model from the place vocabulary. The RobiSEND GUI gives the option of making observations within the field of view of the camera or issuing manual rotation and translation commands. When any one of these actions occurs, the mapper takes all information that influences the current place vocabulary into account and outputs a depiction on the global map. The issued command is also stored on the mapper list so that the optimized map may be output after mapping is complete.

In addition, observations previously made by the operator may be recognized through model feature correspondence. This influences the map update stage. Objects selected and labeled as similar will have high priority in the map optimization process as discussed in Chapter 5.

As a result, the GUI directs the attention of the system for the mapper models as well as the object models, but in a different respect. Object models are identified spatially and tracked over time for model building. Mapper models also accumulate characteristics over time. Inferences made from the qualitative mapping approach are accumulated. The output from the mapper may be triggered by the GUI. Thus, for object models, the operator directs the attention towards what is important. For mapper models, the operator directs attention towards when an update should occur and how long the mapper should accumulate inferences.

6.2.5 RobiSEND Navigation

The navigation process through local observations is the common ground between the MRUAV and human operator. This leverages the spatial relationships between important objects and boundary information as viewed through the sensors of the MRUAV and eyes of the human. Because mapping is the overarching goal of both systems, they contribute to the completion of this task by contributing what each does well. Although other systems in the literature may map environments more accurately in a spatial sense, this system differs in its ability to map and annotate a map in real time and its incorporation of human decision making in collaboration with the decision making of the MRUAV towards maps of more complex environments.

The LIDAR generated map is used as the global reference frame and is built in real time through mapper models. The object model building process identifies objects within the environment and then places them relative to the global map. Object models are influenced by the operator search process. Depending on the observation characteristics, they play a key role in mapping and navigation.

RobiSEND navigation is perception based and allows the MRUAV to be integrated through the control of a single operator. Translational motion decision making is derived from the search process and the resulting functional role of observations in the environment. As mentioned previously, these functional roles are established by the MRUAV and operator uniquely contributing aspects of object and mapper models.

6.3 Operator Blending

Once observations are made, the RobiSEND framework is implemented to deduce observation functional relevance. This relevance is a key aspect of RobiSEND Operator Blending—the modeling of observations by both the robot and human cooperatively and codependently towards a common goal. Unlike a human centered or robot centered framework, the full effort and attention of both MRUAV and operator is needed. The mission takes precedence over the individual needs of the human or robot.

The means for communication between the operator and robot is their shared basis. For an Operator Blended system, this shared basis must also be functionally relevant to both team members. In the case of the MRUAV and human, the navigation process fits this role. For RobiSEND, movement is coupled to the local search process. It is the means in which complex observation models are built and the environment is explored for both the human and the MRUAV. As discussed in Chapter 5, movement is central to intelligent information gathering in complex environments. Local movement is triggered relative to observation models for the RobiSEND navigation process.

6.3.1 Robotic system

Within a human-robot team with a functionally relevant shared basis, challenges persist but may be solved collaboratively. As noted, operator blending involves solving perception and integration challenges of unmanned systems through the intersection of human and robot needs. In this research, these challenges are called the Observational Needs of the human-robot team. For the robotic system, Observational Needs are broken down into 3 areas: ‘What?’, ‘‘When?’’, and ‘How?’. These are requirements of the robotic system in a teaming environment even in the absence of an operator blended approach.

'What?' is derived directly from raw sensor data and its processing to form more complex units of attention. RobiSEND possesses a binocular stereo camera and LIDAR for this purpose. The stereo camera provides raw color information and distance measurements through disparity. The LIDAR gives laser range data based on time of flight. Both of these sensors are able to quantify information available from the surroundings.

''When?'' is similar to 'What?' in that both correspond to bottom-up information. The difference is that, as opposed to measurable quantities in the environment, ''When?'' is related to time. There is a tremendous power in measuring how information changes or the state of the environment may be effected over time. In addition the timing of sensors or alternative sources of information plays a key role in how that data is interpreted. Probabilistic instruments such as Kalman or particle filters are dependent on a thorough understanding of time as well. For robotic system operating in a real environment, time is critical for determining the level of trust and significance of observations.

Lastly, 'How?' involves the methods and algorithms utilized in leveraging available information. This is in essence the black box between inputs and outputs and may come in the form of a computer code or a GUI. This question does not relate to who or what is served by the process, but rather the process of data analysis itself. This falls under an area governed by the robots role in the operator blended system because, regardless of the inputs and outputs, the black box must be coded and programmed. All robots must process raw data through their algorithmic approaches for problem solving.

6.3.2 Human system

The human half of an Operator blended system consists of the 3 Observational Needs 'Who?', 'Where?', and 'Why?' Because a robot has no simple path for understanding who might benefit from its actions, where to extract essential information, or why it even functions at all, these questions are better answered by its human operator. These needs are often transparent to a human operator in teaming situations even when Operator Blending is not involved.

In most missions, 'Who?' evolves quite rapidly. It includes but is not limited to the operator of the system and encompasses all human benefactors. Human benefactors include team members, observers, or unaware bystanders localized to the operational area. Because RobiSEND is a system to support these individuals, the human operator must be aware of who they are and their relevance to the task at hand. This involves safety concerns and general ideas about how to best serve this constituency. Because all robotic teaming situations are designed to support human benefactors, understanding their requirements as they effect the function of the mission is of utmost importance.

‘Where?’ is attention direction towards essential information. Bayesian theories and neural networks have been designed to reach these types of conclusions by reducing the dimensionality of the problem to what is needed to make a decision. For RobiSEND and an Operator Blended approach, it is assumed that the direction of system attention is best done by the system operator. Humans are gifted at siphoning data for functional relevance with relatively limited training data. Although we may be tricked, our perceptual sensors are based on the assumption that we are efficient at these tasks. In systems where the human is decided to be needed in the loop, the human understanding ‘Where?’ is already a need to develop trust and reliability.

‘‘Why?’’ is an area that will never be able to completely be grasped by a robotic system. This is related to our overarching motivations. This insight directs our sensory systems to weight certain inputs over others based on more global mission objectives. For example, a robotic car might sense a log and a prone human both as objects in its path that it may traverse. It must have the ability to determine the human is not acceptable to run over because it would result in the loss of human life. Although the model may be adjusted to accommodate this specific example, the level of certainty needed is best supplied by a human. In tasks that are critical to mission success, the human introduction of answers to the question of ‘‘Why?’’ is needed.

6.3.3 Operator Blending Intersections

Models must be generated that reflect system resources, the specific mission, and operational needs. It is the purpose of this research to demonstrate that coupling of resources may produce more useful perception and integration capabilities. Derivation of object and mapper models makes up the components of operator blending that lead to these goals. The observation model generation process results in deliverables that are meaningful to human benefactors of the technology and relatable to robotic collaborators.

If the Observational Needs of robotic teams and human teams are combined, they result in six total Observational Needs for human-robot teams. Although they may be observed independently and often are traditionally in teaming situations, they all contribute to the components of the operator blended framework. Figure 70 summarizes operator blending relationships with an emphasis on Operational Needs and their relationships to one another. All Observational needs touching one of the designated gold boxes are said to be linked through the operator blending framework. The figure is designed to be a circular 3D figure attached at the top and bottom edges. To show the relationship of the Operational need at the top to the Operational Need at the bottom, 'How?' and 'Who?' are printed twice.

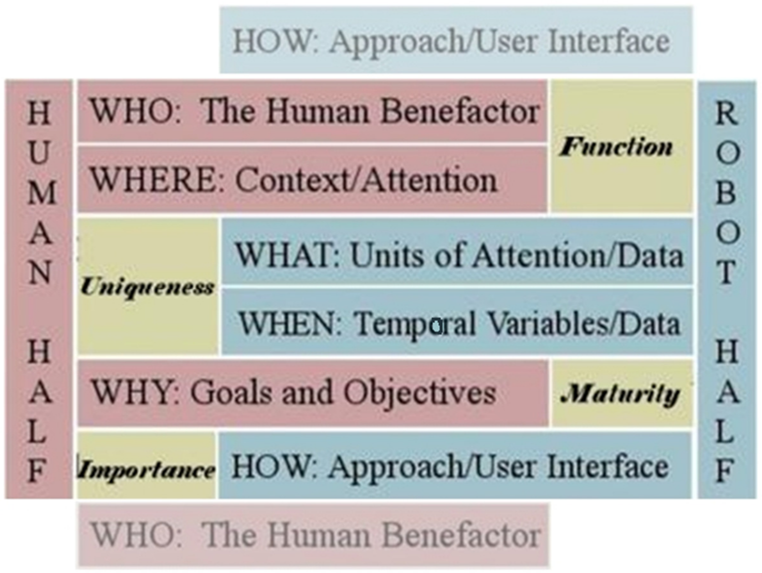

HOW: Approach/User Interface
HUMAN HALF
WHO: The Human Benefactor
WHERE: Context/Attention
Function
ROBOT HALF
Uniqueness
WHAT: Units of Attention/Data
WHEN: Temporal Variables/Data
WHY: Goals and Objectives
Maturity
Importance
HOW: Approach/User Interface
WHO: The Human Benefactor

Figure 70: Operator Blending Human-Robot Collaboration framework and intersections

The 6 Operational Needs have general relevance as they stand alone. The power of Operator Blending is not in their existence as individual contributors, but rather, the intersection of these areas that foster the codependence and cooperation between humans and robots. These intersections accomplish coupling of the human and robot to the observation model generation process. There are four intersections identified for the RobiSEND system: Function, Uniqueness, Maturity, and Importance. All four of these intersections are made up of Operational Needs from both the human and the robot.

6.3.4 Observation Traits

As discussed, six Observational Needs are defined, three for the robot and three for the human, that quantify operator and robotic system perception and integration in teaming situations. Operator blending is identified as Operational Need synthesis that results in codependence and cooperation between robot and human for better mission effectiveness. Four areas of synthesis deemed traits are developed to describe these intersections for the RobiSEND system: Uniqueness, Function, Maturity, and Importance.

Each trait embodies essential elements of codependence and cooperation with components from human and robotic team members. Traits are formulated through codependence because of their reliance on human and robotic sources of information. These traits must provide the means to categorize and combine information from sensors and operator feedback for significance to the observation modeling process.

Their relationships to other traits are leveraged for model significance to the process of navigation. As mentioned, the process of navigation is the shared basis that establishes a functionally relevant communication between robot and operator. The observation models built through this process builds the relationships between the MRUAV and operator that are necessary for mission effectiveness. For environments that lack structure, knowing what observations are is not as important as knowing what they mean, what they are doing, what they might do, or what the observer or robot must do considering their presence. In the RobiSEND operator blending, this is quantified through the presence or lack of traits in object and mapper models.

6.3.5 The Concept of Function

Function is defined as the intersection of 'Who?', 'What?', 'How?', and 'Where?' An observation model possessing function is defined as one whose purpose is relatable to the human benefactor, well defined contextually through operator introduced attention, possesses data that lends itself to the formulation of the necessary units of attention, and may be incorporated regularly in the algorithms or user interface designed for the system.

The Function parameter for RobiSEND governs the ability to differentiate observation internal characteristics. For example, RobiSEND uses SIFT features to quantify these aspects of an object. Object models that produce more than 10 SIFT features are said to possess high Function, while objects with fewer than 10 features are low in Function. Those object models that produce a high number of SIFT features are visually distinctive and their purpose therefore recognizable to humans localized to the MRUAV. The operator, in turn, has directed the attention of the perceptual system towards a boundary based segmentation containing that has a high likelihood of containing the important object and background information. In addition, high numbers of SIFT features also indicate that the other local and global features that make up the object model are available for extraction and, therefore, the necessary units of attention are accessible. Lastly, because SIFT features are such a major component of object recognition and tracking, objects with high numbers of SIFT features are capable of being implemented within the user interface and algorithms necessary for the function of the system.

For observation models not constructed from visual information, function may have a different means of determining whether or not its internal characteristics are well defined for both the human and the robotic agents. As a comparison, mapper model internal characteristics are made up of LIDAR data. As opposed to the internal characteristics of an object defined by SIFT features within an image, Function defines the internal characteristics of clusters of LIDAR points within a scan. Mapper models with features that demonstrate a high degree of linear structure have high Function. This concept is defined further in Chapter 5 on qualitative mapping.

6.3.6 The Concept of Uniqueness

Uniqueness is the intersection of 'Where?', 'What?', ''When?'', and 'Why?' It is defined as the ability to discern important information from background. Once a boundary based segmentation is made, most intelligent systems have the ability to extract the useful parts of the data and discard less useful pieces. Observation features, temporal information, and the ability to introduce goals and objectives to the useful parts of this information then provide observations with unique signatures.

For mapper models, Uniqueness may be determined by the structure present in data determined by principal component analysis along the major and minor axis of a cluster of LIDAR points. A scan with cluster groupings demonstrating a low amount of covariance along one principal axis in comparison to the other is said to possess a high amount of Uniqueness. This must be shown to be consistent over time and as the RobiSEND payload propagates within the environment accomplishing local search related goals.

Function and Uniqueness observed together define the particular class of observation within the operator blending framework. Classes have widely varying functional relevance to the particular human-robot collaborative system. The traits of Importance and Maturity add the finer details that make up the observation model functional significance. A good analogy would be a comparison to biological systems. Function and Uniqueness comparatively determine the gender and sexuality that establish foundational characteristics. Maturity and Importance give an idea of the experiences that shape the individual observation. For object and mapper models, the concepts of class traits and detail traits are further explored in more detail in the following section.

6.3.7 The Concepts of Importance and Maturity

The detail traits of Importance and Maturity both only are defined by three Observational Needs. Importance is made of a majority of human based Observational Needs, while Maturity has mostly robot based Observational Needs. As a result, both define the functional characteristics of an observation as its role evolves during the local search process from different perspectives. Once the class of an observation is identified, Maturity and Importance solidify the functional role considering present perception and integration conditions and the local search context.

Maturity involves the ‘’When?’’, ‘Why?’, and ‘How?’ areas. With knowledge of the mission objectives, the algorithms to process data and the search approach must evolve over time. As mentioned in Chapter 4, this maturation process is linked to the concept of movement. It is the most common means intelligent systems attain additional information. Models may be made more useful, reliable, and diverse over the maturation process.

All observation models may eventually attain maturity if their characteristics remain consistent over time. For object models, this may be linked to the amount of time an object is tracked. For mapper models, this relates whether the spatial relationships captured by the model and its corresponding topological information are recognizable from a previous instance. In both cases, the current observation model has functional traits that are consistently relevant to the RobiSEND system.

Importance involves ‘Why?’, ‘How?’, and ‘Who?’, three Observational Needs that are essential to the present value of an observation model to the human-robot team. The operator must indicate the value of information based on the mission objectives and communicate this to the robotic system through algorithms and the GUI. This, in turn, influences how the information is processed. For RobiSEND, the operator views visual information in the GUI and distinguishes the importance by directing the attention of the system. The definition of the Importance characteristics may change at any time during the local search process depending on the operator control of this process.

6.3.8 Object trait specifics

The traits identified for the RobiSEND system are object Uniqueness, Importance, Function, and Maturity—intersections of the Observational Needs. They are constructed through use of operator interactions and data metrics as defined in Chapter 5. Observation models possess recessive or dominant forms of each trait with 16 possible combinations. Each combination of traits when considered together is the observation COG and corresponds to a specific functional relevance to the task of local search in an unstructured environment. Figure 71 shows the Operator Blending framework with emphasis on the Observational Needs and traits to generate the COGs of observation models.

The Operator Blending framework is presented in this format in order to show the different layers addressed through its framework. The top layer is the existence of the human robot team. This may be broken down into the human and the robot half. Then, the Observational needs of the human and the robot further reduce the perceptual and integration considerations. These considerations produce the intersections of Maturity, Uniqueness, Importance, and Function. Each of these intersections possesses dominant and recessive traits that make up the COD for each object.

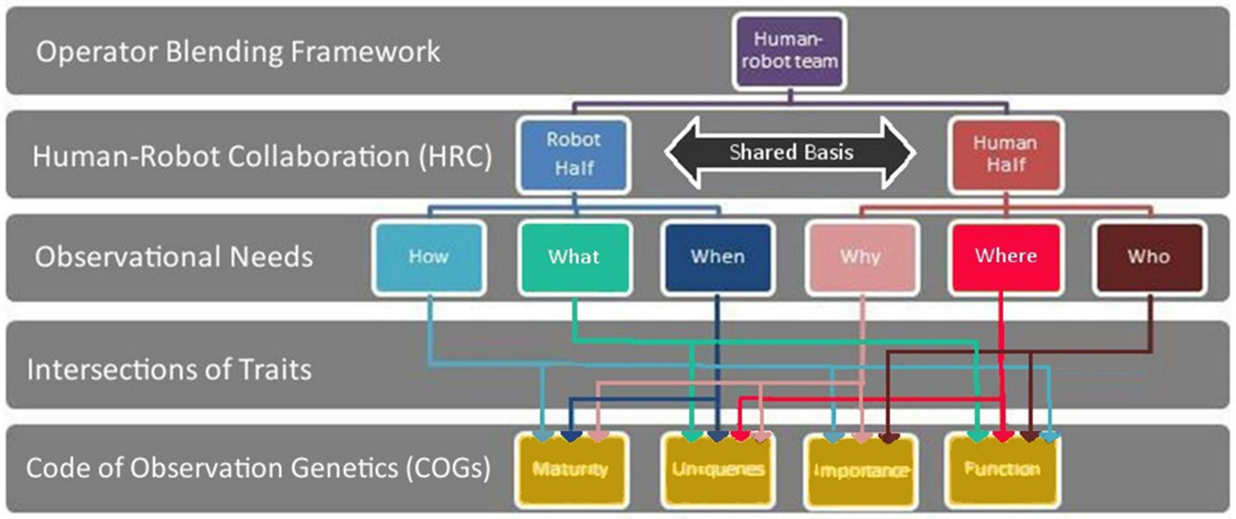


Figure 71: Tree version that shows intersections and how they lead to Code of Observational Genetics from Operator Blending and a shared basis

In terms of object models, data with more dominant traits are best for search and navigation while those with more recessive traits take up less memory and are mainly used for topological annotation and correction. For mapper models, those models with more dominant traits are more metric in nature, while those with more recessive traits, more topological.

Both Uniqueness and Function include data metrics derived from MRUAV sensors. Uniqueness is the ability to make a meaningful segmentation of the observation from the surrounding background. Function, on the other hand, is determined by the observation internal characteristics. If there are adequate internal features for the observation to be distinguished from other observations it is said to possess the dominant Function trait.

Importance is influenced mainly by the system operator and maturity is determined through observation over time. In essence, every observation has a level of significance because it was important enough to be segmented. The operator, however, may signify through his actions a higher importance designation, meaning that he views it as notably relevant to the search or navigation process. Once this selection is complete, if the observed attributes are consistent over time and MRUAV translation, then the object model or mapper model is considered to have dominant Maturity. During the object maturation process, a more detailed model is generated.

Each trait may be found to be two values, one corresponding to dominant or high and the other corresponding to low or recessive. Uniqueness may be Explicit (E), meaning of high uniqueness, or Indefinite (I), meaning of low uniqueness. Importance is either Normal (N), the recessive form of the trait, or Special (S), the dominant form of the trait. Likewise, function may be Temporary (T), meaning there is incomplete functionality, or Full (F), for completeness. Maturity is slightly different. All models begin as Juvenile (J) and have the opportunity to become Permanent (P) if their observed characteristics remain consistent over time and space. During the maturation process, the model is improved because more information is collected to be a part of the object model.

These four traits manifest themselves in a total of 16 COGs for each observation type. Each COG has its own relevance to the search and/or navigation process. As a general rule of thumb, the more dominant traits an object model possesses, the better suited the model is for local search. Likewise, the more dominant traits a mapper model possesses, the more metric its construction. Each computer generated trait is evaluated by comparing the model the total time it is tracked.

Although they have individual significance, these traits are not observed in isolation. For instance, object model class is judged from observing the Uniqueness and Function traits together. All objects are assumed to fit in one of the four defined classes. The COGs for all of the 16 possible observation models allows for object functional classification.

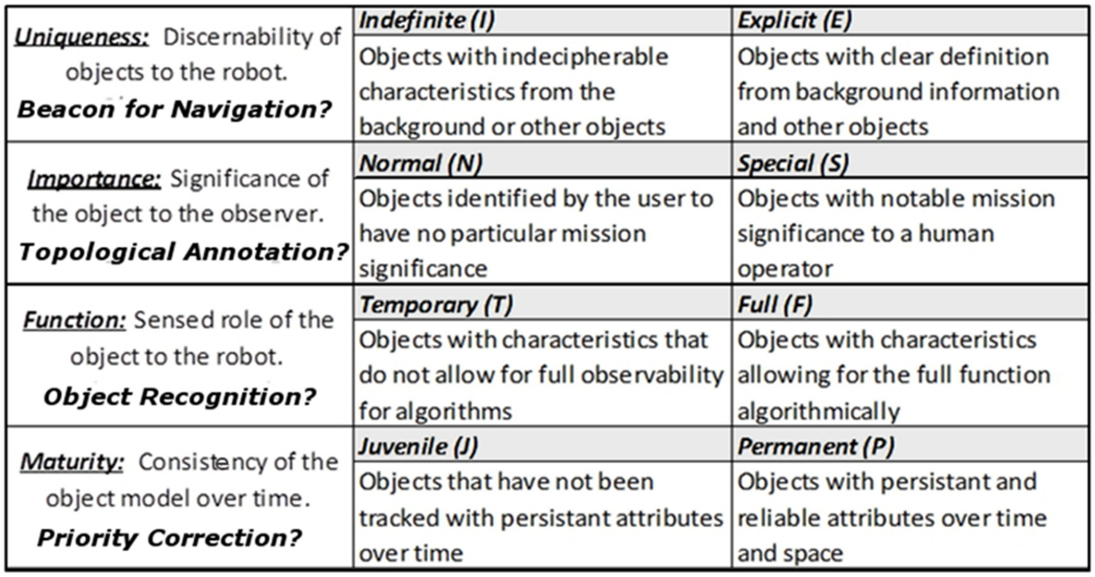
6.4 Models

Object model and mapper model details are given in Chapter 5. How these models fit within the Operator Blending framework is explained in this section. Sub-section 1 explains the different types of object models and what goes into their COG models. Sub-section 2 then does the same for mapper models. The presentation of COG models for this application and the COG concept are novel contributions of this work.

6.4.1 Object Models

The dominant and recessive traits for object models under the Operator Blended framework are given in Table 9. Object models mainly consist of visual information provided by the binocular stereo vision system as discussed in Chapter 5 but may include LIDAR information that overlaps with camera FoV in certain situations. The camera provides three channel color images and disparity images as raw data. From the operator blended approach, local SIFT features, local MSER features, color histograms, and a shape metric are derived from this information for operator selected objects. The operator directs system attention resulting in the COG of the object model. Based on the COG, objects possess a navigational significance tied to the RobiSEND local search process.

Table 9: Table showing Object Model COG Traits



Uniqueness: Discernability of objects to the robot. ***Beacon for Navigation?***	***Indefinite (I)*** Objects with indecipherable characteristics from the background or other objects	***Explicit (E)*** Objects with clear definition from background information and other objects
Importance: Significance of the object to the observer. ***Topological Annotation?***	***Normal (N)*** Objects identified by the user to have no particular mission significance	***Special (S)*** Objects with notable mission significance to a human operator
Function: Sensed role of the object to the robot. ***Object Recognition?***	***Temporary (T)*** Objects with characteristics that do not allow for full observability for algorithms	***Full (F)*** Objects with characteristics allowing for the full function algorithmically
Maturity: Consistency of the object model over time. ***Priority Correction?***	***Juvenile (J)*** Objects that have not been tracked with persistant attributes over time	***Permanent (P)*** Objects with persistant and reliable attributes over time and space

From operator selected objects, Uniqueness is determined from the characteristics of the shape metric and Maximally Stable Extremal Regions (MSER). MSER features are used to differentiate background and foreground distance measurements from the disparity information. Explicit (E) Uniqueness results if and only if the object itself is determined to be a member of the disparity foreground and there is clear disparity background information. Indeterminate (I) Uniqueness is the recessive trait that corresponds to the object being a member of the background or if there is no clear foreground in the object image. Objects must possess Explicit Uniqueness to be used as beacons for the navigation process.

Function, on the other hand, is determined by user selected object internal characteristics from the Scale Invariant Feature Transform (SIFT) and color histogram information. If the object possesses adequate internal features to be distinguishable from other objects, it is said to possess the Full (F) Function trait. A threshold value of 10 SIFT features is used. If the object model has fewer than 10 SIFT features then the object is said to have Temporary (T) Function and possess the recessive trait. For the navigation process, objects with Temporary Function are not saved for object recognition.

The operator decides an objects Importance through the Graphical User Interface (GUI). When selecting an object, the operator decides to left click for Normal (N) Importance or right click for Special (S) Importance. Operators are to select Special Importance when they believe the object of interest is particularly important to the local search process. Once a Special Importance object is selected, it is saved for topological map annotation.

Maturity is determined over time and is found from observing object characteristics during MRUAV propagation. All objects are assumed to possess Juvenile (J) Maturity upon selection. Once objects are selected, the object model is tracked over time, improving the quality of the object model feature representation. As the object maneuvers to this distance, it updates its object model at intervals of time. If object characteristics are consistent over this time, the object then possesses Permanent (P) maturity. Objects that have achieved maturity are weighted higher in the map optimization processes.

The class traits of Function and Uniqueness are the basis for the object model structure. Four classes exist as in all Operator Bleinding COG constructions. Dominant Function and dominant Uniqueness yield objects stored for navigation and object recognition. Recessive Function and dominant Uniqueness are stored just for navigation. Recessive Uniqueness and dominant Function are just stored for object recognition. The case of both recessive Function and Uniqueness is a special case. Because the color and disparity information in this case has proven to be unreliable, the LIDAR information is saved that overlaps with the boundary based segmentation. This information is used to correct the structure of the global map deliverable because it usually corresponds to flat walls with little defining characteristics. Flat walls with little contrast, however, are captured quite efficiently with LIDAR data.

These four traits ultimately yield 16 COGs shown in Figure 72. In combination, they give objects specific roles for RobiSEND. An object with Temporary Function may be used for navigation if it has Explicit Uniqueness, but may not be used for map correction depending on the importance of those objects around it. As shown, the COGs characterize objects based on their functional significance to the operator and MRUAV for local search. Object models, in many respects, are the means in which navigation is possible for RobiSEND. Much like a human being climbs a rock wall or Spiderman travels from building to building, object models are the beacons for local translational motion. This allows for translational motion to be carried out strictly through local commands. As mentioned in Chapter 2, this potentially builds more situational awareness.

ESFP—Object used as a beacon for navigation and stored for object recognition and map correction. High weight given during optimization and used for topological annotation	**ENFP**—Object used as a beacon for navigation and stored for object recognition and map correction. High weight given during optimization but not used to annotate the final map	**INFP**—Object not used as a beacon for navigation but stored for object recognition and map correction. Given high weight for map optimization but not used to annotate the final map.	**ISFP**—Object not used as a beacon for navigation but stored for object recognition and map correction. Given high weight for map optimization an dused to annotate the final map
ESTP—Object used as a beacon for navigation but not stored for object recognition or map correction. High weight given during optimization and used to annotate	**ESFJ**—Object used as a beacon for navigation and stored for object recognition and map correction. Low priority for optimization but used to annotate the final map	**ISTP**—Object not used as a beacon for navigation and not stored for future object recognition. Given a high weight for map optimization and used to annotate the final map	**ISFJ**—Object not used as a beacon for navigation but stored for object recognition and map correction. Low priority for optimization but used to annotate the final map
ENTP—Object used as a beacon for navigation but not stored for object recognition or map correction. Given high weight during optimization but not used for map annotation	**ENFJ**—Object used as a beacon for navigation and stored for object recognition and map correction. Not given a high priority for map correction and not used to annotate the map.	**ISTJ**—Object not used as a beacon for navigation and not stored for object recognition and map correction. Low priority for optimization but used to annotate the final map	**INFJ**—Object not used as a beacon for navigation but stored for object recognition and map correction. Low priority for optimization and not used to annotate the final map
ESTJ—Object used as a beacon for navigation but not stored for object recognition or map correction. Not weighted highly for map correction but used to annotate the final map	**ENTJ**—Object used as a beacon for navigation but not stored for object recognition or map correction. Not weighted highly for map correction and not used to annotate the final map	**INTP**—Object not used as a beacon for navigation and not stored for object recognition and map correctin. High priority for optimization but not used to annotate the final map	**INTJ**—LIDAR information substituted for visual information. Distances used to supplement the performance of mapper models. Information not stored otherwise.

Figure 72: Code of Observational Genetics for Object Models

The detail traits of Maturity and Importance depend on the approach the operator takes to the local search process and are not necessary at all for minimal object model generation. Object models may never be tracked long enough to reach maturity depending on how quickly the next object model is selected. In addition, if the operator chooses not to specify Importance, no object models may be positively weighted for the map update stage. This approach does not ultimately yield properly annotated maps of the environment that result in the operator losing awareness of the surroundings. If object models are not being selected during the search process, the effectiveness of mapping and the ultimate effectiveness of the local search mission are negatively affected.

6.4.2 Mapper Models

As mentioned in Chapter 5, inferences are made from sensor information during an instant in time. Over time, these inferences accumulate and are expressed through plotting a representation from the qualitative map place vocabulary. A mapper is invoked based on the nature of these inferences over the observed time. In order to determine the specific mapper from the place vocabulary, an Operator Blending approach is used to measure functional relevance of accumulated inferences. The operator role in the mapper model creation process is more indirect than the object model creation process. The operator influences mapper models through when object models are identified and their spatial significance to the local search process.

Mappers are invoked in one of two ways. They may either be triggered automatically when accumulated inferences reach a certain threshold value in Safe mode. They may also be invoked manually upon the operator making an observation and building an object model in Shared mode. Both triggering methods are considered derived from operator input. In safe mode, operator decisions on navigation shape the types of observations that influence the accumulator. In Shared mode, operator directed attention directly results in the invocation of mappers.

Chapter 5 gives the five place vocabularies that are used as building blocks for the qualitative mapping approach and make up the place vocabulary. During the local search process, this map is built in real time. Local information is stored and the map is built by mapping from the place vocabulary in the order of invocation. This is mainly an open loop process. The optimization stage takes global information into account and corrects the map for inconsistencies.

The manner in which the place vocabularies are invoked is done through the Operator Blending approach. Inferences as observed over time are used to describe the COGs for mapper models. Once the mapper is invoked, this information is used to build the functional profile of the qualitative observations for the local search global map building process. Table displays the dominant and recessive traits for mapper model COG traits.

Mapper model components with majority dominant traits have more metric characteristics, while those with more recessive traits have more topological characteristics. Both metric and topological characteristics are needed to build an effective qualitative map through Qualitative Spatial Reasoning.

Table 10: Table displaying Mapper Model Code of Observational Genetics Traits

Uniqueness: Discernibility of structure and space ***Hall/Junc vs space_c/space_o***	***Indefinite (I)***	***Explicit (E)***
	Mapper accumulates scans without sensible structure and/or point density	Mapper accumulates scans with sensible structure and/or point density
Importance: Significance of place vocabulary content ***familiarity vs first visit***	***Normal (N)***	***Special (S)***
	Areas not previsited by user or without user selected objects on first visit	Areas previsited by user or with user selected objects on first visit
Function: Sensed linearity of place structure over time ***Hall/space_c vs Junc/space_o***	***Temporary (T)***	***Full (F)***
	Mapper accumulates scans with high covariance between points and/or linearity	Mapper accumulates scans with low covariance between points and/or linearity
Maturity: Consistancy of info over time and/or space ***Large vs small map components***	***Juvenile (J)***	***Permanent (P)***
	Mapper accumulates information over limited space or time	Mapper accumulates information over extended space or time

Mapper model Uniqueness is a measure of the discernability of definitive point clusters in space independent of their actual structure. This is a measure of point density and is used to see if LIDAR points are dispersed over the scan plane corresponding to Indefinite (I) Uniqueness or if LIDAR points are grouped densely together for Explicit (E) Uniqueness. The density of point clusters is independent of their actual shape within the cluster.

This trait is used to decide between two groups of mappers: the mappers that plot definitive elements and the mappers that plot space. Definitive element mappers are Hallway and Junction. Space mappers are the two elements of the place vocabulary that plot OpenSpace and ContentSpace. If the accumulator measures dominant uniqueness over time, then, when invoked, it either plots a Hallway or a Junction, with the inverse being true for OpenSpace grouped with ContentSpace.

The function trait is determined from the linearity or structure of the invocation over time. If the mapper accumulates invocations that are consistently linear for definitive elements or structured for space elements, the dominant Function trait is assigned. Dominant function is considered Full (F) Function, while recessive Function is Temporary (T) Function.

When invoked with a dominate Function trait, the mapper outputs either a Hallway or ContentSpace representation from the place vocabulary. Likewise, a recessive Function trait outputs a OpenSpace or Junction output. This trait, along with Uniqueness, determines the type of mapper invoked for place vocabulary selection.

The Maturity trait is determined by the nature of the accumulator process and how it changes over time. Maturity can either be Juvenile (J) or Permanent (P). Like object models, all new mapper models are considered to begin with Juvenile Maturity. If certain invocations are accumulated over time and space, then this status is switched to Permanent Maturity.

The shift towards permanent maturity for the local search process results in a specific outcome depending on the class of mapper defined from Function and Uniqueness. If a Hallway mapper is invoked, maturity is represented by if the accumulator detected invocations that resulted in the distance covered being more than x_const for the GreaterThanX mapper or greater than x_const for the LessThanX mapper. If a junction has Permanent Maturity, it is represented by the MajorJunction mapper. As mentioned in Chapter 5, major junctions require an angular or translational decision by the operator before they proceed. On the other hand, if the junction is found to have Juvenile Maturity, it is a Minor junction and may be represented without any input from the operator. Lastly, if either OpenSpace or ContentSpace is determined to have Permanent Maturity, they are represented by a larger spatial region. This is because the information accumulated over this time is assumed to represent a larger area of space. This determination for Space place vocabulary is done completely on the invoked amount of displacement in the x direction and y direction and is not related to the actual time traversing the space.

Lastly, Importance is considered a trait that is dominated by the role the user plays in the mapper model creation process. If recessive, the mapper model is considered to have Normal (N) importance and if dominant, the mapper has Special (S) importance. Importance plays a vital role on the optimization stage of the map. Mapper models with high importance take priority in correction.

Importance is determined by considering the topological designations that occur during invocation over time and also where the human-robot team decides to navigate during the search process. If an object correspondence is made and it can be determined that the local search team has been in an area before, this area is considered to have Special importance. This is brought about by the operator's decisions, because it is the operator that issues that translational and rotational commands in Safe mode. In addition, if the operator directs the attention of the system as the accumulator is building the mapper model over time and the object he specifies is important, then the mapper model is also characterized as important. The use of mapper model importance in the optimization phase is discussed further in Section 6.5. Figure 74 summarizes how these traits are used to build COGs for mapper models.

ESFP—*GreaterThanX* Hallway mapper invoked. Mapper model revisited or object/cluster selected during accumulator.	**ENFP**—*GreaterThanX* hallway mapper invoked. Mapper model not matched to a previous model and contains no identifiable cluster or object model.	**INFP**—*ContentSpace* mapper invoked. Wide region. Mapper model not matched to a previous model and contains no identifiable cluster or object model.	**ISFP**—*ContentSpace* mapper invoked. Wide region. Mapper model revisited or object/cluster selected during accumulator.
ESTP—*MajorJunction* mapper invoked. Mapper model revisited or object/cluster selected during accumulator.	**ESFJ**—*LessThanX* Hallway mapper invoked. Mapper model revisited or object/cluster selected during accumulator.	**ISTP**—*OpenSpace* mapper invoked. Wide region. Mapper model revisited or object/cluster selected during accumulator.	**ISFJ**—*ContentSpace* mapper invoked. Thin region. Mapper model revisited or object/cluster selected during accumulator.
ENTP—*MajorJunction* mapper invoked. Mapper model not matched to a previous model and contains no identifiable cluster or object model.	**ENFJ**—*LessThanX* hallway mapper invoked. Mapper model not matched to a previous model and contains no identifiable cluster or object model	**ISTJ**—*OpenSpace* mapper invoked. Thin region. Mapper model revisited or object/cluster selected during accumulator.	**INFJ**—*ContentSpace* mapper invoked. Thin region. Mapper model not matched to a previous model and contains no identifiable cluster or object model.
ESTJ—*MinorJunction* mapper invoked. Mapper model revisited or object/cluster selected during accumulator	**ENTJ**—*MinorJunction* mapper invoked. Mapper model not matched to a previous model and contains no identifiable cluster or object mode.	**INTP**—*OpenSpace* mapper invoked. Wide region. Mapper model not matched to a previous model and contains no identifiable cluster or object model.	**INTJ**—OpenSpace mapper invoked. Thin region. Mapper model not matched to a previous model and contains no identifiable cluster or object model.

Figure 74: Mapper Model Code of Observational Genetics

6.5 Map Optimization

Emphasis on the object recognition problem over the correspondence problem is one of the most important messages of this research. Correspondence may be relied upon in the map optimization elements of mapping to improve the map. The methods for correspondence between mapper and object models are presented in this section. Passive observation interpretation is optimization that involves considering object models and is conducted during the map building process. Active observation interpretation is done at the conclusion of the map building process and involves developing correspondences between object models.

6.5.1 Passive Observation Interpretation

One of the main benefits to building maps based on local information is these maps may be updated easily because all measurements are relative to previous mapper models. If a change is made to the length of a hallway mapper model in the middle of the mapping process, the entire map does not have to be corrected as long as that change is applied to all relevant distances. Map errors can be dealt with quickly and with minimal computational power. Although this is true for scan matching algorithms as well, direct correspondence of features is a much more difficult problem to correct.

For QUICK maps, the latter mapper model match is always corrected for to match a model developed earlier in the mapping process. Correction is done for two reasons. Because QUICK map distances are qualitative, the map correction stage ensures inferences of the same areas in an environment correspond to one distance as opposed to multiple distances. This problem is best understood by Figure 75 in a cyclic environment. Each time RobiSEND goes around this loop, it has a slightly different estimation of distances. Map correction ensures that these features recognized as the same representation and merges them into one. The second reason is, because map correction merges mapper models recognized to be identical in the map, it reduces the size of the map for transmission.

QUICK map optimization involves verifying the location of key points follow a set number of constraints. During map building, there are several build rules established that correct the map. These are known as POI because the operator does not participate or contribute to this process. POI mainly involves condition recognition from the occurrence of certain combinations of mapper models. This optimization is primarily done to ensure that common junctions correspond to one another.

Map building must follow constraints in order to ensure qualitative features correspond to each other once they develop a full COG. The approach primarily revolves around matching junctions. The other mapper models may be converted to their other possible inference values to bring mapper model locations closer together. The rules that govern POI are as follows:

1. If hallways are detected in between two quick 90^{o} rotations, all hallway mapper models are considered important and their length may be altered considering the distance to reach the closest junction
2. Two junctions plotted within a radius *r_const* may be merged as the same junction. This combines junctions with similar characteristic.
3. Placement of important junctions and then mature junctions take precedence in junction placement conflicts

Figure 75 displays the issues with not having POI as part of the real time map building process.

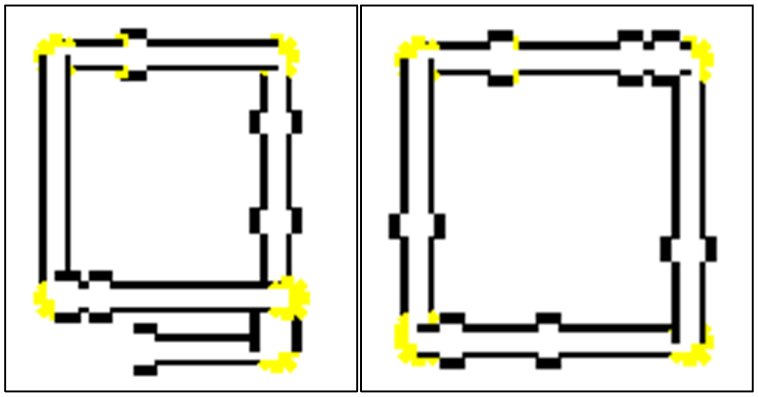


Figure 75: Examples of the issues before and after POI

Hallway length does not correspond to the same total length when traveled in each direction. In addition, multiple junctions are found where only one should represent the actual junction is space. As shown in Figure, after POI, similar junctions are combined into one junction. In addition, hallways and content space features that depend on the amount of propagation, within the unknown environment.

6.5.2 Active Observation Interpretation (AOI)

After mapping is complete, the operator has the choice of implementing AOI that iteratively corrects the map based on object model locations. Observation models that attain importance during the local search process are considered for correction. AOI occurs in two stages. First, observation models that were matched through operator interface selection process are corrected. These are all observation models that are matched with other observation models through their local and global visual features as discussed in Chapter 3. Secondly, observation models with importance are corrected considering their vicinity to other observation models.

Observation model location correction is treated similarly to junction model locations in POI with slightly different correction rules. These rules are as follows:

1. Previous adjacent hallway mapper models may assume alternate inference values in order to move matched observation models closer to correspondence.
2. Once matched observation models are made to have direct spatial correspondence, they are merged as one feature representation. All paths from these features are inherited by the new feature. Any logical discreprency between the two features is resolved in favor of the observation model made earlier in the local search process.
3. Merged models inherit the local features of their constituent models.

4. All previous corrected relationships are maintained if that correction resulted in a direct correspondence in the x or y direction

Figure 76 displays examples of the issues before and after AOI is implemented

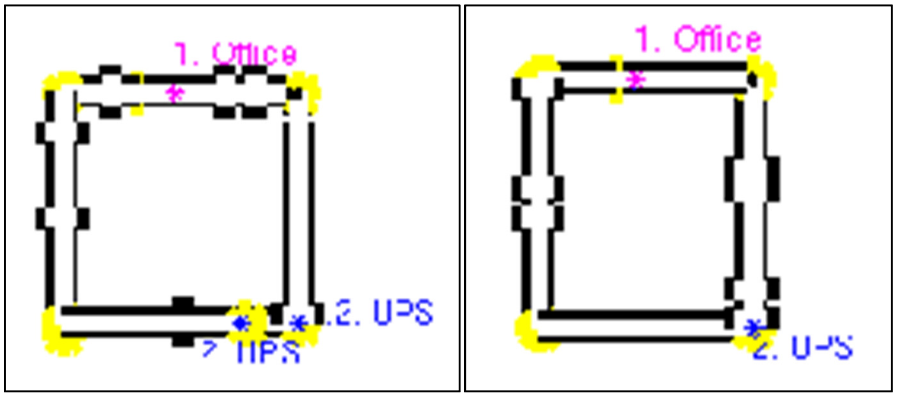


Figure 76: Figure displaying common issues corrected with AOI optimization

At the conclusion of the map optimization process, the map reflects the object recognition process. This leads to correspondence of junctions and matched object and models. The map must then be annotated with topological information to complete the mapping process.

6.6 Finalizing the Map

Once the QUICK map structure is completely updated, its data may be annotated through the text utility in MATLAB. Figure 77 gives an example of map annotation utilizing this approach.

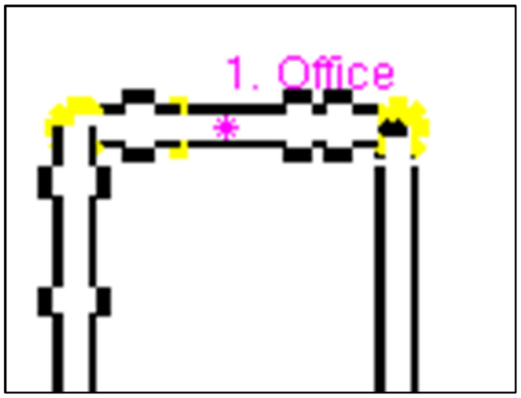


Figure 77: Example of map annotation through the text function

If the operator specified specific labels for objects, they will be reflected in the final map. In the event that the operator would like to add labels to the map at the conclusion of the search process, the edit button may be activated that opens the map up for editing. Once open for editing, clicking on the map pulls up a text dialog box similar to the RobiSEND GUI that allows the operator to enter a specific name for objects or areas in the map.

Once the map is finalized, clicking on a labeled area pulls up the observation information. If the observation was matched with visual features, the visual feature models are displayed in a separate figure window that presents the object background template and location information. Figure displays the graphical depiction of topological information on a map. The window displaying object detail information is shown in Figure.

Chapter 7 introduces the results of the QUICK mapper. The results are discussed with future work possibilities. Mapping presented of areas around the Virginia Tech campus are also given with comparisons to ground truth information. This is followed by a conclusion. Qualitative mapping takes advantage of Operator Blending and the context of its application to MRUAV based local search missions. MRUAV based local search is given as a basic example of Operator Blending concepts because of the unique perception and integration challenges involved in their low altitude operation. Although fly over missions exploit the current capabilities of MRUAVs, this work is about extracting novel capabilities through addressing critical perception and integration issues. The types of deliverables possible are given by QUICK maps because they take advantage of contextually derived local information from the perspective of the sensor platform. The results of this concept, possible because of Operator Blending, are presented in the next Chapter.

7. Results and Conclusion

“Now this is not the end. It is not even the beginning of the end. But it is, perhaps, the end of the beginning.”—Winston Churchill

Human-robot collaboration is growin in importance as robots move towards sharing a more integral part of human lives. In order to make the necessary progress, researchers must first ask the right questions. Many of the answers to these questions are not easy to come by and may require a revised outlook on the role of the human and robot within a team context. Others may require engineers to take on less intuitive challenges. Plenty of opportunity exists to move unmanned systems technology forward.

This research asks questions that are traditionally overlooked but get to the heart of important unmanned systems limitations. The answers provided for these questions are a step forward in solving perception and integration difficulties for MRUAVs conducting local search missions and a step in a different direction for human-robot collaborative teams. At the same time, it is merely a step. Many future steps are needed to move technology forward.

The first few steps along a novel direction, however, are first about finding a new path. Much work must be done, but the establishment of Operator Blending opens up new opportunities. After all, robots have always been intended to be helpful to human beings. Achieving a higher degree of functionality between the two is of utmost importance.

7.1 Contributions

This dissertation has made five key contributions to the field of Human-systems integration as it relates to MRUAVs. These contributions are Operator Blending, Code of Observational Genetics (COGs), Joint Intension Theory applied to unmanned systems, novel computational methods rooted in operator directed sensor attention in an operator blended framework, and Qualitative Unsupervised Intelligent Collaborative Keypoint (QUICK) mapping. A summary is first given in sub-section 1 that details each contribution. The structure of Chapter 7 is then given in sub-section 2.

7.1.1 Summary

Operator Blending, is the foundation for the other developments of this work. Simply stated, it emphasizes cooperative and codependent solutions to the problems of human-robot teams. The most paramount of these issues are deeply rooted in perceptual and integration based teaming concerns often neglected in current research. Both the human and the robot contribute what each does well in solving problems.

For applications where the human and robot are collocated and have limited resources, Operator Blending is a novel approach to building the relationships necessary for mission success. The specific application studied in this research is MRUAV based local search performed by a single operator for support of a small tactical team. The RobiSEND system was developed to demonstrate the concepts presented by emulating the performance of a MRUAV in a hardware-in-the-loop simulation. As a result, the potential of MRUAVs as a technology for human-robot collaborative missions was furthered through this research.

Code of Observational Genetics (COGs), Joint Intension Theory as applied to unmanned systems, and Qualitative Unsupervised Intelligent and Collaborative Keypoint (QUICK) mapping are novel concepts created to support the Operator Blending framework. For the human-robot team, COGs define the semantic relationships of observations in the environment. Joint intension theory establishes the basis for these relationships, taking into account common observational needs and the necessity to establish a shared basis for communication. For the RobiSEND system, this shared basis is the navigation and exploration process that leads to observation model building. The manner in which these observation models produce a representation of a previously unknown environment has resulted in QUICK mapping—a purely qualitative mapping method that interprets real world data through Qualitative Spatial Reasoning.

Computational approaches were also expounded upon through the context of Operator Blending. The presence of operator introduced attention allowed for new ways to extract information. Operator presence algorithmically generated a shift from the data point correspondence problem towards the object recognition problem. In terms of local search, this reduced the computational burden from metric reconstruction of the environment towards the recognition of local features. The AIM-Shift approach to object tracking was introduced to take advantage of operator directed attention through the introduction of boundary based segmentation. AIM-Shift adds additional color histogram information to the object tracking process.

7.1.2 Structure

The final system and deliverables, presented in this Chapter, portray the complete picture of the contributions of Operator Blending. Examples are given that illustrate the functional relevance of object models in the Operator Blending framework. All four COG traits—Function, Uniqueness, Importance, and Maturity—are explained individually with examples of their recessive and dominant forms. How these traits impact the object model role in the local search process is then presented.

Next, examples of object and mapper model COGs are given. The combination of traits to formulate the COG profile for each object defines its functional relevance. Example object model COGs are first discussed that demonstrate how COGs may be used to provide supplemental information to the local search process. This supplemental information drives the navigation process and plays a key role in the topological annotation of maps for Operator Blending. These results are contrasted with traditional approaches that are not concerned with the operational needs of both human and robot team members.

COGs are then discussed from the perspective of mapper models, which provide the structure of global map deliverables through QUICK mapping. Both mapper models and object models are needed to give the full perspective of Operator Blending. Object models are an example of local observations being leveraged to build localized deliverables. Mapper models demonstrate that local observations may be combined to formulate global deliverables. Local and global representations of information are needed when reconstructing the structure of an unknown environment and can potentially be created through the Operator Blending approach.

In the case of MRUAV driven local search operations, the global deliverable produces the qualitative map structure through the combination of mapper models. Object models and topological information is then used to annotate the qualitative map, while simultaneously serving as reference points for navigation. This synthesis of object models and mapper models produces QUICK maps during the local search process.

The COGs for mapper models are presented along with their performance under local search conditions. Joint intension theory results and evidence is given that shows the impact of navigation as a shared basis for the local search process. QUICK mapping models are then presented and discussed in comparison to more traditional approaches to mapping. Results are then discussed and opportunities for future work explored.

7.2 Object Model COG Traits

Object model COG traits are given is Section 7.2 for the specific local search application used to demonstrate Operator Blending concepts. The results of using this framework to map an actual unknown environment are given. Each of the COG traits are discussed as seen through the created GUI. Specific examples are given for the implementation and characteristics of each trait and how they contribute to Operator Blending. The examples given show the functional significance of the object during initial selection. It is noted that if the condition drastically changes from initial selection over the duration of tracking, the object model halts the tracking process. If the camera shifts by 90^o for example, the object being tracked is no longer in the field of view of the camera. As discussed in Chapter 5, the object model is only built as long as feature points can be matched to the ObjectTemplate image. The COG traits discussed in the following sections give an indication of the functional significance of an object model during the duration of the tracking process. It is this functional significance that influences object storage and the role of the object model in the shared basis of the navigation process.

7.2.1 Function

Chapter 6 explains Function as being Temporary in its recessive form or Full in its dominant form and derived from the observation internal characteristics. For RobiSEND object models, this trait is defined by object model SIFT features. For the object in Figure 78 identified by the magenta colored box, 17 SIFT detector features were found within its boundary based segmentation. The total number of detector features exceeds the preset threshold value of 10, resulting in a Full Function COG trait.

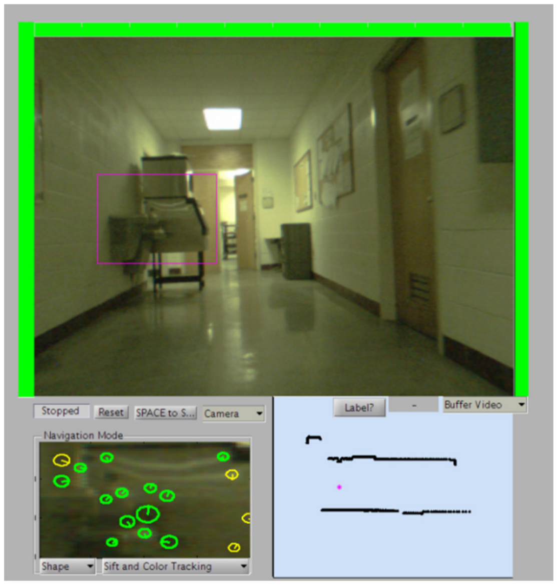


Figure 78: Dominant Function as seen through the RobiSEND GUI

As a result, this object may be considered a good candidate for object recognition. If it displays a dominant Uniqueness trait, would be characterized also as a beacon for navigation. This object model COG in fact has a recessive Uniqueness trait which limits its role in the RobiSEND navigation process.

Figure 79 displays the same relative scene with an observation highlighted in blue. This observation is of a door with a recessive Function trait. As seen at the bottom left corner of the figure, this object only produced 7 SIFT features. Therefore, it would not be stored for object recognition.

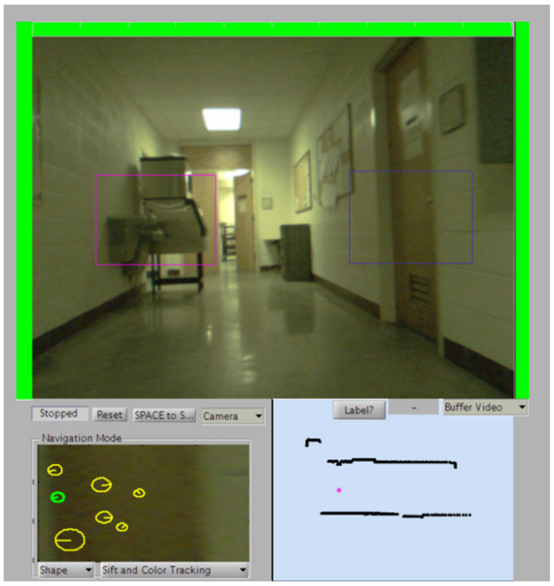


Figure 79: Recessive Function as seen through the RobiSEND GUI

As seen, the COG profile has a direct correlation to the functional role of an observation within the RobiSEND application of local search. The class trait of Function establishes how object models are stored and also how the operator ultimately interacts with object models within his FoV.

7.2.2 Uniqueness

Uniquess, as mentioned in Chapter 6, is the ability to extract object information from background. In continuing the examples provided from object models within the RobiSEND system, the Uniqueness trait is determined by the shape metric derived from disparity information. Figure 80 and Figure 81 give the visual information presented to the operator on the left and the underlying disparity and shape metric information on the right that illustrate these concepts.

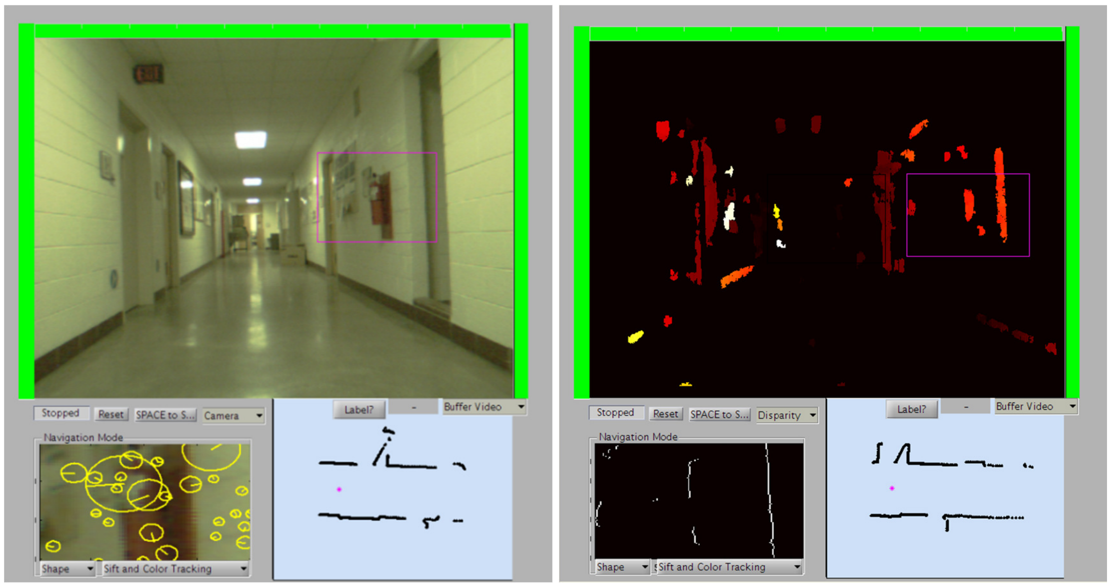


Figure 80: Dominant Uniqueness as seen through the RobiSEND GUI

As shown in the right GUI of Figure 80, a feature vector shape metric of positive shape has a length higher than 30 pixels. Chapter 6 explains that this observation is defined as having Dominant Uniqueness. The object is the fire hydrant highlighted by the magenta boundary based segmentation.

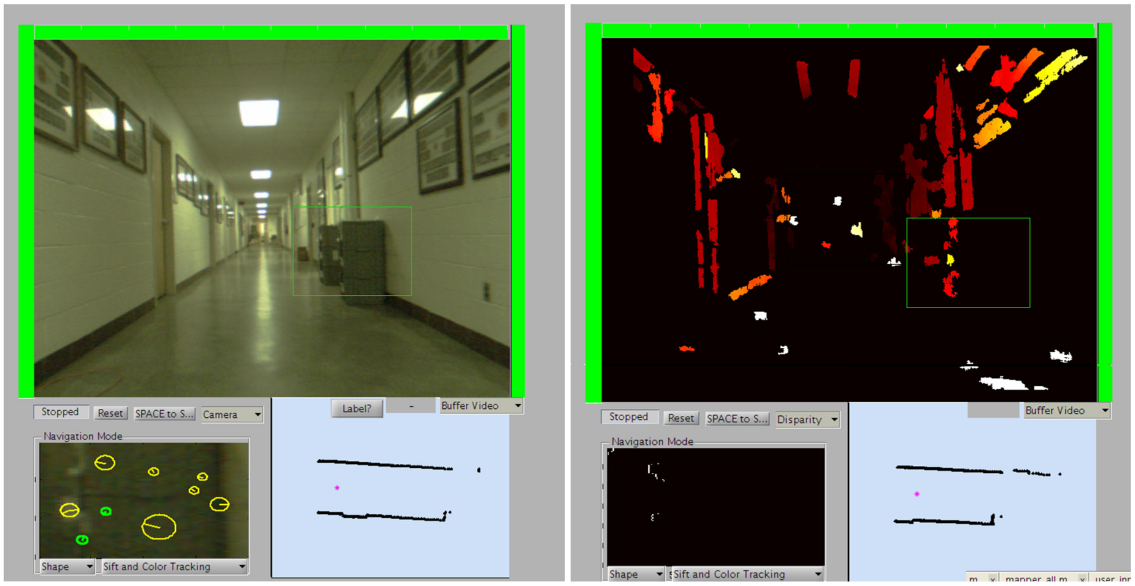


Figure 81: Recessive Uniqueness in the leftmost image as seen through the RobiSEND GUI with additional view of disparity information in the rightmost image

The object of Figure 81 is the recycling bin highlighted by the green boundary based segmentation. The shape metric shown in the right GUI does not meet the requirement for dominant Uniqueness because all of its vector shape metrics are less than 30 pixels in length. As a consequence, this object model is not a beacon for navigation. In addition, its recessive trait in Function due to only defining 9 SIFT features also rules it out for object recognition and tracking. The firehydrant in Figure 80, on the other hand, displays dominant Function and Uniqueness and therefore is a beacon for navigation.

7.2.3 Function and Uniqueness Combination

Based on examples given in Figure 80 and Figure 81, it could be assumed that observations possessing dominant function consequently will have dominant Uniqueness and vice versa. This is not the case as demonstrated in Figure 82. This figure compares the visual information of Figure 79 to its corresponding disparity information. The similar color boxes in both GUIs correspond to the same objects.

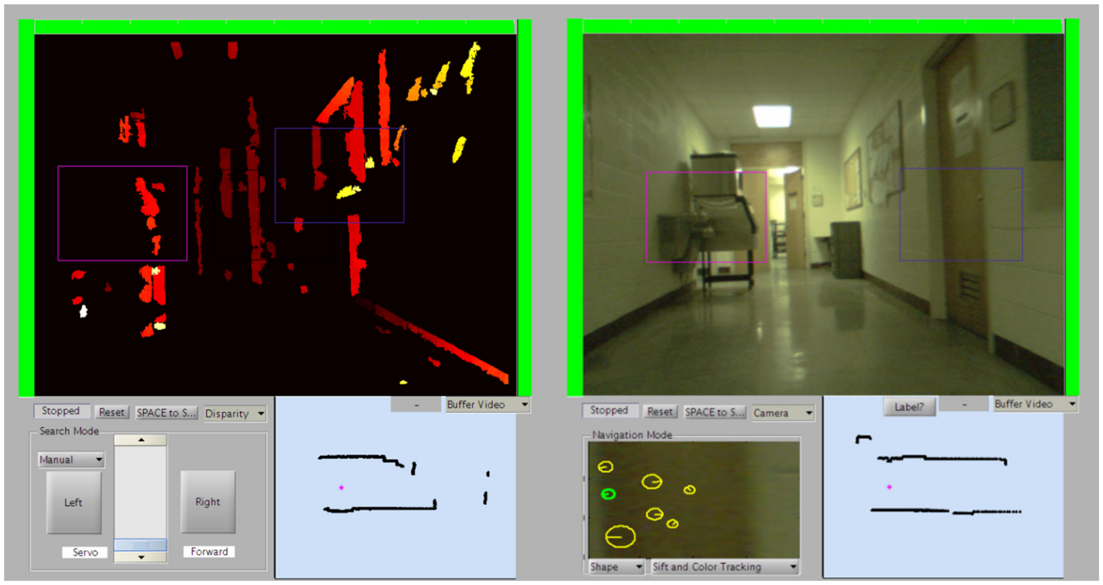


Figure 82: Objects with Dominant Function and Recessive Uniqueness and vice versa as seen though the RobiSEND GUI

As shown, these two objects are examples of an observation with dominant Function but recessive Uniqueness and vice versa. The reason for this in object models is that the vector feature shape metric is extracted largely from vertical edges in an image which require little feature definition outside of those edges. These edges result in a shape metric that meets the requirement for dominant Uniqueness. Likewise, the boundary based segmentation in the magenta box has definitive internal characteristics but weak edges. Therefore, the derived shape metric does not meet the qualification for dominant Uniqueness. As a result, the observation model of the magenta box is stored for recognition and tracking and the observation model is just used for navigation without being stored.

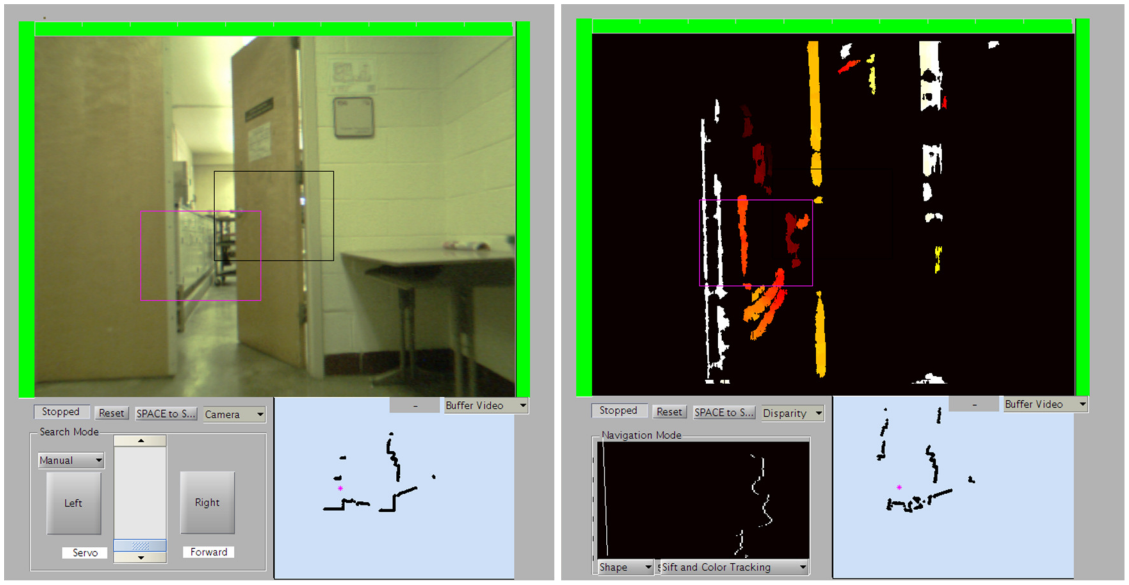


Figure 83: Object with Dominant Function and Uniqueness as seen through the RobiSEND GUI

Objects also may display recessive Function and recessive Uniqueness. One example of this is the observation model of a flat surface with little contrast. This condition is shown in the observation model of Figure 84. As shown, the number of SIFT features is 6 and there is no shape metric information at all, indicated by the orange output for the outer shape image. These object models have no functional significance and as a result should not be used for object recognition and tracking or beacons for navigation. These observations, as mentioned in Chapter 5, are used to correct the global mapper model. The distance information is leveraged to measure how far the RobiSEND payload is from the selected object model. The opposite condition of dominant Function and dominant Uniqueness is shown in the fire hydrant of Figure 80.

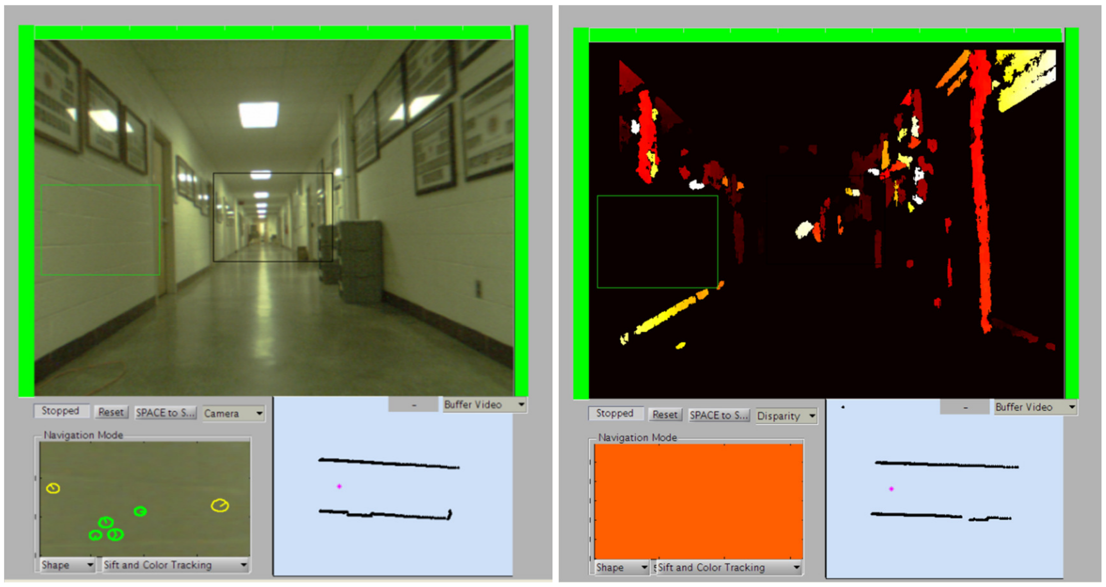


Figure 84: Object wih Recessive Function and Uniqueness as seen through the RobiSEND GUI

7.2.4 Importance

Upon selection, the observation class is determined from the Function and Uniqueness. These traits are supplemented by Importance and Maturity descriptor traits. First, importance is derived from the operator interaction with the GUI. For the RobiSEND system, as mentioned in Chapter 5, the operator has the option of right clicking or left clicking the mouse upon object selection. For object models, a dominant Importance determines whether the object model will be used to topologically annotate the QUICK mapper model. Objects with Normal Importance inherit their class from Function and Uniqueness, but are not displayed visually on the final QUICK map. Object models with Special importance, regardless of class, are displayed on the final map deliverable and uniquely labeled.

Figure 85 gives an example of an object model with dominant Function and Uniqueness but recessive Importance. As shown, there are several frames along the top portion of the walls in the image. The selected frame in Figure 85 is not a particularly unique object. Several frames are similar and so it is logical that the operator would not identify the frames as having dominant Importance. Because the specification of importance determines whether the object model information is used to annotate the final QUICK map, the operator must take great care to only label those objects that are of particular interest to the local search process. If this distinction is not made, then the objects shown on theQUICK map will not have relevance to the small tactical team. An observation model may be given Special Importance at any point it remains in the camera FoV.

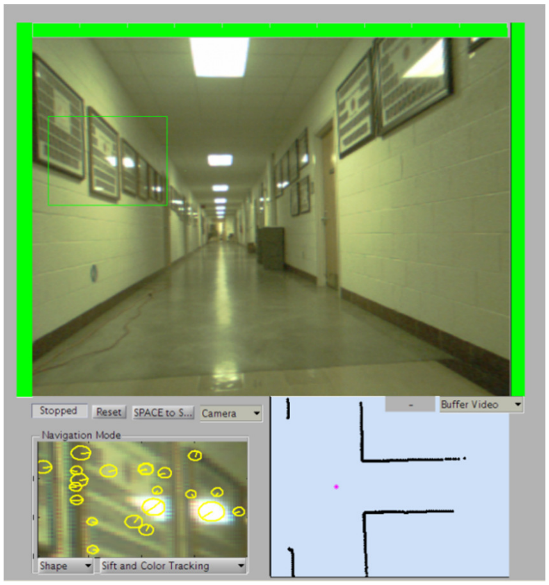


Figure 85: Recessive Importance as seen through the RobiSEND GUI

Dominant importance results when object models are identified through right clicking upon selection. The box in Figure 86 is an example. Its object model information is used to annotate the QUICK map.

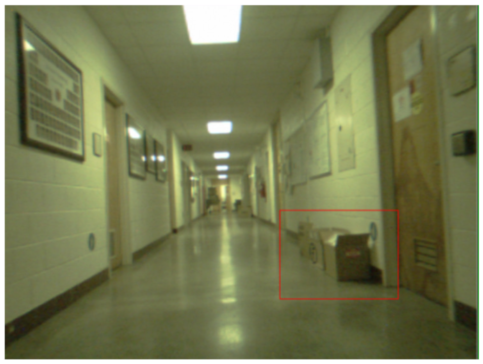


Figure 86: Dominant Importance as seen through the RobiSEND GUI

7.2.5 Maturity

Maturity also is a detail trait for an established class and may be altered at any point during object tracking. All observation models begin as having recessive or Juvenile Maturity. There are two ways in which an observation model gains Dominant or Permanent Maturity. It may either inherit Permanent Maturity over time or be given Permanent Maturity through correspondence. When an observation is selected that has been observed before, it automatically is given dominant Maturity. For RobiSEND object models, the recognition process is either done by a correspondence of features or a correspondence of object model labels. As mentioin in Chapter 5, while an object is being tracked, the operator has the option of labeling the current object.

If two objects have the same label and are spatially placed in the same vicinity, a correspondence occurs and that object is determined to have dominant Maturity. In addition, of an observation is made that results in a feature match, it is given dominant Maturity as well. Figure 87 and Figure 88 give examples for these two conditions. Figure 87 shows the editable box pop-up panel that functions as the means to label object models. Figure 88 shows an object feature correspondence with an object labeled 'BOX' from a previous iteration.

When the operator decides to label an object model, he clicks the label button at the bottom of the direct video feed. The label pop-up box appears and waits for a keyboard command. Once the keyboard command is entered and the operator hits the return key, the pop-up box disappears. This information is automatically stored within the framework of the object model. RobiSEND does not accept labels that are greater than 10 characters long. Names longer than this length are truncated at the 10th character.

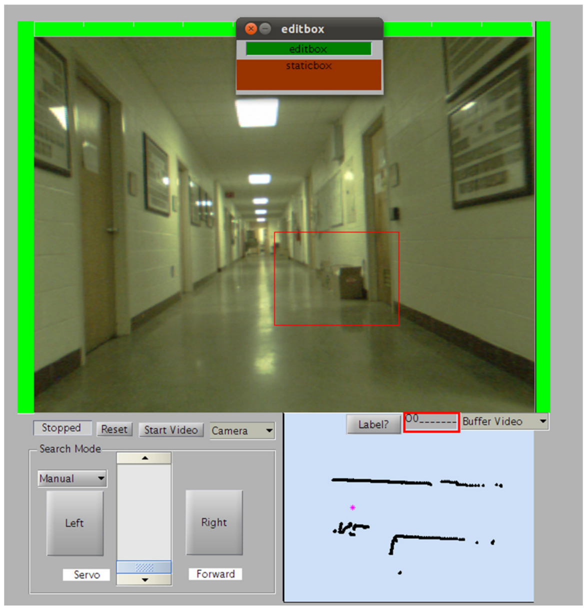


Figure 87: Dominant Maturity Label and Recognition as seen through the RobiSEND GUI. The object model labeling functionality is also displayed with the popup window shown at the top of the image

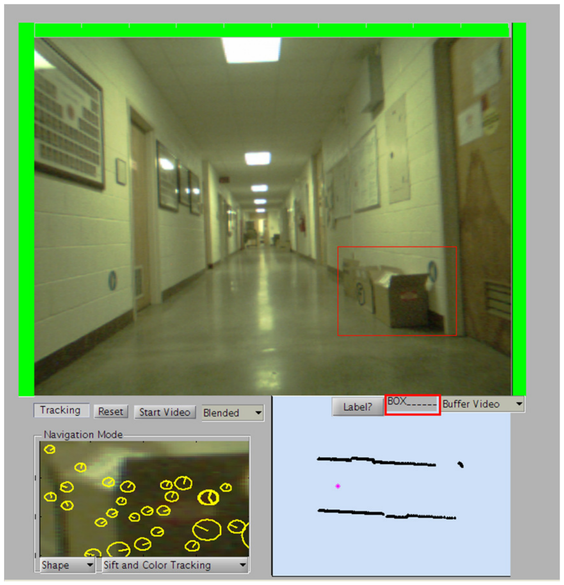


Figure 88: Dominant Maturity Recognition as seen through the RobiSEND GUI. The object model features have been recognized as a previous object model called "BOX". This is displayed to the user as shown with the red highlight in the image.

Dominant Maturity may also be inherited over time. Figure 89 gives an example of an object with initial Juvenile Maturity. For this example, the object in this figure is displayed immediately after operator selection.

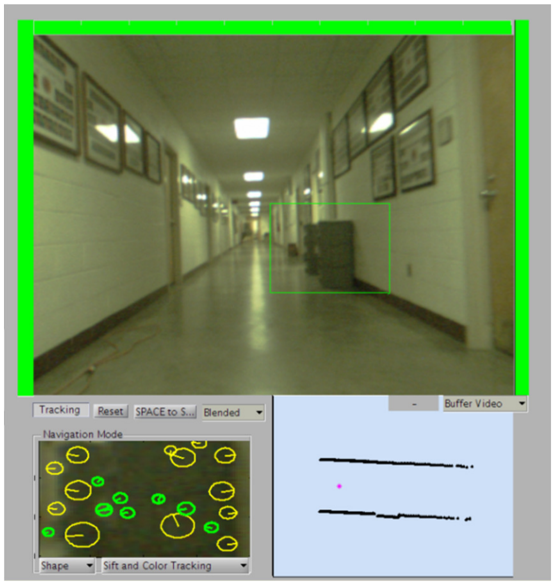


Figure 89: Recessive Maturity as shown in the RobiSEND GUI

Figure 90 demonstrates the maturation process. The observation is tracked for durations of time. Once an object is successfully tracked for 5 seconds, it is considered to have Permanent Maturity.

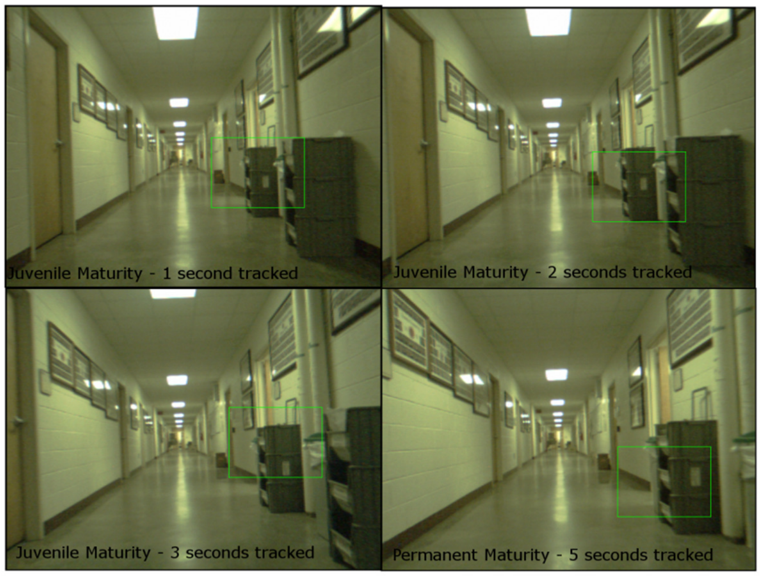

Juvenile Maturity - 1 second tracked
Juvenile Maturity - 2 seconds tracked
Juvenile Maturity - 3 seconds tracked
Permanent Maturity - 5 seconds tracked

Figure 90: Maturity Process over a 5 second span of object tracking

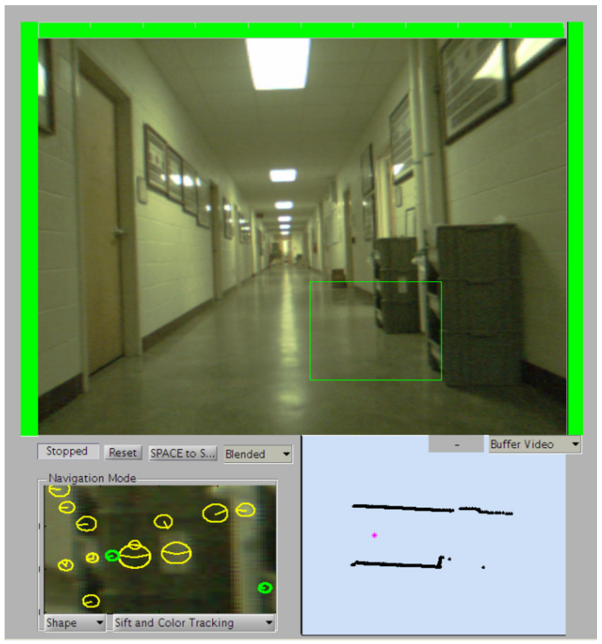


Figure 91: Dominant Maturity as seen through the RobiSEND GUI

Permanent Maturity governs the representative characteristics of the specific observation model class based on Qualitative Spatial Reasoning. For object models, this determines whether or not it may be used to influence other object models. If determined to be mature, the spatial information of the current observation model may be used to achieve a better estimate about previous or future observation models.

7.2.6 COG impact on storage and function

As discussed in Chapter 6, objects are stored, related to, and annotate the QUICK map based on their COG structure. For object models, dominant traits play a more direct role in the local search process. In general, memory is cheap; however, the amount of memory taken to store an observation adds up over time. These observation models are created to be eventually processed on board the MRUAV. As such, only those objects that require full storage are saved as such. As discussed in Chapter 5, all processing is done to captured video information that is streamed to the operator at a rate of 10Hz. Object models with all dominant traits have the highest storage capacity. This processing is done with a Pentium

As a general rule of thumb, objects with recessive traits require less computation and objects with more dominant traits are more useful for the local search process. One example is the fire hydrant of Figure 88. Because the box has been labeled and recognized from a previous instance in time, it inherits a Permanent Maturity. As a result, it is identified as having high priority for the AOI map update stage. The object has Explicit Uniqueness and Full function from observation of its vector feature shape metric and number of SIFT features, respectively. Its SIFT features upon selection totaled above the threshold value of 10 and its shape metric has an uninterrupted length of over 20 pixles. Therefore, the box object model is identified as useful for object tracking and object recognition from the dominant Function trait and participation as a beacon for the navigation process because of its dominant Uniqueness trait. Lastly, assuming the operator right clicked the object upon selection to indicate it is important to the local search process,the box object model has Special Importance, which is also a dominant trait. Because it has been labeled as such, all of the object model information necessary to store the box for QUICK map annotation is also saved. The entire COG object model would be ESFP.

To make an analogy to the amount of computation for each type of object model, we will use the amount of storage space for each object model for an example. The amount of information needed to store the box object model is determined from combining the necessary information for the functionality associated with each dominant trait. The object template image and object background image at each frame requires 14,400 and 57,600 bytes of data for object tracking and QUICK map annotation, respectively. Because 19 SIFT features were found upon object selection, an additional 76 bytes of data are required to store the feature number, scale, size, and rotation for the SIFT detector and 2,432 bytes are needed to store SIFT feature descriptors. The disparity template and disparity background images add an additional 24,000 bytes of information to the model. Assuming there are 10 MSER features identified, 40 bytes of data are required for their storage. For outer shape and inner shape of an object, 240 bytes of data are needed for shape feature vectors.

During the maturation process, the object model template stored 2 other intermediate representations of the box object model. Template images and background images are only stored once, but feature information is stored for each of these intermediate representations for object recognition. Therefore, SIFT ultimately requires 7,534 features and disparity based features add 840 bytes of additional information. As a result, 72,000 bytes of data are needed to be stored from the color pixel information, 19,200 bytes of disparity information, and 8,424 bytes from the feature information for a total of 99,624 bytes for an object with all dominant characteristics.

If Maturity is the recessive Jivenile trait, the COG representation would then be ESFJ. The feature vector information is reduced because no intermediate representations of the object exist. This is the case when the object model is first selected at the first instance in time. For simplicity, the assumption is made that the number of SIFT features and MSER features is consistent with the box object model. Therefore, for recessive Maturity, the total bytes needed for storage is reduced to 94,816. This translates to approximately a 4% reduction in storage requirement if feature information need not be stored. As shown, storing features does not cost a whole lot, therefore, the storage cost for additional objects for recognition is small. This points to the inherent advantage to emphasizing the object recognition problem over the correspondence problem. Features must be stored, however so there is a storage cost allocated

If Importance is determined to be Normal and therefore recessive, a more dramatic reduction in storage requirement occurs. Because the object model is not used to annotate the QUICK map, there is no need to store the object background image, reducing the object box model storage requirement to 37,216 bytes if the object model is ENFJ and 42,024 bytes if it is ENFP. The difference between ENFP and ENFJ object model COGs is in the Maturity trait.

For the class traits of Function and Uniqueness, a difference in storage capacity is also worth addressing. An object model, such as the object box model has dominant Function and Uniqueness. If an object model, such as one derived from the conditions of Figure 82, is analyzed, the storage requirement may be further reduced. The object highlighted in blue with a recessive Temporary Function trait does not require the storage of SIFT features or the object template image, because it will not be used for oject recognition or object tracking. Assuming all other traits are dominant for a COG of

ESTP, the storage requirement for this object would be approximately 82,716 bytes. The object highlighted in magenta with recessive Uniqueness would not require the disparity template image to be stored and no MSER features derived from this template. As a result, 80,424 bytes of data would be needed assuming all other traits to be dominant for a ISFP COG model.

Objects with recessive Uniqueness and recessive Function do not store any stereo camera based information. Instead, the 300 LIDAR points that overlap with the camera FoV are captured at the relative height of the object. The raw LIDAR scan is then displayed on the map as if a CornerJunction were encountered during the building of mapper models. For RobiSEND, LIDAR points are captured at the correct height by issuing an angular change command to the servo tilt platform to adjust the LIDAR scan plane accordingly. If the operator selects an object high in the camera FoV, then the LIDAR level is adjusted upword more drastically than if an object were to be selected from the middle of the FoV. If RobiSEND were to be implemented on an actual MRUAV, the MRUAV height could be adjusted to match that of the selected object. If Importance and Maturity are also recessive, this object has a COG of INTJ.

Dominant Importance and dominant Maturity may slightly add to the storage requirements for this type of object model. The only manner in which an object model with recessive Function and recessive Uniqueness may achieve Dominant maturity is if the operator names it identically to a previous object model with these characteristics. Therefore, it may only inherit maturity if object recognition is achieved through the object labeling process. If this happens, the COG model for both matched object models is INTP, the average distance of the LIDAR information is calculated for each scan, and their global object model locations are given priority during the AOI map optimization stage. Dominant importance may be achieved if the operator choses to right click an object that qualifies, producing an ISTJ. Important objects with dually recessive class traits must capture object background images for map annotation. Therefore, 57,900 bytes of information are needed to be stored.

7.3 Mapper Models

Results from the mapper model creation process are presented in Section 7.3. Three scenarios were fabricated to emphasize the impact of Operator Blending on the mapping process. These scenarios are important to the evaluation of Operator Blending as it relates to mapper model generation because they are used to demonstrate the necessity of each component to mapper model building under the Operator Blending framework. Because QUICK mapping is the only qualitative mapper found in the literature, comparison of scenarios is necessary to communicate the merits of the approach.

7.3.1 Mapper Model Setup

Mapper model COGs also specify their relationships in the local search framework. As mentioned previously, mapper models with more dominant traits are more metric in their characteristics, while objects with recessive traits are more topological. Specific mapper models for the different combinations of traits were discussed in Section 5.9 on QSR and the mapper module. This section discusses how the different mapper models are combined to produce QUICK maps and the general structure of these maps. Examples of QUICK maps are given to demonstrate these concepts.

To apply mapper model techniques discussed in this research, the RobiSEND system was implemented to verify three capabilities: object model integration within the map, operator centered navigation, and QUICK map building. Each capability is isolated through three scenarios that involve varying degrees of operator involvement in local search. Scenario 1 only produces maps based on observed class traits of Function and Uniqueness over time. Scenario 2 adds the Importance trait for topological annotation and Scenario 3 adds the Maturity trait for map optimization. The deliverables produced from Scenario 3 are indicative of the full concept of Operator Blending for local search applications. Because there are no other qualitative mapping techniques in the literatre, comparison is primarily done between these three Scenarios. For each scenario, the RobiSEND system is attached to a cart and moved at 1m/s in the unstructured environment at a constant elevation of 1.5m.

Although it is difficult to make the comparison, effort was made to compare a popular fully metric approach to mapping. To compare QUICK mapping to metric approaches, feature based scan matching is done based on split-and-merge technique as mentioned in Section 4.3.2. This approach is executed through the application of open source software executing a popular approach to occupancy grids called VASCO within the Carnegie Mellon Robot Navigation Toolkit (CARMEN). In addition, a topological map was created based on the saved data for QUICK map building.

Scenario 4 and scenario 5 are needed in order to compare the results of QUICK mapping to the current approaches. CARMEN was chosen because of its ability to produce a purely metric environmental representation through scan-matching with minimal other computational overhead. The map generation process is not done in real-time and there is no Rao-Blackwellized partical filter used to optimize the map. Although it is possible to apply approaches like GMapping (Grisetti, Stachniss, & Burgard, 2007) or GridSLAM (Haehnel, Fox, Burgard, & Thrun) that take advantage of this partical filter, implementation of this type of mapping approach would have required too much computation. No open source implementation of vision based mapping techniques was found that was capable of processing information at or near 10Hz and without extensive post processing. Therefore, metric map comparisons were limited to this LIDAR based technique.

Data is collected on the first floor of Randolph Hall on Virginia Tech campus. The ground truth of this area is shown in Figure 92. The building has a total of four floors with a gross square footage of 165,918 ft^2. The building is not amenable to GPS or magnetometer sensors. Neither produce accurate readings, making a system like RobiSEND based on local information and independent of these types of global sensors necessary.

In Figure 92, all hallways and open spaces are shown outlined in white, with side rooms and spaces shown in blue. The space in which the theoretical MRUAV is assumed to occupy during the search consists of the highlighted wihite area in the image.

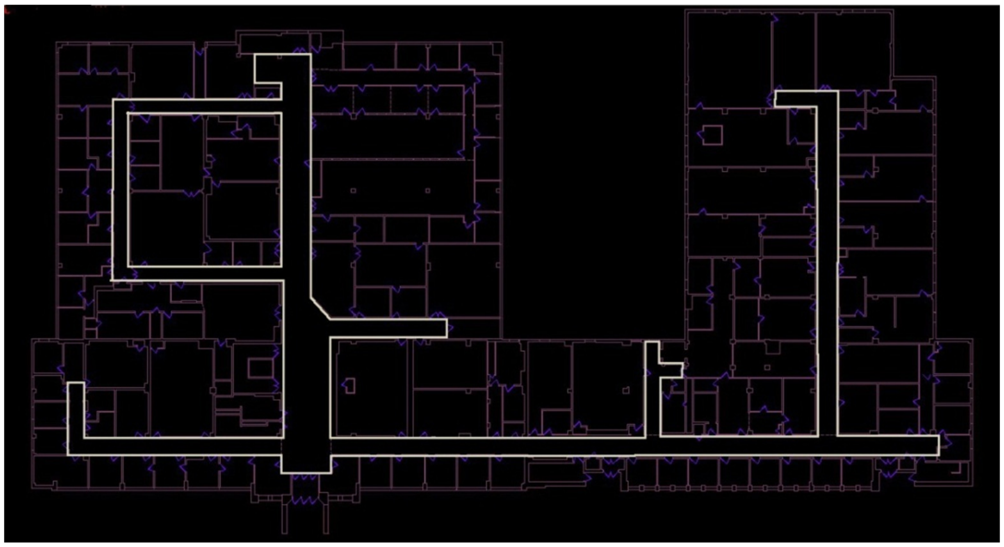


Figure 92: Floor plan of Randolph Hall first floor

Implementations of scenario 1-3 are based on the QUICK mapping framework developed in this research. They are implemented in Matlab and take advantage of the Image Processing and GUIDE toolkits. For visual feature extraction the VLFEAT library (Vedaldi & Fulkerson, 2008) is used to generate MSER and SIFT features. All Matlab code is given in Appendix B. Because there are no qualitative components in scenario 4, its implementation is through the traditional CARMEN structure published and widely used. Metric maps are generated through open source Qt graphical capabilities in Linux, the Carmen data logger, which is also open source, and the VASCO Carmen functions published under the GNU General Public License. Scenario 5 is a representation that is purely topological and based on the operator feedback given during the local search process.

7.3.2 Scenario 1-3 in detail

Data is captured at 10Hz and stored for processing under the conditions subsequently presented for each scenario. This was done so the deliverables from each scenario result from extracting observation models from the same stored video and LIDAR data. Even when the exact same frames of color video, disparity video, and LIDAR scans are input into the mapper model generation process, how these algorithms extract features is dependent on how k-means clusters are initiated and how cluster membership is assigned. Therefore, as mentioned in the Chapter 5 explanation of qualitative maps, because cluster membership is initialized from finding random seed points in the LIDAR data that are grown based on the clustering algorithm, it is highly unlikely that the analyzer and accumulator module process a scan in the exact same way in subsequent processing of the exact same raw sensor data. If the width of a Hallway mapper model is close to the threshold value for instance, the width of the Happer mapper model may not be calculated to be the same in two subsequent QUICK mapping processes of the same data. Figure demonstrates this.

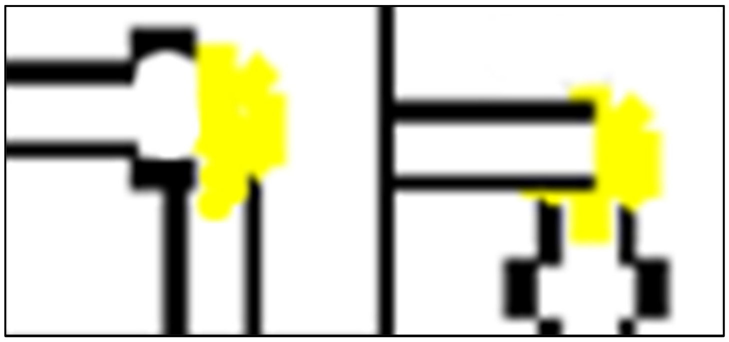


Figure 93: Mapper module variation based on the random initialization of k-means clusters

As shown, a wide Hallway mapper model is shown before the Junction on one trial and after the Junction on the other. These were generated from the same data. The benefit to this is that the random process prevents problematic cluster associations from a particular scan to impacting the associations of a subsequent frame. The impact this has on QUICK maps, however, makes each QUICK map slightly unique based on how these associations are initiated.

For the first scenario, an additional operator is assumed to provide translational motion and heading commands to the MRUAV along the search path. Navigation is not controlled by selecting objects, but object and mapper models are built from the Operator Blending framework. The RobiSEND system propagates along the search path without object selection leading to navigation as the shared basis for communication. As a result, the basic structure of the environment, as portrayed through qualitative mapping techniques, is shown without topological annotation. Typically, these maps would be processed with the AOI techniques of Chapter 6, object template images would be given to the operator at the conclusion of the search, and object model locations would be conveyed to the system operator. These maps are built to contrast with topologically annotated QUICK maps developed in the following sections.

For the second scenario, the QUICK map is built in the exact same manner of Scenario 1, but topological annotation is included in the final map. The specification of important and unimportant objects leads to determining which objects are stored in the RobiSEND memory and used to convey topological information. The label function is disabled because of the lack of control the operator has over the navigation process. The operator acts in the role of payload operator with no control over the translational motion of the theoretical MRUAV. Once again, these commands are assumed to be issued by an additional operator in the loop. At the conclusion of the 10 minute period, the RobiSEND payload has spatially covered the entire search area with observation models developed under operator blending to qualitatively convey the metric and topological characteristics of the searched area.

Scenario 3 controls the playback of the stored video to simulate navigation as a shared basis for MRUAV propagation in the environment. This scenario fully takes advantage of object model COG as discussed in Chapter 6 and require switching from shared mode and safe mode for navigation. In Shared mode, COGs are not just stored for topological annotation as in scenario 1 and scenario 2. They control the playback of the video and LIDAR data to the operator through the GUI. When the operator selects objects in the environment with certain characteristics, the video plays based on LIDAR and visual information to simulate flying towards these objects. The operator has control over how often observations are made and may take additional time in between observations for object model labeling. The operator may also switch the system to a Safe mode simulation where he is able to command the MRUAV to move forward in the environment.

The operator is assumed to have 15 minutes to complete the local search process. At its conclusion, the operator is assumed to consider the same objects important in the environment as scenario 2, but has no restriction on the number of unimportant objects. Unimportant objects are much more important because they are the means of propagation within the simulation. The purpose of scenario 2 is to represent the RobiSEND payload through the implementation of all the concepts presented in this research. It is used in comparison to the map generated in scenario 1 to observe the merits of navigation as a shared basis.

The fourth scenario produces a metric map based on traditional approaches to scan matching and particle filters. It is difficult to compare traditional approaches to those developed in this research because no approach was found that operates in Matlab and maps in real time for comparison or emphasizes the object recognition problem over the correspondence problem. The purpose of the comparison to GridSLAM, which is a C++ based quantitative mapper, is to show the merits of current approaches to produce maps with the hardware of the RobiSEND system. It was necessary to verify that there are approaches that map effectively in real time and are capable of mapping with the assumptions made by the RobiSEND design for payload components. Although the performance of GridSLAM is given in the documentation provided by its creators as being best implemented by LIDAR sensor with more range and accuracy than the LIDAR used in this research, it provided a comparison of popular techniques. Because the approach and computational methods are so drastically different between scenario 4 and scenarios 1-3, the comparison that is made is purely one based on the subjective analysis of the author and may not reflect the optimal conditions of a GridSLAM implementation.

Maps are built in real time as this information is received in accordance with the procedure presented in Chapter 5 and Chapter 6. All QUICK maps are built in the exact same manner with the difference between them the degree of operator involvement in the local search process. These QUICK maps are then discussed and compared to traditional approaches using the GridSLAM generated map as an example. Situational awareness is discussed relative to the degree in which it conforms for each approach to the components of situational awareness for MRUAVs as presented in Chapter 2. These results do not analyze the cognitive load of the operator as these are human factors based problems and outside of the scope of current research. This, however, is an interesting problem and is presented in this Chapter as a future research opportunity.

7.4 Mapper Scenarios

Section 7.4 presents each individual scenario in detail with results from mapping an unknown environment. Sub-section 1 gives the output from scenario 1. Sub-section 2 displays the output from mapping information under scenario 2. Sub-section 3 presents the output from scenario 3. The scenario 3 output is the result under conditions that involve all aspects of Operator Blending.

7.4.1 Automatic Map Generation—Scenario 1

Scenario 1 produces a map that contains the combination of mapper models without the inclusion of object models for topological annotation or map correction. As shown, there is no topological information added to the map in these cases except for the spatial annotation given at 90^{o}

heading changes for CornerJunction mapper models. Figure 94 shows the full QUICK map for the first floor or Randolph Hall.

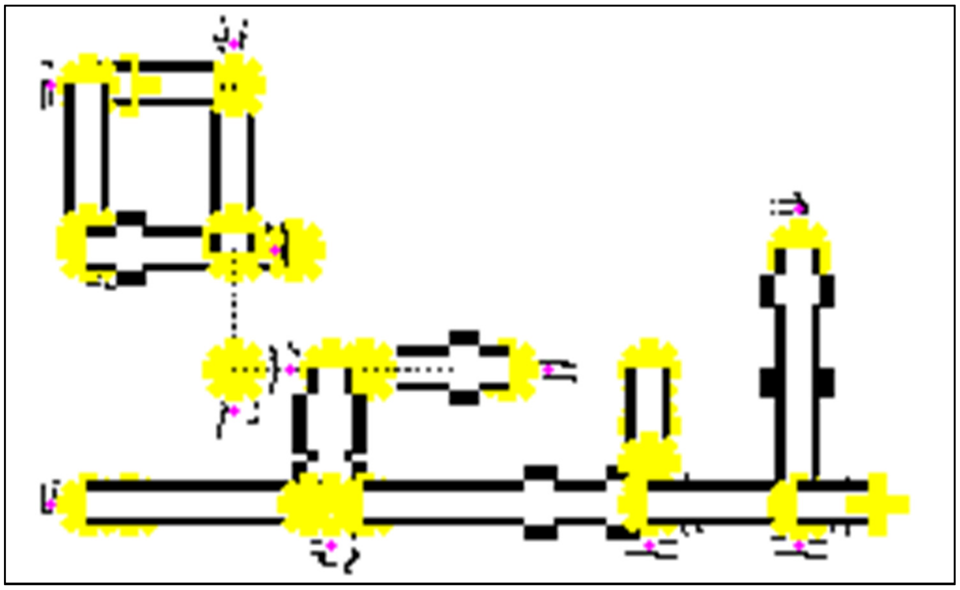


Figure 94: Full QUICK map from Scenario 1

Object recognition can be the basis for a fairly useful map when the operator is not in the loop for full Operator Blending. As the payload was taken throughout the environment, hallways, junctions, and space were idenfified by the mapper. Because the servo angle was not varied during this mapping process to avoid areas with substantial clutter, there are places along the map that show inconsistent width information. The RobiSEND payload was assumed to be at a constant angle throughout.

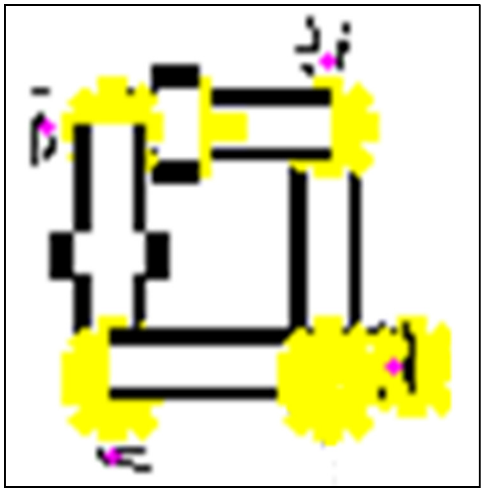


Figure 95: Sub-area 1 QUICK map from Scenario 1

Figure 95 gives another perspective on the cyclic environment at the beginning of the local search process for Scenario 1. At the end of the cycle, the junction in the bottom right corner appears wider than the other three junctions. This is a result of the POI process. The initial estimate for the location of the junction once the loop was completed was short of the actual location. The mapper was able to recognize this inconsistency and shift the junction location to the right for map feature matching.

Figure 96 gives the QUICK map for the second sub-area. Sub-area 2 introduces the ContentSpace component to the map. As RobiSEND moves to an open area, the path of the payload is highlighted by the black dotted line. The actual structure of this area is portrayed by CornerJunctions.

At the right, POI shifts hallway location downward after traversing the vertical hallway. At the end of the search process, a PartialJunction is displayed to demonstrate that at the time the QUICK map was printed, the junction at the end had not yet reached dominant maturity. The QUICK map without POI optimization is shown in Figure 97.

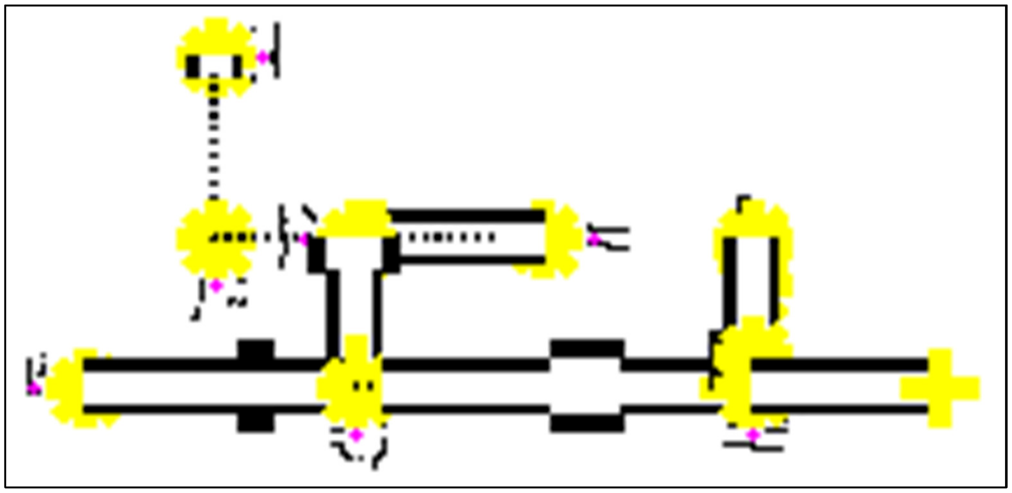


Figure 96: Sub-area 2 QUICK map from Scenario 2 with POI correction

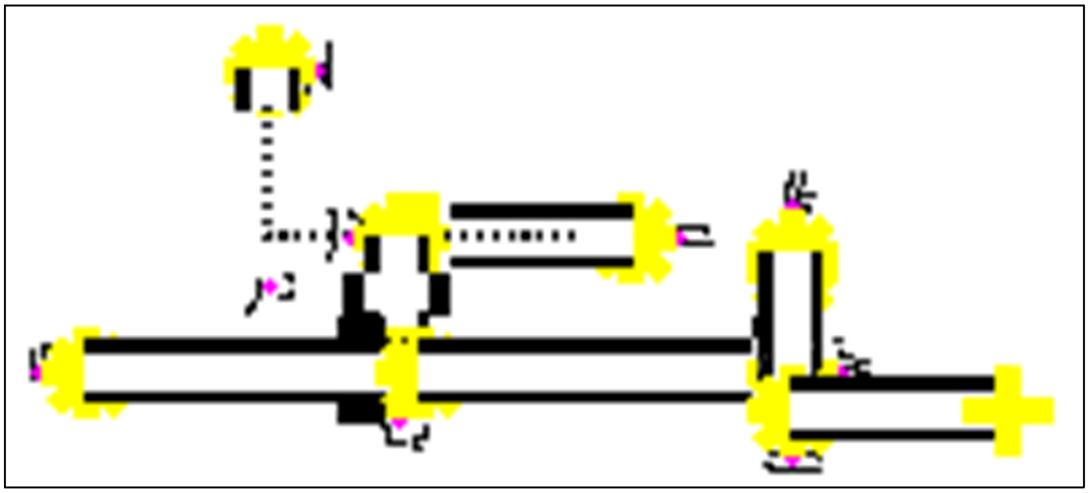


Figure 97: Sub-area 2 QUICK map from Scenario 2 without POI correction

As shown, this results in an inconsistency along an otherwise straight hallway at the right of the figure. The last turn along the search path occurred after it should have. Therefore, the hallway is not straight. Once POI is turned on, this junction is shifted upwards to correspond to the previous junction.

Sub-area 3 is shown in Figure 98. This is the shortest of the 3 areas. Once again inconsistency is present during the qualitative spatial reasoning that occurs to find the width of the hallways. This area does correctly indicate two alternate Partial Junctions along its long hallway. In addition, another Partial Junction is shown at the end of the search path on the far right. This is another junction that has not yet had the opportunity to become mature.

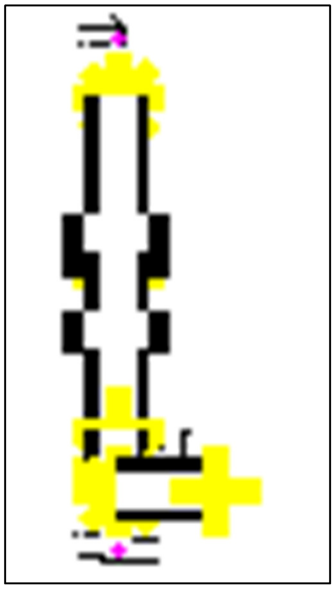


Figure 98: Sub-area 3 QUICK map from Scenario 1

The maps produced by scenario 1 basically omit the Importance trait and topological information because there is no operator in the loop. The RobiSEND payload still constructs mapper models based on perceptual information.

It should be noted that these maps are also not produced in a OHOR scenario. The robot is autonomous along the search path in this simulation. This condition is not adequate in real life scenarios. Therefore, scenario 1 produces maps to demonstrate the perceptual capabilities of the RobiSEND system, but it does not consider the integration challenges.

7.4.2 Payload Controller—Scenario 2

Scenario 2 involves allowing the operator to act in the role of pure payload control. As discussed in Chapter 2, this is the traditional structure of human-robot collaboration with MRUAVs in teaming situations. The object models built under this constraint are presented in the following sections. As mentioned, the label command is disabled for this scenario, because the operator has no control over the translational motion of the MRUAV.

Object models are stored for map annotation and map correction. The object model constituent parts are not made visible to the operator beyond the appearance of the current template image and SIFT features shown in the bottom left hand corner of the GUI. Because the payload operator would not be concerned with object avoidance or possible paths around the theoretical MRUAV for navigation, the LIDAR data is not presented during this process.

In addition, because the label function is disabled, AOI is also disabled for map correction. The functionality of the AOI optimization relies on the ability to recognize the existence of identically labeled objects in the QUICK map. Without the ability to label objects, map correction is limited to POI optimization as in Scenario 1.

The object models generated under the second scenario and their attributes are presented in Table 11. The map was divided into three sub-areas before the start of the local search process. These are labeled as 'start loop', 'statue hallway', and 'long hallway' respectively. Object models, because specific names were not given to them through the object labeling process, are labeled 0-9 on the QUICK map. The object models displayed in Table 11 are from the first sub-area of the map 'start loop' which is of a cyclic environment.

Table 11: Examples of Object models built under Scenario 2

Observation	Uniqueness	Importance	Function	Maturity	COG	Match
#0	1	1	1	0	ESFJ	-
#1	0	0	1	0	INFJ	-
#2	1	0	1	0	ENFJ	-
#3	1	0	1	0	ENFJ	-
#4	0	0	0	0	INTJ	-
#5	1	1	1	0	ESFJ	-
#6	1	1	1	1	ESFP	2
#7	0	0	1	0	INFJ	-
#8	0	1	1	1	ISFP	-
#9	0	0	0	0	INTJ	-

Each object is characterized by the 4 COG traits of operator blending. Object 9 is the only observation that is developed for global map annotation because of recessive Function and Uniqueness.

Additionally to the COG profile for each object, the object template image, capture time, object distance and location from the camera, and object features were stored for each object. This capture time, object distance, and object location from the camera were used for object model placement within the QUICK map. Object features were used for object recognition for the map update AOI portion of the mapping process. These attributes are used for map correction during the AOI portion of QUICK mapping in Scenario 3. Without AOI correction, object models may provide inconsistent information.

Because the operator identifies object models in Scenario 2, the QUICK mapper annotates the map to indicate the relationship of the object to the RobiSEND payload. Figure 98 gives an example of the types of indicators printed on the map.

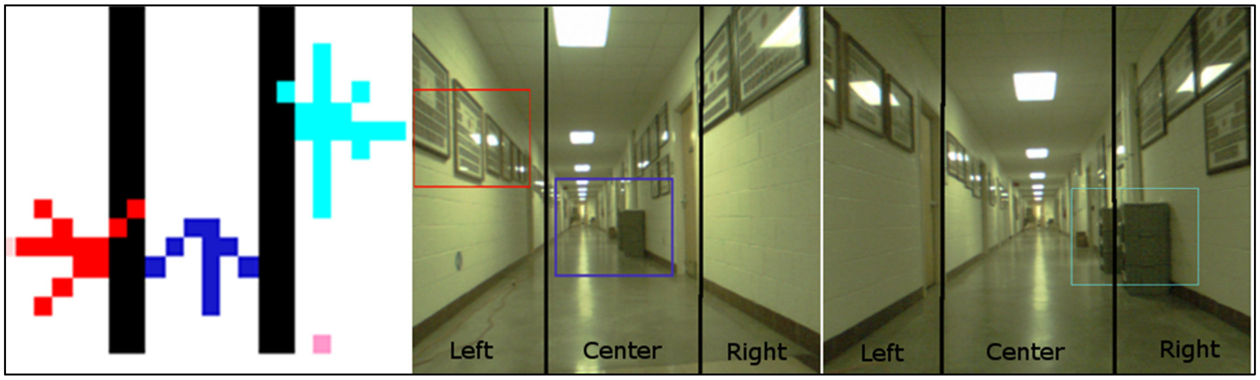


Figure 99: Indicator type map for QUICK map annotation. Center image is of an instant in time before image on the right. The blue object is in the background and so the indicator is an arrow

The red and blue indicator were selected in rapid succession during the local search process. As a result, their placement is during the mapper module output of the same Hallway inference. The red indicator is on the left because the object model it represents was selected from the left side of the FoV of the stereo camera. The blue indicator is an arrow, indicating that although the object model was selected at the same time as the red object model, the actual blue object model location is placed ahead of the selection in the hallway. The cyan indicator is located at a later time and was selected from the right portion of the camera FoV.

Figure 100 displays the QUICK map of the first sub-area generated from scenario 2. Because specific names are not given to object models, the map optimization stage is limited to POI optimization.

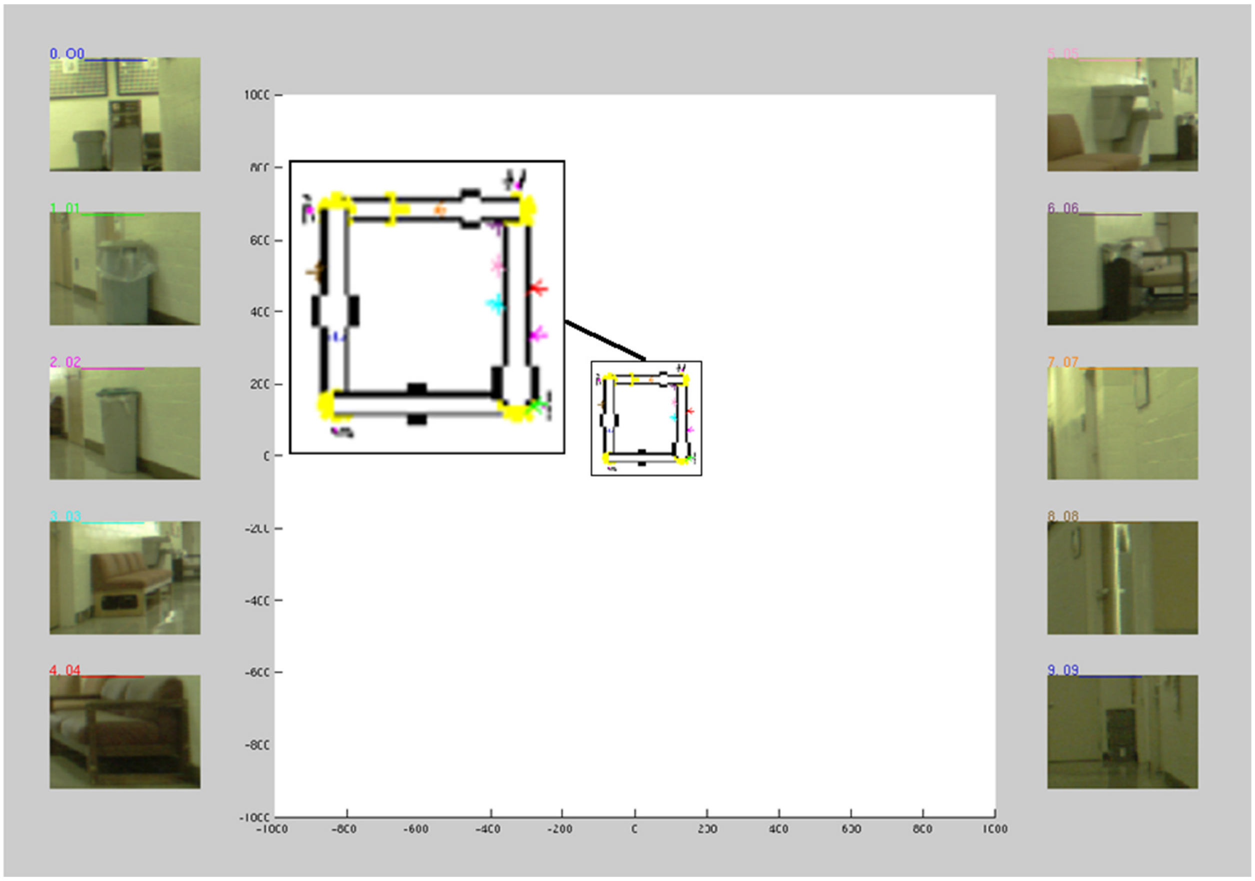


Figure 100: Sub-area 1 labeled in the QUICK framework from Scenario 2 with inference lengths exaggerated

Because the searched area covered in Sub-area 1 is small, the mapper module was adjusted to output longer segments. Therefore, the perceptual units of the area displayed in Figure 100 are twice as large as the perceptual units of Figure 95. This is one distinct advantage to qualitative maps. The relationships displayed between the width and length of segments may be easily adjusted for display purposes. The rest of the QUICK maps discussed in this work will reflect the original mapper model lengths.

The next figure is produced of sub-area 2 as the RobiSEND payload finishes mapping the specified area. Object model annotation is also shown on this QUICK map output for those objects with Dominant Function and Importance.

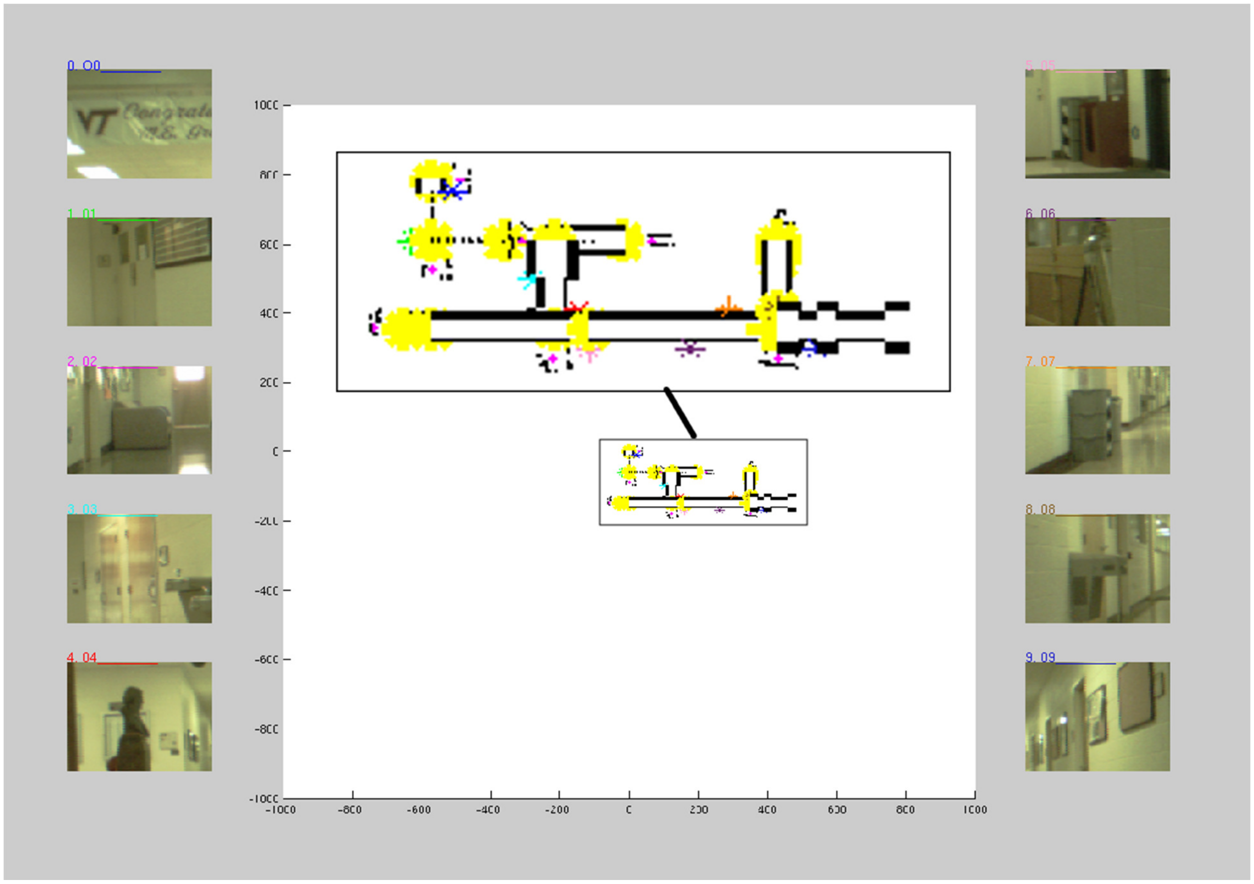


Figure 101: Sub-area 2 within the QUICK framework from Scenario 2

The final sub-area after annotation is given in Figure 102. Because this area did not cover as much distance as the previous areas, the number of selected object models with dominant Function and Importance was only 6 during the local search process.

Because the operator concentrates on the task of payload operation which is independent of MRUAV navigation, the frequency of object selection is impacted. More objects are labeled with less topological annotation and less significance to the local search process. In order to map efficiently, some form of communication is needed between the payload operator and the agent in charge of

MRUAV translational motion. Otherwise, the operator must only label those objects he is given enough time to observe sufficiently. Although a human factors based study would be needed to measure exactly how much time and to what degree communication impacts this process, the general observation that adding detail to the map without this communication is difficult was observed in this research.

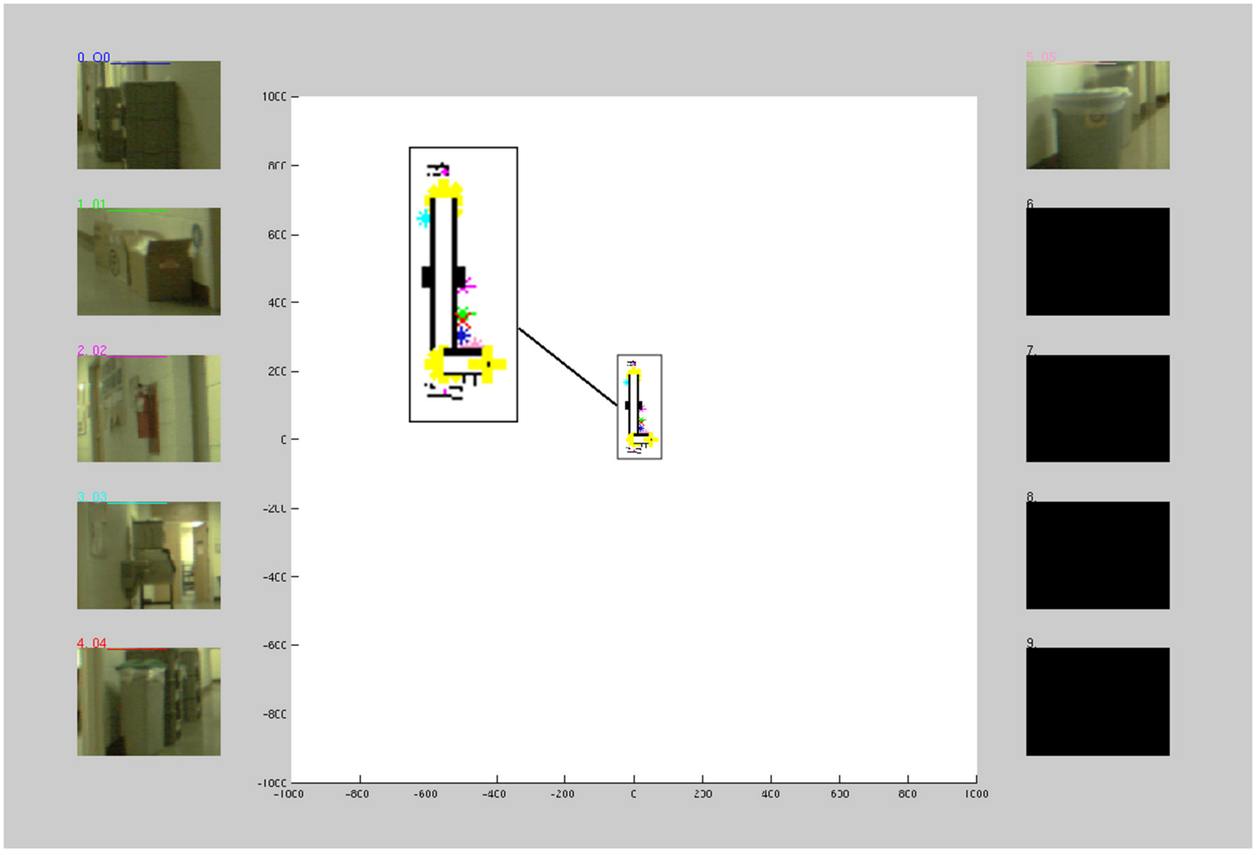


Figure 102: Sub-area 3 Labeled though the QUICK mapping framework for Scenario 2

7.4.3 Full Operator Blending—Scenario 3

Unlike Scenario 1 and Scenario 2, Scenario 3 involves the inclusion of the maturity trait and therefore is a complete application of the theory of Operator Blending. AOI map optimization is enabled along with the ability to topologically annotate mapper models during the local search process. In this scenario, navigation is not controlled by an outside entity and is the shared basis for communicatioin. The operator selects objects that conform with the mission at hand or choses points of departure for navigation. In between object selections, time exists to topologically annotate the map. The selection of objects leads to the translational motion of the theoretical MRUAV and combines this with operator directed attention for observation model generation.

Scenario 2 may be compared to Scenario 3 to observe the impact of navigation as a shared basis. The integration and perception differences between maps generated from each respective scenario are discussed. It is acknowledged that integration challenges are difficult to quantify without a human factors based study to verify results. The perceptual limitations, however, are addressed in a comparison of the methods between the two scenarios. The number of object models and their placement in the final QUICK map is fundamentally impacted by the means of communication.

The 10 minute video captured under scenario 2 was retransmitted to the operator under scenario 3. Based on the objects the operator selects from the direct video feed, the video playback is controlled. Selected objects are "flown towards" by playing the video forward until the disparity distance from the object to the operator is calculated to be 5 meters. Because the navigation process is driven by the ability for the operator to make observations, the search process takes more time in scenario 1. RobiSEND is designed for 15 minute local search. Therefore, the allotted time of 15 minutes was given for the search process in scenario 1. The operator interface for Scenario 1 also periodically displays the current QUICK map to the operator. This GUI is shown in Figure 103.

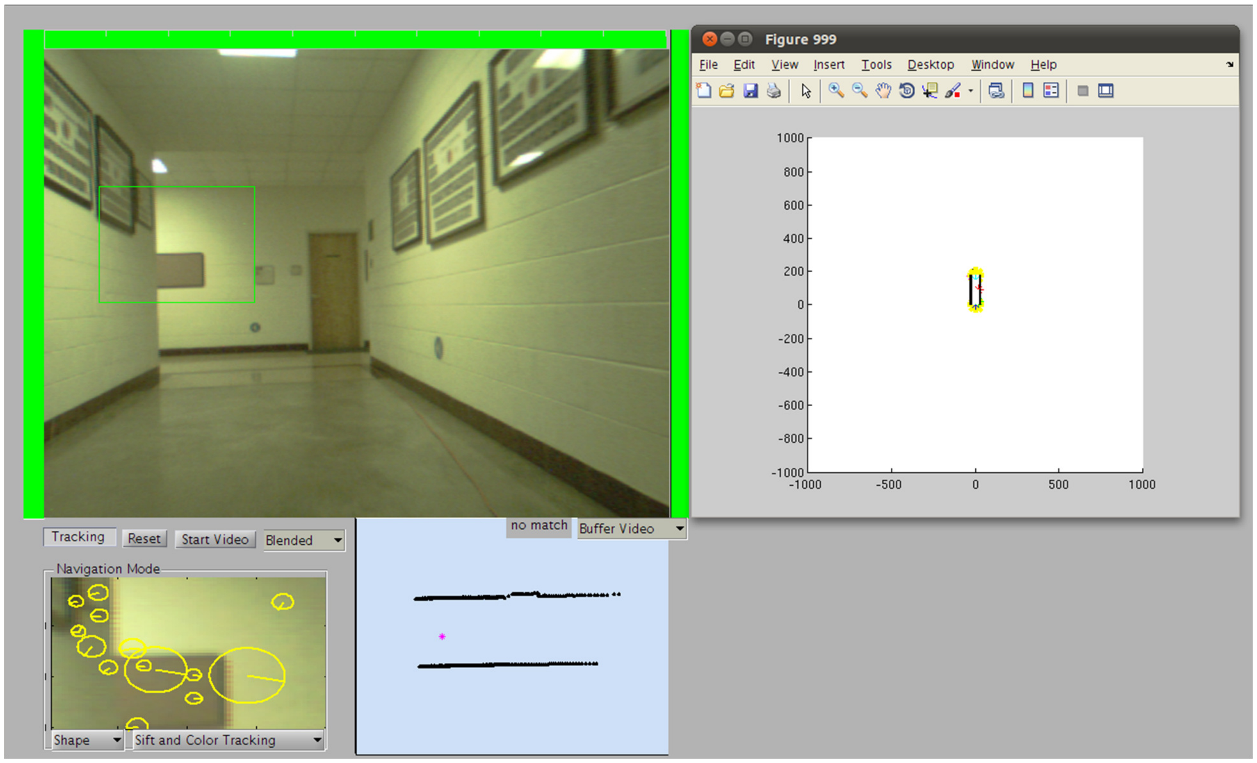


Figure 103: Operator GUI with QUICK mapping display reference graph on

The final QUICk maps are derived for Scenario 3 that portray the full functionality of the Operator Blending framework. Navigation used as a shared basis manages the local search process. Although the local search process takes more time than that of Scenario 2, the objects selected are done for the purpose of navigation and local search. Because there is a conscious separation between objects the operator views for navigation and objects the operator views being important for the search process, object models reflect this distinction. For scenario 1, important objects were chosen to be those an individual would be able to sit on. All sittable objects were right clicked for QUICK map annotation. Along the local search path, the operator used the Normal importance specification to identify navigation beacons. The QUICK maps shown are reprinted after AOI correction for object model correspondence as discussed in Chapter 5.

Figure 104 displays the QUICK map under Scenario 3 for Sub-area 1. The map is enlarged for display purposes so the annotation is visible. In addition, all objects were labeled by the operator with distinct identification for topological annotation.

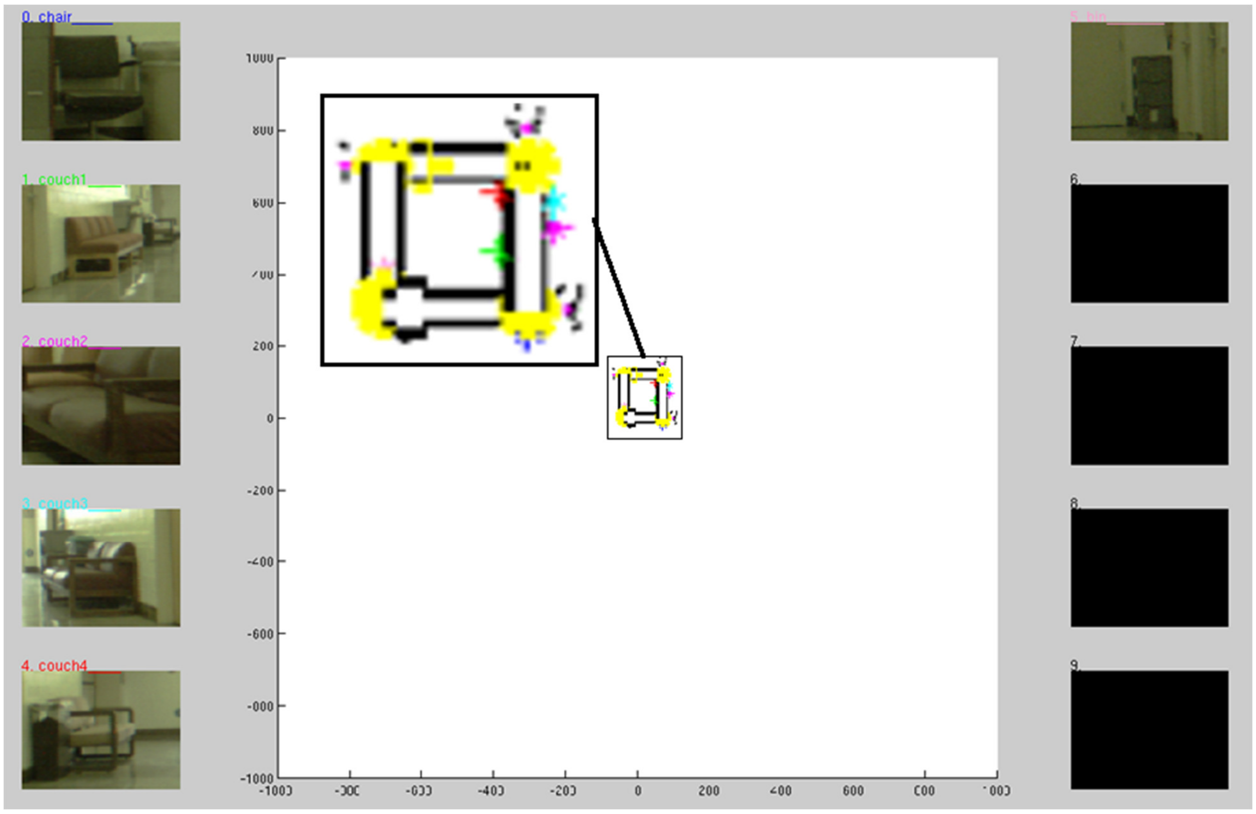


Figure 104: Final QUICK map for Sub-area 1 under full Operator Blending of Scenario 3

Because there was a specific mission for the RobiSEND operator, the number of selected important objects along the local search path was less than that of Scenario 2. The objects selected, however, are believed to have more significance to the local search process. As mentioned previously, the maps of the Sub-areas in this section are enlarged so their features are viewable within this document. The actual QUICK map would have a zoom capability to see finer details of the smaller structure.

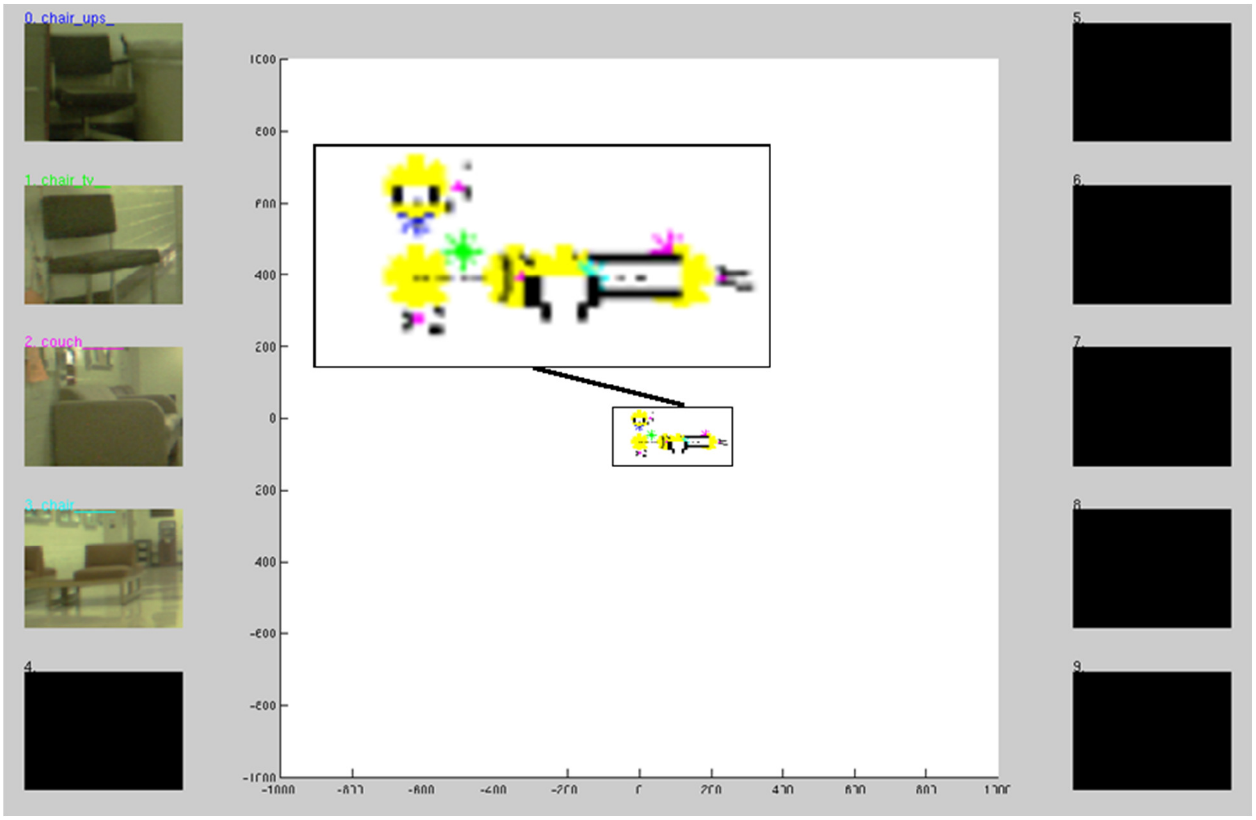


Figure 105: Partial plot of Sub-area 2 QUICK map under full Operator Blending of Scenario 3

As the local search process moved to Sub-area 2, object selection first goes into an area of content space. The objective of QUICK maps, as mentioned previously, is to communicate the local search process and the information present in an unknown environment. As a result, metric information is only valuable if it adds to this purpose. Content space is displayed in the QUICK map of Figure 105 as covering more area than more metric mapper models such as Hallway indicators. The first three Junctions of this figure are realistically in the same general open area of the map. It is through the topological annotation, search path indicated by the placement of ContentSpace mapper models, and CornerJunction mapper model raw scan data that communicates to the small tactical team the structure of the environment.

Also shown in Figure 105, one drawback of the current approach to QUICK map annotation is that the map is always annotated with object model indicators as soon as important objects are selected. The turquois object and the blue object in this figure display how this can be problematic. The object model locations could do a more effective job of communicating the actual locations of these object models. Realistically, the turquois object, for example, would more accurately reflect the environment if it were further down the middle hallway of the map. Because the current structure reflects when objects were selected, however, it has the plus side of reflecting moreso the process taken during local search. Without this approach, when objects are selected would not be encoded within the QUICK map.

The operator, therefore, should be conscious of how information is portrayed in the QUICK map structure. Figure 106 gives the full representation of Sub-area 2 under Scenario 3. The red object of Figure 106 is equivalent to the turquois object of Figure 105. As shown in the Figure 106 example, the operator waited until the object was closer to the RobiSEND payload before designating it as important. The result was a QUICK map indicator that better represents this object placement within the QUICK map.

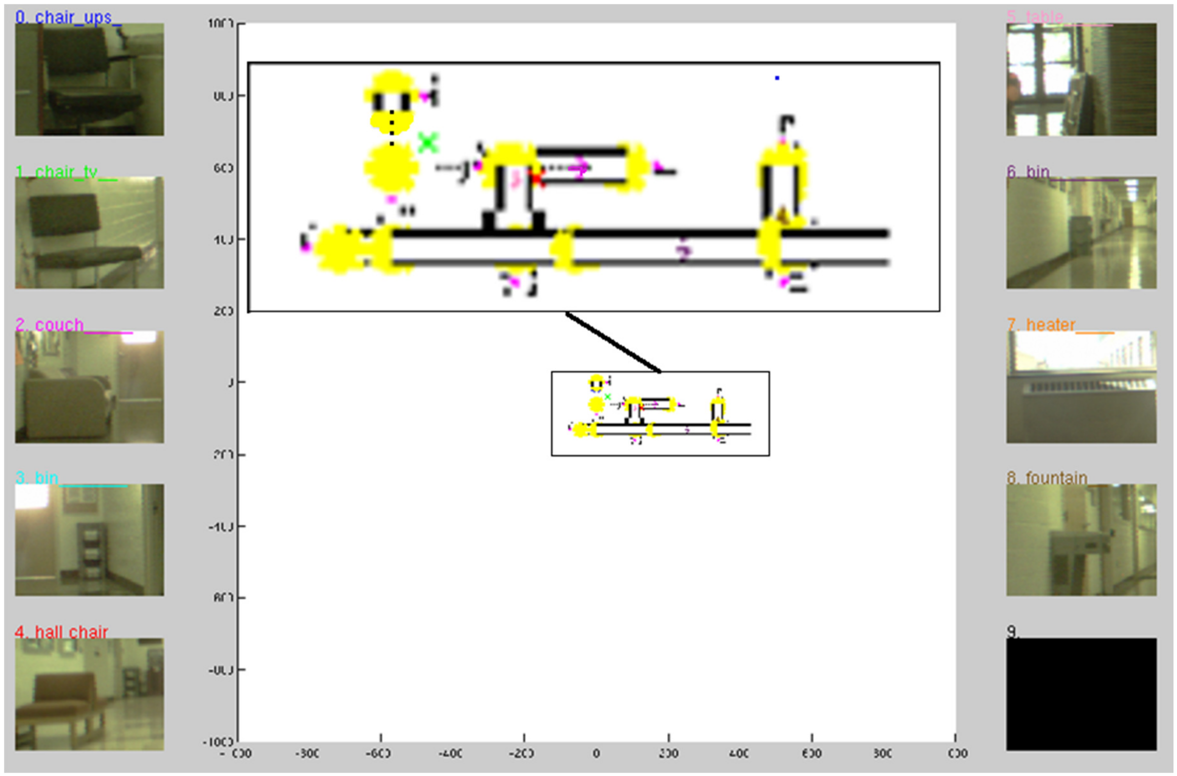


Figure 106: Final QUICK map for Sub-area 2 under full Operator Blending of Scenario 3

Also shown in Figure 106, object 3 and object 6 were both labeled by the operator as “bin”. This is potentially impactful for AOI map optimization. In this case, however, the mapper models are spatially too far apart to impact their individual placement. The chair of object model 1, however, was selected from two different points of view in the process to generate Figure 106. These two views are shown in Figure 107. As RobiSEND moved East in the environment, the initial object selection was made as given in the leftmost image. After turning around and heading back along the same path, the chair object was selected from the opposite direction as in the rightmost image.

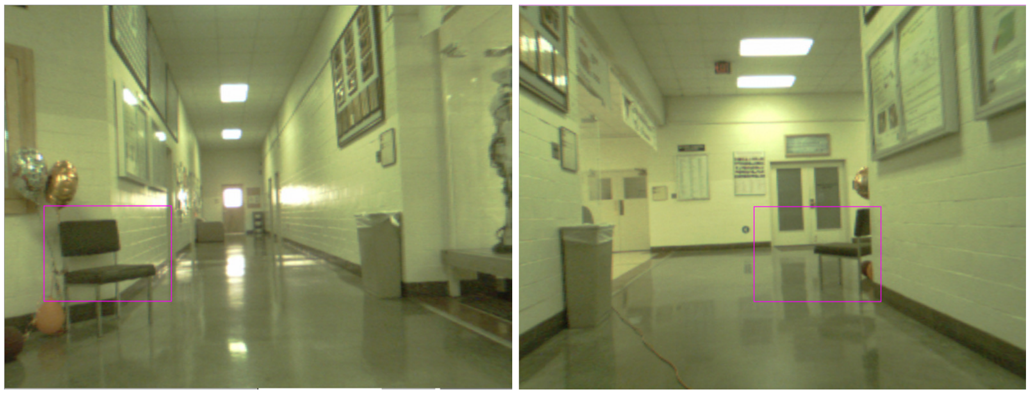


Figure 107: Object selection corrected through AOI map correction. Both objects are given the same label and so descriptors correspond to the same object model

The alternative views of the same objects are corresponded through operator annotation. The object background image that corresponds to the first selection of the mapper model is used for the color information in the QUICK map and the time of selection from the second selection is used to place the object model indicator. If one of the objects was determined to be Mature, then that object indicator would be used for the combined object model. In this case, both objects were not determined to be Mature. If both objects were Mature, then the same process described in this section would be taken.

If future object model selections are corresponded with the features of object model 3 or object model 6 or the operator labels another object model similarly, the new object model would be represented by the object background image of the first selection and its placement would replace the other two object models. Multiple images that have corresponded with previous object models, as a result, are shown only once on the QUICK map after AOI optimization.

Figure 108 displays the QUICK map of Sub-area 3. Much like Figure 106, important objects were selected with the RobiSEND payload going North and wih the RobiSEND payload going South along the local search path. The same procedure for AOI as discussed previously is used. Only indicators from the latter object selection on displayed on the final QUICK map. At the conclusion of the local search process, all sub-areas are combined for the final map structure. This structure is shown in Figure 109.

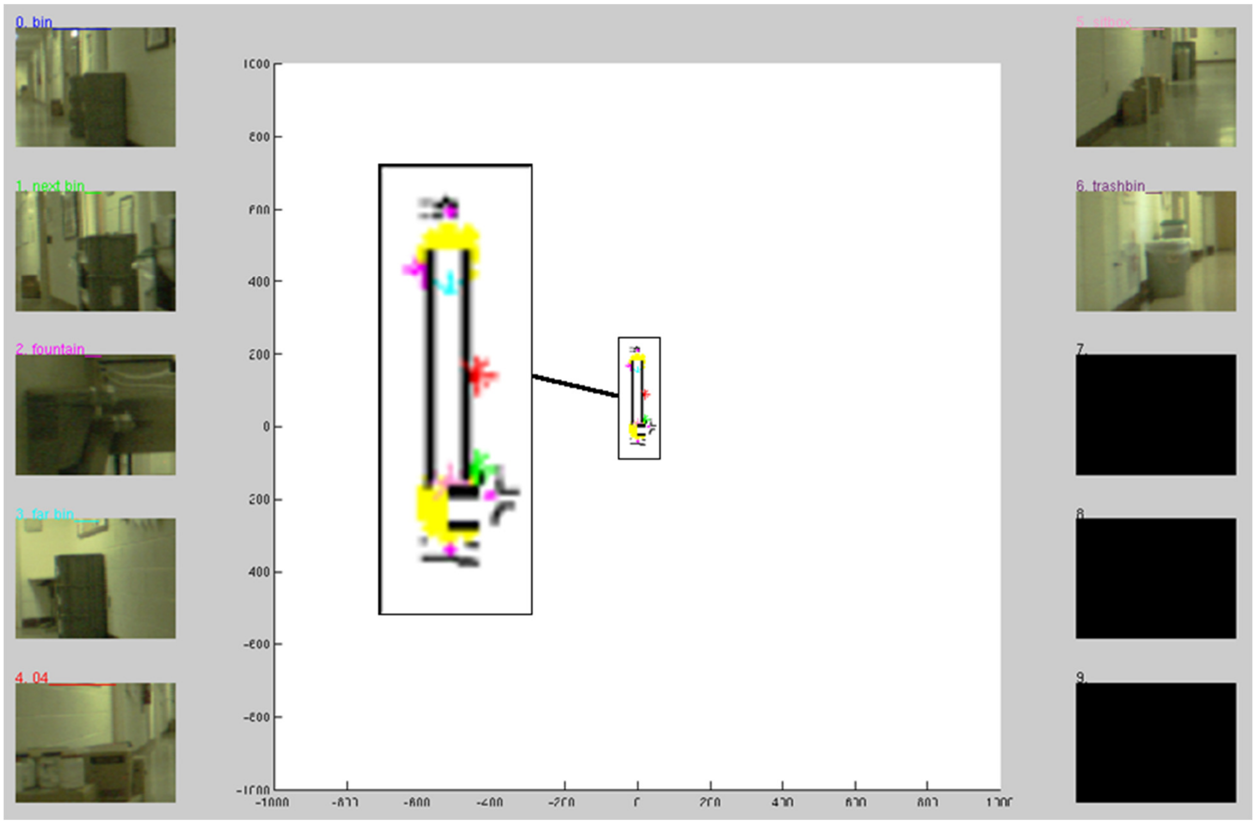


Figure 108: Final QUICK map for Sub-area 3 under full Operator Blending of Scenario 3

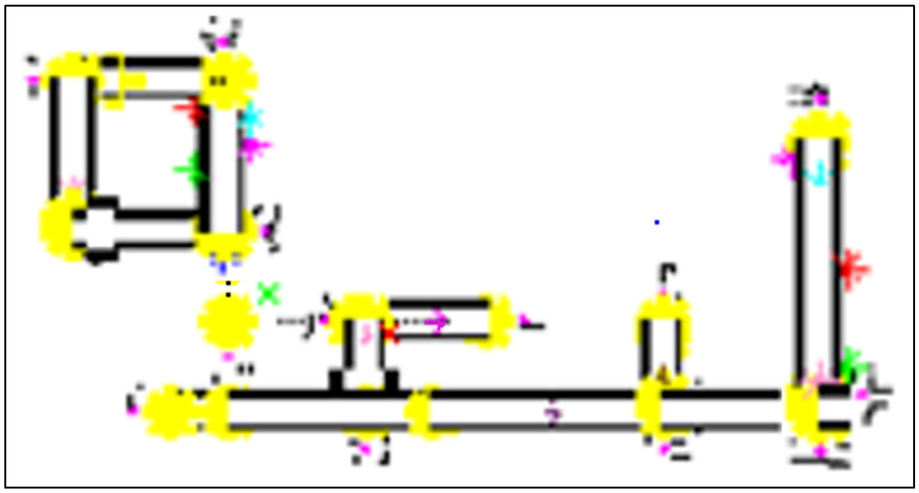


Figure 109: Annotated full QUICK map with indicators

The QUICK map display with object background color image information, as shown in the previous figures of this section is used to display Sub-maps, because of the limited capacity to show 10 objects at a time. At any instance, the end deliverable may be switched between this Sub-map view and the final view that contains how these Sub-map pieces fit together.

7.5 Scenario Comparisons

Scenario comparisons emphasize the need for aspects of Operator Blending for MRUAV based local search missions. Sub-section 1 compares scenario 1 and scenario 4 in order to contrast traditional approaches to metric mapping with the metric components of the QUICK mapping framework. Sub-section 2 compares scenario 2 and scenario 3. This is done in order to demonstrate the merits of the topological components of QUICK maps. As mentioned in Chapter 3, qualitative maps, like quantitiative maps may have both metric and topological components. Operator Blending brings out the benefits of qualitative maps over quantitative maps. How the topological and metric components are manifested and their impact on the quality of qualitative maps are both demonstrated and discussed.

7.5.1 Comparison of Scenario 1 and Scenario 4

As mentioned, purely metric maps were generated for Scenario 4. To compare these maps to the metric components of the QUICK map structure, the Sub-area maps of Scenario 1 were used. These maps provide a basis for comparison because they do not include the topological information introduced to the local search process through the inclusion on the Importance and Maturity traits. It must be noted, that the occupancy grid maps were not generated in real-time. This is an important distinction, because one of the objectives of QUICK maps is to construct a representation of the environment without needing any post processing. The maps generatd in Scenario 1 are not post processed at all. AOI optimization is only conducted based on the topological annotation involved in including object model information within the map as observed in Scenario 2 and Scenario 3.

Figure 110 shows the CARMEN based metric occupancy grid map built from the same raw LIDAR information and interpolated odometry data from operator commands. The blooming at the corners of the occupancy grid map is based on not knowing the exact odometry information. This is because only local information is available for the construction of QUICK map representations. Therefore, although upon initial consideration of the structure of the QUICK map, Junction mapper models seem to be large and not necessarily representative of the map structure, the alternative when dealing with purely metric maps also shows a significant map distortion. Without exact odometry data, this problem is present even in metric map constructions. The QUICK map is a cleaner representation of the map environment, but tends to miss some opportunities for PartialJunction mapper models. One is identified at the Northern edge of the map, but another is completely missed. This may be dealt with through tuning of the accumulator module and analyzer module parameters to be more sensitive.

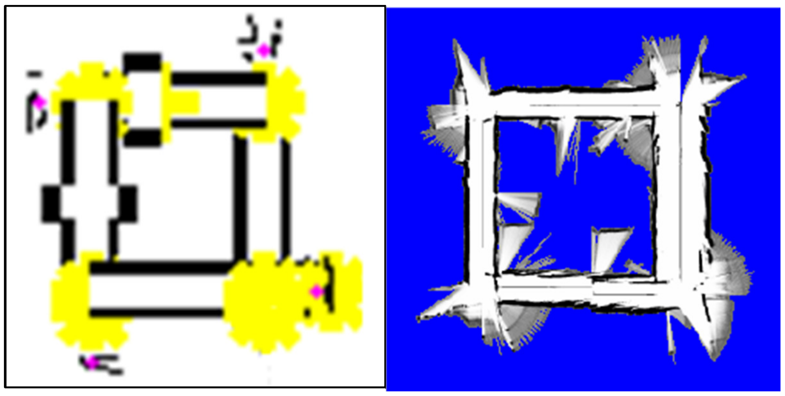


Figure 110: QUICK map from Scenario 1 comparison to purely metric map of Scenario 4

Metric reconstructions without precise odometry information also are susceptible to inconsistencies that could be corrected through a POI optimization approach as in the QUICK map framework. The comparison of Sub-area 2 from Scenario 1 and the CARMEN based map demonstrate this.

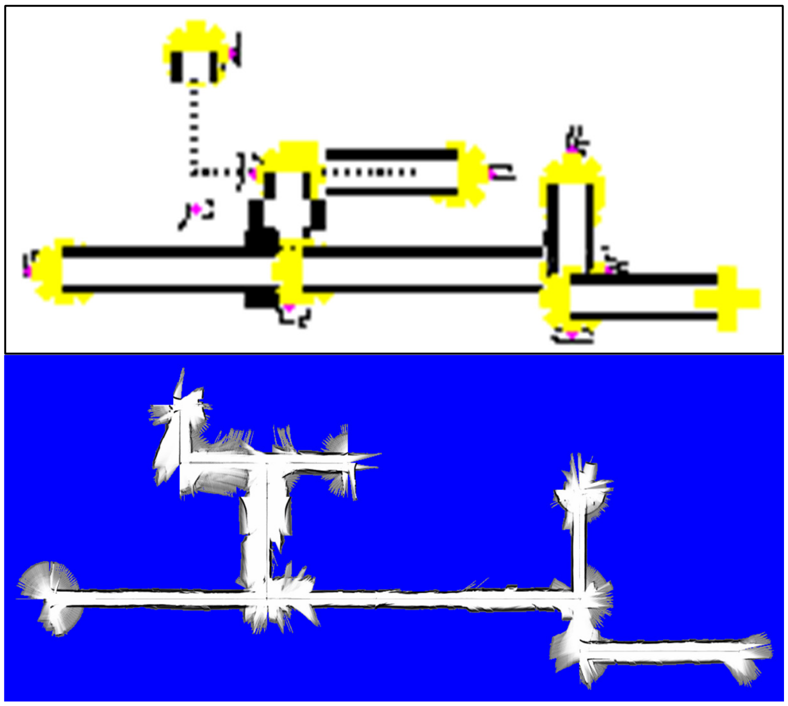


Figure 111: QUICK map of Scenario 1 without POI optimization compared to the purely metric map of Scenario 4

At the Eastern corner of their respective maps, RobiSEND has overestimated the length of the vertical hallway in both cases. Because the QUICK mapping approach integrates multiple measurements over time, this inconsistency is less dramatic in the QUICK mapper representation. The metric map does capture the area mapped as ContentSpace in the QUICK map with more detail. For the function of communicating the general structure of the environoment to a small tactical team, however, the QUICK map gives a cleaner portrayal. In addition, the raw sensor information provided by the QUICK map gives an opportunity to show CornerJunction detail that is useful when viewing paths that have not been fully explored.

Additionally, if a POI optimization stage were to be added to the metric map of Figure 111, correspondence would have to be generated between each of the raw LIDAR scans used as components of the occupancy grid map. The QUICK map integrates several LIDAR scans in the analyzer and accumulator modules before a mapper model is output from the mapper module. Therefore, POI does not require correspondence between scans. The object recognition problem allows for a recognition in the placement of Junction mappe models. Therefore, the POI optimization phase and the mapping component are able to be done in real-time with the QUICK mapping approach.

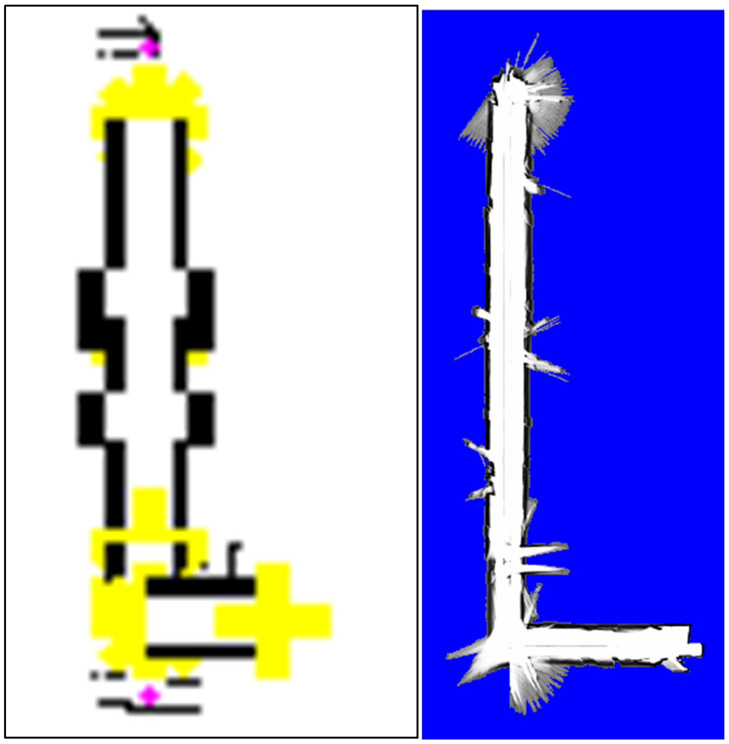


Figure 112: QUICK map comparison from Scenario 1 without POI correction to map of Scenario 4

Figure 112 displays the final Sub-area comparison between Scenario 4 and Scenario 1. Once again, the QUICK mapping approach misses some opportunities for Partial Junction specification, although it does manage to capture their presence in two locations. The CornerJunction mapper model at the bottom of the QUICK map also communicates the hallway structure effectively. The metric map generates the hallway width in a more effective manner. The width of the hallway is potentially useful information for the robot collaborator for obstacle avoidance. This provides another opportunity to point out the difference in purpose between the two maps. Metric maps are built with the purpose of reconstructing the unknown environment spatial relationships as accurately as possible. The purpose of QUICK maps, on the other hand is to communicate the structure of the environment to human and robot collaborators. The map must represent the relationships built in the environment and the local search process that generates these relationships.

7.5.2 Comparison of Scenario 2 and Scenario 3

Scenario 2 and Scenario 3 both communicate more relationships in the environment through the inclusion of topological information. These scenarios, as explained previously, are distinguishable through the existence of navigation as the shared basis for communication. As a reminder, for RobiSEND, this means that for Scenario 3, the local search process is coupled to the navigation process. For Scenario 2, the local search process is independent from the navigation process.

Chapter 2 explains the negatives associated with an independent navigation process. This independence is in line with the current approach to unmanned systems integration. The difficulty is in including multiple operators to run the independent processes, because one individual is not able to multitask. The independent roles of tactical teams are covered in addition to a discussion about the communication difficulties when the agents of a human-robot team are expanded beyond a OHOR scenario. To handle multiple independent processes, autonomy has been found to be one solution, however, for this case it has been ruled out in favor of the advantages given by a human-robot collaborative team. Therefore, it must be pointed out that Scenario 2, if implemented on an actual system, would require independent processes at least to provide translational commands. The operator of Scenario 2 functions in the role of payload control and would need either the autonomous control of robot translation or an additional operator in the pilot role.

The codependence present in the navigation process of Scenario 3 is the opportunity to have the operator impact the observation recognition process and to do so without the introduction of independent processes. In essence, the process of local search is the source of translational commands to the RobiSEND system and the structure of the QUICK map. In addition, actions necessary for topological map annotation are not reliant on communication with a separate agent beyond communication between the human-robot collaborative team. The operator can afford himself time to conduct object label actions under the conditions of Scenario 3 because of complete understanding of the navigation process and what his commands mean to the robot collaborator.

Map optimization is done completely differently as well when considering a comparison between Scenario 2 and Scenario 3. POI optimization is done in both scenarios and is useful to ensure consistency between Junction mapper models based on observation recognition. The AOI optimization possible in Scenario 3, however, adds a consistency to the topological annotation of the QUICK map. Object models that are labeled multiple times are not reprented multiple times under this framework. In addition, priority is given to those correspondences that have achieved Maturity over time. Therefore, more reliable information is emphasized under Scenario 3 as opposed to Scenario 2. Also, the need to identify beacons for navigation through the normal object model designation by the operator forces him to designate objects as important only if they satisfy a specific mission objective. This objective was sittable objects in the example considered previously, but could have any significance depending on the mission profile. Therefore, the additional information included in Scenario 3, even without considereing the integration benefits, makes the full implementation of Operator Blending more accurately reflect the unknown environment.

7.6 Contributions

As mentioned, this dissertation has made five key contributions to the field of Human-systems integration as it relates to unmanned systems. These contributions are Operator Blending, Code of Observational Genetics (COGs), Joint Intension Theory applied to unmanned systems, computational methods related to operator directed attention with AIM-Shift as one example, and Qualitative Unsupervised Intelligent Collaborative Keypoint (QUICK) mapping. The first, Operator Blending, is the foundation for the other developments of this work. Simply stated, it emphasizes cooperative and codependent solutions to the problems of human-robot teams. The most paramount of these issues are deeply rooted in perceptual and integration based teaming concerns often neglected in current research. Both the human and the robot contribute what each does well in solving problems.

7.6.1 Operator Blending

Operator Blending is a branch of human-robot collaboration created in this research to leverage what robotic teammates do well in a collaborative and codepenedent relationship. Human-robot collaboration is not a new concept. The idea, however, that this collaboration does not have to solely serve the needs of the human or robot in this relationship is a key contribution of this dissertation. When both robot and human contribute what each does well, novel means of communication, information sharing, and deliverables are needed to extract the benefits of this new approach. This research not only defines Operator Blending, but it also provides the tools that lay the foundation for Operator Blending as applied to a real application. Ultimately, the new relationships established with Operator Blending create a means to more effectively solve perception and integration challenges. The solutions provided in this research establish the possibility for human-robot collaboration to lead to acceptable performance of MRUAV based local search systems for the support of small tactical teams.

7.6.2 Code of Observational Genetics

Code of Observational Genetics (COGs) are the source of principle components with functional relevance that are the foundation for Operator Blending. Traditionally, human-robot collaboration is seen as either human-centered or robot-centered. Operator Blending emphasizes the need for both to contribute what they do well in solving problems. In order for perception and integration problems to be solved collectively under Operator Blending, observation models must reflect the operational needs of both the human and the robot. COGs define the basic components of observation models needed to address the basic needs of a human-robot collaborative system. Because the robot and the human are collaborating directly, their respective observational needs are not independent. The synthesis of observational needs establishes four traits that are combined to form COGs. These traits are Function, Uniqueness, Importance, and Maturity. This work establishes the theory that the 16 possible combinations of these traits may encompass the functional classifications of observations in an unknown environment. COGs establish a common understanding of observation model characteristics between the human and the robot that is based in the observation significance to shared mission objectives.

7.6.3 Joint Intentsion Theory

Joint intension theory is not a new concept; however, its application to MRUAV based local search systems is the contribution to joint intension theory presented in this work. The basic concept of joint intension theory is that, in the context of an effective team, team motivations dominate individual motivations. Although individual motivations exist, team goals directly influence and determine individual goals. The role of MRUAVs as local search systems is made possible through the inclusion of this team context and its ability to address traditional perception and integration challenges. MRUAVs are not cast as local search systems because these challenges prevent them from being seen as viable contributors to local search. Through joint intension theory and the concept of teaming with a human operator, this work effectively opens up the possibility for MRUAV based local search systems.

7.6.4 AIM-Shift

AIM-Shift is the primary example used to convey the novel approaches to computation developed in this research. Boundary based segmentations are reserved for post processing in previous approaches. Because the operator is in a codependent and cooperative relationships with the robot within the Operator Blending framework, boundary based segmentations are a tool for novel development of computational approaches that involve operator directed attention. Operator directed attention is the foundation for AIM-Shift. Because the operator is able to specify the location of objects of interest, AIM-Shift may localize the feature extraction process. The boundary based segmentation reduces the amount of information processed for tracking simply because only a sixteenth of the image is used for feature extraction based on operator selection. The novel component of AIM-Shift is in the use of boundary based segmentations for the initial object location in real time. Traditional approaches do not have this option because the operator is not a source of information for the object tracking process. Operator directed attention also influences other aspects of computation. AIM-Shift is the most relatable example because it solves a specific problem in the RobiSEND application.

7.6.5 QUICK mapping

QUICK mapping is the first purely qualitative approach to map generation. Although qualitative information has been used in the past for decision making, quantitative map correction, and is the basis for qualitative spatial reasoning, the poverty conjecture makes the case that a purely qualitative method for mapping is not possible. With the inclusion of the codependent and cooperative contributions of the operator under Operator Blending, however, qualitative mapping becomes a possibility. The object recognition process which is guided by operator directed attention is the source of qualitative information in the establishment of COG models. These COG models are synthesized qualitatively through the deliverables created in this work as QUICK maps. Because the mapping process is dominated by object recognition, map optimization is possible through establishing the correspondence of COG models. As opposed to the correspondence problem forcing a feature or point to point correspondence, an object model and mapper model correspondence is used to optimize the QUICK maps. As demonstrated, these optimization steps are not a necessity for the generation of the QUICK map structure and just provide a means to provide more precise environmental representations. The concept of a mapper driven by the object recognition process with no major requirement rooted in correspondence is a novel contribution of this work.

7.7 Discussion

7.7.1 Impact of Integration approach

The success or failure of the Operator Blending approach may be assessed in its ability to affect the perception and integration capability of a human-robot team. The effectiveness of the human agent in this context is related to maintaining situational awareness. A test for situational awareness and cognititve load would be beneficial for proving the effect of the implemented system. This is left for future work. The assessment of situational awareness is done in light of the definition for situational awareness for MRUAVs as defined in Chapter 2.

As mentioned, 7 situational awareness principles govern the integration of Human-robot teams involving MRUAVs. Three of these are perception based and four are integration based. This information is repeated for reference.

1(a). Placement of the MRUAV with respect to terrain and targets
2(a). Knowledge of the MRUAV and points on the earth such as home base or landing zones
3(a). Resilience of the MRUAV to anything that could obscure sensors

1(b). Knowledge of the health or malfunction of the MRUAV
2(b). Understanding of MRUAV relationship to team members and bystanders
3(b). Degree to which the MRUAV can be trusted to respond as expected to user commands
4(b). MRUAV progress towards completing the mission

The perception based challenges for situational awareness of MRUAVs are all addressed by the RobiSEND system. With respect to the placement of the MRUAV with respect to terrain and targets, the QUICK mapping approach is based on local information. Therefore, at any instance in time, it is clear where objects of interest are with respect to the MRUAV location. This is related to the knowledge of the MRUAV and points on the earth. The operator may easily indicate important points on the Earth through the Operator Blending structure. In addition, all mapper models and object models reflect the process of the search as opposed to the actual metric relationships of objects. This process emphasizes important areas during search over those areas with less significance to the operator. Although the argument can be made that more precise metric relationships are possible, RobiSEND takes the approach of using metric relationships for communication only when that information is necessary. Future implementations of Operator Blending may have the ability of more effectively building these relationships, but the fundamental need to have this approach for perceptual purposes is outlined in this work. The last perception based component of situational awareness is resilience of the MRUAV to sensor nuissances. The feature extraction method of RobiSEND coupled with the use of movement to update object templates for the object recognition problem accomplishes this. Therefore, it can be said that RobiSEND successfully addresses all three of these concerns that lead to perception based challenges of MRUAVs.

Integration challenges are not shown as clearly as the perception based challenges through the Scenarios given in Chapter 7. First, information is needed on the RobiSEND GUI that indicates the overall health of the MRUAV system. Because the GUI is mainly dominated by the video feed, room is available to add this functionality if the system were to move from hardware in the loop to an actual implementation outside of hardware-in-the-loop. Secondly, RobiSEND does a good job of conveying its relationship to obstacles qualitatively, but may not necessarily give a good representation of other tactical team members beyond the location of the RobiSEND operator. Unless the operator selects team members or bystanders and indicates them to have importance, the RobiSEND system has no means of extracting their location or significance. The operator does have the capability of selecting team members and tracking their movements in the environment. This ability could potentially be useful in dealing with important dynamic objects. Also, understanding the significance of other individuals is helped by the fact that there is an operator in the loop during the local search process. Because of the shor t duration of local search missions, this operator is able to devote his undivided attention to making sure teammates and bystanders are at a safe distance from the RobiSEND system. In addition, because the MRUAV was implemented in a hardware in the loop scenario and was assumed to travel along a predetermined search path, it is also difficult to assess its reliability to responding to operator commands. In theory, the reduction of the human-robot ratio to OHOR should lead to better predictability of MRUAV function by the operator, although this was not tested in the current research. Lastly, the RobiSEND system conveys the opportunities for future exploration within the environment. Because the value of the MRUAV search system is expendable after the conclusion of the search, the mission is completed when the annotated map is sent back to the operator. The mission role of a MRUAV outfitted by the RobiSEND payload does address this concern with situational awareness. As mentioned in Chapter 1, RobiSEND MRUAVs are designed to be light and transportable enough to be encluded in the gear an individual might operate and carry. Therefore, the tactical team would potentially have access to multiple RobiSEND systems. Each system would be able to be deployed for a specific tactical objective or to continue an objective until mission completion. In the observed scenarios, RobiSEND was able to qualitatively map the structure of the example area in the designated 15 minute mission duration.

7.7.2 Impact of Perception approach

Perception effectiveness is shown in the deliverables from Scenario 1, Scenario 2, and Scenario 3. Without an update stage to a purely qualitative mapping approach rooted in the object recognition problem, the mapper is not capable of producing maps that are consistent. POI optimization is typically all that is needed to correct for many errors and does well to correspond junctions based on their location and features. In addition, when many important objects are selected during a local search, AOI is the only way to ensure that objects observed at different times feed in correctly to the final map.

If the proper correspondences are maintained throughout the duration of the local search process, emphasis on the object recognition problem quickly pays dividends. Features that would be needed on any hand to do object recognition tasks within a quantitative map are used to develop QUICK maps in this research. Without object recognition there is simply no way to relate local information. It is this local information and its many uses that impact the relationship between human and robotic team members. RobiSEND builds these relationships through the features it extracts for the object recognition problem.

QUICK maps annotated in this manner successfully convey the local search process. When comparing the deliverables from Scenario 1 and Scenario 2, the topological information that the RobiSEND system is able to introduce is easily implemented into the final map. As a result a fully annotated and corrected map is available in real time as the local search process is underway. Although it would be beneficial to conduct a human factors based study on the effectiveness of Operator Blended human-robot collaborative teams and how they conduct perception based tasks, there is currenly no other approach to compare this approach to. It is novel in the sense that this combination of resources within the team has never been done.

The perceptual techniques introduces through this research are also novel ways of incorporating boundary based segmentations for perception. Typically, segmentation is conducted through the approaches covered in Chapter 3 that preclude boundary based segmentation because of no means to direct the attention of the system. This means was introduced in this research through Operator Blending. AIM-Shift is an example of what can be done by a system that takes advantage of operator introduced boundary segmentations.

In addition the Code of Observational Genetics captures the inate perception challenges for human-robot collaborative systems. Although this sytems is not the first effort to understand the semantically relevant characteristics of observations, it is the first that allows a human and robotic agent to contribute to their construction. Mapper model effectiveness at generating global deliverables is displayed through the output form Scenario 1. The environmental structure is portrayed adequately through the 8 different outputs form the mapper module, although, for mor complex environments, additional portrayals of the environmental conditions could be beneficial. For object models, COGs managed to encapture the necessary characteristics of objects as they relate to perception for the local search process and human-robot collaborative teams.

7.7.3 Everything Together

Operator Blending is a research area that has the potential to solve many of the current difficulties with perception and integration of human-robot collaborative teams. Although the methods have not been optimized for effectiveness and human factors studies have not been conducted to verify the approach, the theory is the basis of novel research and the usefulness of the aforementioned approach has been demonstrated.

As the iteration of technology begins to taper off and control theory methods continue to improve, the relationships robots force humans to build with technology are of the utmost importance. This dissertation identifies a problem that is traditionally not addressed in human-robot interaction, addresses this problem from the human perspective, frames the problem from a computational perspective, identifies a new means of communication between the constituent agents, and provides a solution for how they address a problem fundamental to human-robot teaming in local search.

Challenges, research fields, and approaches are coupled through this effort. It is established that the semantic formulation of observation models involves must be thought of not as a purely perception based problem but also a problem that involves considering integration concerns. After all, this is what human beings do. If the final deliverables are to truly reflect a teaming relationships and be relatable to both agents, the ability to blend in human processes with robotic processes in essential. For a human-robot team, there must exist a certain level of duality. Prinicple components must be integration and perception based, teams must be robot and human based, maps must be metric and topological based to effectively spark the synergy necessary to achieve some of our most basic goals for robotic systems. The complete picture involves a more comprehensive approach than what is being observed with current research. Although the ideas expressed in this dissertation may be improved upon or implemented in a slightly different manner, the foundation for Operator Blending is a new and valuable concept to the future of human-robot collaborative teams.

7.7.4 Future Work

Operator Blending as a new research field provides many opportunities for future work. Some of these opportunities are direct expansions of other concepts created in this work such as RobiSEND, AIM-Shift, Code of Observational Genetics, MRUAV based Joint Intention Theory, and QUICK Mapping. Other opportunities are more broad-based and encompass ways of implementing an Operator Blended approach to solving problems. Because so many challenges exist and must be solved to realize the potential of localized robotic systems in teaming situations, the expansion of Operator Blending as a field will lead to new and exciting questions being asked that go well beyond the scope of this dissertation.

In order to advance the ideas of the RobiSEND system, the first and perhaps most difficult opportunity for future work is to implement the conceptualized RobiSEND payload on an actual MRUAV. Although this research had access to MRUAVs for this purpose, many challenges characterize this task as outside of the scope of work for this thesis. Collaboration with those in the fields of control theory and human factors and the ability to perform flight operations, MRUAV upkeep, and maintenance would be needed to safely perform the necessary tests to verify the approach. Operator Blending is an approach that does not necessarily need to be limited to MRUAV based local search systems. The expansion of the approach to ground based vehicles may be more feasible for the actual implementation of the RobiSEND payload and is another opportunity for future work.

The computational methods developed in this work may also be expanded upon for implementation within other Operator Blended systems. AIM-Shift is a fairly simple approach to object tracking that takes advantage of novel boundary based segmentations possible from the Operator Blended approach. This technique can potentially be expanded upon or implemented through other Operator Blended unmanned systems. In addition, a shift away from the correspondence problem for mapping towards the object recognition problem also provides an opportunity to implement these techniques in diverse conditions. Embedded systems were explored for the development of Operator Blended approaches and would be ideal for future systems because of their light weight and form factor. The computational techinques developed in this thesis are potentially portable to systems with computational power not sufficient for more traditional approaches.

The Code Observational Genetics concept is also a new approach to classification that has potential in future human-robot collaborative studies. On a fundamental level, the base concept of COGs is the existence of sematically relevant principle components for the description of objects. Although this is not a new concept, it is one that is often neglected in the advancement of robotic systems capabilities. In addition, the semantic formulation of observation descriptors is not typically looked at as a problem in human-robot collaborative theory. The four COG traits identified in this research have potential to be expanded upon and improved for future Operator Blended systems.

QUICK mapping is also an exciting opportunity for future work. There is a lot of improvement possible to more efficiently map an unknown area through qualitative spatial reasoning. The concept of the QUICK map, to the author's knowledge, is the first attempt to concretely apply these concepts outside of a means to reinforce current quantitative methods. The QUICK mapping technique has potential to expand the field of robotic mapping. Future work would consist of porting its ideas to a software other than MATLAB for cosmetic and computational purposes and fine tuning the mapper place vocabulary. The ability for this mapping technique to communicate the process of local search over the metric relationships in the environment is a new concept and has potential within the field of human-robot collaboration.

Lastly, the MRUAV's place in local search based missions may also be expanded upon. The implementation of this technology has not quite been optimized in current systems. Although there are control theorists that are able to map unknown areas with MRUAVs, these sytems do not typically operate in autonomy well outside of controlled circumstances and their user interfaces are not optimal for operator in the loop systems. The integration considerations addressed in this research are important for understanding the limitations and challenges for MRUAV implementation. The exploration of local search in general as an application for this technology is an area of research that could provide for new and exciting opportunities.

7.8 Conclusion

Operator Blending is a novel research field created in this research as an alternative to traditional human-robot collaborative approaches to teaming. The methods presented to demonstrate Operator Blending concepts are used in this dissertation to communicate ideas and the merits of the approach. MRUAV local search conducted by small tactical teams was used as the muse for the development of this new and far reaching field.

The genesis of Operator Blended is presented as the marriage of the robot to the human through a cooperative and codependent relationship. The operator and the human both must contribute what each does well to effectively accomplish mission objectives. This is a starkly different approach to that studied in current human-robot collaborative fields where the emphasis is on the needs of the robot or the human depending on the circumstance. Opportunities for new computational methods, approaches to mapping, system integration, and relationships between human and robotic teammates were made possible through this approach.

Qualitative Unsupervised Intelligent Collaborative Keypoint mapping, Code of Observational Genetics, Area Intersecting Mean-SHIFT, and MRUAV based joint intension theory were all concepts developed by this work to support Operator Blending. Although there are several opportunities to optimize these approaches and more comprehensively verify their effectiveness, the theory has opened the door to a new way of understanding human-robot collaborative relationships.

Bibliography

Adams, J. A. (2002). Critical considerations for human–robot interface development. *2002 AAAI Fall Symp.* (pp. 1-8). Menlo Park, CA: Human–Robot Interaction.

Adams, R., & Bischof, L. (1994). Seeded region growing. *IEEE Trans. On PAMI*, 16(6).

Amigoni, F., Gasparini, S., & Gini, M. (2006). Building segment-based maps without pose informations. *In Proceedings of the IEEE, 94*(7), 1340-1359.

Bajcsy, R., Kamberova, G., & Nocera, L. (2000). 3D reconstruction of environments for virtual reconstruction. *In Proc. of the 4th IEEE Workshop on Applications of Computer Vision*.

Bajracharya, M., Moghaddam, B., Howard, A., Brennan, S., & Matthies, L. (2009). Results from a real-time stereo-based bedestrian detection system on a moving vehicle. *In Proceedings of the IEEE ICRA 2009 workshop on people detection and tracking.*

Barnes, M. J., Knap, B. G., Tillman, B. W., Walters, B. A., & Velicki, D. (2000). Crew systems analysis of unmanned aerial vehicle (UAV) future job and tasking environments. Aberdeen Proving Ground, MD: U. S. Army Research Laboratory.

Bar-Shalom, Y., & Fortmann, T. E. (1998). Tracking and Data Association. *Academic Press*.

Batkiewicz, T., Dohse, K., Kalivarapu, V., Dohse, T., Walter, B., Knutzon, J., et al. (2006). Multimodal UAV ground control system. *In 11th AIAA/ISSMO multidisciplinary analysis and optimization conference* (pp. 6–8). Portsmouth, VA: AIAA.

Besl, P. J., & Jain, R. C. (1988). Segmentation through variable-order surface fitting. *IEEE Trans. On Pattern Analysis and Machine Intelligence*, 10(2) 167-192.

Bratman, M. (1992). Shared cooperative activity. *The Philosophical Review, 101*(2), 327-341.

Bruemmer, D., Few, D., Boring, R., Marble, J., Walton, M., & Nielsen, C. (2005). Shared understanding for collaborative control. *Systems, Man and Cybernetics, Part A: Systems and Humans*, 494-504.

Burgard, W., Fox, D., Jans, H., Matenar, C., & Thrun, S. (1999). Sonar-based mapping of large-scale mobile robot environments using EM. *In Proceedings of the International Conference on Machine Learning.* Bled, Slovenia.

Byrne, E. A., & Parasuraman, R. (1996). Psychophysiology and adaptive automation. *Biol. Psychol.*, vol. 42, 249-268.

Cacciola, S. (2007). *Fusion of Laser Range-Finding and computer vision data for traffic detection by autonomous vehicles.* Blacksburg, VA: Master's Thesis, Virginia Tech.

Castellanos, J. A., & Tardios, J. D. (2000). Mobile robot localization and map building: A multi sensor fusion approach. Boston, MA: Kluwer academic publishers.

Chang, Y., Kuwabara, H., & and Yamamoto, Y. (2008). Novel Application of a Laser Range Finder with Vision System for Wheeled Mobile Robot. *Proc. of IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics*, 280-285.

Chatila, R., & Laumond, J. P. (1985). Position referencing and consistent world modeling for mobile robots. *In Proceedings of the 1985 IEEE International Conference on Robotic sand Automation*.

Chen, J., Haas, E., Pillalamarri, K., & Jaconbson, C. (2006). *Human-robot interface: Issues in Operator performance, interface design, and technologies.* Aberdeen Proving Ground, MD: U. S. Army Research Laboratory.

Cheng, Y. (1998). Mean shift, mode seeking, and clustering. *IEEE Trans on pattern analysis and machine intelligence*, 17(8): 790-799.

Choi, Y., and Hong, J., & Park, K. (2007). Obstacle avoidance using active window and flexible vector field with a laser range finder. *Int. Conf. on Control, Automation and Systems*, 2123-2128.

Clark, H. (1996). *Using language.* New York, NY: Cambridge University Press.

Cohen, P., & Levesque, H. (1991). Teamwork. *Nous*, 35.

Comaniciu, D., Ramesh, V., & Meer, P. (2000). Real-time tracking of non-rigid objects using mean shift. *IEEE Trans on computer vision and pattern recognition*, 673-678.

Cook, K. L. (2007). The silent force multiplier: The history and role of UAVs in warfare. *Aerospace conference, 2007 IEEE*, (pp. 1-7).

Cooke, N., DeJoode, J., Pedersen, H., Gorman, J., O.Connor, O., & AnKiekel, P. (2004). The role of individual and team cognition in uninhabited air vehicle command-and-control. *American: Arizona State University*, 44-53.

Cooke, N., Pedersen, H., Connor, L., Gorman, J., & Andrews, D. (2006). Acquiring team-level command and control skill for uav operation. *Human Factors of Remotely Operated Vehicles*, vol. 7, 285-297.

Cooper, J., & Goodrich, M. A. (2008). Towards combining UAV and sensor operator roles in UAV-enabled visual search. *In Proceedings of the 3rd ACM/IEEE International conference on Human-Robot Interaction*, (pp. 351-358). New York, NY.

Cowings, P., Toscano, W., DeRoshia, C., & Tauson, R. (1999). Effects of command and control vehicle (C2V) operational environment on Soldier health and performance. San Jose, CA: National Aeronautics and Space Administration Ames Research Center.

Cummings, M., & Guerlain, S. (n.d.). *Human performance issues in supervisory control of autonomous airborne vehicles.* Retrieved August 4, 2008, from http://web.mit.edu/aeroastro/ www/ people/missyc/pdfs/AUVSI_Cummings.pdf

Darken, R., & Peterson, B. (2001). Spatial Orientation, Wayfinding and Representation. *Handbook of Virtual Environment Technology.* Mahway, NJ: Lawrence Erlbaum Associates.

De Visser, E. J. (2008). Designing an adaptive automation system for human supervision of unmanned vehicles: A bridge from theory to practice. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 52(4)*, 221-228.

De Visser, E., LeGoullon, M., Freedy, A., Freedy, E., Weltman, G., & Parasuraman, R. (2008). A design methodology for contralling, monitoring, and allocating unmanned vehicles. *3rd International Conference on Human Centered Processes.* Delft: Proceeding of Supervisory Control in Critical Systems Management Workshop.

Dempster, A. P., Laird, A. N., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of The Royal Statistical Society ,Series B,39(1)*, 1-38.

Department of Defense. (2004). *Defense Acquisition Guide.* Washington, DC: Office of the Secretary of Defense.

Department of Defense. (2009). *Unmanned Systems Integrated Roadmap 2009-2034.* Washington, DC: Office of the Secretary of Defense.

Dixon, S. R., & Wickens, C. D. (2003). Imperfect Automation in Unmanned Aerial Vehicle Flight Control. Savoy, IL: Institute of Aviation.

Drury, J. L., Keyes, B., & Yanco, H. A. (2007). LASSOing HRI: Analyzing Situation Awareness in Map-Centric and Video-Centric Interfaces. *Proceedings of the Second Annual ACM/IEEE Conference on Human-Robot Interaction*.

Drury, J., Scholtz, J., & Yanco, H. (2003). Awareness in human-robot interactions. *In Proceedings of th eIEEE Conference on Systems, Man and Cybernetics.* Washington, DC.

Elinas, P. (2007). Stereo Vision SLAM near real-time learning of 3D point-landmark and 2D occupancy grid maps using particle filters.

Ellis, S. R., Mania, K., Adelstein, B. D., & Hill, M. I. (2004). Generalizeability of Latency Detection in a Variety of Virtual Environments. *Proceedings of the 48th Annual Meeting of the Human Factors and Ergonomics Society*, 2083-2087.

Endsley, M. R. (1988). Design and evaluation for situation awareness enhancement. *in Proceedings of the Human Factors Society 32nd Annual Meeting* (pp. 97-101). Santa Monica, CA: Human Factors and Ergonomics Society.

Everett, H. (1995). Sensors for Mobile Robot Systems: Theory and Application. Wellesley, MA: A. K. Peters.

Faltings, B. (1987). Qualitative kinematics in mechanisms. *In Proceeding of the International Joint Conference on Artificial Intelligence 87.* Milan, Italy.

Faugeras, O. D. (1992). What can be seen in three dimension swith an uncalibrated stereo rig? *Second European Conference on Computer Vision*, (pp. 563-578). Santa Margherita Ligure, Italy.

Ferguson, R. W., Rasch, R. A., Turmel, W., & Forbus, K. D. (2000). Qualitative Spatial Interpretation of Course-of-Action Diagrams . *Proceedings of the 14th International Workshop on Qualitative Reasoning.* Morelia, Mexico.

Ferland, F., Pomerleau, F., Dinh, C., & Michaud, F. (2009). Egocentric and exocentricteleoperation interface using real-time, 3D video projection. *In Proceedings of the 4th ACM/IEEE International Conference on Human-Robot Interaction* (pp. 37-44). La Jolla, CA: ACM.

Fong, T., Thorpe, C., & Baur, C. (2002). Multi-robot remote driving with collaborative control. *IEEE Transactions on Industrial Electronics*.

Forbus, K. D. (1995). Qualitative spatial reasoning framework and frontiers. In J. Glasgow, N. Narayanan, & B. Chandrasekaran, *Diagrammatic Reasoning: Cognitive and Computational Perspectives* (pp. 183-202). MIT Press.

Fox, D., Ko, J., Konolige, K., Limketkai, B., & Stewart, B. (2006). Distributed multi-robot exploration and mapping. Proc.oftheIEEE,94(7),2006.SpecialIssueonMulti-Robot Systems. *in Proceeds of IEEE 2006.Special Issue on Multi-Robot Systems*.

French, J., Ghirardelli, T. G., & Swoboda, J. (2003). The effect of bandwidth on operator control of an unmanned ground vehicle. *in Proceeds of I/ITSEC.* Orlando, FL.

Gage, D. W. (1995). UGV History 101: A Brief History of Unmanned Ground Vehicle (UGV) Development Efforts. *Unmanned Systems Magazine 13(3)*, 9-16.

Gockley, R., Bruce, A., Forlizzi, J., Michalowski, M., Mundell, A., Rosenthal, S., et al. (2005). Designing robots for long-term social interaction. *Intelligent Robots and Systems*, 1338-1343.

Goodrich, M. A., & Schultz, A. C. (2007). Human-Robot Interaction: A Survey. *Foundations and Trends in Human-Computer Interaction, 1(3)*, 203-275.

Grimson, W. E. (1993). *Why Stereo Vision Is Not Always About 3D Reconstruction.* Cambridge, MA: Technical Report A.I. Memo No. 1435, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Grisetti, G., Stachniss, C., & Burgard, W. (2007). Improved techniques for grid mapping with Rao-Blackwellized particle filters. *IEEE Transactions or Robotics, 23*(1), 34-46.

Haehnel, D., Fox, D., Burgard, W., & Thrun, S. (n.d.). A highly efficient Fast-SLAM algorithm.

Hager, G. (1990). *Task directed sensor fusion and planning.* Kluwer Academic Publishers.

Hahnel, D. (2004). *Mapping with mobile robots.* Freiburg, Germany: Unpublished PhD Thesis, University of Freidburg.

Hahnel, D., Fox, D., Burgard, W., & Thrun, S. (2003). An efficient Fast SLAM Algorithm for Gerating Maps of Large0scale cyclic environments from raw laser range measurements. *in proceedins of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 206-211.

Hebert, M., Unnikrishnan, R., & Pantofaru, C. (2007). Toward objective evaluation of image segmentation algorithms. *IEEE Transactions on pattern analysis and machine intelligence, 29(6)*, 929-944.

Hitec. (2011). Retrieved 2009, from www.hitecrcd.com

Hoffman, G., & Breazeal, C. (2004). Collaboration in human-robot teams. *1st AIAA Intelligent Sysgtems Conference.* Chicago, IL.

Hokuyo. (2011). *Hokuyo Automatic Co., LTD.* Retrieved 2009, from www.hokuyo-aut.jp/02sensor

Hollands, J., & Wickens, C. D. (1999). *Engineering psycology and human performance, 3rd edition.* New Jersey: Prentice Hall.

Howard, A., Wolf, D. F., & Sukhatme, G. S. (2004). Towards 3d mapping in large urban environments. *in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots andSystems (IROS).* Sendai, Japan.

Ikeda, S., & Miura, J. (2006). 3D indoor environment modeling by a mobile robot with omnidirectional stereo and laser range finder. *in Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (pp. 3435-3440).

Inagaki, T. (2003). Adaptive automation: Sharing and trading of control. In *Handbook of Cognitive Task Design* (pp. 147-169).

Iocchi, L., & Pellegrini, S. (2007). Building 3D maps with semantic elements integrating 2D laser, stereo vision and INS on a mobile robot. *in Proceedings from ISPRS International Workshop 3D-ARCH*.

Kaber, D. B., & Endsley, M. R. (2004). The effects of level of automation and adaptive automation on human performance, situation awareness and workload in a dynamic control task. *Theoretical issues iin Ergonomics Science, 5(2)*, 113-153.

Kaber, D. B., & Riley, J. M. (1999). Adaptive automation of a dynamic control task based on secondary task workload measurement. *International Journal of Cognitive Ergonomics*, 169-187.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transaction of the ASME-Journal of Basic Engineering*, 35-45.

Kanade, T. (1989). *Three-dimentsional machine vision.* Boston, MA: Kluwer Academic Publishers.

Kaupp, T., Makarenko, A., & Durrant-Whyte, H. (2010). Human-robot communication for collaborative decision making - a probabilistic approach. *Robotics and Autonomous Systems* , 444-456.

Klappstein, J., Vaudrey, T., Rabe, C., Wedel, A., & Klette, R. (2009). Moving object segmentation using optical flow and depth information. *In 34e Pacific-Rim Symposium on Image and Video Technology (PSIVT)*, 611-623.

Kogut, G. e. (2007). Sensor fusion for intelligent behavior on small unmanned ground vehicles. *in Proceedings from SPIE*.

Kohler, C., Ottlik, A., Nagel, H.-H., & Nebel, B. (2004). Qualitative reasoning feeding back into quantitative model-based tracking. *European Conference of Artificial Intelligence*, (pp. 1041-1042).

Krueger, G. (1991). Sustaining military performance in continuous operations: combatant fatigue, rest and sleep needs. *Handobook of Military Psychology* (pp. 255-277). New York, NY: Wiley and Sons.

Kuipers, B. (2000). The spatial semantic hierarchy. *Artificial Intelligence 119*, 191-233.

Lane, J. C., Carignan, C. R., Sullivan, B. R., Akin, D. L., Hunt, T., & Cohen, a. R. (2992). Effects of time delay on telerobotic control of neutral buoyancy vehicles, 2002, vol. 3, pp. 2874–2879. *in Proceedings of 2002 IEEE International Conference Robotic Automation*, 2874-2879.

Lee, H. K., & Cho, H. S. (2007). Stereo moire technique: A novel 3D measurement method using a stereo camera and a digital pattern projector. *International Journal of Optomechatronics, 1(2)*, 209-230.

Lee, J. D., & See, K. A. (2004). Trust in automation: designing for appropriate reliance. *Human Factors, 46*, 50-80.

Liang, C. K., & Chen, H. N. (2005). Rolling shutter distortion correction. *in Proceedings from SPIE Visual Communications and Image Processing.* Beijing, China.

Lingemann, K., & al., e. (2005). High-speed laser localization for mobile robots. *Robotics and autonomous systems*, 275-296.

Liu, J., & Daneshmend, L. (2003). *Spatial reasoning and planning: geometry, mechanism, and motion.* Berlin, Germany: Sprierng.

Liu, Y., Emery, R., Chakrabarti, D., Burgard, W., & Thrun, S. (2001). Using EM to learn 3D models of indoor environments with mobile robots. *in Proceedings from 18th Conference on Machine Learing.* Williamstown, MA: Williams College.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. *Proceedings of the International Conference on Computer Vision. 2*, (pp. 1150-1157).

Lucas, B., & Kanade, T. (1981). An iterative image registratioin technique with an application to stereo vision. *Proceedings from DARPA Image Understanding Workshop*, 121-130.

Lundberg, C., Christensen, H. I., & Hedstrom, A. (2005). The use of robots in harsh and unstructured field applications. 143-150.

MacKenzie, I., & Ware, C. (1993). Lag as a determinant of human performance in interactive systems. *Conference on Human Factors in Computing Systems*, (pp. 488-493). New York, NY.

Meehan, M., Razzaque, S., Whitton, M., P., F., & Brooks, J. (2003). Effect of Latency on Presence in Stressful Virtual Environments. *in Proceedings of the IEEE Virtual Reality (IEEE Computer Society)*.

Meyer, F., & Beucher, S. (1990). Morphological segmentation. *Journal of Visual Communication and Image Representation, 1*, 21-46.

Mohr, B. B. (2008). Micro air vehicle navigation system. *IEEE Aerospace and Electronic Systems Magazine, 23(4)*, 19.

Molloy, R., & Parasuraman, R. (1996). Monitoring an automated system for an single failure: vigilance and task complexity effects. *Human Factors, 38*, 311-322.

Moravec, H. P. (1988). Sensor fusion in certainty grids for mobile robots. *AI Magaizine, 9(2)*, 61-74.

Moravec, H. P., & Martin, M. C. (1994). Robot navigation by 3D spatial evidence grids. *Mobile Robot Laboratory, Robotics Institute, Carnegie Mellon University*.

Mouloua, M., Gilson, J., Kring, R., & Hancock, P. (2001). Workload, situation awareness, and teaming issues for uav/ucav operations. *in Proceedings of the Human Factors and Ergonomics Society, 45*, 162-165.

Murphy, R. (2000). *Introduction to AI Robotics.* Cambridge, MA: MIT Press.

Murphy, R. (2004). Human-robot interaction in rescue robotics. *IEEE Trans. Systems, Man, Cybernetics, 34(2)*, 138-153.

Murphy, R., Pratt, K., & Burke, J. (2008). Crew roles and operational protocols for rotary-wing micro-UAVs in close urban environments. *ACM SIGCHI/SIGART Human-Robot Interaction*, 73-80.

Nevado, M. M., Garcia-Bermejo, J. G., & Casanova, E. Z. (2004). Obtaining 3D models of indoor environments with a mobile robot by estimating local surface directions. *Robotics and Autonomous Systems, 48*, 131-143.

Nguyen, V., Martinelli, A., Tomatis, N., & Siegwart, R. (2005). A comparison of line extractio nalgorithms using 2D laser rangefinders for indoor mobile robotics. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (pp. 1929-1934).

Nuechter, A., Wulf, O., Lingemann, K., Hertzberg, J., Wagner, B., & Surmann, H. (2006). 3D mapping with semantic knowledge. *Bredenfeld, A., Jaco, A., Noda, I. Takahashi, Y., eds: Robo Cub-2005: Robot Soccer World Cup IX, vol. 4020 of Lecture Notes in Artificial Intelligence.* Springer Verlag.

Okubo, Y., Ye, C., & Borenstein, J. (2009). Characterization of the Hokuyo URG-04LX laser rangefinder for mobile robot obstacle negotiation. *Proceedins of SPIE, 7332*, 733212.

Olsen, D., & Goodrich, M. (2003). Metrics for evaluating human-robot interactions. *in Proceedings of NIST Performance Metrics for Intelligent Systems Workshoop*.

Oppermann, R. (1994). *Adaptive User Support.* Hillsdale, NJ: Lawrence Erlbaum.

Orr, M., Rasmussen, S., Karni, E., & Blake, W. (2005). Framework for developing and evaluating MAV control algorithms in a realistic urban setting. *American Control Conference 2005.* Portland, OR.

Parasuraman, R., Sheridan, T., & Wickens, C. (2000). A model for types and levels of human interaction with automation. *IEEE Transactions on Systems, Man and Cybernetics B, 30*, 286-297.

Phidgets. (2011). Retrieved 2010, from www.phidgets.com

Poppinga, J., Birk, A., & Pathak, K. (2008). *A characterization of 3D sensors for response robots.* in http://www.robocup2009.org.

Premebida, C., & Nunes, U. (2005). Segmentation and geometric primitives extraction from 2D laser range data for mobile robot applications. *in Robotica 2005 Scientific meeting of the 5th national Robotics Festival.* Coimbra, Portugal.

Raskin, J. (2000). *The humane Interface: New Directions for designing interactive systems.* Reading, MA: Addison-Wesley.

Rastogi, A. (1996). *Design of an interface for teleoperation in unstructured environments using augmented reality displays.* Toronto, CA: Master's Thesis, University of Toronto.

Renz, J., & Mitra, D. (2004). Qualitative direction calculi, with arbitrary granularity. *Trends in Artificial Intelligence, 8th Pacific Rim International Conference on Artificial Intelligence*, (pp. 65-74).

Sarter, N. B., Woods, D. D., & Billings, C. E. (1997). Automation Surprises. *in Salvend, G. ed. Handobook of Human Factors and Ergonomics 2nd e.* (pp. 1926-1943). New York, NY: John Wiley & Sons.

Scholtz, J. (2003). Theory and evaluation of human robot interactions. *in Proceeds of Hawaii International COnference on System Science, 36*.

Skubic, M., Bailey, C., & Chronis, G. (2003). A sketch interface for mobile robots. *in Proceedings of IEEE Conference Systems, Man and Cybernetics*, (pp. 918-924). Washington, D.C.

Smith, R., Self, M., & Cheesman, P. (1990). Estimating uncertain spatial relationships in robotics. In I. J. Cox, & G. T. Wilfong, *Autonomous Robot Vehicles* (pp. 167-193). New York, NY: Springer-Verlag.

Sohn, H., & Kim, B. (2005). A robust localization algorithm for mobile robots with laser range finders. *in Proceedins of IEEE International Conference on Robotics and Automation*, 3545-3550.

Sohn, H., & Kim, B. (2008). An efficient localization algorithm based on vector matching for mobile robots using laser range finders. *Journal for Intelligent Robotic Systems, 51*(4), 461-488.

Stamos, I., & Allen, P. K. (2002). Geometry and texture recovery of scenes of large scale. *Computer Vision and Image Understanding, 88*, 94-118.

Surmann, H., Nuchter, A., & Hertzberg, J. (2003). An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. In *J. Robotics and Autonomous Systems, 45(3-4)* (pp. 181-198).

Talukder, A., Goldberg, S., Matthies, L., & Ansar, A. (2003). Real-time detection of moving objects in a dynamic scene from robotic vehicles. *in Proceedings of the International Conference on Intelligent Robots and Systems, 2*, 1308-1313.

Task Force Devil Combined Arms Assessment Team. (2003). *The modern warior's combat load.* Washington, DC: U.S. Army Center for Army Lessons Learned .

Tate, D., & Smith, A. (1995). A genetic approach to the quadratic assignment problem. *Computers and Operations Research, 22*(1), 73-78.

Taylor, T. (2009). *Mapping of indoor environments by robots using low-cost vision sensors.* Unpublished PhD Thesis, QUT.

Thakoor, N., Gao, J., & Jung, S. (2010). Embedded planar surface segmentation system for stereo images. *Machine Vision and Applications, 21(2)*, 189-199.

Thau, R. S. (1997). *Reliably mapping a robot's environment using fast vision and local, but not global, metric data.* Unpublished PhD Thesis, MIT.

Thomas, L. C., & Wickens, C. D. (2000). *Effects of display frames of reference on spatial judgements and chage detection.* Champaign, IL: University of Illinois Urbana.

Thrun, S. (2002). Robotic mapping: A survey. *Exploring artificial intelligence in the new millenium.* Morgan Kaufmann.

Thrun, S., Martin, C., Liu, Y., Hahnel, D., Emery-Montemerlo, R., Chakrabarti, D., et al. (2004). A real-time expectation maximization algorithm for acquiring mulit-planar maps of indoor environments with mobile robots. *IEEE Transactions on Robotics and Automation, 20(3)*, 433-443.

Tittle, J. S., Roesler, A., & Woods, D. D. (2002). The remote perception problem. *in Human Factors and Ergonomics Society 46th Annual Meeting.* Baltimore, MD.

Trucco, E., Isgro, F., & Bracchi, F. (2003). Plane detection in disparity space. *in International Conference on Visual Information Engineering*, 73-76.

Tvaryanas, A. P., Thompson, W. T., & Constable, S. H. (2005). *The U. S. military unamnned aerial vehicle (UAV) experience: Evidence-based human systems integration lessons learned.* Brooks City-Base, TX: USAF 311th Performance Enhancement Directorate.

Ueda, T., & al., e. (2006). Mobile SOKUKI sensor system: accurate range data mapping system with sensor motion. *International Conference on Autonomous Robots and Agents*, 309-314.

United States Air Force Scientific Advisory Board. (2004). *Human systems integration in Air Force weapon systems development and acquisition.* Washington, DC: Department of the Air Foce, Department of Defense.

Van Erp, J. B., & Padmos, P. (2003). Image parameters for driving within direct viewing systems. *Ergonomics, 46*, 1471-1499.

Vedaldi, A., & Fulkerson, B. (2008). *VLFeat: An open and portable library of computer vision algorithms*. Retrieved 7 13, 2011, from http://www.vlfeat.org

Veltkamp, R. C., & Hagendoorn, M. (2000). State-of-the-art in shape matching. *Multimedia Search: State of the Art.* Springer-Verlag.

Videre-design. (2010). Retrieved 2009, from www.videredesign.com

Weiss, G., Wetzler, C., & Puttkamer, E. (1994). Keeping track of position and orientatin of moving indoor systems by correlation of range-finder scans. *In proceds of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 12-16.

Williams, K. W. (2004). *A summary of unmanned aerial aircraft accident/incident data: Human factors implications.* Washington, DC: U. S. Department of Transportation, Federal Aviation Administration, Office of Aerospace Medicine.

Wilson, G. F., & Russell, C. (2003). Real-time assessment of mental workload using psychophysiological measures. *Human Factors, 45*, 635-643.

Wulf, O., Brenneke, C., & Wagner, B. (2004). Colored 2D maps for robot navigation with 3D sensor data. *in IEEE/RSJ International Conference on Intelligent Robots and Systems, 3*, 2991-2996.

Yamauchi, B., & Beer, R. (1996). Spatial learning for navigation in dynamic environments. *in IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics, Special Issue on Learning Autonomous Robots*.

Yamauchi, B., & Langley, P. (1997). Place recognition in dynamic environments. *Journal of Robotic Systems, 14*(2), 107-120.

Yanco, H. A., & Drury, J. L. (2002). *A taxonomy for human-robot interaction.* Falmouth, MA: Proceedings of the AAAI 2002 Fall Symposium on Human-Robot Interaction.

Yanco, H. A., Drury, J. L., & Scholtz, J. (2004). Beyond usability evaluation: analysis of human-robot interaction at a major robotics competition. *Journal of Human-Computer Interaction, 19(1-1)*, 117-149.

Ye, C., & Borenstein, J. (2003). A new terrain mapping method for mobile robots obstacle negotiation. *in Proceedings of SPIE*, 52-62.

Yuen, D. C., & MacDonald, B. A. (2004). Theoretical considerations of multiple particle filters for simultaneous localisation and map-building. *Knowledge-based Intelligent Information and Engineering Systems, Lecture Notes in Artificial Intelligence* (pp. 203-209). Berlin, Germany: Springer Verlag.

Zeng, S., & Weng, J. (2007). Online-learning and attention-based approach to obstacle avoidance using a range finder. *Journal of intelligent robot systems, 50*, 219-239.

Zhang, H., Fritts, J. E., & Goldman, S. A. (2008). Image segmentation evaluation: a survey of unsupervised methods. *Computer Vision Image Understanding, 110*(2), 260-280.

Zhang, L., & Ghosh, B. K. (2000). Line segment based map building and localization using 2D laser range finder. *in Proceedings of the 2000 IEEE Internationa Conference on Robotics adn Automation*, 2538-2543.

Appendix A: MATLAB Code

The following algorithms are included for understanding of the methods discussed in this dissertation. Certain components needed to run this code are not necessarily included. Please contact the author if further explaination is needed or for help with code implementation. This code is the original work of the author and cannot be used without permission, citation, and acknowledgement.

GUI

MainGUI—videre_lite_SAVE.m

```
function varargout = videre_lite_SAVE(varargin)
% Last Modified by GUIDE v2.5 15-May-2011 11:47:22

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                'gui_Singleton',  gui_Singleton, ...
                'gui_OpeningFcn', @videre_lite_SAVE_OpeningFcn, ...
                'gui_OutputFcn',  @videre_lite_SAVE_OutputFcn, ...
                'gui_LayoutFcn',  [] , ...
                'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before videre_lite_SAVE is made visible.
function videre_lite_SAVE_OpeningFcn(hObject, eventdata, handles, varargin)

set(handles.startvideo, 'String', 'Start Video')
cmat('init');%initialize stereo camera
cmat('readParamFile', '/home/gaines90/svs/data/gaines2.ini');%init
handles.openerror=cmat('open');
%initialize array for data storage
handles.left=zeros(480, 640, 3);%image output must be 3 channel
handles.right=zeros(480, 640);%right image is only used for disparity
handles.disparity=zeros(480, 640);
handles.detailscan=zeros(14,182);
handles.dist_disp=0;
disp('Opening Function line 67')
% Choose default command line output for videre
handles.output = hObject;
set(handles.output, 'CurrentCharacter', '1');%exit condition on keyboard input
%setup figures for hokuyo
axis(handles.rhokuyo, [-2500 6500 -4500 4500]);%the axis set is to
%optimize for continuous view
% Update handles structure
```

```
guidata(hObject, handles);
% --- Outputs from this function are returned to the command line.

function varargout = videre_lite_SAVE_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;
% --- Executes on button press in camswitch.

function camswitch_Callback(hObject, eventdata, handles)

function hokuyoangle_Callback(hObject, eventdata, handles)

function hokuyoangle_CreateFcn(hObject, eventdata, handles)

if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor',[.9 .9 .9]);
end

% --- Executes on button press in startvideo.
function startvideo_Callback(hObject, eventdata, handles)

set(handles.click_location, 'String', ['80';'60']);
handles.labels=repmat('_', 10);
handles.labels_k=1;

if get(handles.servopower, 'Value')==2
%Setup parameters for servo control
   handles.handle=servo_init;
end
if get(handles.hokuyopower, 'Value')==2
   xy1=hokuyo_init;
   if get(handles.servopower, 'Value')==2
      set(handles.hokuyo_detail, 'Visible', 'on');
   end
end
compnum=0;
handles.angles=(compnum-135):.3515625:(compnum+135);% what it is
%inform user how to stop video
set(handles.startvideo, 'String', 'SPACE to Stop')
disp('Camera Running...........')
cmat('start');
   dispalt=0;%This is the toggle variable that alternates disparity
   handles.currentIm=imagesc(handles.left, 'Parent',handles.leftcam);
   colormap(handles.leftcam,hot)
   axes(handles.leftcam)%makes leftcam axes current
   axis off
   hr=rectangle('Parent', handles.leftcam,'Position', [240 180 160-1 120-1]);
   set(hr, 'EdgeColor', 'k');
set(handles.click_location, 'String', ['80';'60']);
set(handles.currentIm, 'ButtonDownFcn', @(hObject, eventdata)videre_lite_SAVE(...
   'leftcam_ButtonDownFcn',hObject,eventdata,guidata(hObject)));
r=zeros(769, 1);%initialize servo values
x=zeros(769,1);
y=zeros(769,1);
scanparam=25;%%These values are for the sweeping behavior of the servo
scanwindow=scanparam*-1;%sets up scan window for 20 degrees
%OPTIONAL COMPASS
compassport='/dev/ttyUSB0';
   %setup compass
   if get(handles.compasspower, 'Value')~=1
      ports=instrfind;
      delete(ports);
      s=serial(compassport, 'BaudRate', 9600);
      fopen(s);
      fprintf(s, 'b=8');%this sets the BaudRate
      ports=instrfind;
```

```
        delete(ports);
        s=serial(compassport, 'BaudRate', 57600);
        fopen(s);

        fprintf(s, 'c?');
        fscanf(s);
        compval=fscanf(s);
        compval=textscan(compval, '$C%4c');
        set(handles.compass, 'String', compval);
    end
%%Only adjust these values when system is stopped
set(handles.compasspower, 'Visible', 'off');
set(handles.hokuyopower, 'Visible', 'off');
set(handles.servopower, 'Visible', 'off');
%So the plot function is always replace for the hokuyo
set(handles.rhokuyo, 'NextPlot', 'replacechildren');
click_location_num=[320;240];
set(handles.click_location, 'String', ['0';'0']);
xoffset=0;
yoffset=0;
set(handles.status, 'String', 'Pose');

if handles.openerror==-1
    set(handles.status, 'String', 'No Data');
    disp('No camera detected')
end

%initialize from first image
[localIm_background localRect_background]=imcrop(handles.left, [0 0 160-1 120-1]);
[handles.fprev handles.dprev]=vl_sift(single(rgb2gray(localIm_background)*255));
%initialize importance values for 10 stored importance objects
localIm_template_important=zeros(60,80,3,10);
localIm_dprev_important=uint8(zeros(128, 100, 10));
localIm_fprev_important=zeros(4,100,10);
val_important=zeros(30,10);
freq_important=zeros(30,10);
full='no';
set(handles.output, 'UserData', '0')

object(10,3).deltax=0;
object(10,3).deltay=0;
object(10,3).template=zeros(60,80,3);
object(10,3).dprev=uint8(zeros(128,100));
object(10,3).fprev=zeros(4,100);
object(10,3).colorval=zeros(30,1);
object(10,3).colorfreq=zeros(30,1);
object(10,3).detail=0;
object(10,3).label='no label';
object(10,3).label_match='no match';
beacon(20).number=0;
beacon(20).deltax=0;
beacon(20).deltay=0;

color_weight=[2;3;5];
match_timeout=0;

n=1;%importance index for storing features of important objects
k=1;%object storage index for objects found to be important
p=1;%beacon storage index for beacons.  All unimportant objects
t=1;%time index for time from file
handles.z=1;%area storage index
centroid=[0,0];
val=0;%
freq=0;%
min_Bhatt=0;
%set location for buffer video
```

```
buffer_loc='/home/gaines90/Desktop/FINAL1';
fid_color = fopen(fullfile(buffer_loc, 'left.bin'), 'r+');
fid_disparity = fopen(fullfile(buffer_loc,'disparity.bin'), 'r+');
fid_hokuyo = fopen(fullfile(buffer_loc, 'hokuyo.bin'), 'r+');
fid_right = fopen(fullfile(buffer_loc, 'right.bin'), 'r+');

while get(handles.output, 'CurrentCharacter') == '1'
  modeswitch = get(handles.camswitch, 'Value');
     click_location=get(handles.click_location, 'String');
     if click_location(1)~='0'
        tic
        flag5=0;
        flag30=0;
        click_location_num=str2num(click_location);
        set(handles.click_location, 'String', ['0';'0']);
        [localIm_template, localRect_template]=imcrop(handles.left,...
           [click_location_num(1)-40 click_location_num(2)-30 80-1 60-1]);
        [localIm_background localRect_background]=imcrop(handles.left,...
           [click_location_num(1)-80 click_location_num(2)-60 160-1 120-1]);

        if get(handles.buffer, 'Value')==1
           cmat('getDisparity', handles.disparity);%SEGMENT ADDED TO CAPTURE DISPARITY OF TEMPLATE
        else
           %%%disparity already from file
        end
        localIm_disparity=imcrop(handles.disparity, localRect_template);
        localIm_disparity_background=imcrop(handles.disparity,...
           localRect_background);
        mean_disparity=mean(mean(localIm_disparity_background));

        if mean_disparity<=0
           wallscan=1;
        else
           wallscan=0;
        end

        [region,feat, M, handles.dist_disp, maxf, indf, num_feat]=...
           mser_detector_cont(localIm_disparity);
         set(handles.compass, 'String', 'Forward');

        set(handles.rhokuyo, 'Visible', 'on')%turn map on so we can see object placement
        cla(handles.hokuyo_detail_axis)

        if get(handles.hokuyopower, 'Value')==2 && ...
              get(handles.servopower, 'Value')==2
           if handles.dist_disp >=0
              theta=asin((click_location_num(2)-240)/...
                 ((handles.dist_disp)^.5*550))*180/pi;
           else
              handles.dist_disp=2;
              theta=asin((click_location_num(2)-240)/...
                 ((handles.dist_disp)^.5*550))*180/pi;
              disp('no angle')
           end

           if wallscan==1
              posn=80+theta+25;%80 is neutral
              calllib('phidget21', 'CPhidgetServo_setPosition', ...
                 handles.handle, 0, posn);
              str = num2str(posn-25);
              set(handles.angleval, 'String', str);
              %Compass querry
              if get(handles.compasspower, 'Value')~=1
                 s=instrfind;
                 fprintf(s, 'c?');%querry magnetometer
                 fscanf(s);%read blank line
```

```
            compval=fscanf(s);%read data line
            compval=textscan(compval, '$C%4c');%convert data
            set(handles.compass, 'String', compval);%display value
            compnum=str2double(compval);%convert frm string to number
            final_angle=handles.angles+compnum;%magnetometer
        else
            final_angle=handles.angles;
        end

        pause(.3)

        %after servo moves, this command takes a scan
        r=hokuyoAPI('getScan','range', 0, 768, 1, 3);
        r_out=r(311:492, :)';
        %plot the partial scan
        x=r_out.*cosd(final_angle(:,311:492));
        y=r_out.*sind(final_angle(:,311:492));
        [k_val, centroid, kdist]=kmeans([x',y'], 10,'EmptyAction','drop');
    else
        %Compass querry
        if get(handles.compasspower, 'Value')~=1

            fprintf(s, 'c?');%querry magnetometer
            fscanf(s);%read blank line
            compval=fscanf(s);%read data line
            compval=textscan(compval, '$C%4c');%convert data
            set(handles.compass, 'String', compval);%display value
            compnum=str2double(compval);%convert frm string to number
            final_angle=handles.angles+compnum;%magnetometer angle over range

        else
            final_angle=handles.angles;
        end

        %after servo moves, this command takes a scan
        r=hokuyoAPI('getScan','range', 0, 768, 1, 3);
        r_out=r(311:492, :)';

        %plut the partial scan
        x=r_out.*cosd(final_angle(:,311:492));
        y=r_out.*sind(final_angle(:,311:492));

        %plot the entire scan x=r'.*cosd(final_angle);
        %y=r'.*sind(final_angle);
        [k_val, centroid, kdist]=kmeans([x',y'], 10,'EmptyAction','drop');
    end
end
if handles.dist_disp>0
else
    disp('test No disparity value')
end
[template_fprev template_dprev]=vl_sift(single(rgb2gray(localIm_template)*255));%initialize fprev and dprev
template_fprev_default=template_fprev;%sets default values
template_dprev_default=template_dprev;
localIm_template_default=localIm_template;
xoffset=80;
yoffset=60;
set(handles.status, 'String', 'Tracking');
%find the color histogram values
[val, freq]=createColorHistograms_cont(localIm_template);
%%Right click designates objects aas important
if strcmp('alt', get(gcf, 'SelectionType'))%categorize objects importance
    if k~=1
        str_label=get(handles.output, 'UserData');
        size_str_label=size(str_label,2);
        if ~strcmp(str_label(1), '0')
        if size_str_label <10
            handles.labels(k-1, 1:size(str_label,2))=str_label;
        else
            handles.labels(k-1, :)=str_label(1,1:10);
        end
    end
elseif k==1
    handles.labels(k, 1:2)='O0';%
end
set(handles.label, 'Visible', 'on')
importance='r';
%Save features of important objects
localIm_template_important(:,:,:,n)=localIm_template;
localIm_dprev_important(1:size(template_dprev,1),...
    1:size(template_dprev,2), n)=template_dprev;
localIm_fprev_important(1:size(template_fprev,1),...
    1:size(template_fprev,2), n)=template_fprev;
val_important(1:size(val,1), 1:size(val,2), n)=val;
freq_important(1:size(freq,1), 1:size(freq,2), n)=freq;
if n<10
    n=n+1;
else
    n=2;
    full='yes';
end
if strcmp(full, 'yes')
    m=10;
else
    m=n-1;%minus 1 because n is the next iteration so the current # is -1
end
    template_id=match_compare(template_dprev, localIm_dprev_important, 8, m);
    if template_id==0%
        if k~=1;
            handles.labels(k, 1:2)=['0' num2str(k-1)];%
            set(handles.output, 'UserData', handles.labels(k,:))
        end
        object(k,1).deltax=click_location_num(1)-320;
        object(k,1).deltay=handles.dist_disp;%disparity distance
        object(k,1).template=localIm_template;%60x80x3 color template of object
        object(k,1).background=localIm_background;%120x160
        object(k,1).dprev=template_dprev;%descriptors derived from color template
        object(k,1).fprev=template_fprev;%features derived from color template
        object(k,1).colorfreq=freq;%color bin values
        object(k,1).colorval=val;%color bin types for each color freq
        object(k,1).detail=1;%was a detail scan conducted
        object(k,1).label_match='no match';%Which object in 10 object queu does the object match with, default is 'no match'

        if k<10
            k=k+1;
        else
            k=1;
        end
        set(handles.label_display, 'String', handles.labels(k))
    else
        set(handles.label_display, 'String', handles.labels(template_id, :))
        n=n-1;
        object(k,1).label=handles.labels(template_id, :);
        object(template_id, 1).label=object(k,1).label;
        object(template_id,1).deltax=click_location_num(1)-320;
        object(template_id,1).deltay=handles.dist_disp;
        object(k,1).deltax=handles.dist_disp;
        object(k,1).deltay=handles.dist_disp;
        object(k,1).label_match=template_id;

    end
```

```
        else
            importance='g';
            set(handles.label, 'Visible', 'off')
            beacon(p).number=p;
            beacon(p).deltax=click_location_num(1)-320;
            beacon(p).deltay=handles.dist_disp;

            if p<20
                p=p+1;
            else
                p=1;
            end

        end
    end

if(modeswitch~=2)

    %check to see if buffer or live video
    if get(handles.buffer, 'Value')==1 && modeswitch==1
        %set Cdata property of image object
        cmat('getImage', handles.left, handles.right);%capture left image
    elseif get(handles.buffer, 'Value')==2 && modeswitch==1
            [handles.left, handles.disparity, handles.right, ...
                count]=video_play_bin_cont_test(...
                fid_color, fid_disparity, fid_right);
            handles.left=handles.left/255;
            t=t+1;%read next compass value next iteration
            if count==0
                fid_color = fopen(fullfile(buffer_loc, 'left.bin'), 'r+');
                fid_disparity = fopen(fullfile(buffer_loc, 'disparity.bin'), 'r+');
                fid_hokuyo = fopen(fullfile(buffer_loc, 'hokuyo.bin'), 'r+');
                fid_right = fopen(fullfile(buffer_loc, 'right.bin'), 'r+');
                t=1;%reset compass from file
            end
                    r=fread(fid_hokuyo, 769, 'double');
                    x=r'.*cosd(handles.angles);
                    y=r'.*sind(handles.angles);
    elseif get(handles.buffer, 'Value')==2 && modeswitch==3
        handles.dist_disp=handles.dist_disp-.1;
        if handles.dist_disp >15
            handles.dist_disp=15;
        elseif handles.dist_disp <= 0
            handles.dist_disp=0;
            if strcmp(get(handles.compass, 'String'), 'Turn')
                handles.dist_disp=15;
                set(handles.compass, 'String', 'Forward')
            end
        end

        if handles.dist_disp>=5
            [handles.left, handles.disparity, handles.right,...
                count]=video_play_bin_cont_test(fid_color, fid_disparity, fid_right);
            handles.left=handles.left/255;
            t=t+1;%read next compass value next iteration
            if count==0
                %return
                fid_color = fopen(fullfile(buffer_loc, 'left.bin'), 'r+');
                fid_disparity = fopen(fullfile(buffer_loc, 'disparity.bin'), 'r+');
                fid_hokuyo = fopen(fullfile(buffer_loc, 'hokuyo.bin'), 'r+');
                fid_right = fopen(fullfile(buffer_loc, 'right.bin'), 'r+');
                t=1;%reset compass from file
            end
                    r=fread(fid_hokuyo, 769, 'double');
                    x=r'.*cosd(handles.angles);
                    y=r'.*sind(handles.angles);
    else
    end

elseif get(handles.buffer, 'Value')==1 && modeswitch==3
    set(handles.buffer, 'Value',2)
    disp('line 645 set to buffer mode')

end

set(handles.currentIm, 'CData', handles.left)

if xoffset==80%Condition for template matching based on selected objects.
    if get(handles.tracking_mode, 'Value')~=2%Not on the first iteration with color info and not on SIFT mode
        [ind_Bhatt, octal, Bhatt, min_Bhatt, dist_d]=shiftIm_octal(...
            localIm_background, val, freq);%%%%%%%%%%%%%%%%
        if min_Bhatt<=.1
            color_const=0;
        elseif min_Bhatt<=.2
            color_const=color_weight(1);
        elseif min_Bhatt<=.3
            color_const=color_weight(2);
        elseif min_Bhatt<=.4
            color_const=color_weight(3);
        else
            color_const=0;
        end

        switch ind_Bhatt
            %%First 5 cases are for central N, S, E, and W.
            %%Last 4 cases are for corners, when applicable
            %%from shiftIm_octal
            case 1%North image more prevalen correct up
                change_color=[0;color_const];%method 1
                %                       disp('/\')
            case 2%East Image more prevalent correct to the right
                change_color=[-color_const;0];%method 1
                %                       disp('--->')
            case 3%middle image more prevalent therefore do nothing
                change_color=[0;0];
                %disp('O')
            case 4%West Image more prevalent correct to the left
                change_color=[color_const;0];%method 1
                %                       disp('<---')
            case 5%South image more prevalen correct down
                change_color=[0;-color_const];%method 1
                %                       disp('\/')
            case 6%North West image more prevalent
                change_color=[color_const;color_const];%met1
                %                       disp('/\<---')
            case 7%North East image more prevalent
                change_color=[-color_const;color_const];%met1
                %                       disp('/\--->')
            case 8%South East image more prevalent
                change_color=[-color_const;-color_const];%m1
                %                       disp('\/--->')
            case 9%South West image more prevalent
                change_color=[color_const;-color_const];
                %                       disp('\/<---')
        end

        %MODE to see color contribution
        click_location_num=click_location_num-change_color;

        [localIm_background localRect_background]=imcrop(handles.left, ...
            [click_location_num(1)-80 click_location_num(2)-60 160-1 120-1]);
        delete(hr)
    hr=rectangle('Parent', handles.leftcam,'Position', localRect_background);
    set(hr, 'EdgeColor', importance);

end

%crops the new image for SIFT features
[localIm_background localRect_background]=imcrop(handles.left, ...
    [click_location_num(1)-80 click_location_num(2)-60 160-1 120-1]);

if get(handles.tracking_mode, 'Value')~=3%
    [handles.fprev, handles.dprev, match, change, change_all]=sift_detector_cont(...
        rgb2gray(localIm_background)*255, template_fprev, template_dprev, ...
        localRect_background, xoffset, yoffset);
    change=kmean_change(change_all, change);
    if isnan(change)%if no features available use color information
        if get(handles.tracking_mode, 'Value')==1
            change=change_color;%method1
        else
            change=[0;0];
        end
    end
    delete(hr)
    hr=rectangle('Parent', handles.leftcam,'Position', localRect_background);
    set(hr, 'EdgeColor', importance);

    click_location_num=click_location_num-change;
end
set(handles.status, 'String', 'Tracking');

if size(match, 2)==0 && flag5==1 && flag30==0
    disp('reset clock')
    flag5=0;
    tic
end

if flag5==0 && floor(toc) == 5 && importance=='r'
    if size(match,2)>=2
        if k==1
            q=10;%q is previous k
        else
            q=k-1;
        end
        disp('5 seconds mature')
        object(q,2).deltax=click_location_num(1)-320;

        if get(handles.buffer, 'Value')==1
            cmat('getDisparity', handles.disparity);
        else
            %%%disparity already from file
        end

        [localIm_q localRect_q]=imcrop(handles.left, ...
            [click_location_num(1)-40 click_location_num(2)-30 80-1 60-1]);
        localIm_disparity_q=imcrop(handles.disparity, localRect_q);
        [region,feat, M, handles.dist_disp_q, maxf, indf, num_feat]=...
            mser_detector_cont(localIm_disparity_q);
        [template_fprev_q template_dprev_q]=vl_sift(...
            single(rgb2gray(localIm_q)*255));%initialize fprev and dprev
        [val_q, freq_q]=createColorHistograms_cont(localIm_q);

        object(q,2).deltay=handles.dist_disp_q;%
        object(q,2).template=localIm_q;%60x80x3 color template of object
        object(q,2).dprev=template_dprev_q;%descriptors derived from color template
        object(q,2).fprev=template_fprev_q;%features derived from color template
        object(q,2).colorfreq=freq_q;%color bin values
        object(q,2).colorval=val_q;%color bin types for each color freq
```

```
            object(q,2).detail=1;%was a detail scan conducted of the object
            object(q,2).label=object(k,1).label;
            flag5=1;%ensures this loop is only checked after 5 seconds has passed

        else
            flag5=2;%designates object no longer mature
            flag30=2;%designates object no longer mature
        end
    end

    if flag30==0 && floor(toc) == 30 && importance=='r'
        if size(match,2)>=2
            if k==1
                q=10;%q is previous k
            else
                q=k-1;
            end
            disp('30 seconds mature')
            object(q,3).deltax=click_location_num(1)-320;%

            if get(handles.buffer, 'Value')==1
                cmat('getDisparity', handles.disparity);
            else
                %%%disparity already captured from file
            end

            [localIm_q localRect_q]=imcrop(handles.left, [click_location_num(1)-40 ...
                click_location_num(2)-30 80-1 60-1]);
            localIm_disparity_q=imcrop(handles.disparity, localRect_q);
            [region,feat, M, handles.dist_disp_q, maxf, indf, num_feat]...
                =mser_detector_cont(localIm_disparity_q);
            [template_fprev_q template_dprev_q]...
                =vl_sift(single(rgb2gray(localIm_q)*255));%initialize fprev and dprev
            [val_q, freq_q]=createColorHistograms_cont(localIm_q);

            object(q,3).deltay=handles.dist_disp_q;%the distance based on disparity
            object(q,3).template=localIm_q;%60x80x3 color template of object
            object(q,3).dprev=template_dprev_q;%descriptors derived from color template
            object(q,3).fprev=template_fprev_q;%features derived from color template
            object(q,3).colorfreq=freq_q;%color bin
            object(q,3).colorval=val_q;%color bin types for each color freq
            object(q,3).detail=1;%was a detail scan conducted of the object
            object(q,3).label=object(q,2).label;
            %pause(5) close(editbox)%box used to label objects
            %                   object(k).label=str_label;
            %object(q,3).label_match='no match';%Which object in
            %10 object queu does the object match with, default
            %is 'no match'
            flag30=1;%ensures this loop is only checked after 30 seconds have passed

        else
            flag30=2;%designates object no longer mature
        end
    end

else %condition for locked window in the middle of the image
    set(handles.label, 'Visible', 'off')
    [localIm_background localRect_background]=imcrop(handles.left, ...
        [320-80 240-60 160-1 120-1]);
    [handles.fprev, handles.dprev, match, change, change_all]...
        =sift_detector_cont(rgb2gray(localIm_background)*255, handles.fprev, ...
        handles.dprev, localRect_background, xoffset, yoffset);
    set(handles.status, 'String', 'Pose');
end

%condition to check if the box exits the visible area
```

```
    if localRect_background(1)<=-20 || localRect_background(2) <= -20 ...
            || localRect_background(1)>=500 ||localRect_background(2)>=380 ...
            || min_Bhatt>=.4%When boundary is reached, these values are now stored
        localRect_background=[240 180 160-1 120-1];
        delete(hr)
        hr=rectangle('Parent', handles.leftcam,'Position', localRect_background);
        set(hr, 'EdgeColor', 'k');
        click_location_num=[360;240];
        xoffset=0;
        yoffset=0;
        disp('Pose Estimation...........')
    end
%simple disparity mode part 1
elseif(modeswitch==2&&dispalt==0)%disparity runs at half the frame speed

    if get(handles.buffer, 'Value')==1%if Live Video
        %set Cdata property of image object
        match=zeros(11,11);%this makes sure low match condition is not run
        disp('line 856 I dont know what this match does')
        disparitytype = get(handles.disparitytype, 'Value');%Gives two options for disparity
        if disparitytype==1
            cmat('readParamFile', '/home/gaines90/svs/data/gaines_shape.ini');
         else
             cmat('readParamFile', '/home/gaines90/svs/data/gaines_placement.ini');
         end
     set(handles.currentIm, 'CData', handles.disparity)
     cmat('getImage', handles.left, handles.right);%capture stereo pair
     dispalt=1;

    else
        match=zeros(11,11);%this makes sure low match condition is not run
            [handles.left, handles.disparity, handles.right, count]...
                =video_play_bin_cont_test(fid_color, fid_disparity, fid_right);
            set(handles.currentIm, 'CData', handles.disparity)
            handles.left=handles.left/255;
            t=t+1;%read next compass value next iteration
            if count==0
                %return
                fid_color = fopen(fullfile(buffer_loc, 'left.bin'), 'r+');
                fid_disparity = fopen(fullfile(buffer_loc, 'disparity.bin'), 'r+');
                fid_hokuyo = fopen(fullfile(buffer_loc, 'hokuyo.bin'), 'r+');
                fid_right = fopen(fullfile(buffer_loc, 'right.bin'), 'r+');
                t=1;%reset compass from file

            end

    end
                    r=fread(fid_hokuyo, 769, 'double');
                    x=r'.*cosd(handles.angles);
                    y=r'.*sind(handles.angles);

 %simple disparity mode part 2
 elseif(modeswitch==2&&dispalt==1)
     if get(handles.buffer, 'Value')==1%if Live Video
         cmat('getDisparity', handles.disparity);%capture disparity
         dispalt=0;
     else
            [handles.left, handles.disparity, handles.right, count]...
                =video_play_bin_cont_test(fid_color, fid_disparity, fid_right);
            handles.left=handles.left/255;
            t=t+1;%read next compass value next iteration
            if count==0
                %return
                fid_color = fopen(fullfile(buffer_loc, 'left.bin'), 'r+');
                fid_disparity = fopen(fullfile(buffer_loc, 'disparity.bin'), 'r+');
                fid_hokuyo = fopen(fullfile(buffer_loc, 'hokuyo.bin'), 'r+');
                fid_right = fopen(fullfile(buffer_loc, 'right.bin'), 'r+');
                t=1;%reset compass from file

              end
        end

                    r=fread(fid_hokuyo, 769, 'double');
                    x=r'.*cosd(handles.angles);
                    y=r'.*sind(handles.angles);
    end

%    %servo control
if get(handles.servopower, 'Value')==2
    if get(handles.scanmode, 'Value')==1
        posn=get(handles.hokuyoangle, 'Value')+25;
        calllib('phidget21', 'CPhidgetServo_setPosition', handles.handle, 0, posn);
        str = num2str(posn-25);
        set(handles.angleval, 'String', str);
    elseif get(handles.scanmode, 'Value')==2 %Scan towards ground
        if scanwindow==0 || scanwindow==scanparam%counts values up and back within window
            scanwindow=scanwindow*-1;
        end
        posn=90+25+abs(scanwindow);%%the value of 90 is around neutral
        scanwindow=scanwindow+1;
        calllib('phidget21', 'CPhidgetServo_setPosition', handles.handle, 0, posn);
        str = num2str(posn-25);
        set(handles.angleval, 'String', str);
    else  %%Scan within visual range
        if scanwindow==0 || scanwindow==scanparam%counts values up and back within window
            scanwindow=scanwindow*-1;
        end
        posn=90+25-abs(scanwindow);
        scanwindow=scanwindow+1;
        calllib('phidget21', 'CPhidgetServo_setPosition', handles.handle, 0, posn);
        str = num2str(posn-25);
        set(handles.angleval, 'String', str);
%       handles.left(uint16(posn), :)= zeros(1, 1920);%view position of Hokuyo
    end
end

if size(match,2)<=10 && size(match,2)>=5 && xoffset==80
        localIm_template=imcrop(handles.left, [click_location_num(1)-40 ...
            click_location_num(2)-30 80-1 60-1]);
        [template_fprev template_dprev]=vl_sift(single(rgb2gray(localIm_template)*255));
        imagesc(localIm_template, 'Parent', handles.localobj)
        h2=vl_plotframe(handles.fprev, 'Parent', handles.localobj);
        set(h2, 'color', 'y', 'Linewidth', 2)
        set(hr, 'EdgeColor', 'm');
        match_timeout=0;
end
if size(match,2)==0
    if xoffset==80
        %occlusion detection through disparity information
        if get(handles.buffer, 'Value')==1
            cmat('getDisparity', handles.disparity);
        else
            %%%Disparity already captured from file
        end
        localIm_disparity=imcrop(handles.disparity, [click_location_num(1)-40 ...
            click_location_num(2)-30 80-1 60-1]);
        [region,feat, M, dist_occlusion, maxf, indf, num_feat]...
            =mser_detector_cont(localIm_disparity);
      template_fprev=template_fprev_default;
      template_dprev=template_dprev_default;
      imagesc(localIm_template_default, 'Parent', handles.localobj)
      h2=vl_plotframe(template_fprev, 'Parent', handles.localobj);
```

```
        set(h2, 'color', 'y', 'Linewidth', 2)
        match_timeout=match_timeout+1;
        set(handles.status, 'String', 'Idle');
        set(hr, 'EdgeColor', 'y');
            if match_timeout == 5%if not generating matches, condition to return
                xoffset=0;
                yoffset=0;
                click_location_num=[360;240];
                set(handles.click_location, 'String', ['0';'0']);
                delete(hr);
                hr=rectangle('Parent', handles.leftcam,'Position', [240 180 160-1 120-1]);
                set(hr, 'EdgeColor', 'k');
                match_timeout=0;
                set(handles.status, 'String', 'Pose');
                disp('Pose Estimation............')
            end
    end
else
    match_timeout=0;
end
plot(handles.rhokuyo, x,y, 'k.', centroid(:,1), centroid(:,2), 'm*');
drawnow
    %update handles structure each time through loop
     guidata(hObject, handles);
end
   str_label=get(handles.output, 'UserData');%label the current object.
   size_str_label=size(str_label,2);
   if k~=1
       if size_str_label <10
           handles.labels(k-1, 1:size(str_label,2))=str_label;
       else
           handles.labels(k-1, :)=str_label(1,1:10);
       end
   end
cmat('stop');
cmat('close');
disp('Camera STOPPED')
set(handles.status, 'String', 'Stopped');
set(handles.hokuyo_detail, 'Visible', 'off');
if get(handles.servopower, 'Value')==2
    calllib('phidget21', 'CPhidget_close', handles.handle);
    calllib('phidget21', 'CPhidget_delete', handles.handle);
end
clear all%%CHANGE

% --- Executes during object creation, after setting all properties.
function camswitch_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in reset.
function reset_Callback(hObject, eventdata, handles)
clear handles.rhokuyo
cmat('init');
cmat('readParamFile', '/home/gaines90/svs/data/gaines2.ini');
handles.openerror=cmat('open');
if handles.openerror==-1
    set(handles.status, 'String', 'No Data');
    disp('Failed to initialize stereo camera')
end
set(handles.output, 'CurrentCharacter', '1');%stop on key input
set(handles.startvideo, 'String', 'Start Video')
set(handles.compasspower, 'Visible', 'on')
set(handles.hokuyopower, 'Visible', 'on');
set(handles.servopower, 'Visible', 'on');
```

```
% --- Executes on mouse press over axes background.
function leftcam_ButtonDownFcn(hObject, eventdata, handles)
set(handles.search_mode, 'Visible', 'off');
pause(.002)
set(handles.navigation_mode, 'Visible', 'on');
%wait until a double click is done on new rectangle image object
[click_location1, click_location2]=ginput(1);%this in place so the rectangle need not be drawn
if get(handles.camswitch, 'Value')~=2%test to see if not in disparity mode
[localIm_object localRect]=imcrop(handles.left, [click_location1-40 ...
   click_location2-30 80-1 60-1]);
set(handles.click_location, 'String',[click_location1;click_location2]);
imagesc(localIm_object, 'Parent', handles.localobj)
dim=size(size(localIm_object));
if dim(2)==3 %test to see the # of channels in the image
   localIm_object=rgb2gray(localIm_object)*255;
end
      [handles.fprev, handles.dprev, matches]=sift_detector_cont(localIm_object, ...
         handles.fprev, handles.dprev, localRect, 80, 60);
      h2=vl_plotframe(handles.fprev, 'Parent', handles.localobj);
      h3=vl_plotframe(handles.fprev(:,matches(2,:)), 'Parent', handles.localobj);
      set(h2, 'color', 'y', 'Linewidth', 2)
      set(h3, 'color', 'g')
else
   localIm_disparity=imcrop(handles.disparity, [click_location1-40 ...
      click_location2-30 80-1 60-1]);%needed for feature extraction
   backgroundIm_disparity=imcrop(handles.disparity, [click_location1-80 ...
      click_location2-60 160-1 120-1]);
   localIm_object=imcrop(handles.left, [click_location1-40 click_location2-30 80-1 60-1]);

   set(handles.click_location, 'String',[click_location1;click_location2]);

   %display local image object in handles.localobj
   imagesc(backgroundIm_disparity, 'Parent', handles.localobj)
   [region,feat, M, Intensity, maxf, indf]=mser_detector_cont(localIm_disparity);
end
% Update handles structure
guidata(hObject, handles);

% --- Executes on mouse press over axes background.
function rhokuyo_ButtonDownFcn(hObject, eventdata, handles)
set(handles.navigation_mode, 'Visible', 'off');
pause(.002)
set(handles.search_mode, 'Visible', 'on');

% --- Executes on mouse press over figure background.
function figure1_ButtonDownFcn(hObject, eventdata, handles)
set(handles.navigation_mode, 'Visible', 'off');
pause(.01)
set(handles.search_mode, 'Visible', 'on');

function Right_Callback(hObject, eventdata, handles)
if get(handles.compasspower, 'Value')~=1
   s=instrfind;
   fprintf(s, 'c?');
   fscanf(s);
   compval=fscanf(s);
   compval=textscan(compval, '$C%4c');
   set(handles.compass, 'String', compval);
   compnum=str2double(compval);
   cumangle=handles.angles+compnum;
   if get(handles.compasspower, 'Value')==3%if in cumulative mode, plot every click
      r=hokuyoAPI('getScan','range', 0, 768, 1, 3);
      x=r'.*cosd(cumangle);
      y=r'.*sind(cumangle);
      plot(handles.rhokuyo, x, y, 'k.');
      set(handles.rhokuyo, 'NextPlot', 'add')
```

```
    else
        set(handles.rhokuyo, 'NextPlot', 'replacechildren');
    end
end
set(handles.compass, 'String', 'Turn')
% Update handles structure
guidata(hObject, handles);

% --- Executes on button press in Left.
function Left_Callback(hObject, eventdata, handles)
if get(handles.compasspower, 'Value')~=1
    s=instrfind;
    fprintf(s, 'c?');
    fscanf(s);
    compval=fscanf(s);
    compval=textscan(compval, '$C%4c');
    set(handles.compass, 'String', compval);
    compnum=str2double(compval);
    cumangle=handles.angles+compnum;
    if get(handles.compasspower, 'Value')==3%if in cumulative mode, plot every click
        r=hokuyoAPI('getScan','range', 0, 768, 1, 3);
        x=r'.*cosd(cumangle);
        y=r'.*sind(cumangle);
        plot(handles.rhokuyo, x, y, 'k.');
        set(handles.rhokuyo, 'NextPlot', 'add')
    else
        set(handles.rhokuyo, 'NextPlot', 'replacechildren');
    end
    % Update handles structure
    guidata(hObject, handles);
end
set(handles.compass, 'String', 'Turn')
disp('Left callback')

function edit1_Callback(hObject, eventdata, handles)

function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --------------------------------------------------------------------
function search_mode_ButtonDownFcn(hObject, eventdata, handles)

function navigation_mode_ButtonDownFcn(hObject, eventdata, handles)

function Right_ButtonDownFcn(hObject, eventdata, handles)

function Left_ButtonDownFcn(hObject, eventdata, handles)

function disparitytype_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function disparitytype_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function compasspower_Callback(hObject, eventdata, handles)

function compasspower_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function scanmode_Callback(hObject, eventdata, handles)
function scanmode_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function status_Callback(hObject, eventdata, handles)

function status_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function hokuyopower_Callback(hObject, eventdata, handles)

function hokuyopower_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function servopower_Callback(hObject, eventdata, handles)

function servopower_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function tracking_mode_Callback(hObject, eventdata, handles)

function tracking_mode_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function hokuyo_detail_Callback(hObject, eventdata, handles)
r=servo_sweep(65, 105, 3 , .0001,handles.handle);
 hokuyo_resize=imresize(r/5000*255, [240, 320], 'nearest');
 handles.detailscan=r;
 hokuyo_edge=edge(hokuyo_resize,'log');
[Hv, Tv, Rv]=hough(hokuyo_edge, 'Theta',0:5); %With a restriction of Theta
 Pv=houghpeaks(Hv,3);
 linesv = houghlines(hokuyo_edge,Tv,Rv,Pv,'FillGap',60,'MinLength',100);
 axes(handles.hokuyo_detail_axis);
  hold on
 imagesc(hokuyo_resize, 'Parent', handles.hokuyo_detail_axis);
 XVsize=size(linesv, 2);
 if XVsize>1;
     XVcomp=zeros(XVsize, 2);
     XVcomp2=zeros(XVsize, 2);
     for m=1:XVsize
         XYv=[linesv(m).point1; linesv(m).point2];%stores coordinates in format
         XVavg=uint8((linesv(m).point1+linesv(m).point2)/2);
         XVcomp(m, 1:2)=[hokuyo_resize(XVavg(2), XVavg(1)-5), hokuyo_resize(XVavg(2),XVavg(1)+5)];
         XVcomp2(m, 1:2)=[hokuyo_resize(XVavg(2), XVavg(1)-10), hokuyo_resize(XVavg(2),XVavg(1)+10)];
         plot(handles.hokuyo_detail_axis, XYv(:,1),XYv(:,2),'LineWidth',2,'Color','green');%plots coordinates
         plot(handles.hokuyo_detail_axis, XVavg(1), XVavg(2), 'm*')
         plot(handles.hokuyo_detail_axis, XVavg(1)+10, XVavg(2), 'k')
         plot(handles.hokuyo_detail_axis, XVavg(1)-10, XVavg(2), 'k')
     end
     XVflip=fliplr(XVcomp);
     door_detect(XVcomp, XVcomp2, XVflip, XVsize, 30);
 end
  hold off
  axes(handles.leftcam);
 axis(handles.hokuyo_detail_axis, 'off');
 set(handles.rhokuyo, 'Visible', 'off');

function label_Callback(hObject, eventdata, handles)
editbox

function object_out_Callback(hObject, eventdata, handles)

function buffer_Callback(hObject, eventdata, handles)

function buffer_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Buffer Video Read Function—video_play_bin_cont_test.m

```
function [left, disparity, right, count]=video_play_bin_cont_test(...
   fid_color, fid_disparity, fid_right)

[left_temp count]=fread(fid_color, 921600);

if count==0
   left=zeros(480,640,3);
   disparity=zeros(480,640);
   right=zeros(480,640);

else
   left=reshape(left_temp, 480,640,3);
   right=fread(fid_right, [480 640]);
   disparity=fread(fid_disparity, [480 640], 'int16');
end

end
```

Hokuyo Read Function—hokuyo_singlescan.m

```
function [r, xy]=hokuyo_singlescan(comp_power, plotflag)

if nargin==0
   comp_power=0;
   plotflag=0;
elseif nargin==1
   plotflag=0;
end

if strcmp(comp_power, 'on')
   compnum=compass_singlescan;
else
   compnum=0;
end

   angles=-1*((compnum-135):.3515625:(compnum+135));% what it is
   r=hokuyoAPI('getScan','range', 0, 768, 1, 3);
   x=r'.*cosd(angles);
   y=r'.*sind(angles);
   xy=[x;y];

if strcmp(plotflag, 'on')
   angles_ref=1000*ones(1, 769);
   angles_ref(1)=0;
   angles_ref(769)=0;
   xref=angles_ref.*cosd(angles);
   yref=angles_ref.*sind(angles);
   xyref=[xref;yref];
   figure(109), plot(xy(1,:), xy(2,:), 'b.'),
   hold on, plot(0,0,'r*'), hold off
end

end
```

Hokuyo Initialization—hokuyo_init.m

```
function xy1=hokuyo_init()

dev='/dev/ttyACM0';%initializing Hokuyo
baud=115200;      %does not matter for USB

   if ~hokuyoAPI('open',dev,baud);%failure to open Hokuyo API results in exit
      hokuyoAPI('open', dev, baud);
   end

      [r1,xy1]=hokuyo_singlescan();
      xy1=unique(xy1', 'rows')';
end
```

Hokuyo Read Function From File—hokuyo_scanread.m

```
function [hokuyo_r, hokuyo_xy, hokuyo_mat]=hokuyo_scanread(file_path, readnum)

fid=fopen(file_path, 'r+');

A=fread(fid, 'double');
sizeA=size(A,1);
readings=sizeA/769;
hokuyo_mat=reshape(A, 769, readings);

angles=(-135):.3515625:(135);

hokuyo_r=hokuyo_mat(:,readnum);%readnum in seconds

%sensor is reading in data backwards for some odd reason
hokuyo_r=flipud(hokuyo_r);

hokuyo_x=hokuyo_r'.*cosd(angles);
hokuyo_y=hokuyo_r'.*sind(angles);

hokuyo_xy=[hokuyo_x; hokuyo_y]';

%hold off
%figure(100),plot(hokuyo_r, '.')

%hold off
%figure(101), plot(hokuyo_x, hokuyo_y, '.')

fclose(fid);

end
```

Observation Labeling During Search—editbox.m

```
function varargout = editbox(varargin)

%This function allows for real-time labeling of observations

% Edit the above text to modify the response to help editbox_display

% Last Modified by GUIDE v2.5 15-Jan-2011 11:33:19

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @editbox_OpeningFcn, ...
                   'gui_OutputFcn',  @editbox_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

function varargout = editbox_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
varargout{2} = handles.str;
% --- Executes just before editbox_display is made visible.
function editbox_OpeningFcn(hObject, eventdata, handles)
handles.output = hObject;
handles.str='';
handles.test='';
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes editbox_display wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function editbox_display_Callback(hObject, eventdata, handles, varargin)
str = get(handles.editbox_display, 'String');
%set(handles.staticbox, 'String', handles.str);

%set(editbox, 'UserData', str)
set(videre_lite_SAVE, 'UserData', str)
 close(editbox)

% --- Executes during object creation, after setting all properties.
function editbox_display_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Data Acquisition

Information Capture—both_mapping2.m

```
function both_mapping2(seconds)

if nargin ==0
   seconds=10;
%    save_dir='/home/gaines90/Desktop/imagestore';
end
hokuyo_init;
scans=floor(seconds*10);
cmat('init');
cmat('open');
cmat('start');
left=zeros(480, 640, 3);%image output must be 3 channel image for color
right=zeros(480, 640);%right image is only used for disparity and is black and white
disparity=zeros(480, 640);

fid_left = fopen('/home/gaines90/Desktop/imagestore/left.bin', 'w+');
fid_right = fopen('/home/gaines90/Desktop/imagestore/right.bin', 'w+');
fid_hokuyo = fopen('/home/gaines90/Desktop/imagestore/hokuyo.bin', 'w+');
fid_disparity = fopen('/home/gaines90/Desktop/imagestore/disparity.bin', 'w+');

if scans>9000
   disp('exceded maximum map time of 15 minutes')
   disp('default value of 15 minutes selected')
   scans=9000;
elseif scans<0
   disp('time cannot be negative. Exiting')
   return
end
time=zeros(scans,1);
disp('Begin capture')
tic
for i=1:scans
   cmat('getImage', left, right);%capture left image
   fwrite(fid_left, left*255, 'uint8');
   cmat('getDisparity', disparity);
   fwrite(fid_right, right, 'uint8');
   fwrite(fid_disparity, disparity, 'int16');
   r_vect=hokuyo_singlescan;
   fwrite(fid_hokuyo, r_vect, 'double');

%    fwrite(fid4, compval);

   time(i)=toc;
end
toc

cmat('stop');
cmat('close');
fclose(fid_left);
fclose(fid_right);
fclose(fid_disparity);
fclose(fid_hokuyo);
zero_time=time-time(1);
dlmwrite('/home/gaines90/Desktop/imagestore/time.csv', zero_time, ',');

end
```

Import User Input Information—user_input.m

```
function [translation, heading]=user_input(submap_number)

if nargin==0
   submap_number=1;
end
if submap_number == 1
   translation=zeros(3000,1);
   heading=zeros(3000,1);
   translation(26:350,1)=.1*ones(325,1);
   translation(410:680,1)=.2*ones(271,1);
   translation(750:1030,1)=.2*ones(281,1);
   translation(1100:1360,1)=.2*ones(261,1);
   translation(1430:1630,1)=.1*ones(201,1);
   translation(1700:1820,1)=.2*ones(121,1);
   heading(1:350,1)=ones(350,1)*0;
   heading(351:760,1)=ones(410,1)*90;
   heading(680:1110,1)=ones(431,1)*180;
   heading(1031:1440,1)=ones(410,1)*270;
   heading(1361:1800,1)=ones(440,1)*0;
elseif submap_number==2
   translation=zeros(3000,1);
   heading=zeros(3000,1);
   translation(20:60,1)=.1*ones(41,1);
   translation(105:190,1)=.1*ones(86,1);
   translation(230:365,1)=.2*ones(136,1);
   translation(440:570,1)=.1*ones(131,1);
   translation(610:660,1)=.1*ones(51,1);%no turn
   translation(710:860,1)=.1*ones(151,1);
   translation(900:1250,1)=.1*ones(351,1);
   translation(1325:1850,1)=.1*ones(526,1);
   translation(1950:2280,1)=.1*ones(331,1);
   translation(2320:2440,1)=.2*ones(121,1);
   translation(2500:2620,1)=.1*ones(121,1);
   translation(2740:2996,1)=.1*ones(257,1);
   heading(1:60,1)=ones(60,1)*0;
   heading(61:190,1)=ones(130,1)*270;
   heading(191:365,1)=ones(175,1)*0;
   heading(366:386,1)=ones(21,1)*270;%180
   heading(387:570,1)=ones(184,1)*180;
   heading(571:860,1)=ones(290,1)*270;
   heading(861:1250,1)=ones(390,1)*180;
   heading(1251:1271,1)=ones(21,1)*90;%180
   heading(1272:1850,1)=ones(579,1)*0;
   heading(1851:2300,1)=ones(450,1)*0;
   heading(2301:2400,1)=ones(100,1)*90;%180
   heading(2401:2420,1)=ones(20,1)*0;
   heading(2421:2620,1)=ones(200,1)*270;
   heading(2621:2996,1)=ones(376,1)*0;
elseif submap_number==3
   translation=zeros(3000,1);
   heading=zeros(3000,1);

   translation(21:435,1)=.1*ones(415,1);
   translation(530:978,1)=.1*ones(449,1);
   translation(1020:1140,1)=.1*ones(121,1);
   heading(1:430,1)=ones(430,1)*90;
   heading(431:478,1)=ones(48,1)*0;
   heading(479:978,1)=ones(500,1)*270;%180
   %heading(979:1200,1)=ones(182,1)*0;

end

end
```

Convert Binary Data to CARMEN Log Format—carmen_format.m

```
function carmen_format(file_path, submap_number, append_conversion, conversion_path)

if nargin==1
   submap_number=1;
   append_conversion='no';
   conversion_path='/home/gaines90/Desktop/ExampleData/conversion1.log';
elseif nargin==2
   append_conversion='no';
   conversion_path='/home/gaines90/Desktop/ExampleData/conversion1.log';
elseif nargin ==3
   conversion_path='/home/gaines90/Desktop/ExampleData/conversion1.log';
end
 [translation, heading]=user_input(submap_number);

[~,~, hokuyo_mat]=hokuyo_scanread(file_path, 1);
hokuyo_scaled=hokuyo_mat'/1000;%each row must be a reading
num_scans=length(hokuyo_scaled);
hokuyo_time=repmat((1:num_scans)',1,3)/10;
hokuyo_time(:,2)=zeros(num_scans,1);
odometry=zeros(num_scans, 9);
reading_column=769*ones(num_scans, 1);%number of readings
%reading_column=repmat('769', num_scans, 1);

total_x=0;
total_y=0;

flaser='FLASER ';
odom='ODOM ';
%space_column=repmat(' ', num_scans, 1);
%space_column=ones(num_scans, 1)*769;

%Generate the odometry information in loop
for i=1:num_scans
   delta_x=translation(i)*cosd(heading(i));
   delta_y=translation(i)*sind(heading(i));
   total_x=total_x+delta_x;
   total_y=total_y+delta_y;
   odometry(i, 1)=total_x;
   odometry(i, 2)=total_y;
   odometry(i, 3)=heading(i)*pi/180;
end
odometry(:,7:9)=hokuyo_time;%complete odometry information
readings=[reading_column, hokuyo_scaled, odometry(:,1:3), odometry(:, 1:3), hokuyo_time];%complete readings
if exist(conversion_path, 'file') && ~strcmp(append_conversion, '-append')
   delete (conversion_path)
   disp('REPLACED EXISTING LOG FILE')
else
   disp('CREATED NEW LOG FILE')
end

fid=fopen(conversion_path, 'a');
for i=1:num_scans
   fwrite(fid, flaser);
   dlmwrite(conversion_path, readings(i,:), '-append', 'delimiter', ' ');
   fwrite(fid,odom);
   dlmwrite(conversion_path, odometry(i,:), '-append', 'delimiter', ' ');

end

fclose(fid);

end
```

Mapping Construction

Main Mapper Test—mapper_test.m

```
function mapper_test(translation, heading)

close all

numF=0;
numU=0;
%heading=0;
x_pos=0;
y_pos=0;
%translation=ones(readings, 1);
%heading=zeros(readings, 1);
delta_x=0;
delta_y=0;
delta_t=0;
last='init';
count=0;
match=0;
heading_prev=0;
delta_path=0;
type='START';
heading_flag=false;
space_flag=false;
space_x=0;
space_y=0;
mapper_number=0;
junction_number=0;

%mapper_log columns are 1. mapper_number, 2. flagF, 3. flagU, 4. heading,
%5. x_pos, 6. y_pos, 7. delta_path which is previous heading, 8. delta_x,
%9. delta_y, 10. flagM, 11. flagI, 12. iteration 13. adjustmentx 14.
%adjustmenty
junction_log=zeros(100, 7);%Junction log stores heading in x location and y location of junctions 1. junction number 2. junction heading 3. xpos
mapper_log=zeros(1000,13);%   4. ypos 5. mapper number 6. xdiff 7. ydiff

for i=1:readings
    [read_scan, hokuyo_xy]=hokuyo_scanread('/home/gaines90/Desktop/FINAL1/hokuyo.bin', i);

iteration=i;
pause(.1)
[numF, numU, heading, heading_prev, heading_flag, space_flag,space_x, space_y, x_pos, y_pos, delta_path, delta_h,…
    delta_x, delta_y, delta_t, last, type, count, mapper_log, junction_log, mapper_number, junction_number, match, iteration]=accumulator2(…
   hokuyo_xy, numF, numU, heading(i), heading_prev, heading_flag, space_flag, space_x, space_y, x_pos, y_pos, translation(i), delta_path,…
   delta_x, delta_y, delta_t, last, type, count, mapper_log, junction_log, mapper_number, junction_number, match, read_scan, iteration);

end

end
```

Mapper Variance Module—mapper_variance.m

```
function [centroid, covariance, k_norm]=mapper_variance(hokuyo_xy, num_clusters, iterations)

%set defaults
if nargin==0
   disp('must input hokuyo scan')
   return
elseif nargin ==1
   num_clusters=20;
   iterations=10;%%this may be improved with more iterations.  Depends on computational time
end

hokuyo_xy=unique(hokuyo_xy, 'rows');%eliminate repeated points

[n,d]=size(hokuyo_xy);

   %kmeans calculates clusters from data
   %k_dist finds the squared euclidean distance
   [k_val, centroid, k_dist]=kmeans(hokuyo_xy, num_clusters, 'Start','uniform','Maxiter',iterations ,'EmptyAction','drop');

V_clust = repmat(struct('count',0,'hokuyo_clust',zeros(n,d)),1,num_clusters);
for i=1:n, % Separate cluster points
   V_clust(k_val(i)).count = V_clust(k_val(i)).count + 1;%keeps a count of each point as filed
   V_clust(k_val(i)).hokuyo_clust(V_clust(k_val(i)).count,:) = hokuyo_xy(i,:);%separates out points from the group in each file
end
covariance = zeros(d,d,num_clusters);
index_Max=zeros(num_clusters, 1);
clust_weight=zeros(num_clusters,1);

for i=1:num_clusters,
   %percentage of each type of categorized point
   clust_weight(i)=V_clust(i).count/n;
   covariance(:,:,i) = cov(V_clust(i).hokuyo_clust(1:V_clust(i).count,:));

   if max(max(covariance(:,:,i)))>2000
      index_Max(i,1)=1;

   end
end

index_Max=logical(index_Max);
%filter out too small values

%could add a filter based on k_norm
k_norm=k_dist.*clust_weight;

k_norm=k_norm(index_Max);
covariance=covariance(:,:,index_Max);
centroid=centroid(index_Max, :);

%filter out NaN values
index_NaN=any(isfinite(centroid),2);
k_norm=k_norm(index_NaN);
covariance=covariance(:,:,index_NaN);
centroid=centroid(index_NaN, :);

figure(200), plot(hokuyo_xy(:,1), hokuyo_xy(:,2), 'b.', centroid(:,1), centroid(:,2), 'm.')
%mapper_ellipse(centroid, covariance)%OPTIONAL plot of covariance ellipse on centroids of covariance
pause(.01)
```

Mapper Ellipse Plot Function—mapper_ellipse.m

```
%adapted from example code written by Patrick P. C. Tsui
%PAMI research group
%Department of Electrical and Computer Engineering
%University of Waterloo
%March, 2006

function mapper_ellipse(centroid,variance)

d = length(centroid);
hold on
for i=1:d

  [Ev,D] = eig(variance(:,:,i));

  iV = inv(variance(:,:,i));
  % Find the larger projection
  P = [1,0;0,0];  % X-axis projection operator
  P1 = P * 2*sqrt(D(1,1)) * Ev(:,1);
  P2 = P * 2*sqrt(D(2,2)) * Ev(:,2);
  if abs(P1(1)) >= abs(P2(1)),
    Plen = P1(1);
  else
    Plen = P2(1);
  end
  count = 1;
  step = 0.001*Plen;
  Contour1 = zeros(2001,2);
  Contour2 = zeros(2001,2);
  for x = -Plen:step:Plen,
    a = iV(2,2);
    b = x * (iV(1,2)+iV(2,1));
    c = (x^2) * iV(1,1) - 1;
    Root1 = (-b + sqrt(b^2 - 4*a*c))/(2*a);
    Root2 = (-b - sqrt(b^2 - 4*a*c))/(2*a);
    if isreal(Root1),
      Contour1(count,:) = [x,Root1] + centroid(i,:);
      Contour2(count,:) = [x,Root2] + centroid(i,:);
      count = count + 1;
    end
  end
  Contour1 = Contour1(1:count-1,:);
  Contour2 = [Contour1(1,:);Contour2(1:count-1,:);Contour1(count-1,:)];

  %    plot(centroid(i,1),centroid(i,2),'k+');
  plot(Contour1(:,1),Contour1(:,2),'k-');
  plot(Contour2(:,1),Contour2(:,2),'k-');

end

hold off

end
```

Analyzer Module—analyze2.m

```
function [voteF,voteU, partial]=analyzer2(covariance, k_norm)

   num_points=length(covariance);
   U_totalX=0;
   U_totalY=0;
   U_totalD=0;
   F_totalS=0;
   F_totalC=0;
   F_totalD=0;
   for i=1:num_points
      [~,D] = eig(covariance(:,:,i));
      U_param=covariance(2,2, i)/covariance(1,1, i);
      if U_param<.1
         U_totalX=U_totalX+1;%X directioin for horizontal line
      elseif U_param>5
         U_totalY=U_totalY+1;%Y direction for vertical line
      else
         U_totalD=U_totalD+1;%Diagonal or uncorrelated
      end
      F_param=D(2,2)/D(1,1);
      if F_param>=10%Structure
         F_totalS=F_totalS+1;
      elseif F_param < 10%clutter
         F_totalC=F_totalC+1;
      else
         F_totalD=F_totalD+1;
      end
   end
    partial=sum(k_norm/1000)/num_points;

   %vote for HALLWAY
   if U_totalX/num_points>=.5   && F_totalS/num_points >= .5%&& U_totalY/num_points<.2 && U_totalC/num_points<.3

      voteU=1;
      voteF=1;

   %vote for JUNCTION
   elseif U_totalY/num_points>=.2 && F_totalS/num_points >= .5

      voteU=0;
      voteF=1;

%vote for CONTENT SPACE
   elseif F_totalS/num_points <= .7 && U_totalD/num_points>=.4

      voteU=1;
      voteF=0;

   %vote for OPEN SPACE
   elseif F_totalS/num_points <= .3 && (U_totalD+U_totalY)/num_points > .5

      voteU=0;
      voteF=0;
   else

      disp('nothing')

      voteU=0;
      voteF=0;

   end
end
```

Accumulator Module—accumulator2.m

```
function [numF, numU, heading, heading_prev, heading_flag, space_flag, space_x, space_y, x_pos, y_pos, …
   delta_path, delta_h, delta_x, delta_y, delta_t, last, type, count, mapper_log, junction_log,mapper_number, junction_number,…
    match, iteration]=accumulator2 (hokuyo_xy, numF, numU, heading, heading_prev, heading_flag, space_flag, space_x, space_y,…
    x_pos, y_pos, translation, delta_path,delta_x, delta_y, delta_t, last, type, count, mapper_log, junction_log, …
   mapper_number, junction_number, match, read_scan, iteration)

%INPUTS/OUTPUTS
%hokuyo_xy--current LIDAR scan
%numF--accumulated votes for F trait
%numU--accumulated votes for U trait
%flagM--maturity flag to determine maturity trait
%flatI--importance flag to determine importance trait
%heading--operator input global heading
%translation--operator input of translation
%delta_x--accumulated local displacement in the local x direction
%delta_y--accumulated local displacement in the local y direction
%delta_t--accumulated  time of current accumulation
%matched features during accumulation
%delta_h--difference in heading from previous iteration

%start local counter
if delta_t==0
%    disp('test')
%    heading_i=heading;
   delta_x=0;
   delta_y=0;
   numF=0;
   numU=0;
   count=0;
%    flag_M=0;
   tic
end

%%has heading changed??
delta_h=heading-heading_prev;%temporary fix
%local heading for accumulation

%heading and translation from user input vector.  tracks how far moved
%during accumulation set delta_x and delta_y as zero outside of loop IF IT

%SHOULD BE GLOBAL
space_x=space_x+translation*cosd(heading);
space_y=space_y+translation*cosd(heading);
%SHOULD BE LOCAL
delta_y=delta_y+translation;
delta_x=0;
delta_t=toc;
num_clusters=20;%default values
iterations=10;%default values
[~, covariance, k_norm]=mapper_variance(hokuyo_xy, num_clusters, iterations);%from individual scans
%covariance and k_norm used to find votes for F and U, delta_x, delta_y,
%and delta_t used to find votes for M, observatins used to find votes for I
[voteF,voteU, partial]=analyzer2(covariance, k_norm);%match must be a vector of all observation matches
flagM=0;
flagI=1;

if voteF==1 && voteU==1
   if strcmp(last, 'h') && translation==.1
      count=count+1;
   elseif strcmp(last, 'h') && translation==.2
      count=count+2;
   elseif strcmp(last, 'h') && translation ==0
      count=0;%was do nothing
    else
        count=0;%ie previous reading not hallway
    end
    last='h';
elseif voteF==1&&voteU==0
    if strcmp(last, 'j')% && flagM==1
        flagM=1;
        count=count+4;
    elseif strcmp(last, 'h') && partial>1000%&& flagM==0
        flagM=0;
        count=count+20;
    else
        count=0;%ie previous reading not junction
    end
    last='j';

elseif voteF==0 && voteU==1
    if strcmp(last, 'c') && translation==.1
        count=count+20;
    elseif strcmp(last, 'c') && translation==.2
        count=count+20;
    elseif strcmp(last, 'c') && translation ==0
        count=0;%was do nothing
    else
        count=0;%ie previous reading not content space
    end
    last='c';

elseif voteF==0 &&voteU==0
    if strcmp(last, 'c') && translation==.1
        count=count+20;
    elseif strcmp(last, 'c') && translation ==.2
        count=count+20;
    elseif strcmp(last, 'c') && translation ==0
        count=0;%was do nothing
    else
        count=0;%ie previous reading not content space
    end
    last='c';
end

%set numF and numU outside of the loop with accumulator as zero initially
numF=numF+voteF;
numU=numU+voteU;

    mapscale=30;

    if delta_h ~= 0
%       space ('down', 'n', heading_prev, 'on',read_scan*mapscale, heading_prev, x_pos+20*cosd(heading_prev),
y_pos+20*sind(heading_prev));

        if heading_flag==false;
            space ('down', 'n', heading_prev, 'on',read_scan*mapscale, heading_prev, x_pos+200*cosd(heading_prev),
y_pos+200*sind(heading_prev));
            plot(x_pos+200*cosd(heading_prev), y_pos+200*sind(heading_prev), 'm*', 'MarkerSize', 2)

            disp('HEADING CHANGE****************************************')
            heading_flag=true;

            delta_path=heading_prev;

            mapper_number=mapper_number+1;
            last='a';
            %logger code
            [mapper_log, junction_log, junction_number, heading, x_pos, y_pos]=mapper_logger(mapper_number, junction_number,mapper_log,
junction_log, -1, -1, heading, x_pos, y_pos, delta_path, delta_x, delta_y, flagM, flagI, iteration);
```

```
            mapper_number=mapper_number+1;

        else

        end

    end

        testspacex=space_x;
    testspacey=space_y;

    heading_prev=heading;

if count >= 20; %|| movement_Flag==true

    flagF=voteF;
    flagU=voteU;

    if ~strcmp(type, 'JUNCTION')%do not log multiple junctions
        mapper_number=mapper_number+1;
    end

    last_type=type;

[heading, x_pos, y_pos, type]=mapper(last, heading, x_pos, y_pos, delta_path, delta_x, delta_y, flagM, flagI, read_scan, partial);

disp(type);

if ~strcmp(type, 'JUNCTION') && strcmp(last_type, 'JUNCTION')%log hallway or space after a junction missed initially
    mapper_number=mapper_number+1;
end

[mapper_log, junction_log, junction_number, heading, x_pos, y_pos]=mapper_logger(mapper_number, junction_number, mapper_log,
junction_log, flagF, flagU, heading, x_pos, y_pos, delta_path, delta_x, delta_y, flagM, flagI, iteration);

if strcmp(type, 'HALLWAY')
    heading_flag=false;
    delta_path=heading;
elseif strcmp(type, 'CONTENT SPACE')
    space_flag=true;
elseif strcmp(type, 'JUNCTION')
    if space_x^2 + space_y^2 >5

        space_flag=false;
        space_x=0;
        space_y=0;
    end
end

    numF=0;
    numU=0;
    match=0;
    %movement_Flag=false;
    delta_h=0;
    delta_x=0;
    delta_y=0;
    delta_t=0;%reinitialized on tic through the function
 %   tic

end

end
```

Mapper Module—mapper.m

```
function [heading_new, x_pos_new, y_pos_new, type]=mapper(last, heading, x_pos, y_pos, delta_path, delta_x, delta_y, flagM, flagI,
read_scan, partial)
angle_fw='fw';
width='n';
hokuyo_map='on';
heading_new=heading;

%delta_path
if strcmp(last, 'h')

    [class, width]=hallway_select(flagM, flagI, partial);

    [heading_new, x_pos_new, y_pos_new]=hallway (angle_fw,  class, width, heading, x_pos, y_pos);

    type='HALLWAY';

elseif strcmp(last, 'j')

    class=junction_select(flagM);

    [heading_new, x_pos_new, y_pos_new]=junction (class, width, delta_path, heading, x_pos, y_pos);

    type='JUNCTION';

elseif strcmp(last, 'c')

    [class, heading_new] = region_select(heading, delta_path, delta_x, delta_y);
class='up';
    [heading_new, x_pos_new, y_pos_new]=space (class, width, heading_new, hokuyo_map,read_scan, heading, x_pos, y_pos);

    type='CONTENT SPACE11';

elseif strcmp(last, 'o')

    [class, heading_new] = region_select(heading, delta_path, delta_x, delta_y);

    [heading_new, x_pos_new, y_pos_new]=space (class, width, heading_new, hokuyo_map,read_scan, heading, x_pos, y_pos);

    type='OPEN SPACE';

elseif strcmp(last, 'a')

    heading_new=heading;
    x_pos_new=x_pos;
    y_pos_new=y_pos;
    type='ANGLE';
%    class=angle_select(heading);

%    [heading_new, x_pos_new, y_pos_new]=angle(class,heading, x_pos, y_pos);

else

    disp('invalid structure command in mapper')

end

end
```

Map Hallway—hallway.m

```
function [heading_new, x_pos_new, y_pos_new]=hallway (angle,  class, ...
width, heading, x_pos, y_pos)

figure(999), hold on, axis([-1000 1000 -1000 1000]), axis square
scrnsz=get(0,'ScreenSize');
set(999, 'Position', scrnsz)
%Set arguments
if nargin ==0
   angle='fw';
   class='eq';
   width='n';
   heading=0;
   x_pos=0;
   y_pos=0;
elseif nargin ==1
   class='eq';
   width='n';
   heading=0;
   x_pos=0;
   y_pos=0;
elseif nargin == 2
   width='n';
   heading=0;
   x_pos=0;
   y_pos=0;
elseif nargin ==3
   heading=0;
   x_pos=0;
   y_pos=0;
end

%set lengths of segments
if strcmp(class, 'g')
   length=20;
elseif strcmp(class, 'eq')
   length=10;
elseif strcmp(class, 'l')
   length=5;
else
   disp('Invalid length class')
   return
end

%set angle of segments
if strcmp(angle, '45l')
   rotation=45;
elseif strcmp(angle, '45r')
   rotation=-45;
elseif strcmp(angle, '90l')
   rotation=90;
elseif strcmp(angle, '90r')
   rotation=-90;
elseif strcmp(angle, 'fw')
   rotation=0;
else
   disp('Invalid angle class')
   return
end

%new global heading
heading_new=heading+rotation;

%evaluate width of segment
```

```
if strcmp(width, 'w')
   out_diameter=16;
   in_diameter=10;
   length=length*2;
elseif strcmp(width, 'n')
   out_diameter=10;
   in_diameter=6;
elseif strcmp(width, 't')
   out_diameter=5;
   in_diameter=3;
else
   disp('Invalid width class');
   return
end

%plot length x and y are a function of heading
length_x=length*cosd(heading_new);
length_y=length*sind(heading_new);

%set new end point locations
x_pos_new=x_pos+length_x;
y_pos_new=y_pos+length_y;

%plot model outer diameter
plot([x_pos,x_pos_new], [y_pos, y_pos_new], 'k-', 'LineWidth', out_diameter)

%plot model inner diameter
plot([x_pos,x_pos_new], [y_pos, y_pos_new], 'w-', 'LineWidth', in_diameter)

end
```

Select Hallway Type—hallway_select.m

```
function [class, width]=hallway_select(flagM, flagI, partial)
if flagM==1
   class = 'g';
else
   class = 'eq';
end
if partial>1250
   width='w';
else
   width='n';
end
if flagI==1
   disp('mature function')
end
```

Select Junction—junction_select.m

```
function class =junction_select(flagM)
if flagM==1
   class='fl';
elseif flagM==0
   class ='pl';
end
end
```

endMap Junction—junction.m

```
function [heading, x_pos, y_pos]=junction (class, width, heading_prev,heading, x_pos, y_pos)

figure(999), hold on, axis([-1000 1000 -1000 1000]), axis square

scrnsz=get(0,'ScreenSize');
set(999, 'Position', scrnsz)
%Set arguments
if nargin ==0
   class='fl';
   width='w';
   heading=0;
   x_pos=0;
   y_pos=0;
elseif nargin ==1
   width='w';
   heading=0;
   x_pos=0;
   y_pos=0;
elseif nargin == 2
   heading=0;
   x_pos=0;
   y_pos=0;
end

%set type of junction
if strcmp(class, 'fl')
   junc='*';
   size_adjust=1;%needed for corner
elseif strcmp(class, 'pl')&& mod(heading, 90)==0 %check modulus
   junc='+';
   size_adjust=1;
elseif strcmp(class, 'pl')&& mod(heading, 90)==45
   junc='x';
   size_adjust=1;
elseif strcmp(class, 'cl')&& mod(heading, 90)==0 %check modulus
   junc='s';
   size_adjust=3;
```

```
elseif strcmp(class, 'cl')&& mod(heading, 90)==45 %check modulus
    junc='d';
    size_adjust=3;
else
    disp('Invalid junction class')
    return
end

junc_color='y';
%evaluate width of segment
if strcmp(width, 'w')
    out_diameter=56/size_adjust;%size of junction must size adjust for corner junctions
    size=14/size_adjust;%width of individual segments
    path_width=12;
    path_length=20;
elseif strcmp(width, 'n')
    out_diameter=28/size_adjust;
    size=8/size_adjust;
    path_width=6;
    path_length=35;%from 10
elseif strcmp(width, 't')
    out_diameter=18/size_adjust;
    size=5/size_adjust;
    path_width=3;
    path_length=5;
else
    disp('Invalid width class');
    return
end

%clear path calculation
path_length_x=path_length*cosd(heading_prev);
path_length_y=path_length*sind(heading_prev);

size=size/2;
out_diameter=out_diameter/2;

%plot model outer diameter
plot(x_pos, y_pos, [junc_color junc], 'MarkerSize', size,'LineWidth', out_diameter)

%corner path
if strcmp(class, 'cl')&& mod(heading, 90)==0
    plot([x_pos-path_length_y/2, x_pos+path_length_y/2], [y_pos-path_length_x/2, y_pos+path_length_x/2], 'b-', 'LineWidth',
path_width)%shows possible directions from a corner
elseif strcmp(class, 'cl')&& mod(heading, 90)==45
    plot([x_pos-path_length_x/2, x_pos+path_length_x/2], [y_pos+path_length_y/2, y_pos-path_length_y/2], 'b-', 'LineWidth',
path_width)%shows possible directions from a corner
end

%clear path
plot([x_pos, x_pos-path_length_x], [y_pos, y_pos-path_length_y], 'w-', 'LineWidth', path_width)


end
```

Map Angle—angle.m

```
function [heading, x_pos, y_pos]=angle(class,heading, x_pos, y_pos)

if nargin==0
   heading=0;
   x_pos=0;
   y_pos=0;
elseif nargin ==1
   x_pos=0;
   y_pos=0;
end

figure(999), hold on, axis([-1000 1000 -1000 1000])

if strcmp(class, '45l')
   rotation=45;
elseif strcmp(class, '45r')
   rotation=-45;
elseif strcmp(class, '90l')
   rotation=90;
elseif strcmp(class, '90r')
   rotation=-90;
elseif strcmp(class, 'fw')
   rotation=0;
else
   disp('Invalid angle class')
   return
end

heading=heading+rotation;

end
```

Angle Select from Heading Change—angle_select.m

```
function class = angle_select (delta_h)

if delta_h==0
    class='fw';
elseif delta_h==45
   class='45l';
elseif delta_h==90
   class='90l';
elseif delta_h==-45
   class='45r';
elseif delta_h==-90
   class='90r';
end

end
```

Map Space—space.m

```
function [heading_new, x_pos_new, y_pos_new]=space (class, width,...
    heading_new, hokuyo_map,read_scan, heading, x_pos, y_pos)

figure(999), hold on, axis([-1000 1000 -1000 1000]), axis square

scrnsz=get(0,'ScreenSize');
set(999, 'Position', scrnsz)
%Set arguments
if nargin ==0
    class='up';
    width='w';
    heading_new=0;
    hokuyo_map='on';
    read_scan=0;
    heading=0;
    x_pos=0;
    y_pos=0;
elseif nargin ==1
    width='w';
    heading_new=0;
    hokuyo_map='on';
    read_scan=0;
    heading=0;
    x_pos=0;
    y_pos=0;
elseif nargin == 2
    hokuyo_map='on';
    heading_new=0;
    read_scan=0;
    heading=0;
    x_pos=0;
    y_pos=0;
elseif nargin ==3
    hokuyo_map='on';
    read_scan=0;
    heading=0;
    x_pos=0;
    y_pos=0;
end

space_color=[.9 .9 .9];
space_type='c';
%set the square space width
if strcmp(width, 'w')
    space_width=5;
elseif strcmp(width, 'n')
    space_width=4;
elseif strcmp(width, 't')
    space_width=3;
else
    disp('invalid space width value')
end

base45=(sqrt(2)/4)*space_width;
base90=(1/2)*space_width;

%set type of space exit location
%delta_x and delta_y are local distances inside of space

if sind(heading)>0 && cosd(heading)>0%45 degree heading

    if strcmp(class, 'up')
```

```
        delta_x=base45*2;
        delta_y=base45*2;

    elseif strcmp(class, 'c')

        delta_x=base45*1;
        delta_y=base45*1;

    elseif strcmp(class, 'down')

        delta_x=base45*0;
        delta_y=base45*0;

    elseif strcmp(class, 'upl')

        delta_x=base45*1;
        delta_y=base45*3;

    elseif strcmp(class, 'left')

        delta_x=base45*0;
        delta_y=base45*2;

    elseif strcmp(class, 'dl')

        delta_x=base45*-1;
        delta_y=base45*1;

    elseif strcmp(class, 'dr')

        delta_x=base45*1;
        delta_y=base45*-1;

    elseif strcmp(class, 'right')

        delta_x=base45*2;
        delta_y=base45*0;

    elseif strcmp(class, 'upr')

        delta_x=base45*3;
        delta_y=base45*1;

    else
        disp('Invalid space exit class')
        return
    end

elseif sind(heading)>0 && cosd(heading)<0%%135 degree heading
    if strcmp(class, 'up')

        delta_x=base45*-2;
        delta_y=base45*2;

    elseif strcmp(class, 'c')

        delta_x=base45*-1;
        delta_y=base45*1;

    elseif strcmp(class, 'down')

        delta_x=base45*0;
        delta_y=base45*0;

    elseif strcmp(class, 'upr')
```

```
        delta_x=base45*-1;
        delta_y=base45*3;

    elseif strcmp(class, 'right')

        delta_x=base45*0;
        delta_y=base45*2;

    elseif strcmp(class, 'dr')

        delta_x=base45*1;
        delta_y=base45*1;

    elseif strcmp(class, 'dl')

        delta_x=base45*-1;
        delta_y=base45*-1;

    elseif strcmp(class, 'left')

        delta_x=base45*-2;
        delta_y=base45*0;

    elseif strcmp(class, 'upl')

        delta_x=base45*-3;
        delta_y=base45*1;

    else
        disp('Invalid space exit class')
        return
    end


elseif sind(heading)<0 && cosd(heading)<0%%225 degree heading
    if strcmp(class, 'up')

        delta_x=base45*-2;
        delta_y=base45*-2;

    elseif strcmp(class, 'c')

        delta_x=base45*-1;
        delta_y=base45*-1;

    elseif strcmp(class, 'down')

        delta_x=base45*0;
        delta_y=base45*0;

    elseif strcmp(class, 'upl')

        delta_x=base45*-1;
        delta_y=base45*-3;

    elseif strcmp(class, 'left')

        delta_x=base45*0;
        delta_y=base45*-2;

    elseif strcmp(class, 'dl')

        delta_x=base45*1;
        delta_y=base45*-1;

    elseif strcmp(class, 'dr')
```

```
        delta_x=base45*-1;
        delta_y=base45*1;

    elseif strcmp(class, 'right')

        delta_x=base45*-2;
        delta_y=base45*0;

    elseif strcmp(class, 'upr')

        delta_x=base45*-3;
        delta_y=base45*-1;

    else
        disp('Invalid space exit class')
        return
    end

elseif sind(heading)<0 && cosd(heading)>0%315 degree heading

    if strcmp(class, 'up')

        delta_x=base45*2;
        delta_y=base45*-2;

    elseif strcmp(class, 'c')

        delta_x=base45*1;
        delta_y=base45*-1;

    elseif strcmp(class, 'down')

        delta_x=base45*0;
        delta_y=base45*0;

    elseif strcmp(class, 'upl')

        delta_x=base45*3;
        delta_y=base45*-1;

    elseif strcmp(class, 'left')

        delta_x=base45*2;
        delta_y=base45*0;

    elseif strcmp(class, 'dl')

        delta_x=base45*1;
        delta_y=base45*1;

    elseif strcmp(class, 'dr')

        delta_x=base45*-1;
        delta_y=base45*-1;

    elseif strcmp(class, 'right')

        delta_x=base45*0;
        delta_y=base45*2;

    elseif strcmp(class, 'upr')

        delta_x=base45*1;
        delta_y=base45*-3;
    else
        disp('Invalid space exit class')
        return
    end

elseif sind(heading)==0 && cosd(heading)==1%0 degree heading

    if strcmp(class, 'up')

        delta_x=base90*2;
        delta_y=base90*0;

    elseif strcmp(class, 'c')

        delta_x=base45*1;
        delta_y=base45*0;

    elseif strcmp(class, 'down')

        delta_x=base90*0;
        delta_y=base90*0;

    elseif strcmp(class, 'upl')

        delta_x=base90*2;
        delta_y=base90*1;

    elseif strcmp(class, 'left')

        delta_x=base90*1;
        delta_y=base90*1;

    elseif strcmp(class, 'dl')

        delta_x=base90*0;
        delta_y=base90*1;

    elseif strcmp(class, 'dr')

        delta_x=base90*0;
        delta_y=base90*-1;

    elseif strcmp(class, 'right')

        delta_x=base90*1;
        delta_y=base90*-1;

    elseif strcmp(class, 'upr')

        delta_x=base90*2;
        delta_y=base90*-1;

    else
        disp('Invalid space exit class')
        return
    end

elseif sind(heading)==1 && cosd(heading)==0%%90 degree heading
    if strcmp(class, 'up')

        delta_x=base90*0;
        delta_y=base90*2;

    elseif strcmp(class, 'c')

        delta_x=base45*0;
        delta_y=base45*1;

    elseif strcmp(class, 'down')

        delta_x=base90*0;
        delta_y=base90*0;

    elseif strcmp(class, 'upr')

        delta_x=base90*1;
        delta_y=base90*2;

    elseif strcmp(class, 'right')

        delta_x=base90*1;
        delta_y=base90*1;

    elseif strcmp(class, 'dr')

        delta_x=base90*1;
        delta_y=base90*0;

    elseif strcmp(class, 'dl')

        delta_x=base90*-1;
        delta_y=base90*0;

    elseif strcmp(class, 'left')

        delta_x=base90*-1;
        delta_y=base90*-1;

    elseif strcmp(class, 'upl')

        delta_x=base90*-1;
        delta_y=base90*2;

    else
        disp('Invalid space exit class')
        return
    end


elseif sind(heading)==0 && cosd(heading)==-1%%180 degree heading
    if strcmp(class, 'up')

        delta_x=base90*-2;
        delta_y=base90*0;

    elseif strcmp(class, 'c')

        delta_x=base45*-1;
        delta_y=base45*0;

    elseif strcmp(class, 'down')

        delta_x=base90*0;
        delta_y=base90*0;

    elseif strcmp(class, 'upl')

        delta_x=base90*-2;
        delta_y=base90*-1;

    elseif strcmp(class, 'left')
```

```
        delta_x=base90*-1;
        delta_y=base90*-1;

    elseif strcmp(class, 'dl')

        delta_x=base90*0;
        delta_y=base90*-1;

    elseif strcmp(class, 'dr')

        delta_x=base90*0;
        delta_y=base90*1;

    elseif strcmp(class, 'right')

        delta_x=base90*-1;
        delta_y=base90*1;

    elseif strcmp(class, 'upr')

        delta_x=base90*-2;
        delta_y=base90*1;

    else
        disp('Invalid space exit class')
        return
    end

elseif sind(heading)==-1 && cosd(heading)==00%270 degree heading

    if strcmp(class, 'up')

        delta_x=base90*0;
        delta_y=base90*-2;

    elseif strcmp(class, 'c')

        delta_x=base45*0;
        delta_y=base45*-1;

    elseif strcmp(class, 'down')

        delta_x=base90*0;
        delta_y=base90*0;

    elseif strcmp(class, 'upl')

        delta_x=base90*1;
        delta_y=base90*-2;

    elseif strcmp(class, 'left')

        delta_x=base90*1;
        delta_y=base90*-1;

    elseif strcmp(class, 'dl')

        delta_x=base90*1;
        delta_y=base90*0;

    elseif strcmp(class, 'dr')

        delta_x=base90*-1;
        delta_y=base90*0;

    elseif strcmp(class, 'right')
```

```
        delta_x=base90*-1;
        delta_y=base90*-1;

    elseif strcmp(class, 'upr')

        delta_x=base90*-1;
        delta_y=base90*-2;

    else
        disp('Invalid space exit class')
        return
    end

end

% %set new end point locations
 x_pos_new=x_pos+delta_x*2;
 y_pos_new=y_pos+delta_y*2;

%plot model square from center point considering heading
if mod(heading, 90)==0&&~strcmp(space_type, 'c')
    plot(x_pos+(space_width/2)*cosd(heading), y_pos+space_width/2*sind(heading), 's', 'MarkerSize', 10*space_width/2,'LineWidth', 5,
'MarkerEdgeColor', 'none', 'MarkerFaceColor', space_color)
elseif mod(heading,90)~=0 && ~strcmp(space_type, 'c')
    plot(x_pos+(space_width/2)*cosd(heading), y_pos+space_width/2*sind(heading), 'd', 'MarkerSize', space_width/2,'LineWidth', 1,
'MarkerEdgeColor', 'none', 'MarkerFaceColor', space_color)
end

%%plot hokuyo scan
if strcmp(hokuyo_map, 'on')
    hokuyo_mapper('map', 'on', read_scan/5000*2, heading, x_pos, y_pos);
end

end
```

Space Class Select—region_select.m

```
function [class, heading_new] = region_select(heading, delta_h, delta_x, delta_y)

heading_new=delta_h;
if delta_y >=.75 && delta_x >=-.25 && delta_x <= .25
    class='up';
elseif delta_y >=.75 && delta_x >=.25
    class='upr';
elseif delta_y >=.75 && delta_x <= -.25
    class='upl';
elseif delta_y >=.25 && delta_y<=.75 && delta_x >=-.25 && delta_x <= .25
    disp('c')
    class='c';
elseif delta_y >=.25 && delta_y <= .75 && delta_x >=.25
    class='right';
elseif delta_y >=.25 && delta_y <= .75 && delta_x <= -.25
    class='left';
elseif delta_y <=.25 && delta_y >= -.25 && delta_x >=-.25 && delta_x <= .25
    class='down';
elseif delta_y <=.25 && delta_y >= -.25 && delta_x >=.25
    class='dr';
elseif delta_y <=.25 && delta_y >= -.25 && delta_x <= -.25
    class='dl';
end
```

Print Raw Sensor Data from LIDAR—hokuyo_mapper

```
function [heading, x_pos,y_pos]=hokuyo_mapper(class, plotflag, read_scan, heading, x_pos, y_pos)

%set default values
if nargin==0
   class='sim';
   plotflag='on';
   read_scan=0;
   heading=0;
   x_pos=0;
   y_pos=0;
elseif nargin==1
   plotflag='on';
   read_scan=0;
   heading=0;
   x_pos=0;
   y_pos=0;
elseif nargin==2
   read_scan=0;
   heading=0;
   x_pos=0;
   y_pos=0;
end

%get hokuyo data
if strcmp(class, 'sim')% map the range of the hokuyo sensor
   r=6*ones(769,1);
elseif strcmp(class, 'map')%map a stored hokuyo scan
   r=read_scan;
elseif strcmp(class, 'scan')%hardware is on and data updated realtime
   r=hokuyoAPI('getScan', 'range', 0, 768, 1, 3);
end

%handles.angles=(compnum-120):.3125:(compnum+120);% what it should be
angles=(heading-135):.3515625:(heading+135);% what it is

%local values of x and y from the scan point
x_local=r'.*cosd(angles);
y_local=r'.*sind(angles);

%global values of x and y
x=x_local+x_pos;
y=y_local+y_pos;


if strcmp(plotflag, 'on')
%    angles_ref=1000*ones(1, 769);
%    angles_ref(1)=0;
%    angles_ref(769)=0;
%    xref=angles_ref.*cosd(angles);
%    yref=angles_ref.*sind(angles);
%    xyref=[xref;yref];
plot(x,y, 'k.', 'MarkerSize', 5)

end

end
```

QUICK

QUICK Figure—maximize_figure.m

```
function map= maximize_figure(handle)
figure(handle)
scrnsz=get(0, 'ScreenSize');
set(handle,'Position', scrnsz);
figure(handle)
map = frame2im(getframe(handle));
```

Mapper Logger for Optimization—mapper_logger.m

```
function [mapper_log, junction_log, junction_number, heading, x_pos, y_pos]=mapper_logger(mapper_number, junction_number, ...
   mapper_log, junction_log, flagF, flagU, heading, x_pos, y_pos, delta_path, delta_x, delta_y, flagM, flagI, iteration)

mapper_log(mapper_number,1)=mapper_number;
mapper_log(mapper_number,2)=flagF;
mapper_log(mapper_number,3)=flagU;
mapper_log(mapper_number,4)=heading;
mapper_log(mapper_number,5)=x_pos;
mapper_log(mapper_number,6)=y_pos;
mapper_log(mapper_number,7)=delta_path;
mapper_log(mapper_number,8)=delta_x;
mapper_log(mapper_number,9)=delta_y;
mapper_log(mapper_number,10)=flagM;
mapper_log(mapper_number,11)=flagI;
mapper_log(mapper_number,12)=iteration;

if flagF==1&&flagU==0&&flagM==1%see if junction occured and see if junction stats are different than previous junction

    junction_number=junction_number+1;
    junction_log(junction_number, 1)=junction_number;
    junction_log(junction_number, 2)=heading;
    junction_log(junction_number, 3)=x_pos;
    junction_log(junction_number, 4)=y_pos;
    junction_log(junction_number, 5)=mapper_number;

    if junction_number>1 && all(junction_log(junction_number, 2:4)==junction_log(junction_number-1, 2:4))

       disp('not unique')

       junction_number=junction_number-1;
    else
       disp('unique')
    end

   [mapper_log, junction_log, heading, x_pos, y_pos]=POI(mapper_log, junction_log, junction_number, heading, x_pos, y_pos);

end


end
```

Passive Observation Interpretation—POI.m

```
function [mapper_log, junction_log, heading, x_pos, y_pos] = POI (mapper_log, junction_log, junction_number, heading, x_pos, y_pos)

  %values of the current junction to compare to previous junctions
  heading_compare=junction_log(junction_number, 2);
  x_pos_compare=junction_log(junction_number, 3);
  y_pos_compare=junction_log(junction_number,4);
for i=1:junction_number-1

  %iteratively calculate the distance between current junction and
  %previous junctions
  x_diff=x_pos_compare-junction_log(i,3);
  y_diff=y_pos_compare-junction_log(i,4);

  %if junction are a distance of 50 apart or less
  if heading_compare==junction_log(i, 2) && abs(x_diff)<=60 && x_diff ~= 0 && y_diff<=100

    %the new x_pos is shifted the difference
    x_pos=x_pos-x_diff;
    junction_log(junction_number, 6)=x_diff;%shift ammount is stored in junction_log for troubleshooting


    mapper_number_start=junction_log(junction_number-1,5);%find the mapper model that begins at the previous junction
    mapper_number_end=junction_log(junction_number, 5);%find the mapper model that ends at the current junction
    mapper_log(mapper_number_start:mapper_number_end,5)=mapper_log(mapper_number_start:mapper_number_end,5)-x_diff; %shift
all mapper models from junction to junction an ammount x_diff

    %mapper_number_end=junction_log(junction_number,5);
    %junction_log(i,3)=rand*10000;
    junction_log(junction_number,3)=junction_log(i,3);
  end

  if heading_compare==junction_log(i,2) && abs(y_diff)<=60 && y_diff ~= 0 && x_diff<=100% if y_diff is less than 50

    %the new y_pos value is shifted the difference
    y_pos=y_pos-y_diff;
    junction_log(junction_number, 7)=y_diff;%stored in junction_log for troubleshooting
    mapper_number_start=junction_log(junction_number-1,5);%find the mapper model that begins at the previous junction
    mapper_number_end=junction_log(junction_number, 5);%find the mapper model that ends at the current junction
    mapper_log(mapper_number_start:mapper_number_end,6)=mapper_log(mapper_number_start:mapper_number_end,6)-y_diff;% shift
all mapper models from junction to junction an amount y_diff

    %junction_log(i,4)=rand*10000;
    junction_log(junction_number, 4)=junction_log(i, 4);

  end

  %visual plot function for the fact that we are moving the x_pos

end

end
```

Initialize QUICK Map—QUICK_init.m

```
function map=QUICK_init(map, submap1,submap2,submap3)

if nargin==1
   submap1=zeros(120,160,3);
   submap2=zeros(120,160,3);
   submap3=zeros(120,160,3);
elseif nargin ==2
   submap2=zeros(120,160,3);
   submap3=zeros(120,160,3);
elseif nargin ==3
   submap3=zeros(120,160,3);
end

map(31:150,41:200,:)=submap1;
map(195:314,41:200,:)=submap2;
map(359:478,41:200,:)=submap3;
map(523:642,41:200,:)=zeros(120,160,3);
map(687:806,41:200,:)=zeros(120,160,3);
map(31:150,1101:1260,:)=zeros(120,160,3);
map(195:314,1101:1260,:)=zeros(120,160,3);
map(359:478,1101:1260,:)=zeros(120,160,3);
map(523:642,1101:1260,:)=zeros(120,160,3);
map(687:806,1101:1260,:)=zeros(120,160,3);

figure,imshow(map)

text(41, 31-1, '1.', 'Color', 'k');
text(41, 195-1, '2.', 'Color', 'k');
text(41, 359-1, '3.', 'Color', 'k');
text(41, 523-1, '4.', 'Color', 'k');
text(41, 687-1, '5.', 'Color', 'k');
text(1101, 31-1, '6.', 'Color', 'k');
text(1101, 195-1, '7.', 'Color', 'k');
text(1101, 359-1, '8.', 'Color', 'k');
text(1101, 523-1, '9.', 'Color', 'k');
text(1101, 687-1, '10.', 'Color', 'k');

end
```

QUICK Mapper—QUICK.m

```
function map=QUICK(map, object, object_log, mapper_log, labels)

clear all

map(31:150,41:200,:)=object(1, 1).localIm_background;
map(195:314,41:200,:)= object(2, 1).localIm_background;
map(359:478,41:200,:)= object(3, 1).localIm_background;
map(523:642,41:200,:)= object(4, 1).localIm_background;
map(687:806,41:200,:)= object(5, 1).localIm_background;
map(31:150,1101:1260,:)= object(6, 1).localIm_background;
map(195:314,1101:1260,:)= object(7, 1).localIm_background;
map(359:478,1101:1260,:)= object(8, 1).localIm_background;
map(523:642,1101:1260,:)= object(9, 1).localIm_background;
map(687:806,1101:1260,:)= object(10, 1).localIm_background;

figure,imshow(map)

text(41, 31-1, ['1. ' labels(1,:)], 'Color', 'b');
text(41, 195-1, ['2. ' labels(1,:)], 'Color', 'g');
text(41, 359-1, ['3. ' labels(1,:)], 'Color', 'r');
text(41, 523-1, ['4. ' labels(1,:)], 'Color', 'c');
text(41, 687-1, ['5. ' labels(1,:)], 'Color', 'm');
text(1101, 31-1, ['6. ' labels(1,:)], 'Color', 'b');
text(1101, 195-1, ['7. ' labels(1,:)], 'Color', 'g');
text(1101, 359-1, ['8. ' labels(1,:)], 'Color', 'r');
text(1101, 523-1, ['9. ' labels(1,:)], 'Color', 'c');
text(1101, 687-1, ['10. ' labels(1,:)], 'Color', 'm');

end

%object(k,1).deltax=click_location_num(1)-320;
%object(k,1).deltay=handles.dist_disp;%disparity distance
 %object(k,1).template=localIm_template;%60x80x3 color template of object
plot_objects(object_log, mapper_log, object)
```

Feature Extraction

Color Histogram—createColorHistogram_cont.m

```
%==============================================================
% createColorHistograms(im_str)
% im_str can be an image file location or a three-dimensional array
%==============================================================

%==============================================================
% Copyright (C) 2007-2008 Chaitanya Sai Gaddam (gsc at cns.bu dot edu)
% http://www.discerniblepreferences.com
% Redistribution and use in source and binary forms, with or without
% modification, are permitted provided that the following conditions
% are met:
% 1. Redistributions of source code must retain the above copyright
%    notice, this list of conditions and the following disclaimer.
% 2. Redistributions in binary form must reproduce the above copyright
%    notice, this list of conditions and the following disclaimer in the
%    documentation and/or other materials provided with the distribution.
% 3. Neither the name of the Chaitanya Sai Gaddam nor the names of its contributors
%    may be used to endorse or promote products derived from this software
%    without specific prior written permission.
%
% THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND
% ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
% IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
% ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
% FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
% DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
% OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
% HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
% LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
% OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
% SUCH DAMAGE.
%==============================================================

function [col_val col_freq col_RGB]=createColorHistograms_cont(Im)

Im=Im*255;

%res_val: Binning resolution. Increasing the value gives coarsers bins
%===
res_val=64;

%===
%t_count: Color triplets are converted to a single value for purposes of
%binning
%===
t_count=res_val*floor(Im(:,:,1)/res_val)+256*(res_val*floor(Im(:,:,2)/res_val))+256*256*(res_val*floor(Im(:,:,3)/res_val));
t_count=sort(t_count(:));

%===
% Use unique to calculate the number of triplets (ind_last-ind_first) in each bin
%===
[col_val,ind_first]=unique(t_count,'first');
[~,ind_last]=unique(t_count,'last');
col_freq=(ind_last-ind_first+1)/(size(Im,1)*size(Im,2));%normalize based on the number of color pixels to get probabilities..

col_RGB=([rem(col_val,256) floor(rem(col_val,256*256)/256) floor(col_val/(256*256))]);
```

MSER Feature Detection—mser_detector_cont.m

```
function [r,f, M, dist_disp, max_f_radius, index_f_radius, num_feat]=mser_detector_cont(Im)

se = strel('rectangle',[3,3]);%Morphological operation to get rid of inconsistencies in the disparity

Im=imdilate(Im, se);
Im_feat=uint8(Im*.3391);

[r,f] = vl_mser(Im_feat,'MinDiversity',.01,'MaxVariation',.99,'Delta',1, 'BrightOnDark',1,'DarkOnBright',0 ) ;%adjust diversity down to more
features and get overlap of features.  Adjust MaxVariation to consider more features as the same and merge more
feat=uint8(vl_ertr(f));%for small image indexing
f = vl_ertr(f);
dim_feat=size(f);
num_feat=dim_feat(2);
f_temp=f;
[max_f_radius, index_f_radius]=max(f_temp(5, :), [], 2);
f_temp(5,index_f_radius)=0;
[max2_f_radius, index2_f_radius]=max(f_temp(5,:), [], 2);
ftemp(5, index2_f_radius)=0;
[max3_f_radius, index3_f_radius]=max(f_temp(5,:), [], 2);

M = zeros(size(Im_feat)) ;
  for x=r'
    s = vl_erfill(Im_feat,x) ;
    M(s) = M(s) + 1;
  end

if max_f_radius>=10
  Intensity=Im(feat(2,index_f_radius), feat(1, index_f_radius));
  Intensity2=Im(feat(2,index2_f_radius), feat(1, index2_f_radius));
  Intensity3=Im(feat(2,index3_f_radius), feat(1, index3_f_radius));

  if Intensity>0
    dist_disp=800/Intensity;%from disparity information get dist val
  elseif Intensity2>0
    dist_disp=800/Intensity2;
  else
    dist_disp=800/Intensity3;
  end
else
  Intensity=-2;%negative 2 indicates there was not significant feature point data to extract disparity
  dist_disp=-2;
end

end
```

Feature Shift for Tracking—kmean_change.m

```
function [change, centroid, k_val]=kmean_change(change_all, change)

change_unique=unique(change_all', 'rows');

if size(change_unique, 1)>=3

    [k_val, centroid, kdist]=kmeans(change_unique, 1,'EmptyAction','drop');

    kdist(kdist==0)=50;

    [num_change, indch]=min(kdist, [], 1);
    change_final1=centroid(indch,:);
    change=change_final1';

end

end
```

Matching Descriptors—match_compare.m

```
function template_id=match_compare(d, d_matrix, thresh, n)

%test1=d_matrix
%    d_matrix_ind=any(shiftdim(d_matrix, 1));%eliminate zero planes
%    d_matrix=d_matrix(:,:,d_matrix_ind(:,:,1));%eliminate zero planes
%test2=d_matrix

%test1=d_matrix
%d_matrix=d_matrix(:,:,1:n);
%test2=d_matrix

num_match=zeros(1,n);
template_id=0;
for m=1:n
    d_prev=d_matrix(:, :, m);%compare to previous saved templates


    d_prev=d_prev(:,any(d_prev,1));%eliminate zero columns

    matches=vl_ubcmatch(d, d_prev, 2);
    if size(matches, 2)>=thresh && m~=n
        disp('template match')
        num_match(m)=size(matches,2);
        [~,template_id]=max(num_match, [],2);
    end

end
```

Octal Image Creation—shiftIm_octal.m

```
function [ind_Bhatt, octalIm, Bhatt, min_Bhatt, dist]=shiftIm_octal(Im, temp_vect, temp_freq)
%%Took the "Stat parameter away and replaced with Bhattacharyya coefficient

if nargin==1
   temp_vect=0;%zero inputs
   temp_freq=0;

end

sizeIm=floor(size(Im)/2);
halfsizeIm=floor(sizeIm/2);
octalIm=zeros(sizeIm(1), sizeIm(2), 3, 9);
ind_Bhatt=0;
Bhatt=0;
min_Bhatt=0;
dist=0;

octalIm(:,:,:,1)=Im(1:sizeIm(1), 1+halfsizeIm(2):sizeIm(2)+halfsizeIm(2), :);%%ImN
octalIm(:,:,:,2)=Im(1+halfsizeIm(1):sizeIm(1)+halfsizeIm(1), 1+sizeIm(2):sizeIm(2)+sizeIm(2), :);%%ImE
octalIm(:,:,:,3)=Im(1+halfsizeIm(1):sizeIm(1)+halfsizeIm(1), 1+halfsizeIm(2):sizeIm(2)+halfsizeIm(2), :);%%ImM
octalIm(:,:,:,4)=Im(1+halfsizeIm(1):sizeIm(1)+halfsizeIm(1), 1:sizeIm(2), :);%%ImW
octalIm(:,:,:,5)=Im(1+sizeIm(1):sizeIm(1)+sizeIm(1), 1+halfsizeIm(2):sizeIm(2)+halfsizeIm(2), :);%%ImS
octalIm(:,:,:,6)=Im(1:sizeIm(1), 1:sizeIm(2), :);%%ImNW
octalIm(:,:,:,7)=Im(1:sizeIm(1), 1+sizeIm(2):sizeIm(2)+sizeIm(2), :);%%ImNE
octalIm(:,:,:,8)=Im(1+sizeIm(1):sizeIm(1)+sizeIm(1), 1+sizeIm(2):sizeIm(2)+sizeIm(2), :);%%ImSE
octalIm(:,:,:,9)=Im(1+sizeIm(1):sizeIm(1)+sizeIm(1), 1:sizeIm(2), :);%%ImSW

if temp_vect(1)~=0||temp_freq(1)~=0

   %Stat_mat=zeros(size(temp_freq,1), 9);
   Bhatt_mat=zeros(size(temp_freq,1), 9);
   for k=1:9
      [col_vect freq_vect]=createColorHistograms_cont(octalIm(:,:,:,k));%for each octIm find color hist vectors
      [compare_vect, Itemp,Icol]=intersect(temp_vect, col_vect);%find and equate common color bins

      for m=1:size(compare_vect,1)
        Stat_mat(m,k)=abs(temp_freq(Itemp(m))-freq_vect(Icol(m)));%MUST DEAL IN PROBABILITIES therefore divide by 4800
         Bhatt_mat(m,k)=sqrt(temp_freq(Itemp(m))*freq_vect(Icol(m)));
      end

   end
   Stat=sum(Stat_mat);%Sum absolute differences mean-shift similarity function
   Bhatt=sum(Bhatt_mat);
   dist=sqrt(1-Bhatt);
   %[min_Stat, ind_Stat]=min(Stat);
   dist(3)=1.5*dist(3);
   [min_Bhatt, ind_Bhatt]=min(dist);

end
```

SIFT Feature Extractor—sift_detector_cont.m

```
function [fprev, dprev, matches, change, change_all, scores, comparison, background_feat, object_feat]=sift_detector_cont(Im, fprev, dprev,
localRect, xoffset, yoffset)
Im=single(Im);
d=dprev;
f=fprev;

[fprev, dprev]=vl_sift(Im);%default PeakThresh is 0 and default EdgeThresh is 10
%fprev=fprev;

[matches, scores]=vl_ubcmatch(d,dprev, 2);

[change, comparison, background_feat, object_feat, change_all]=feature_compare(f, fprev, matches, localRect, xoffset, yoffset);

end
```

Shape Feature Extractor—disp_seg.m

```
function [outlineO, outlineI]=disp_seg(disp, val_min, val_max)

disp_logical=val_max>=disp & val_min<=disp;

halfsize=floor(size(disp, 2)/2);

disp_leftsideO=disp_logical(:,1:halfsize);
disp_rightsideO=fliplr(disp_logical(:, halfsize+1:size(disp, 2)));
disp_rightsideI=disp_logical(:,halfsize+1:size(disp,2));
disp_leftsideI=fliplr(disp_logical(:, 1:halfsize));
outlineO=zeros(size(disp));
outlineI=zeros(size(disp));
for i=1:size(disp_leftsideO, 1)-1
  leftflagO=0;
  rightflagO=0;
  leftflagI=0;
  rightflagI=0;

  for j=1:size(disp_leftsideO, 2)
    if outlineO(i,j)==0 &&disp_leftsideO(i,j)~=0&&leftflagO==0;
      outlineO(i,j)=1;
      leftflagO=1;
      %j=size(disp_leftside,2);
    end

    if outlineI(i,j+halfsize)==0 &&disp_rightsideI(i,j)~=0&&rightflagI==0;
      outlineI(i,j+halfsize)=1;%UNCOMMENT FOR INNER VALUES
      rightflagI=1;
      %j=size(disp_leftside,2);
    end

    if outlineO(i,size(disp, 2)-j)==0 && disp_rightsideO(i,j)~=0&&rightflagO==0;
      outlineO(i,size(disp, 2)-j)=1;
      rightflagO=1;
    end

    if outlineI(i,size(disp, 2)-halfsize-j+1)==0 &&disp_leftsideI(i,j)~=0&&leftflagI==0;%UNCOMMENT FOR INNER VALUES
      outlineI(i,size(disp, 2)-halfsize-j+1)=1;
      leftflagI=1;
    end

  end
end
```

Servo Functions

Initialize Servo—servo_init.m

```
function handle=servo_init()

  if ~libisloaded('phidget21')%if library is not loaded, load library
     loadlibrary('phidget21', 'phidget21matlab.h');
  end

     ptr = libpointer('int32Ptr', 0);
     calllib('phidget21', 'CPhidgetServo_create', ptr);

     %ptr points to a CPhidgetServoHandle sturcture but we use it's int value
     %for control functions - handle

     handle = get(ptr, 'Value');

     calllib('phidget21', 'CPhidget_open', handle, -1);

        if calllib('phidget21', 'CPhidget_waitForAttachment', handle, 2500) == 0
           disp('Opened servo')
        else
           disp('Could not Open Servo')
        end
end
```

Single Servo Reading—servo_singlescan.m

```
function servo_singlescan(location, delay, handle, close)


if nargin==3
    close=0;%keep open unless close specified
end

if ~strcmp(close, 'close')

posn=location+25;

calllib('phidget21', 'CPhidgetServo_setPosition', handle, 0, posn);
pause(delay)

else

  calllib('phidget21', 'CPhidget_close', handle);
  calllib('phidget21', 'CPhidget_delete', handle);

end

end
```

Pitch Servo Function—servo_sweep.m

```
function r_out=servo_sweep(angle1, angle2, step, delay, handle)

hokuyo_init;
%angle1=25+angle1;%offset for servo implimentation
%angle2=25+angle2;

if angle1/angle2>=1
   step=-step;
end

sizem=floor((angle2-angle1)/step);

r=zeros(769,sizem+1);

servo_singlescan(angle1,.4, handle);
r(:,1)=hokuyo_singlescan();

k=2;
for m=(angle1+step):step:angle2
   servo_singlescan(m,delay,handle);
   r(:,k)=hokuyo_singlescan();
   k=k+1;
end

%servo_singlescan(80,.4, handle);

r_out=r(311:492, :)';

%figure , imagesc(r_out)


End
```

Appendix B: Fair Use

Images in this document fall under three categories. Images that are the original creation of the author, copyrighted images under Section 106 or Section 106A of US Copyright Law, and images that are accessed as part of the Public Domain of free materials controlled under Section 107 defining Fair Use. The US Copyright law pertaining to these distinctions is as follows:

Section 106. Exclusive rights in copyrighted works

Subject to sections 107 through 122, the owner of copyright under this title has the exclusive rights to do and to authorize any of the following:

(1) to reproduce the copyrighted work in copies or phonorecords;

(2) to prepare derivative works based upon the copyrighted work;

(3) to distribute copies or phonorecords of the copyrighted work to the public by sale or other transfer of ownership, or by rental, lease, or lending;

(4) in the case of literary, musical, dramatic, and choreographic works, pantomimes, and motion pictures and other audiovisual works, to perform the copyrighted work publicly;

(5) in the case of literary, musical, dramatic, and choreographic works, pantomimes, and pictorial, graphic, or sculptural works, including the individual images of a motion picture or other audiovisual work, to display the copyrighted work publicly; and

(6) in the case of sound recordings, to perform the copyrighted work publicly by means of a digital audio transmission.

Section 106A. Rights of certain authors to attribution and integrity

(a) Rights of Attribution and Integrity. — Subject to section 107 and independent of the exclusive rights provided in section 106, the author of a work of visual art —

(1) shall have the right —

(A) to claim authorship of that work, and

(B) to prevent the use of his or her name as the author of any work of visual art which he or she did not create;

(2) shall have the right to prevent the use of his or her name as the author of the work of visual art in the event of a distortion, mutilation, or other modification of the work which would be prejudicial to his or her honor or reputation; and

(3) subject to the limitations set forth in section 113(d), shall have the right —

(A) to prevent any intentional distortion, mutilation, or other modification of that work which would be prejudicial to his or her honor or reputation, and any intentional distortion, mutilation, or modification of that work is a violation of that right, and

(B) to prevent any destruction of a work of recognized stature, and any intentional or grossly negligent destruction of that work is a violation of that right.

(b) Scope and Exercise of Rights. — Only the author of a work of visual art has the rights conferred by subsection (a) in that work, whether or not the author is the copyright owner. The authors of a joint work of visual art are coowners of the rights conferred by subsection (a) in that work.

(c) Exceptions. — (1) The modification of a work of visual art which is the result of the passage of time or the inherent nature of the materials is not a distortion, mutilation, or other modification described in subsection (a)(3)(A).

(2) The modification of a work of visual art which is the result of conservation, or of the public presentation, including lighting and placement, of the work is not a destruction, distortion, mutilation, or other modification described in subsection (a)(3) unless the modification is caused by gross negligence.

(3) The rights described in paragraphs (1) and (2) of subsection (a) shall not apply to any reproduction, depiction, portrayal, or other use of a work in, upon, or in any connection with any item described in subparagraph (A) or (B) of the definition of "work of visual art" in section 101, and any such reproduction, depiction, portrayal, or other use of a work is not a destruction, distortion, mutilation, or other modification described in paragraph (3) of subsection (a).

(d) Duration of Rights. — (1) With respect to works of visual art created on or after the effective date set forth in section 610(a) of the Visual Artists Rights Act of 1990, the rights conferred by subsection (a) shall endure for a term consisting of the life of the author.

(2) With respect to works of visual art created before the effective date set forth in section 610(a) of the Visual Artists Rights Act of 1990, but title to which has not, as of such effective date, been transferred from the author, the rights conferred by subsection (a) shall be coextensive with, and shall expire at the same time as, the rights conferred by section 106.

(3) In the case of a joint work prepared by two or more authors, the rights conferred by subsection (a) shall endure for a term consisting of the life of the last surviving author.

(4) All terms of the rights conferred by subsection (a) run to the end of the calendar year in which they would otherwise expire.

(e) Transfer and Waiver. — (1) The rights conferred by subsection (a) may not be transferred, but those rights may be waived if the author expressly agrees to such waiver in a written instrument signed by the author. Such instrument shall specifically identify the work, and uses of that work, to which the waiver applies, and the waiver shall apply only to the work and uses so identified. In the case of a joint work prepared by two or more authors, a waiver of rights under this paragraph made by one such author waives such rights for all such authors.

(2) Ownership of the rights conferred by subsection (a) with respect to a work of visual art is distinct from ownership of any copy of that work, or of a copyright or any exclusive right under a copyright in that work. Transfer of ownership of any copy of a work of visual art, or of a copyright or any exclusive right under a copyright, shall not constitute a waiver of the rights conferred by subsection (a). Except as may otherwise be agreed by the author in a written instrument signed by the author, a waiver of the rights conferred by subsection (a) with respect to a work of visual art shall not constitute a transfer of ownership of any copy of that work, or of ownership of a copyright or of any exclusive right under a copyright in that work.

Section 107. Limitations on exclusive rights: Fair use

Notwithstanding the provisions of sections 106 and 106A, the fair use of a copyrighted work, including such use by reproduction in copies or phonorecords or by any other means specified by that section, for purposes such as criticism, comment, news reporting, teaching (including multiple copies for classroom use), scholarship, or research, is not an infringement of copyright. In determining whether the use made of a work in any particular case is a fair use the factors to be considered shall include —

(1) the purpose and character of the use, including whether such use is of a commercial nature or is for nonprofit educational purposes;

(2) the nature of the copyrighted work;

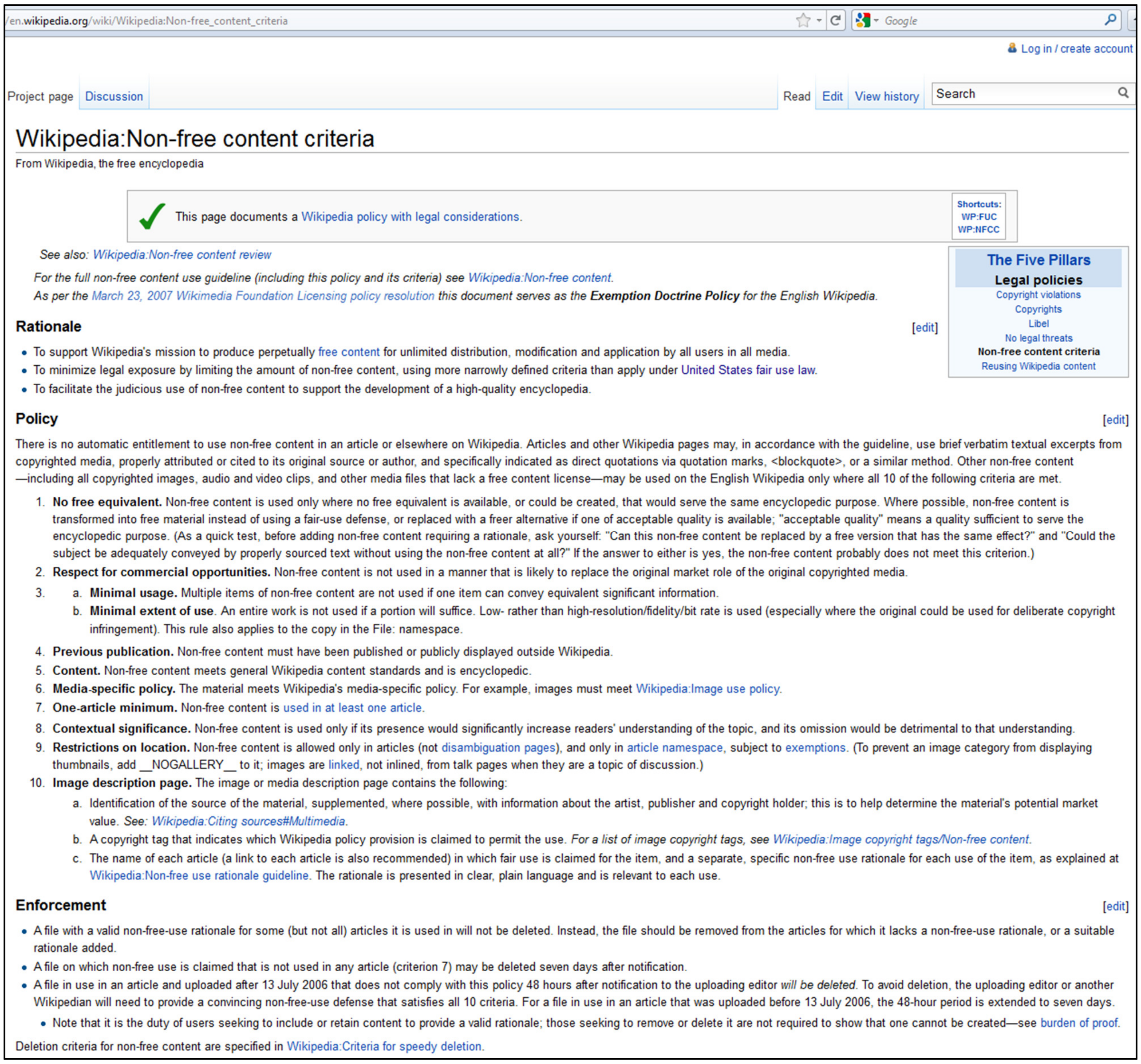
(3) the amount and substantiality of the portion used in relation to the copyrighted work as a whole; and

(4) the effect of the use upon the potential market for or value of the copyrighted work.

The fact that a work is unpublished shall not itself bar a finding of fair use if such finding is made upon consideration of all the above factors.

Wikipedia Content Criteria

All content claiming to fit under Public Domain was obtained from Wikipedia, which has the explicit mission to produce perpetually free content for unlimited distribution, modification and application. Although non-free content is rare, it may only be published considering the content complies with Fair Use. This policy is shown in Figure 113.


/en.wikipedia.org/wiki/Wikipedia:Non-free_content_criteria

Log in / create account

Project page Discussion Read Edit View history Search

Wikipedia:Non-free content criteria

From Wikipedia, the free encyclopedia

This page documents a Wikipedia policy with legal considerations.

Shortcuts: WP:FUC WP:NFCC

See also: Wikipedia:Non-free content review

For the full non-free content use guideline (including this policy and its criteria) see Wikipedia:Non-free content.

*As per the March 23, 2007 Wikimedia Foundation Licensing policy resolution this document serves as the **Exemption Doctrine Policy** for the English Wikipedia.*

The Five Pillars
Legal policies
Copyright violations
Copyrights
Libel
No legal threats
Non-free content criteria
Reusing Wikipedia content

Rationale [edit]

- To support Wikipedia's mission to produce perpetually free content for unlimited distribution, modification and application by all users in all media.
- To minimize legal exposure by limiting the amount of non-free content, using more narrowly defined criteria than apply under United States fair use law.
- To facilitate the judicious use of non-free content to support the development of a high-quality encyclopedia.

Policy [edit]

There is no automatic entitlement to use non-free content in an article or elsewhere on Wikipedia. Articles and other Wikipedia pages may, in accordance with the guideline, use brief verbatim textual excerpts from copyrighted media, properly attributed or cited to its original source or author, and specifically indicated as direct quotations via quotation marks, <blockquote>, or a similar method. Other non-free content —including all copyrighted images, audio and video clips, and other media files that lack a free content license—may be used on the English Wikipedia only where all 10 of the following criteria are met.

1. **No free equivalent.** Non-free content is used only where no free equivalent is available, or could be created, that would serve the same encyclopedic purpose. Where possible, non-free content is transformed into free material instead of using a fair-use defense, or replaced with a freer alternative if one of acceptable quality is available; "acceptable quality" means a quality sufficient to serve the encyclopedic purpose. (As a quick test, before adding non-free content requiring a rationale, ask yourself: "Can this non-free content be replaced by a free version that has the same effect?" and "Could the subject be adequately conveyed by properly sourced text without using the non-free content at all?" If the answer to either is yes, the non-free content probably does not meet this criterion.)
2. **Respect for commercial opportunities.** Non-free content is not used in a manner that is likely to replace the original market role of the original copyrighted media.
3. a. **Minimal usage.** Multiple items of non-free content are not used if one item can convey equivalent significant information.
   b. **Minimal extent of use**. An entire work is not used if a portion will suffice. Low- rather than high-resolution/fidelity/bit rate is used (especially where the original could be used for deliberate copyright infringement). This rule also applies to the copy in the File: namespace.
4. **Previous publication.** Non-free content must have been published or publicly displayed outside Wikipedia.
5. **Content.** Non-free content meets general Wikipedia content standards and is encyclopedic.
6. **Media-specific policy.** The material meets Wikipedia's media-specific policy. For example, images must meet Wikipedia:Image use policy.
7. **One-article minimum.** Non-free content is used in at least one article.
8. **Contextual significance.** Non-free content is used only if its presence would significantly increase readers' understanding of the topic, and its omission would be detrimental to that understanding.
9. **Restrictions on location.** Non-free content is allowed only in articles (not disambiguation pages), and only in article namespace, subject to exemptions. (To prevent an image category from displaying thumbnails, add __NOGALLERY__ to it; images are linked, not inlined, from talk pages when they are a topic of discussion.)
10. **Image description page.** The image or media description page contains the following:
    a. Identification of the source of the material, supplemented, where possible, with information about the artist, publisher and copyright holder; this is to help determine the material's potential market value. *See: Wikipedia:Citing sources#Multimedia.*
    b. A copyright tag that indicates which Wikipedia policy provision is claimed to permit the use. *For a list of image copyright tags, see Wikipedia:Image copyright tags/Non-free content.*
    c. The name of each article (a link to each article is also recommended) in which fair use is claimed for the item, and a separate, specific non-free use rationale for each use of the item, as explained at Wikipedia:Non-free use rationale guideline. The rationale is presented in clear, plain language and is relevant to each use.

Enforcement [edit]

- A file with a valid non-free-use rationale for some (but not all) articles it is used in will not be deleted. Instead, the file should be removed from the articles for which it lacks a non-free-use rationale, or a suitable rationale added.
- A file on which non-free use is claimed that is not used in any article (criterion 7) may be deleted seven days after notification.
- A file in use in an article and uploaded after 13 July 2006 that does not comply with this policy 48 hours after notification to the uploading editor *will be deleted*. To avoid deletion, the uploading editor or another Wikipedian will need to provide a convincing non-free-use defense that satisfies all 10 criteria. For a file in use in an article that was uploaded before 13 July 2006, the 48-hour period is extended to seven days.
  - Note that it is the duty of users seeking to include or retain content to provide a valid rationale; those seeking to remove or delete it are not required to show that one cannot be created—see burden of proof.

Deletion criteria for non-free content are specified in Wikipedia:Criteria for speedy deletion.

Figure 113: Wikipedia policy on Fair Use, Public Domain media, and Non-free content

Fair Use Checklist

All images claimed to fall under Public Domain meet the following stipulations:

(1) The purpose and character of all images is for nonprofit educational purposes;

(2) Copyrited work is contained within Wikipedia and, in the event it is not non-free content, complies with Wikipedia's policy on non-free media;

(3) The amount and substantiality of the portion used in relation to the copyrighted work is minimal;

(4) The effect of the use upon the potential market for or value of the copyrighted work is not substantial.