

Local Bundling of Disparity Maps for Improved Dense 3D Visual Reconstruction

Jason Gassaway

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

Master of Science
In
Mechanical Engineering

Kevin Kochersberger, Chair
A. Lynn Abbott
Harvey Rhody
Mike Roan

July 11, 2011
Blacksburg, VA

Keywords: Stereo Correspondence
Bundle Adjustment
Computer Vision
Terrain Mapping
UAV
Stereo Vision

Copyright 2011, Jason C. Gassaway

Local Bundling of Disparity Maps for Improved Dense 3D Visual Reconstruction

Jason Gassaway

Abstract

This thesis presents a new method for improved resolution of stereoscopic 3D terrain mapping by local dense bundling of disparity maps. The Unmanned Systems Lab (USL) at Virginia Tech is designing an unmanned aerial vehicle (UAV) first-response system capable of 3D terrain mapping in the wake of a nuclear event. The UAV is a helicopter, and is equipped with a stereo boom imaging system, GPS, and an inertial measurement system (IMU) for low-altitude aerial mapping. Previous 3D reconstruction algorithms on the project used two-frame rectified stereo correspondence to create a local 3D map, which was geo-located by raw GPS and IMU data. The new local dense bundling algorithm combines multiple pairs of stereo images by SURF feature point matching, image rectification, matching of dense points with semi-global block matching, and optimization of camera pose and dense 3D point location using a stereo-constrained local bundle adjustment. The performance of the algorithm is evaluated numerically on synthetic imagery and qualitatively on real aerial flight data. Results indicate the algorithm produces marked improvement in accuracy and vertical resolution, given a proper helicopter flight path and sufficient image overlap. With synthetic imagery and precise pose supplied, the algorithm shows a 1.2x to 6x reduction in vertical error.

*To my God, who sustains me through all things,
Thanks.
And praise be to you,
because your love is better than life.*

^ ^
—

Acknowledgements

I would first like to thank my advisor, Dr. Kevin Kochersberger. He have paved the way for my time here in Virginia Tech's graduate program. He provided me means to live, a direction to research, and a lab to call home. Thanks, I am so very grateful. Thanks to Dr. Abbott, for his broad knowledge on computer vision and guidance in my research. Thanks to Dr. Harvey Rhody, for joining me on this venture all the way from New York, I appreciate your new insights into my work. Thanks to Dr. Mike Roan for being on my committee, I appreciate your willingness to join this club. Thanks to Dr. Mani Golparvar-Fard for sharing with me from your extensive knowledge and experience with visual reconstruction.

A big thanks go out to all the guys at the Unmanned Systems Lab. A lab without them would have been a lonely graduate existence. Thanks for all the conversations, distractions, encouragement, troubleshooting, brainstorming, and general wackiness that kept life interesting. And in no particular order: Thanks to Kenny Kroeger, for a perfect record. Thanks to Bryan Krawiec, for a boisterous kindness that took me by surprise when I was down. Thanks Jerry Towler, for edits and helpful advice, for more information about more things than I suspect is legal. Thanks to Brian McCabe, for talks on hopes and dreams, the infrared monocle and for being real (Bully for you good sir). Thanks to Pete Fanto, for a computer in dire times and for being an awesome officemate who was always behind me. Thanks to Nathan Short, for first handing me the torch of USL stereo vision. Thanks to Kevin Stefanik, for giving another boost in computer vision and for breakfasts at Gillies. And a special thanks to Shajan, my constant companion during all those late nights. I appreciate your endurance and companionship my friend, and I know your steadfastness helped me to carry on as well. And of course, thanks to all everyone else at the USL: Eric Gustafson, the Bromley, Matt Torok, Collin Lutz, Mike Couch.

My thanks also go to Aaron Marcus, for a deep friendship that has been a true blessing this entire time of college. Thanks to my roommate Dan, for listening to my woes and worries and encouraging me during the roughest spots. Thanks also to Marc Hutcheson, for creating an amazing 3D blender model for testing. Thanks to Charlie McRaven, one of the wiser men I know, for all the talking, growing, and praying each week. Thanks to Sandy Young for sharing an abounding life wisdom and a fuller perspective.

And especially, to my Mom and Dad, thank you ever so much. Thanks for listening ears and words of wisdom, for encouragement and steadfastness, insight and faithfulness. And especially for your love and your prayers; I never would have made it but for both. Thanks Stephen for good talks, and being an awesome brother.

And most of all, thanks God. You got me through this, and you are the reason I pressed through all this. Let this honor you above all else. I acknowledge you as my God and my strength, my refuge and my father. Thanks.

Table of Contents

Abstract	ii
Acknowledgements	iv
List of Figures.....	viii
List of Tables.....	x
List of Acronyms	xi
List of Symbols.....	xii
1. Introduction.....	1
1.1. Purpose.....	1
1.2. Contribution	2
1.3. Overview of Thesis	3
2. Literature Review	4
2.1. Sparse Feature Matching.....	4
2.1.1. Kanade-Lucas-Tomasi	4
2.1.2. SIFT	5
2.1.3. SURF.....	6
2.2. Dense Feature Matching	7
2.2.1. Window-based Matching	7
2.2.2. Dynamic Programming Methods.....	9
2.2.3. Planar-constrained Matching	10
2.2.4. Sequential Stereo Pair Matching	10
2.3. Comprehensive 3D Reconstruction Algorithms	11
2.3.1. Bundle adjustment	11
2.3.2. Applications	13
2.3.3. Algorithm Frameworks	14
2.3.4. Evaluation of current 3D vision solutions.....	16
3. Background.....	17
3.1. Project Mission	17
3.1.1. Description of mission	17
3.1.2. Description of equipment.....	18
3.2. Overview of Vision-Based 3D Reconstruction.....	20

3.2.1.	Projective Geometry.....	20
3.2.2.	The Epipolar Constraint.....	22
3.2.3.	Rectification.....	23
3.2.4.	Stereo Vision.....	25
3.3.	Theoretical Challenges	26
3.3.1.	Radiometry	26
3.3.2.	Physical Environment	27
3.3.3.	Hardware Limitations	29
3.4.	Assumptions and Constraints.....	33
4.	Methodology	37
4.1.	Existing Code	37
4.1.1.	Pose refinement with grid search and IRLS.....	38
4.1.2.	Semi-Global Block Matching.....	38
4.2.	Local Dense Bundling Algorithm.....	41
4.2.1.	Operational Framework	41
4.2.2.	Sparse Feature Detection and Matching.....	42
4.2.3.	Dense Feature Matching	44
4.2.4.	Local Bundle Adjustment.....	46
4.2.5.	Error checking.....	48
4.3.	Blender Synthetic Image Creator	49
5.	Results	52
5.1.	Synthetic Imagery.....	52
5.1.1.	Ideal Performance	52
5.1.2.	Performance with Noise.....	56
5.1.3.	3D Reconstruction Analysis	58
5.1.4.	Computational Time	66
5.2.	Real Flight Imagery	68
5.2.1.	Datasets.....	68
5.2.2.	3D Reconstruction	68
5.2.3.	Failure Analysis.....	72
6.	Conclusion and Future Recommendations	75
6.1.	Local Dense Bundling.....	75

6.2. Future Works	76
6.2.1. Robust Rectification Between Frames	76
6.2.2. Optimized Code	78
6.2.3. Non-sequential Matching	78
6.2.4. Global Bundle Adjustment	79
6.2.5. Multi-view SGBM	79
6.2.6. 3D Hypothesis modeling	80
Bibliography	82

List of Figures

Figure 2-1. SIFT feature descriptor.	6
Figure 2-2. Sum of squared differences (SSD) algorithm.	8
Figure 3-1. (A) Stereo vision system mounted to RMAX. (B) RMAX in flight, carrying stereo vision system.	19
Figure 3-2. Pinhole camera geometry. The camera center is at C and p is the principal point.	20
Figure 3-3. Image screen (x_s, y_s) and camera pixel (x, y) coordinate systems.	21
Figure 3-4. The fundamental matrix F. The fundamental matrix describes how a point $\mathbf{x}$ in one image is transferred by the plane π to a matching point $\mathbf{x}_2$	23
Figure 3-5. Stereo rectification process.	24
Figure 3-6. Depth calculation from rectified stereo images.	25
Figure 3-7. (A) Diffuse and (B) specular surface reflection.	27
Figure 3-8. Effect of discontinuity on a planar patch assumption. Scene varies depending on viewpoint, which invalidates the planar assumption.	28
Figure 3-9. Effect of low texture on match correlation from left (A) camera to right (B,C). Unique points (B) have strong correlation in the SAD score with a distinctive match; while low texture (C) shows multiple matches with similar appearance.	29
Figure 3-10. Vertical accuracy across the image at a 40 m distance from stereo boom. Image is 1600x1200 with 8 mm focal length, 1.5 meter baseline and 1 pixel disparity change.	30
Figure 3-11. Accuracy at center pixel over distance (or altitude). Image is 1600x1200 with 8 mm (1980-pixel) focal length, 1.5 m baseline and 1-pixel disparity change.	31
Figure 3-12. Horizontal resolution for both x and y axes as a function of Z (altitude AGL).	31
Figure 4-1. Cost function for dissimilarity in SGBM.	39
Figure 4-2. SGBM uses multiple dynamic programming paths.	40
Figure 4-3. Operational framework of the local dense bundling algorithm.	42
Figure 4-4. Raw images (A) are first rectified by stereo calibration (B). SURF points are extracted (C), matched, and used for secondary rectification of the left images (D).	44
Figure 4-5. Blender 3D modeling and rendering software.	50
Figure 4-6. Example rendered image.	50
Figure 5-1. Side-view of ground truth blender model.	53
Figure 5-2. Average error over 5 frames (in meters) relative to distance (in meters) from left camera as produced by Raw SGBM and local dense bundling algorithms.	53
Figure 5-3. Average error across 5 frames with respect to distance from camera in the presence of 1 m 5-degree noise. Local bundling and SGBM were run at 60 m AGL with a virtual baseline of 4 m.	57
Figure 5-4. Ground truth 3D model with texture overlay.	59
Figure 5-5. 3D reconstruction using ideal pose with 8 m virtual baseline at 80 m AGL (A) and noisy pose with 8 m virtual baseline at 60 m AGL(B).	60
Figure 5-6. Close-up of peak (ground truth model shown) used for comparing 3D reconstruction.	61

Figure 5-7. Close up where terracing occurs over a peak in the raw SGBM point cloud from 60 m AGL imagery.	62
Figure 5-8. Close up where terracing over a peak is reduced by local dense bundling.	62
Figure 5-9. Terracing in SGBM cloud (A) is reduced by local dense bundling (B) with a virtual baseline of 8 m at 60 m AGL	63
Figure 5-10. 3D reconstructed point clouds from Gaussian corrupted imagery/pose. (A) Raw SGBM with no pose correction. (B) Pose corrected local dense bundling. All data at 60 m AGL	64
Figure 5-11. Angle of discretization for two dense bundled point clouds with different camera orientations.	65
Figure 5-12. 3D reconstructed model of the shed in April 14 dataset using local dense bundling (A) and raw, uncorrected SGBM (B).	70
Figure 5-13. Close-up of 3D reconstructed shed from April 14 flight. SGBM (A) and local dense bundling (B).	71
Figure 5-14. Close-up detail of increased resolution between raw SGBM result (A) and local dense bundling result (B).	72
Figure 5-15. Improper rectification occurs when original the previous (A) and current (B) left images undergo projective distortion (C),(D) due to incorrect fundamental matrix calculation. Distortion in this example results from low texture and poor overlap.	73

List of Tables

Table 3-1. RMAX Specifications	19
Table 5-1. Average error (in meters) for 3D reconstructed model constructed from ideal imagery across varying altitudes and virtual baselines.	54
Table 5-2. Factor of reduction in average error relative to basic SGBM for different virtual baselines at varying altitude.	54
Table 5-3. Loss of points and percent overlap for imagery at 60 m AGL and variable virtual baselines.	56
Table 5-4. Average offset errors in camera pose before and after correction by local bundling.	58
Table 5-5. Computational time for various functions in the local dense bundling algorithm.	67

List of Acronyms

AGL	Above ground level
CCD	Charge-coupled device
DoG	Difference of Gaussians
DSM	Digital surface model
GPS	Global positioning system
IMU	Inertial measurement unit
INS	Inertial navigation system
IRLS	Iteratively reweighted least squares
KLT	Kanade-Lucas-Tomasi Tracker
LM	Levenberg-Marquardt
LiDAR	Light detection and ranging
SBA	Sparse bundle adjustment
SIFT	Scale-invariant feature transform
SGBM	Semi-global block matching
SURF	Speeded-up robust features
SAD	Sum of absolute differences
SNCC	Sum of normalized correlation coefficient
SSD	Sum of squared differences
SAR	Synthetic aperture radar
UAV	Unmanned aerial vehicle
USL	Unmanned systems lab (at Virginia Tech)

List of Symbols

Sparse Bundle Adjustment	
J	Jacobian of SBA parameter vector
ϵ	SBA error matrix
p	SBA parameter vector
x	SBA measurement vector
$\hat{x}$	SBA vector of reprojected points
f_n	SBA functional relation
δ_p	Incremental step in linearized SBA parameter vector
Projective Geometry	
x_s	2D screen coordinate (physical space)
x	2D pixel coordinate
X	3D spatial coordinate (world coordinates)
P	Projection matrix
(f_x, f_y)	Focal length in x and y directions
K	Camera calibration matrix
(c_x, c_y)	Camera center location in pixels
R	Rotation between camera frame and world coordinate frame
T	Translation between camera frame and world coordinate frame
C	Camera center
l_1, l_2	Epipolar lines in images 1 and 2
H_π	Epipolar homography
F	Fundamental matrix
B	Baseline distance between cameras
D	Disparity map

d disparity

Local Dense Bundling Algorithm

f_i	The frame at time step i
N	Total number of frames
W	Local window of frames
I'_{iL}	Raw left image in frame i
I'_{iR}	Raw right image in frame i
I_{iL}	Rectified (within frame) left image in frame i
I_{iR}	Rectified (within frame) right image in frame i
H_1, H_2	Rectification homographies for left images in frame i and $i - 1$
$I_{L1(i)}$	Left image in frame i rectified to left image in frame $i - 1$
$I_{L2(i)}$	Left image in frame $i - 1$ rectified to left image in frame i
G_d	Dense list of points and camera pose
A_j	A dense point j containing a 3D coordinate and 2D coordinates for all cameras in the local frame in which A_j is visible
$\mathbf{X}_j$	3D coordinate for point j in world coordinates
$\mathbf{x}_{ijL}, \mathbf{x}_{ijR}$	2D observed location of point j in left and right rectified (within frame) images of frame i
D_{iLR}	Disparity map between left and right rectified images for frame i
D_{iLL}	Disparity map between left images of frame i and frame $i - 1$
Q	Fast 3D backprojection matrix
$\mathbf{X}_{jc}$	3D coordinate for point j in camera coordinates for left camera of current frame

C_{xR}	Camera center x -axis coordinate for right rectified image
RT_i	Combined rotation and translation matrix
D_{ind}	Valid index map
G'_d	Optimized dense list
A'_k	Optimized dense point
$\mathbf{x}'_{ikL}, \mathbf{x}'_{ikR}$	2D observed location of point k in left and right rectified (within frame) images of frame i
$\mathbf{x}_m$	Measurement vector for local dense bundling
$\hat{\mathbf{x}}_{ijL}$	2D reprojected coordinates of 3D point j in left camera of frame i
$\hat{\mathbf{x}}_{ijR}$	2D reprojected coordinates of 3D point j in right camera of frame i
$\Delta \mathbf{X}_{max,j}$	Maximum acceptable translation between original location and optimized location of point j
$T_{z,i-N}$	z component of 3D location of left camera in frame $i - N$
T_{i-N}	3D location of left camera in frame $i - N$

1. Introduction

This thesis presents a method of increasing both accuracy and resolution for visual reconstruction of 3D terrain models from low-altitude aerial stereo imagery. The work is part of a larger project at the Unmanned Systems Lab (USL) for robotic first response and analysis in the wake of a nuclear event. The full-scale project consists of an unmanned aerial vehicle (UAV) equipped with variety of interchangeable pods for radiation mapping, 3D imaging, and collection of ground samples using a tethered robot. The 3D terrain mapping serves as a tool for operators to analyze a post-nuclear disaster site. This 3D terrain map is generated through visual reconstruction from stereo imagery. Current software on the project can rapidly generate a local 3D terrain model; however, the model accuracy is limited by the altitude of the UAV and the vertical resolution of the stereo system onboard. This thesis presents a technique for improving 3D terrain mapping accuracy and resolution through local dense bundling of disparity maps. The algorithm also reduces the altitude limitations imposed by the stereo system, allowing for higher-altitude, higher-speed flights, which can efficiently cover more terrain, thus extending the range and capability of the USL robotic nuclear response system.

1.1. Purpose

Computer vision and 3D reconstruction is an art studied by many and perfected by none. The challenges are many, since it is a heavily under-constrained problem with few general solutions; however, in the last 40 years, the art has progressed much, buoyed by the ever-increasing computational power of computers and enhanced by incremental improvements of each subsequent age of researchers. The research within this thesis combines a variety of current methods to improve visual 3D terrain reconstruction. The research is driven by the goal of robotic first-response to nuclear disasters. In the event of nuclear disaster, a single unmanned aerial vehicle could scan the affected site and provide both aerial radiation data and a 3D terrain map of the area. Responders to the disaster could use this data to assess the situation, determine likely fallout locations, and guide rescue or nuclear response crews through the aftermath of the disaster. To accomplish this task, an unmanned aerial helicopter was equipped with a stereo camera system to fly

low-altitude sweeps over the terrain. Previously, an open-source computer vision solution was used to generate dense terrain data from the stereo imagery [1]; however, this approach had multiple limitations. First, the distance between stereo cameras limited the vertical resolution at each altitude. Second, the previous software could only generate local 3D information from a single pair of stereo images. This local 3D data was converted to world coordinates by the raw GPS and IMU data; however, the raw data had misalignment errors between stereo pairs of up to 3 meters, much of the data was redundant, and there were no checks for erroneous data. The purpose of this thesis is to provide an algorithm that overcomes these issues by increasing vertical resolution of 3D terrain data, decreasing misalignment errors between subsequent stereo point clouds, and reducing erroneous data.

1.2. Contribution

This thesis presents two contributions to the field of robotics and computer vision. Primarily, this thesis develops an improved method for generating dense terrain maps from stereo vision imagery during low altitude UAV flight. This method is called *local dense bundling* and integrates feature point detection and matching for image rectification, dense feature creation from the semi-global block matching algorithm and refinement of dense matches via local optimization. Normally, the depth, or distance, resolution of a given stereo vision system is limited by baseline, or distance from left camera to right camera; however, the local dense bundling algorithm optimizes data from multiple sets of stereo images to increase resolution. Given two or more stereo pairs of images with sufficient overlap, the resolution of the 3D data can be increased and average model error reduced by combining multiple pairs of stereo images taken at different times. If the stereo vision system is moving, such as during UAV flight, subsequent images from the same camera can be combined together to form a stereo pair. This time separated stereo pair can be used to link information from multiple stereo pairs together. If the UAV movement between stereo pair capture is greater than the fixed baseline of the stereo vision system, the local dense bundling algorithm can increase resolution and reduce error in the resulting 3D model. The dense algorithm also uses the increased information from multiple stereo images to detect matching mistakes and remove erroneous data.

The second contribution is a new technique/tool for evaluating mapping and 3D visual reconstruction algorithms on local or global scale using synthetic imagery. Blender [2], an open-source 3D modeling software tool, was used in conjunction with Python scripts to model a large-scale 3D environment and generate synthetic imagery. Since the camera pose and 3D terrain structure are precisely known in synthetic imagery, this data can be used to evaluate the accuracy and robustness of vision-based 3D reconstruction algorithms. Such data also enables comparison between different algorithms for reconstructed structure and camera pose.

1.3. Overview of Thesis

This thesis begins by examining previous work along the same research lines. The thesis reviews the basic building blocks used by the local dense bundling algorithm, such as sparse and dense feature matching, along with current applications of visual 3D reconstruction and comprehensive reconstruction frameworks. The thesis provides a theoretical overview of stereo vision, discusses the challenges faced in visual 3D terrain mapping, and establishes assumptions and constraints. The thesis describes previous algorithms for on project, presents the new local dense bundling algorithm, and describes the new synthetic imagery tool. The thesis evaluates the performance of the algorithm on both real and synthetic data. Finally, the thesis concludes with a summary of results and possible extensions for future work.

2. Literature Review

Three-dimensional reconstruction from imagery depends primarily on finding corresponding points between two or more images. For a given 3D point in world space $\mathbf{X}$, the 2D corresponding point in a given image is $\mathbf{x}$. If $\mathbf{x}$ can be located in multiple images, it can be used in conjunction with camera *pose*, or position and rotation, information to compute the original world point $\mathbf{X}$. The challenge is to match these points between images. In computer vision, two primary techniques accomplish matching of 2D camera points between images: One locates unique features in an image and then matches points between images based on the similarity of these features. This approach determines sparse, but fairly reliable point matches between images. The second applies a dense, pixel-to-pixel matching technique, usually based on the area surrounding each pixel along epipolar lines (explained in Section 3.2). Most 3D reconstruction algorithms commonly incorporate both sparse and dense matching techniques to create a global mapped environment. This chapter examines methods for sparse and dense matching, as well as current comprehensive vision-based 3D reconstruction algorithms.

2.1. Sparse Feature Matching

Over the last thirty years, researchers have developed a variety of approaches to detect and match sparse features. The three most widely used algorithms are the Kanade-Lucas-Tomasi tracker, SIFT, and SURF.

2.1.1. Kanade-Lucas-Tomasi

The Kanade-Lucas-Tomasi (KLT) tracker was originally developed for optical flow, or the tracking of points with arbitrary motion from one image to another. By assuming a linear approximation for small movement of pixels within a window, centered on some pixel p , Lucas and Kanade [3] develop an optical flow equation for the displacement vector that pixels within the window must satisfy:

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} d_x \\ d_y \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}, \quad (2-1)$$

where $[d_x, d_y]^T$ is the displacement vector, x and y are the axes of the camera coordinate system, and t is time, such that each subsequent image is an incremental step in time. The partial derivatives of the image I with respect to x , y , and t (summed in the window surrounding point p) are I_x , I_y , and I_t . Solving for the displacement vector gives

$$\begin{bmatrix} d_x \\ d_y \end{bmatrix} = [A^T A]^{-1} A^T b. \quad (2-2)$$

Using this definition, good features for tracking are found at pixels where $A^T A$ has values of sufficiently high magnitude to overcome image noise, represented as a predefined threshold. In practice, the solution to this displacement of a feature is found iteratively using a Newton-Raphson method to solve the linearization of image intensities for small motions. The basic Lucas-Kanade flow equation (2-2) operates on the assumption of small displacement of pixels between images. This holds true for small camera movements and changes within a scene; however, to operate on images with large displacements, the KLT tracker is applied in a pyramidal scheme. The image is down-sampled repeatedly to form a Gaussian pyramid and the KLT is applied at each layer from lowest resolution to highest resolution. Shi and Tomasi [4] also improve the tracker with an additional verification of feature quality over time. An affine transformation is fit between the original image of the feature and a later, non-sequential image. If the result is too dissimilar, the feature is dropped and no longer used for tracking.

2.1.2. SIFT

One of the most common feature detection algorithms is the scale-invariant feature transform (SIFT) developed by Lowe [5]. SIFT extracts features invariant to both scale and rotation by locating maxima and minima in the difference of Gaussians filter (DoG) across multiple scales. A 2x2 Hessian kernel is used at each point to distinguish between edge responses to the DoG, which are undesirable and single point responses. With a point located at some position (x, y) and scale in the image, the algorithm assigns the feature a primary orientation by generating an orientation histogram from the local gradients and selecting the dominant peak. All descriptor coordinates are referenced to this dominant orientation to provide rotation invariance. The descriptor itself is also constructed

from a histogram of gradients, where the gradients are summed into a 4x4 array of histograms with eight orientation bins each. This produces a 128-element descriptor for comparison against other features. A 2x2 example of this method is shown in Figure 2-1.

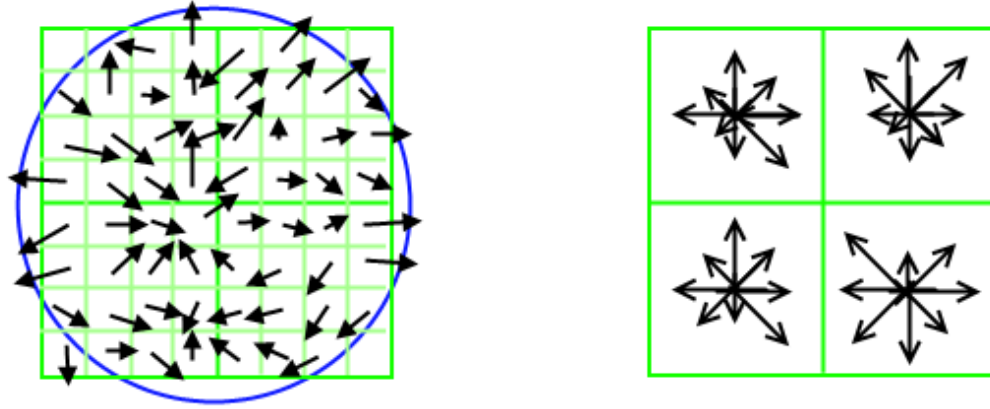


Figure 2-1. SIFT feature descriptor.

SIFT points from all images are usually stored in a fast nearest-neighbor searching tree, such as a k -dimensional (kD) tree, to provide quick access to similar features for matching between images.

2.1.3. SURF

Bay et al. [6] develops a fast approximation to SIFT, called Speeded Up Robust Features (SURF). SURF follows the same overall method of SIFT by locating maxima and minima using the Hessian matrix across multiple scales in the image and creating a feature descriptor from the relative orientations of the local image gradient. The SURF algorithm achieves much faster calculations than SIFT by generating an integral image [7] and approximating Gaussian partial derivatives with a box filter. For the descriptor generation, SURF determines the dominant angle of a feature with Haar wavelet responses, also calculated by box filters from the integral image. The descriptor is created by summing Haar wavelet responses d_x and d_y for a 4x4 set of sub-regions around the point into a vector $v = [\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|]$ for each sub-region, which results in a 64-element descriptor. The wavelet responses are invariant to bias in illumination, and normalizing the vector makes the descriptor invariant to contrast in the image. Bay states that SURF per-

forms nearly 3x faster than SIFT and has an equivalent feature matching rate between images of 82.6% compared to 78.1%.

2.2. Dense Feature Matching

Sparse feature matching can provide an excellent set of features, recognizable from almost any orientation and scale; however, these methods often utilize only a fraction of the data available in an image. Conversely, dense matching seeks to maximize 3D information by determining a pixel-to-pixel match for every point visible between images. To accomplish this dense matching, the powerful *epipolar* constraint is used. The epipolar constraint relates the points between any two images, such that a point x in an initial image will be found on line l in the second image. A more rigorous explanation follows in Chapter 3, but in brief, this constraint reduces a 2D search for correspondence to a 1D challenge. Nearly all dense matching methods rely heavily on the epipolar constraint. Many algorithms also make the assumption of *rectified stereo* correspondence. *Stereo* indicates two or more cameras are used, while *rectification* is the process of aligning images by their epipolar lines, such that equivalent epipolar lines run along the same y -coordinate (for horizontal stereo cameras). Scharstein [8] presents an excellent overview and evaluation of such dense two-frame stereo correspondence algorithms. Brown et al. [9] presents a similar “big picture” look at computational stereo algorithms, drawing distinctions among key ideas and approaches. This section presents existing dense matching methods based on local windows, dynamic programming, planar constraint techniques, and sequential stereo pair matching.

2.2.1. Window-based Matching

Given rectified left and right images, I_L and I_R , pixel p in the left image and matching pixel q in the right image are related by the disparity $d = D(x, y)$ where D is the disparity map such that $p = I_L(x, y)$ and $q = I_R(x - d, y)$. The goal of all dense two-frame stereo correspondence algorithms is to calculate this disparity map. A common algorithm for performing pixel-to-pixel correspondence is to minimize the sum of squared differences (SSD) cost function. The SSD cost function takes all pixel intensity values within a window surrounding pixel p in the first image and subtracts a similar window around a

potential candidate pixel q on the epipolar line in the second image, as shown in Figure 2-2. The resulting differences are squared and summed together to form a score, or cost, function. The pixel q with the lowest score is considered a match.

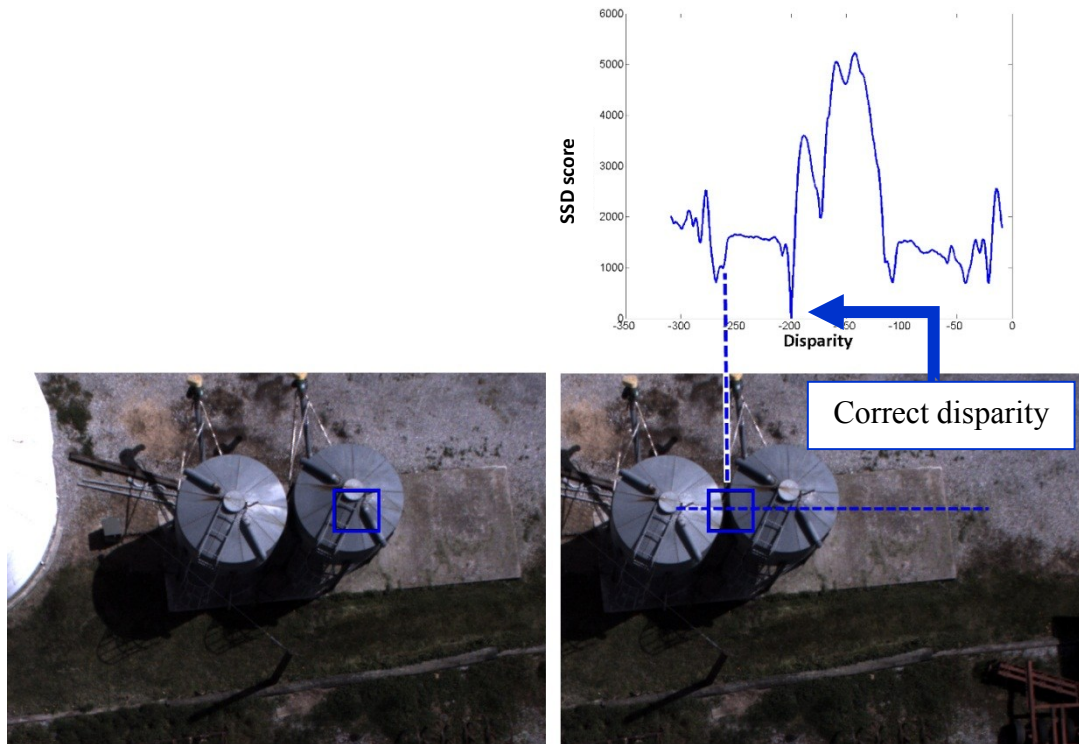


Figure 2-2. Sum of squared differences (SSD) algorithm.

A variant of the SSD, called the sum of absolute differences (SAD), is implemented in the open-source computer vision library, OpenCV. SAD requires less computation, and the OpenCV implementation includes checks to reduce errors resulting from low texture or repeated patterns [10].

Another windowing method is adaptive windowing. SangUn [11] uses a combination of directional windows and boundary detection in a hierarchal framework. SangUn first detects edge boundaries using a predicted disparity map and Sobel edge filters, followed by selection of the window size and shape to maximize matching information in textureless regions, while keeping the window from crossing boundaries. These results are used in a hierarchal framework, where the image is repeatedly down-sampled and lower-resolution results guide the disparity search and window adaptation at higher-resolution samples. Kanade [12] drives adaptive windowing by estimating disparity uncertainty as a statisti-

cal model. Kanade iteratively expands the window in the least-uncertain direction until some threshold is reached. Zhang [13] uses a similar approach to Kanade, but instead of guiding window size by uncertainty, Zhang uses the “quality” of a sum of normalized correlation coefficient (SNCC) cost function. Zhang calculates the quality of an SNCC function as the ratio of the first and third-strongest peaks in the SNCC function. The window around a point of interest is directionally expanded and a new SNCC cost function is computed. If the ratio increases, indicating a higher quality or stronger match, the expansion is accepted; otherwise all growth in that direction is halted.

Zabih [14] develops an area-based matching technique called “rank and census”, that relies on non-parametric local transforms. The rank transform is simply the total number of pixels in a window with a higher intensity than the center pixel p . The census transform unwraps the window surrounding p as a bit string, with each pixel as a 1 if the pixel has a higher intensity than p and 0 otherwise. These bit strings are compared using the Hamming distance to determine a match. By using a non-parametric transform instead of raw or even normalized pixel intensities, the “rank and census” method reduces the effects of edge discontinuities, while maintaining speed of operation.

2.2.2. Dynamic Programming Methods

Birchfield [15] presents a combination of dynamic programming and an image sampling insensitive dissimilarity measure to solve stereo correspondence. Similar to SSD, dynamic programming calculates optimal disparity d to minimize a cost function between pixel p in the first image and q in the second image. However, dynamic programming uses the previous best disparity value to optimally select the next disparity. Instead of using raw pixel intensity values as a cost function, Birchfield compares a small, linearly interpolated intensity function surrounding each pixel. This sub-pixel measure reduces the errors of discretization of camera pixels. Discretization results from imaging a continuous scene onto the charge couple device (CCD) of a camera, where the light intensity is sampled into discrete pixels. Sub-pixel sampling can restore some of the data lost due to this discretization.

Hirschmuller [16], uses dynamic programming from multiple directions to create semi-global matching, with “mutual information” as a base cost function. Hirschmuller first calculates the individual costs of each pixel p for a given range of disparities using “mutual information”, a probabilistic method where pixel cost decreases as the likelihood of correct prediction between the two images increases. This base cost is combined with a surface continuity constraint in the semi-global framework. The semi-global framework takes previous optimal matches along each direction and penalizes the base cost of pixel p if the new disparity differs from the previous directions. Hirschmuller [17] extends the method to work more effectively in structured environments using intensity-consistent disparity selection, to overcome low-texture areas, and discontinuity-preserving interpolation to fill holes cause by filtering. Haller et al. [18] use the census transform in a semi-global framework and implemented the algorithm in real-time with a current generation graphics processing unit.

2.2.3. Planar-constrained Matching

Baillard and Gallup both construct a 3D model by creating piecewise planar models. Baillard [19] locates 3D lines by edge detection, creates plane hypotheses, tests these hypotheses by matching texture on the planes, and finally merges similar planes together. Gallup [20] uses a more complex scheme, designed for extension to multi-view (two or more frame) stereo. The algorithm segments the scene into planar and non-planar regions, using an initial RANSAC plane labeling method to create hypotheses. It then links these hypotheses across multiple images and classifies and refines them using energy minimization by a graph-cut labeling process.

2.2.4. Sequential Stereo Pair Matching

Miled et al. [21] perform matching between multiple pairs of stereo images by jointly calculating both motion fields—motion of each pixel between two temporally separated images—and disparity maps. They use a quadratic minimization of a cost function which is based on total variation [22]—a measure of pixel intensity oscillation through the image—and subject to convex motion constraints. Gong and Yang [23] use sequential stereo images to calculate disparity flow, which encapsulates 3D structure and motion

information as a combination of disparity and optical flow. Gong and Yang incorporate a trinocular stereo system with orthogonal reliability-based dynamic programming to match between images and determine disparity flow. Koch [24] reconstructs 3D surfaces from stereo image pairs by creating disparity maps for each pair through dynamic programming, approximating these discrete maps as 3D parametric models for each pair of images, and meshing each dense model together through scene segmentation and depth interpolation.

2.3. Comprehensive 3D Reconstruction Algorithms

Sparse feature point methods and dense correspondence methods are usually combined in a global framework to maximize their strengths while mitigating weaknesses. These frameworks must handle numerous camera positions and point matches, while also providing validation and robustness of resulting 3D structure. Global camera pose is usually provided by 3D position/orientation measurement systems such as GPS and IMUs (inertial measurement units), optimization of 3D projective geometry, or some combination of the two. Accurate agglomeration of dense matches into 3D structure uses a variety of techniques including local and global optimization, planar estimation, segmentation, classification and filtering. This section examines the optimization technique of bundle adjustment, current applications for 3D reconstruction, various comprehensive algorithm frameworks, and evaluates current visual reconstruction solutions for use in the Unmanned Systems Lab (USL) project.

2.3.1. Bundle adjustment

Bundle adjustment (BA) is the problem of simultaneously optimizing both 3D point location and camera viewing parameters (camera pose and or calibration) for visual reconstruction. Bundle adjustment is so named because it describes the rays of light emanating from each 3D point which then converge on a camera optical center as a “bundle” and all the “bundles” for a set of cameras are “adjusted” optimally for point position camera pose and camera calibration parameters. Bundle adjustment can prove to be very complicated due to the nonlinear nature of projective geometry, parameterization of rotations, and camera intrinsics. Usually, bundle adjustment is used as a final step to

determine optimal camera poses and 3D point locations from a set of sparse feature points. Dense 3D structure can be computed afterwards using the optimized camera locations. Triggs et al. [25] give a broad overview of BA including choice of cost function and robustness, as well as numerical methods of optimization. Dickscheid [26] provides a way to benchmark different BA algorithms based on consistency of form deviation and precision as related to a reference set of data.

Lourakis and Argyros [27] developed an excellent C++ package for performing sparse bundle adjustment (SBA), taking advantage of the sparse nature of the Jacobian matrix in BA to speed up calculations. This sparseness arises from the lack of interaction between numerous points and cameras, since not all points are visible in all cameras. Sparse bundle adjustment functions by minimizing the reprojection error between observed and predicted points, represented as the sum of squares of a large number of nonlinear real-valued functions. The minimization is achieved through the Levenberg-Marquardt (LM) algorithm, which acts like a combination of steepest decent and the Gauss-Newton method. Let f be a functional relation that correlates a parameter vector $p \in \mathcal{R}^m$, containing the point positions and camera pose, to a measurement vector $\hat{x} = f(p)$, $\hat{x} \in \mathcal{R}^n$, where x is the observed 2D image coordinates of the points and $\hat{x}$ is the computed 2D point locations of the same 3D features. It is desired to find the vector p' that locally minimizes the squared distance $\epsilon^T \epsilon$, where $\epsilon = x - \hat{x}$. For a small change $\|\delta_p\|$, f is approximated in the neighborhood of p by

$$f(p + \delta_p) \approx f(p) + J\delta_p, \quad (2-3)$$

where J is the Jacobian of f . The Levenberg-Marquardt algorithm is an iterative process, which steps the solution by δ_p to converge to the local minimizer p' . Consequently, δ_p is the solution to a least-squares problem, found when $J\delta_p - \epsilon$ is orthogonal to the column space of J . One can then construct

$$J^T J \delta_p = J^T \epsilon, \quad (2-4)$$

where $J^T J$ is the first order approximation to the Hessian of $\frac{1}{2} \epsilon^T \epsilon$ and δ_p is the Gauss-Newton step. The Levenberg-Marquardt algorithm uses a slight variation of this by introducing a damping term μ in the form

$$(J^T J + \mu I) \delta_p = J^T \epsilon, \quad (2-5)$$

where I is the identity matrix. The damping term controls the size of the Gauss-Newton step adaptively, enabling LM to alternate between a steepest decent method when far from the minimum and the Gauss-Newton method for fast convergence near a minimum.

2.3.2. Applications

The design of most visual reconstruction software is driven by the intended use of the resulting 3D model. Three-dimensional visual reconstruction is used in a slew of fields including remote sensing, building construction documentation, heritage reconstruction, robotic perception and path planning. On the architectural front, El-Hakim [28] and Remondino [29] examine the use of visual reconstruction for detailed documentation of historic artifacts and sites. Gloparvar-Fard [30] develops a 3D reconstruction technique from arbitrary photographs to automate progress monitoring of a construction site over time. Snavely [31] shows the potential for photo tourism of famous locations and landmarks, using models built by unstructured photo collections, such as online image searches.

Historically, much of 3D visual reconstruction originates from the fields of Geographic Information Sciences (GIS) and remote sensing, where even before computers, photogrammetry was used to manually build elevation maps from aerial photos. Geographic Information Sciences and remote sensing continue to drive much computer vision research, seeking to automate and improve 3D mapping from aerial and satellite imagery. Zhang [13] uses a comprehensive visual reconstruction algorithm to develop 3D models from linear-array satellite imagery, available in the commercial product VirtuoZo. Li et al. [32] use high resolution satellite imagery to perform topographic mapping of the Martian surface. The researchers at Urban Reconstruction [33] use an integrated multi-camera vehicle system to map building exteriors in an urban environment. Mills et al. [34] com-

pare aerial photography stereo vision against light detection and ranging (LiDAR) for use in vegetation management.

A more recent but burgeoning use for 3D visual reconstruction is robotics. Moghadam et al. [35] use both stereo vision and LiDAR to create probabilistic maps for 3D path planning through an environment. Stephanik et al. [1] and Eisenbeiss [36] explore the use of unmanned aerial vehicles for performing visual terrain mapping. Stephanik et al. examines the potential for rapid 3D terrain mapping with stereo vision, while Eisenbeiss considers applications in archaeology and agriculture. Rusu et al. [37] show the full gamut of robotic 3D vision by using a mobile robot, equipped with stereo camera and IMU, to perform visual odometry, map the local environment, classify the environment (wall, stairs, floor, etc.), and perform path planning within the 3D map, all in real-time.

2.3.3. Algorithm Frameworks

There are numerous paths to solving the 3D visual reconstruction problem and many different global algorithms have been developed by mixing the basic building of computer vision within a larger framework. The most common framework consists of two stages. A sparse stage first detects and matches feature points between images, estimates camera pose and 3D feature point location, then finally optimizes the results. This sparse procedure is usually followed with a dense stage, which uses the optimized camera positions and sparse data for determining dense structure of the scene.

A prime example of this framework is Bundler, an open-source spin-off from the photo tourism project by Snavely, in conjunction with patch multi-view stereo (PMVS) [38], a dense reconstruction method. Bundler first detects and matches SIFT points between all images, then estimates and optimizes camera pose and sparse structures. These results are fed into PMVS, which performs dense feature detection with Harris and DoG filters, followed by expansion of neighboring features. The features are matched by a photometric consistency constraint using normalized cross correlation and a visibility constraint. Finally, the results are filtered for outliers and a dense polygon surface is reconstructed.

The Urban Reconstruction project also uses such a framework, by detecting features with a hierarchical KLT tracker, estimating camera pose, and generating dense results. Camera pose is estimated either by 2D to 3D world point correspondences or an integrated inertial navigation system (INS) and GPS system. The entire algorithm works in real-time on a GPU, so all imagery is in sequential order. Dense reconstruction is accomplished by multi-view stereo, first using a plane-sweep algorithm for each frame, then fusing the resulting disparity maps based on a confidence and stability measure for each pixel.

Li et al. [39] skip the sparse step by using solely dense DAISY[40] feature points to calculate dense depth maps. Li et al. rectify images and apply dynamic programming along the epipolar lines with dense DAISY feature points to create an initial set of dense depth maps. The dense features are collected into tracks, or sets of dense features in different images associated with a single 3D point. These tracks are optimized to remove erroneous matches and final 3D structure is optimized by a bound constrained algorithm.

Another popular global framework begins with a stereo assumption, where each frame consists of two images taken simultaneously from a pair of stereo cameras. The overall framework still resembles the sparse/dense two-step method; however, a stereo camera constraint provides extra information, allowing for different approaches to 3D reconstruction. Jung and Lacroix [41] use a wide-baseline stereo system on board a blimp for low-altitude aerial mapping. Their algorithm first generates a dense disparity map for each stereo frame by a correlation-based pixel matching algorithm. Vehicle and camera motion is determined by detecting and matching Harris features between sequential frames in 2D and tracking landmark feature points with an extended-Kalman filter. The algorithm aligns the dense disparity maps using the landmark feature points in a simultaneous localization and mapping (SLAM) framework.

Rusu et al. [37] also use a stereo assumption to generate depth data, in this case as point cloud data. The system operates in real time, so for each stereo frame the algorithm computes point cloud data and estimates motion with visual odometry and IMU data. The current point cloud data is decomposed into a dynamic octree and converted into a local

polygonal model. The local model updates and refines the global model, similar to a SLAM framework.

Finally, Zhang develops a comprehensive multi-stage framework for generating digital surface models (DSMs) from linear array imagery. Zhang uses multiple primitive matching from feature points, edges, and area-based techniques to generate a dense set of matches. Matching is iteratively conducted using geometrically constrained cross-correlation in a hierarchical structure, so each previous set of matches defines a DSM, which is used to guide the next matching procedure. Zhang then applies relational matching with probabilistic relaxation labeling to globally align points and remove mismatches. Finally, multiphoto geometrically constrained matching is used to achieve sub-pixel accuracy.

2.3.4. Evaluation of current 3D vision solutions

Despite the advanced research found in 3D visual reconstruction, no current commercial off-the-shelf product exists that can meet the USL nuclear response project's constraints and goals. The overall goal is to generate 3D terrain maps of a post-disaster area using low-altitude stereo aerial imagery from a UAV, with potential implementation in real time. Most commercial systems for aerial and satellite mapping require user interactivity, preventing full autonomy of the system. These systems, like SOCET SET [42], VirtuoZo, and SAT-PP are designed for processing large batches of imagery with a trained human operator. They are also designed exclusively for post-processing and no real-time implementations exist. Though Urban Reconstruction has demonstrated a real-time mapping system, no commercial version has been released. Open-source solutions, like Bundler and Clustering Views for Multi-view Stereo show promise in producing 3D dense structure from arbitrary collections of photos; however, they are still designed for post-processing, which prevents real-time consideration, and they require a large number of pictures with high overlap for accurate results. Also, current open-source methods do not incorporate any form of GPS or IMU data, which is available on the USL system. For this reason, the USL has developed a UAV stereo vision system along with the local dense bundling algorithm to produce dense 3D terrain data for an emergency nuclear situation.

3. Background

The work presented in this thesis was driven by the overarching nuclear response project at the Unmanned Systems Lab at Virginia Tech. This chapter first examines the specifics of the mission including goals, selection of stereo vision for 3D mapping and mission hardware. The mission details are followed by an overview of the theory behind vision-based 3D reconstruction and the assumptions and constraints used in the project

3.1. Project Mission

3.1.1. Description of mission

The Unmanned Systems Lab (USL) at Virginia Tech is designing and fabricating an unmanned aerial vehicle (UAV) system to assist analysis in the wake of a nuclear disaster. This includes response to intentional events, such as dirty bombs, improvised nuclear devices or full-yield nuclear weapons, as well as unintentional events, such as the emergency situations in Chernobyl, Three-Mile Island and even the more recent disaster at the Fukushima Daiichi nuclear plant in Japan. Since such a post-disaster environment could be highly radioactive, a first-response unmanned vehicle provides a method of keeping humans safe during investigation. The UAV system will perform a variety of tasks, including mapping the environment in 3D, mapping aerial radiation counts, and collecting radioactive samples by means of a deployable ground robot, tethered to the UAV.

The 3D mapping is essential for several reasons. First, post-disaster terrain could be substantially altered due to the nuclear event. Mapping the new terrain and recording the environment in 3D would allow an operator to analyze the scene from afar. The operator could determine likely locations of radioactive material, which could be collected for analysis either remotely by robot, or by sending in human teams. The 3D map could also provide ingress and egress paths for such a human team, reducing the time spent in a radioactive area. Similarly, the 3D terrain map would allow the tethered ground robot to perform 3D path planning during sample collection. Thus the 3D terrain map provides several highly useful abilities to an operator.

Several different methods exist for attaining a 3D terrain map via aerial scanning, given the assumptions of a small UAV and low-altitude flight. The foremost system is light detection and ranging (LiDAR), followed by monocular (single camera) vision, stereo vision, and synthetic aperture radar (SAR). LiDAR is generally the most accurate, with precision and standard deviation of less than 5-cm vertically; however, such systems are usually heavy, expensive, and power-consuming. SAR is still in development for lightweight UAVs, and while 3D terrain mapping methods exist for larger aircraft, current lightweight commercial SAR systems can only perform 3D mapping by multiple flights. Due to weight, power, and cost considerations, LiDAR was ruled out, and SAR was not selected due to the need for multiple passes. Vision-based methods, like monocular and stereo vision are lighter weight and much less expensive, with similar resolution to LiDAR at low altitudes. Vision systems depend much more heavily on software to generate precision and accuracy. Stereo vision was selected for this project in place of monocular vision to bootstrap the 3D mapping process and provide the potential for real-time 3D mapping. For a more detailed analysis and comparison of these technologies see Stefanik [1].

3.1.2. Description of equipment

A Yamaha RMAX autonomous helicopter was selected for the mission to carry all systems, providing 28 kg of payload capacity for a total 94 kg helicopter system. Helicopter specifications are shown in Table 3-1 and Figure 3-1 shows the RMAX helicopter with stereo system installed.

Table 3-1. RMAX Specifications

Specification	Item	Value
Performance	Maximum payload	28 kg
	Gross weight	94 kg
	Flight duration	60 minutes
	Range (visual observation)	10 km
Dimensions	Main rotor diameter	3.1 m
	Tail rotor diameter	0.55 m
	Overall length (including rotors)	3.6 m
	Overall height	1.1 m
Engine	Water-cooled, 2 stroke, horizontally opposed 2-cylinder	
	Displacement	246 cc
	Power	21 HP
	Fuel	Gasoline/oil mixed 50:1



(a)



(b)

Figure 3-1. (A) Stereo vision system mounted to RMAX. (B) RMAX in flight, carrying stereo vision system.

The stereo vision hardware consists of three cameras, an IMU/GPS receiver, and a carbon fiber/epoxy tube shown in Figure 3-1. Two of the cameras are used in stereo correlation, both of which are grayscale Sony XCD-U100 FireWire cameras. A third color camera, XCD-U100CR, is used for color overlay. This approach was chosen because the Bayer pattern of color cameras reduce their effective resolution. Each camera has a resolution of 1600x1200 pixels and a wide-angle lens with a focal length of 8 mm. A Microstrain

3DM-GX3-35 with GPS antenna provides both IMU and GPS data. The cameras are connected to an embedded PC/104 computer with an Atom processor for synchronization and data collection. For rigidity, the three cameras and IMU are mounted within the carbon fiber tube.

The carbon fiber-epoxy tube is attached to the helicopter landing gear using a vibration damping mount. The mount suspends the stereo system between two square steel brackets using a combination of elastic tubing and foam to reduce vibrations transferred from the helicopter body to the more sensitive cameras. Collectively, the stereo cameras, IMU/GPS, and carbon fiber/epoxy tube assembly is called the *stereo boom* and weighs 3.1 kg.

3.2. Overview of Vision-Based 3D Reconstruction

This section provides a basic overview of the fundamentals necessary for understanding the local dense bundling algorithm. The four primary concepts are projective geometry, the epipolar constraint, the rectification process, and stereo vision.

3.2.1. Projective Geometry

Vision-based 3D reconstruction relies heavily on the mathematics of projective geometry. A camera can be represented using the theoretical pin-hole model, shown in Figure 3-2.

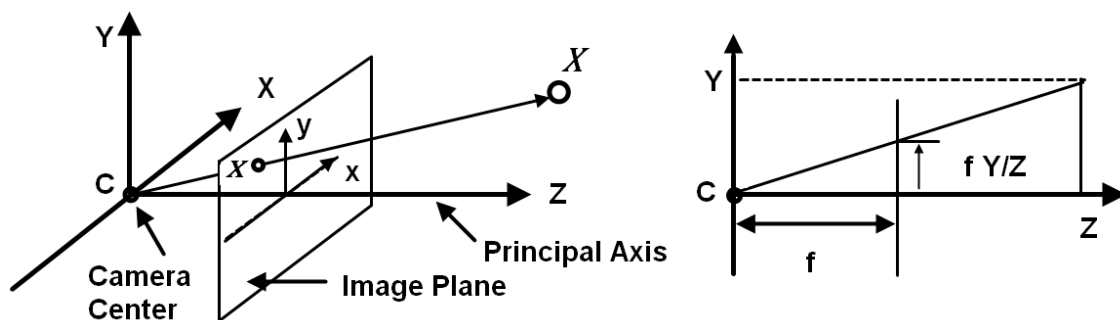


Figure 3-2. Pinhole camera geometry. The camera center is at C and p is the principal point.

In this model, the lens of the camera is represented as a “pinhole” through which light rays pass. These rays are projected onto the camera’s CCD array, which is represented as the image plane. Most projective math is represented using homogenous coordinates,

which allow for simultaneous translation and rotation. A homogenous 3D coordinate, $\mathbf{X} = [X \ Y \ Z \ 1]^T$ can be projected onto the image plane as $\mathbf{x}_s = [x_s \ y_s \ 1]^T$, a homogeneous screen coordinate on the image plane, using (3-1) and (3-2),

$$\mathbf{x}_s = P\mathbf{X} \quad (3-1)$$

$$w \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (3-2)$$

where P is called the camera projection matrix, f is the distance from the camera center to the screen, called focal length, and w is a scaling factor. Currently, (x_s, y_s) represent the physical screen coordinates (in meters) of the projected point on the charge-coupled device (CCD) array of the camera. The discretization from physical coordinates on the CCD array to image pixel locations can be represented as shown in Figure 3-3.

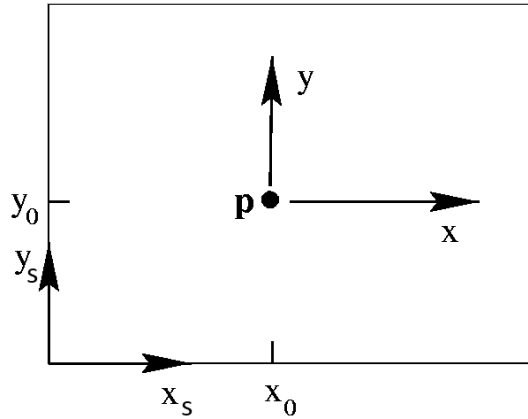


Figure 3-3. Image screen (x_s, y_s) and camera pixel (x, y) coordinate systems.

This coordinate conversion can be accomplished by modifying the camera projection matrix to

$$P = K [I \ 0] \quad (3-3)$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \alpha & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} [I \ 0] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \quad (3-4)$$

where (x, y) refer to pixel coordinates, K is the camera calibration matrix, composed of the camera center point (c_x, c_y) in pixels, where the center axis of the camera intersects the image plane, the focal length and scale factor in pixels (f_x, f_y) , and α , a parameter which accounts for skew but is zero in almost all modern cameras. The terms (f_x, f_y) ac-

count for focal length and pixels on the sensor array that are not exactly square. This definition of P works for 3D points referenced to the camera frame; however, for world coordinate points, P is expanded to the final definition of

$$P = K [R \quad T] \quad (3-5)$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = K [R \quad T] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \quad (3-6)$$

where R is a 3x3 orthogonal rotation matrix and T is a 3x1 translation vector. Here the camera calibration matrix holds *intrinsic* parameters, while rotation and translation together hold *extrinsic* parameters, also referred to as intrinsics and extrinsics.

In this thesis, *projection* means converting from 3D coordinates to 2D image coordinates, and *back-projection* converts from image coordinates to 3D coordinates. Unfortunately, given a known 2D camera point $\mathbf{x}$, moving backwards from image to 3D coordinates requires more information than is available in the raw image coordinates x and y , because scale w is unknown. To obtain scale, some other piece of information must be acquired. This is most commonly accomplished by locating the same point 3D point in a second image or camera. Also, for the remainder of the thesis, *image* and *camera* will be used interchangeably, though preference will be given for *image* in the case when only 2D pixel information is under discussion and *camera* when pose information is also involved. Another key term is *reprojection*, which is the process of projecting a previously back-projected 3D point onto the original camera. The difference between the original measured location coordinates $\mathbf{x}$ and the reprojected coordinates $\hat{\mathbf{x}}$ is called reprojection error, which is used as a measure of 3D reconstruction accuracy.

3.2.2. The Epipolar Constraint

Given a 3D point $\mathbf{X}$ viewed from two different camera locations, the corresponding 2D image points $\mathbf{x}_1$ and $\mathbf{x}_2$ are related by the respective camera projection matrices such that $\mathbf{x}_1 = P_1 \mathbf{X}$ and $\mathbf{x}_2 = P_2 \mathbf{X}$. There exists a geometric constraint between $\mathbf{x}_1$ and $\mathbf{x}_2$ called the epipolar constraint, which is shown in Figure 3-4 and described in (3-8).

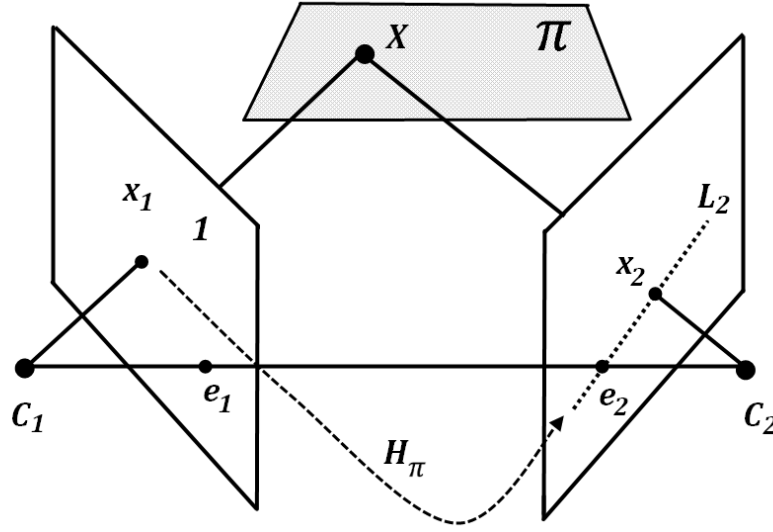


Figure 3-4. The fundamental matrix F . The fundamental matrix describes how a point $\mathbf{x}$ in one image is transferred by the plane π to a matching point $\mathbf{x}_2$.

As shown in Figure 3-4, the camera centers C_1 , C_2 and the 3D point X determine a plane π , called the epipolar plane, upon which both $\mathbf{x}_1$ and $\mathbf{x}_2$ must lie. Each camera center can be projected onto the opposing camera's screen as *epipoles* e_1 and e_2 and the line formed by the intersection of plane π with the camera screen forms the epipolar line l_2 , for camera 2. The plane forms a 2D homography mapping H_π between points $\mathbf{x}_1$ and $\mathbf{x}_2$. This relationship between points can be represented as a 3×3 matrix of rank 2 called the fundamental matrix.

$$l_2 = [e_2] \times H_\pi \mathbf{x}_1 = F \mathbf{x}_1 \quad (3-7)$$

$$\mathbf{x}_2^T F \mathbf{x}_1 = 0. \quad (3-8)$$

This epipolar constraint can be used to greatly reduce the search space when trying to find corresponding points between two images without prior knowledge of the imaged 3D point.

3.2.3. Rectification

Most stereo correspondence algorithms rely on the assumption that images are both *rectified* and *undistorted*. Rectification transforms a pair of stereo images to align their epipolar lines parallel to one another along the same y -axis value (for horizontal stereo).

Undistortion removes the nonlinear effect of radial lens distortion and CCD skew. The procedure is visualized in Figure 3-5.

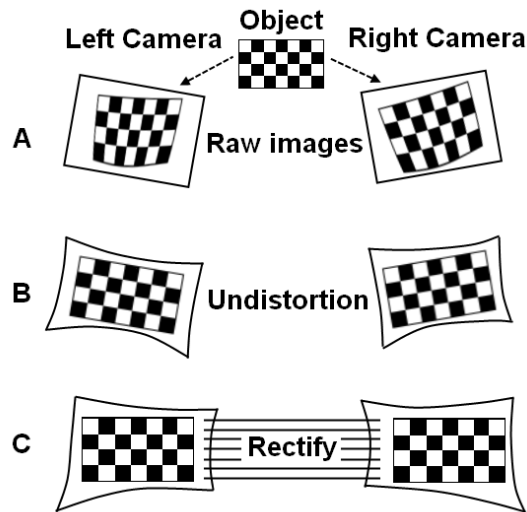


Figure 3-5. Stereo rectification process.

The main advantage of rectification is that after image correction, searching for correspondence between pixels requires only a 1D search along the epipolar line, which now is the row in the opposite image at the same y value. Rectification can be accomplished in several ways, but this thesis uses the two techniques available in the open-source computer vision library, OpenCV [10]. The first technique requires a fixed set of cameras, calibrated *a priori* to determine camera intrinsics (focal length and camera center) and extrinsics (rotation and translation relative to one another). The second requires only the fundamental matrix between two images, which can be calibrated for any arbitrary pair of cameras using matched image features. For the remainder of this thesis, the left and right images, captured simultaneously from the stereo boom, along with camera pose (as observed by GPS and an inertial measurement unit) will be referred to as a *frame*. The first rectification method works *within* a frame, to rectify left and right images, while the second method works *between* two different frames, to rectify the first left image to the second left image.

3.2.4. Stereo Vision

Assuming that the two images have been rectified and undistorted, they can be represented as shown in Figure 3-6.

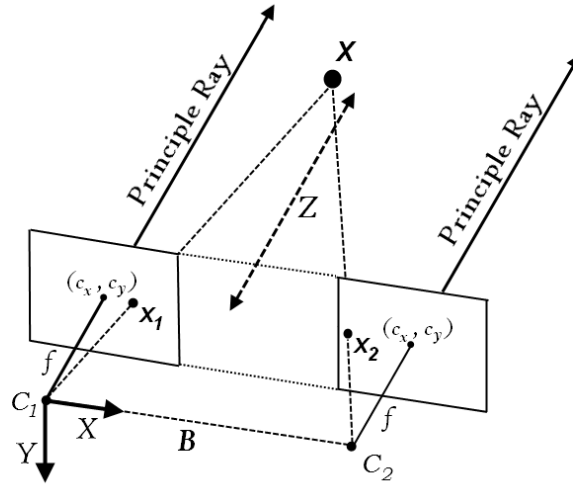


Figure 3-6. Depth calculation from rectified stereo images.

In this configuration, the distance Z from the 3D point X to the stereo system can be found by similar triangles as evidenced in

$$Z = \frac{fB}{x_1 - x_2} = \frac{fB}{d}, \quad (3-9)$$

where B is the *baseline*, or distance between the cameras, f is the focal length (in pixels), and x_1 and x_2 are the x coordinates of the point X in the first and second images, respectively. This equation shows that disparity is inversely proportional to the distance, where disparity is defined as the difference in pixel coordinates $d = x_1 - x_2$. For calibrated stereo systems the baseline is constant and usually known *a priori*; however, stereo vision can also operate on monocular imagery from a moving camera, which results in a variable baseline as defined by camera motion. To delineate, *baseline* refers to the fixed distance between left and right cameras on the stereo boom, while *virtual baseline* refers to any variable baseline caused by camera or vehicle movement.

3.3. Theoretical Challenges

Vision-based reconstruction is an ill-posed problem [43], which is the main reason that no fully automatic, precise, and reliable general solution exists. Reconstruction depends primarily on relating variations in pixel intensity of the 2D images (which is proportional to the light received by the CCD array) to geometry in the physical world. In an ideal case, these pixel intensity patterns could be directly related to world geometry and solved by projective geometry. However, pixel intensity changes actually result from several different real-world phenomena, not only geometry but also surface material properties and illumination. The challenge lies in distinguishing changes in geometry from other changes. A basic visual reconstruction model for photogrammetry assumes the following:

- The overall scene is rigid and static.
- The terrain surface is piecewise smooth (or piecewise planar) and opaque.
- Constant illumination of the scene.
- All surfaces are diffuse (non-reflective or matte).
- All surfaces have *texture* or visible variations and patterns of emitted light intensity.

These assumptions attempt to convert the ill-posed vision-based reconstruction problem into a well-posed problem. However, these assumptions are broken at different times and different ways, depending on the scene being viewed and the algorithm used for reconstruction. A successful algorithm is founded on appropriate assumptions, sufficient constraints, and proper handling when assumptions are violated. This section explores possible violations resulting from radiometry, the physical environment, and challenges rising from hardware limitations.

3.3.1. Radiometry

Light-related or radiometric challenges include radiosity changes, shadows, specular reflection, and translucent materials. Changes in radiosity occur when the illumination of a scene changes between images, either from moving light sources or varying light intensities. In an outdoor environment, radiosity can change due to clouds moving across the sun, or dynamic local lighting such as lamps and spotlights. Shadows are a subset of radiosity changes and cause difficulty both by reducing contrast within shadowed regions

and producing non-static scene texture if the light source is moving. *Specular* or mirror-like reflections occur when light is reflected away from a surface in a single direction, as differentiated from *diffuse* reflections where reflected light is scattered in all directions. Specular reflections cause objects to appear differently from different directions, especially in the case of specular highlights, such as those in Figure 3-7B.

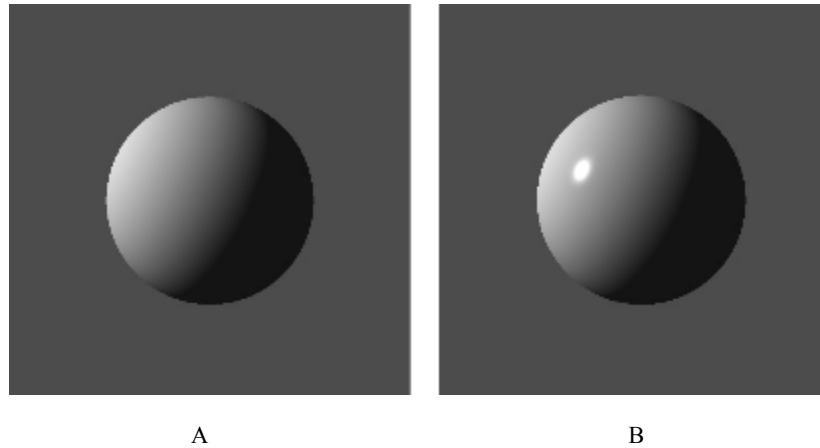


Figure 3-7. (A) Diffuse and (B) specular surface reflection.

A diffuse surface, also called a Lambertian surface, is desirable for vision-based reconstruction because it removes the ambiguity that results from a view dependent surface reflectivity function. Together, these radiometric issues pose a significant challenge to any visual reconstruction algorithm.

3.3.2. Physical Environment

The physical environment challenges for vision include geometric issues, like occlusion, dynamic environments, and scene discontinuity, along with surface properties like low/repetitive textures and translucent materials. Occlusion occurs when a 3D point is visible in one camera but blocked from view in the other camera by some object or surface. Obstruction often causes mismatches, where two independent but visually similar 3D points are matched as the same point between two images. Another geometric challenge is a non-static or dynamic environment, which breaks the rigid scene assumption. A dynamic environment means the shape and structure of a scene is dependent on time. In an outdoor scene changes can occur with trees and grass blowing in the wind or moving

cars. Discontinuity is a break in the piecewise smooth assumption. A true physical break in continuity is very rare, for example a blimp, completely unattached to the environment; however, image discontinuities can easily occur at steep terrain variation below camera resolution, or due to occlusion. Discontinuities can cause problems in local point-based feature descriptors, which usually assume a flat planar patch surrounds the point in interest, as well as area-based dense algorithms which often assume the scene is piecewise planar. Such issues are visualized in Figure 3-8.

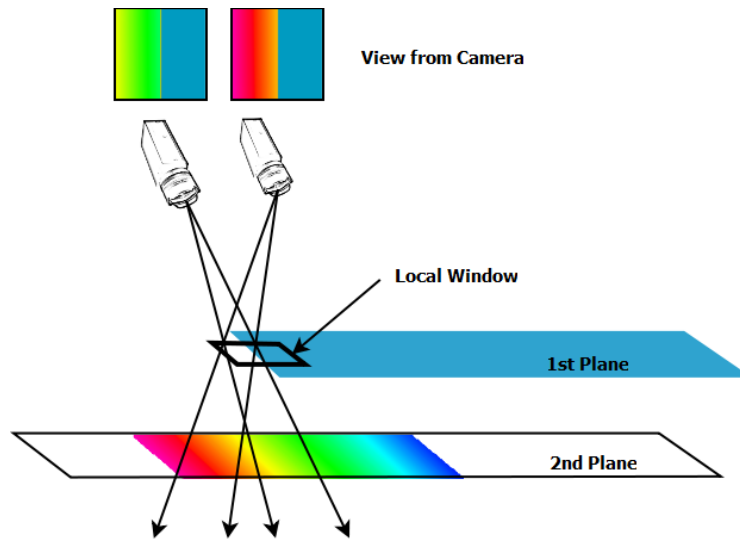


Figure 3-8. Effect of discontinuity on a planar patch assumption. Scene varies depending on viewpoint, which invalidates the planar assumption.

The challenges resulting from material surfaces are repetitive texture or structure, textureless regions, and translucency. Most vision reconstruction algorithms, either feature or area-based, use a similarity measure to match image points. This implicitly assumes for some target point x_1 in image 1, that the correct match x_2 in image 2 will look “most similar” and be uniquely identifiable, such as Figure 3-9A. However if the texture is repetitive or textureless, multiple potential matches will occur, as shown in Figure 3-9B.

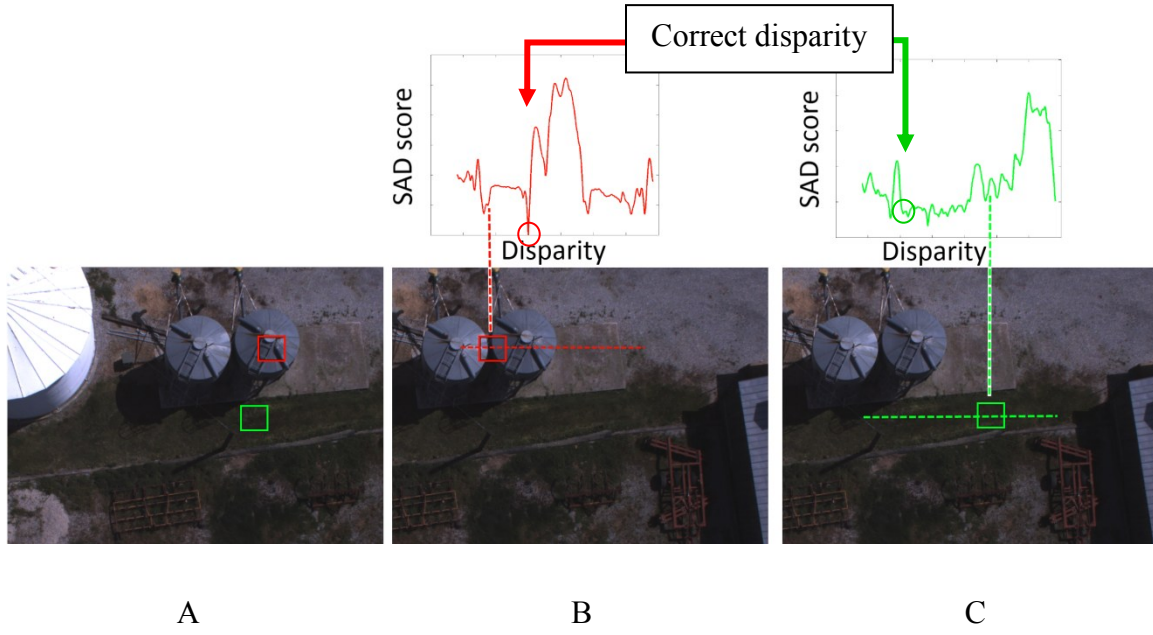


Figure 3-9. Effect of low texture on match correlation from left (A) camera to right (B,C). Unique points (B) have strong correlation in the SAD score with a distinctive match; while low texture (C) shows multiple matches with similar appearance.

Multiple matches can also result from translucent surfaces, which have viewpoint dependent appearance, similar to specular materials. Multiple potential matches must be dealt with using some other technique, such as another similarity measure, a different constraint, or a global optimization step.

3.3.3. Hardware Limitations

Hardware also causes challenges due to limited stereo camera resolution, poor lens quality, non-simultaneous stereo image capture, motion blur, GPS/IMU errors and percent overlap. Stereo camera resolution determines the spatial accuracy of the terrain map and results from camera resolution, the baseline between cameras, and distance to the imaged object or surface. According to the pinhole model in Figure 3-2, the depth, or distance, accuracy for the estimation of a 3D point relative to the camera $\mathbf{X}_c = [X, Y, Z]$ is given by

$$\sqrt{\Delta X^2 + \Delta Y^2 + \Delta Z^2} = \frac{Z \Delta d}{Bf} \sqrt{X^2 + Y^2 + Z^2}, \quad (3-10)$$

where B is the baseline between cameras, f is the focal length in pixels, and Δd is the change in disparity in pixels. From this equation it is evident that the accuracy is linearly related to the Δd and quadratically related to the distance from an imaged object. The po-

sition in the image also affects accuracy, though it can be shown that the center of the image ($X = Y = 0$), which corresponds to the camera center point ($x_1 = c_x, y_1 = c_y$), is the most accurate location. For the stereo boom parameters in this project and at a height of 60 meters above ground level (AGL), the vertical accuracy varies horizontally across the image according to Figure 3-10.

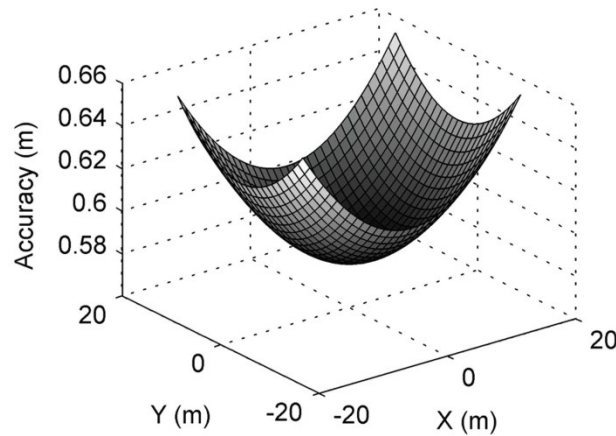


Figure 3-10. Vertical accuracy across the image at a 40 m distance from stereo boom. Image is 1600x1200 with 8 mm focal length, 1.5 meter baseline and 1 pixel disparity change.

If measurements are restricted to the center pixel, Figure 3-11 shows the vertical resolution at different distances from the camera which, for a nadir camera, corresponds to altitude AGL. Figure 3-12 shows the horizontal resolution at different altitudes AGL, which is simply a linear relationship.

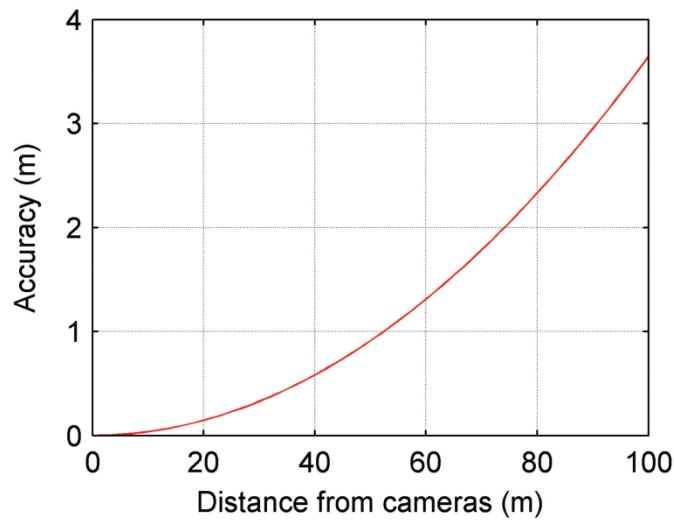


Figure 3-11. Accuracy at center pixel over distance (or altitude). Image is 1600x1200 with 8 mm (1980-pixel) focal length, 1.5 m baseline and 1-pixel disparity change.

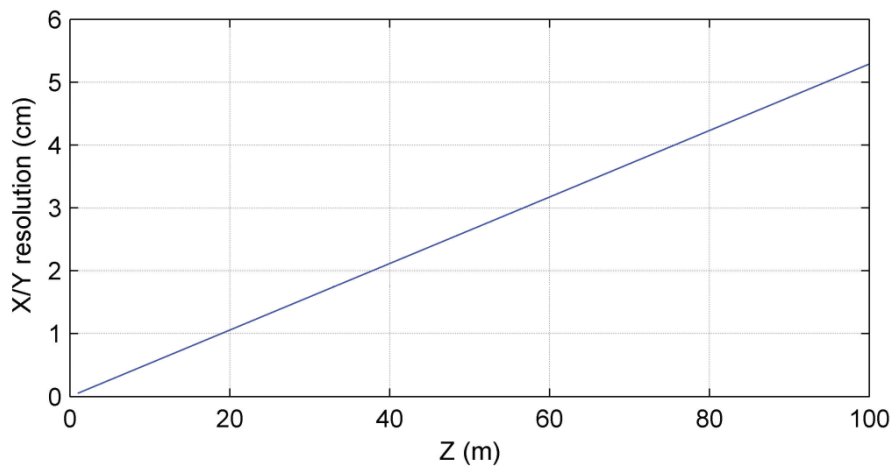


Figure 3-12. Horizontal resolution for both x and y axes as a function of Z (altitude AGL).

As expected, since accuracy decreases quadratically with altitude, high accuracy occurs at low altitudes and, conversely, high altitudes decrease the vertical accuracy. This loss of accuracy creates a significant tradeoff between vertical accuracy and area coverage. At higher altitudes, the field of view is greater, so mapping can be performed faster and more area can be covered. However, these higher altitude runs have lower accuracy, due to the stereo boom resolution. This trade-off is the primary reason for performing low-altitude aerial surveying with the UAV, since the stereo boom accuracy drops off much more rapidly after 40 meters.

Poor lens quality, asynchronous image capture, and motion blur can also present significant challenges to the visual reconstruction algorithm. No lens is perfect, and most lenses cause significant distortion to an image. This distortion can be accounted for, as mentioned in Section 3.2.3; however, distortion errors can still occur, due to miscalibration or an insufficient distortion model. An important assumption in stereo imagery is that the images are captured simultaneously. However, the current stereo boom system experiences a delay between left and right image capture, observed by experimentation. This issue is believed to result from using a software trigger to activate the cameras, instead of a hardware trigger [44]. Such a delay is insignificant when the cameras and scene are static; however, since the majority of imagery is acquired during UAV flight, this delay results in an apparent motion between the two stereo cameras along the flight path, which can corrupt *a priori* stereo boom calibration. Non-static scenes or cameras can also cause motion blurring, when an imaged object moves across more than one pixel during the exposure of the image. In practice, exposure times usually range from 0.4 ms to 40 ms in sunlight outdoor scenes and helicopter flight is relatively slow, ranging from 2 m/s to 8 m/s. Despite these conservative parameters, motion blurring can still occur. Translational blurring, which results from speed, can be reduced by increasing altitude, which reduces spatial resolution accordingly. However, the majority of motion blur occurs by changes in orientation, such as when the helicopter turns. Blurring, either by turning or translation, adds to the ambiguity of the visual reconstruction problem, since the intensity at each pixel results from imaging several patches of terrain geometry, instead of just one.

Finally, visual reconstruction algorithms based on aerial imagery must deal with errors in GPS/IMU positioning and percent overlap requirements. The Microstrain 3DM-GX3-35 provides position and orientation data at a rate of 4 Hz and 50Hz, respectively. The horizontal accuracy is 2.5 m for position and 0.5° for heading. This data provides an initial estimate of absolute camera pose; however, these errors are well above the resolution of the stereo boom and must be dealt with for acceptable final results.

Percent overlap also heavily affects visual reconstruction. For an aerial stereo imagery, two types of overlap exist: the overlap between left and right cameras, called *stereo overlap*; and the overlap from sequential images left 1 to left 2, called *sequential overlap*. Left

to right image overlap is fixed for the stereo boom and is very high at normal flight altitudes, while sequential overlap depends on flight speed and altitude which can vary by flight. This thesis refers to sequential overlap simply as overlap, while stereo overlap is mentioned specifically. The need for sufficient percent overlap creates a trade-off between information gain and matching quality. Small overlap allows imagery to be captured at higher speeds, so a larger area can be mapped. Higher altitudes also generate a larger viewing area and allow greater virtual baselines, which increases depth resolution according to (3-10). Conversely, sufficient overlap is required to properly relate two images and greater overlap provides more potential matches between images, which increases the quality of matches through redundancy. Likewise, greater overlap increases the amount data shared between the images, which can be used for 3D mapping. In summary, low overlap requirements increase the depth resolution and rate at which imagery can be acquired, while high overlap requirements increase the density of information gained between frames.

3.4. Assumptions and Constraints

The work in this thesis overcomes these challenges through a combination of assumptions, applied constraints, and error checking. This section sets forth the assumptions and constraints as well as the validity of assumptions for the local dense bundling algorithm. The assumptions reduce many of the challenges faced in visual reconstruction, and the remaining difficulties are dealt with by algorithmic application of constraints and error checking, which is explained in more detail in Chapter 4.

This thesis will assume the following:

- All imagery will be taken during daylight hours
- The exposure of the left and right stereo cameras is identical within each stereo frame.
- Any scene illumination changes occur uniformly across the surface through time.
 - The main illumination for outdoor, daylight flight is the sun, and the only change in illumination comes from movement of the sun and clouds. At low altitudes the camera view is sufficiently small to cause cloud changes to be

relatively uniform across a single instantaneous stereo frame. An auto-exposure algorithm is in place to maintain identical exposure within a stereo frame and similar light levels between sequential stereo frames. The helicopter mission will also keep radiometric changes locally consistent. Also, due to the helicopter flight path, adjacent images are never captured more than 10 minutes apart. The entire mission itself will last only 30 minutes. As a result, scene changes due to shadow movement are locally consistent and can be ignored. The lowered contrast in shadowed regions will be dealt with by texture enhancing measures within the dense stereo algorithm.

- The overall scene is rigid.
 - This assumption is broken by movement, such as non-stationary cars or trees blowing in the wind. However, since the two stereo images are taken simultaneously, they provide 3D instantaneous information. As long as the majority of the scene remains static, multiple sets of 3D instantaneous information can be combined and moving objects will appear as matching errors to be removed.
- The terrain surface is piecewise linear.
 - This assumption will be broken at discontinuities. To deal with surface discontinuities, the dense algorithm depends on robust methods like RANSAC for sparse feature matching and the dense matching optimizes a combined surface continuity constraint and similarity constraint.
- The terrain surface has primarily diffuse reflection and is opaque.
 - The overall assumption is valid since the majority of terrain will exhibit low specularity, especially in non-urban scenes. For urban environments, this assumption is invalidated by cars, windows, and metallic roofs, which have specular and/or transparent surfaces. For these instances, specular surfaces will be removed by error detection during local bundling of frames.
- The helicopter altitude will remain constant, independent of terrain height.
- The terrain has a known maximum and minimum height.
 - The maximum and minimum height act as geometric cutoffs which limit the required search space for the dense algorithm. Any data outside these limits

should be removed as erroneous. The maximum height constraint is justified because the helicopter flies at a constant altitude and will not fly closer than 10 meters to any terrain due to safety concerns. The minimum height constraint is set to 100 m away from the helicopter and is justified by stereo camera resolution, since distances further than 100 meters have such poor vertical resolution that the information is almost unusable for an operator.

- The calibration of the stereo boom is known and constant.
 - This constraint enables many of the advantages of the local dense bundling algorithm. The prior calibration is performed pre-mission and determines all intrinsic and extrinsic information between left and right cameras. The stereo boom is isolated from the helicopter vibrations with a damping mount to keep the calibration parameters constant.
- There exists 81.25% or greater overlap between sequential images
 - This constraint is applied by the helicopter flight path and—in conjunction with the helicopter speed and altitude—defines the maximum search area. The actual value of 81.25% overlap is derived from the experimental performance of the local dense bundling algorithm, given in Chapter 5.
- The terrain has sufficient texture to enable both detection of feature points and matching dense pixels.
 - This assumption is required for any form of visual reconstruction. Though some terrain, both in urban and non-urban environments, will have low-texture and repetitive patterns, these regions will be overcome through the optimization within the dense algorithm or removed during the error checking of the bundling process.
- The system will not be used to map large bodies of water, such as lakes or rivers.
 - Larger bodies of water are highly specular, non-rigid, and often textureless. As a result, they will simply be ignored due to the high likelihood of errors. It is worthwhile to note that LiDAR also cannot map such regions, due to the complex and dynamic specular surface properties of water.
- The majority of camera motion will be translational.

- This assumption will be fulfilled by using an appropriate helicopter flight path. Forward translational flight provides the maximum information gain for stereo vision, and prevents rectification anomalies. Though the stereo imagery is not perfectly simultaneous, as mentioned before, experimental results have shown that given translational motion, feature points can be used to correct the rectification process.
- Very little motion blur is present.
 - Though motion blur occurs, especially during turns, the local dense bundling algorithm uses both a feature point algorithm and area-based algorithm that are resistant to blurring. The feature point algorithm down-samples initially, which reduces noise and motion blurring, and feature points are matched using a robust method. Similarly, the optimization within the area-based algorithm reduces its sensitivity to motion blurring. Finally, images that experience extreme motion blurring are unused.

Though many of these assumptions are broken during the imaging process, the exceptions are usually detected and removed by proper algorithmic design. Together these assumptions and constraints enable the local dense bundling algorithm to enhance existing techniques and provide better vertical resolution and representation of 3D terrain.

4. Methodology

The local dense bundling algorithm is founded upon many building blocks common to the vision processing community. These include sparse feature detection and matching, dense two-frame stereo algorithms, and sparse bundle adjustment. The contribution of this thesis is a local method for combining these elements to optimize dense 3D structure from aerial stereo imagery. The algorithm fuses the advantages of two-camera stereo with motion stereo to enhance vertical resolution of the 3D terrain model. This thesis also presents a new tool for evaluating 3D visual reconstruction algorithms by simulating a full helicopter mapping mission with synthetic imagery. This chapter covers the project's existing code for stereo matching, the framework and inner workings of the local dense bundling algorithm, and a tool for creating synthetic imagery using open-source 3D modeling and rendering software with Python scripts.

4.1. Existing Code

The local dense bundling algorithm builds upon several previous algorithms, both for sparse reconstruction and camera positioning, as well as for fast dense matching. The earliest version of the 3D stereo mapping algorithm [45] consisted of software to capture images, calculate the disparity maps using a fast SAD window algorithm, and then reproject 3D data locally for each stereo frame. This local information was mapped to a global coordinate system by raw GPS/IMU values. The second iteration of the 3D mapping algorithm initializes data with GPS and IMU, but performs fast local pose refinement with 3D feature points and a grid-based iteratively reweighted least squares (IRLS) approach [44]. The dense matching is performed with an open-source variant of semi-global matching called semi-global block matching (SGBM). The local dense bundling algorithm forms the third component of the mapping algorithm, combining both of grid-based IRLS and SGBM into a global framework with improved resolution. All three components are designed to work frame-to-frame, to allow for real-time implementation.

4.1.1. Pose refinement with grid search and IRLS

For a given frame, composed of a left and right image, the pose refinement first extracts 2D SURF features from both images. The algorithm rectifies these points using the stereo boom calibration and then matches the points both using a nearest-neighbor search and an epipolar-confined search space. These matches are back-projected into 3D coordinates relative to the frame, using the stereo calibration. The result is a set of sparse 3D points for each frame, which are related to one another by rigid 6-degree-of-freedom transformations composed of rotation and translation. The algorithm proceeds to determine this rigid transformation between the current frame and subsequent frame. The search space is first divided into a multi-dimensional grid. This creates a six-dimensional grid with three dimensions of translation (x, y, z) and three dimensions of rotation (pitch, roll, and yaw). Since the helicopter will experience only small rotations for pitch and roll, these dimensions, these two dimensions can be represented as additional translational uncertainty. Also, altitude (z) is assumed to be constant for helicopter flight, so instead of six dimensions, only three dimensions, x , y , and yaw, are used for the multi-dimensional grid search. For each grid point, the cost is computed by summing the dissimilarity between matched SURF descriptors. This grid scheme is then used in a robust IRLS algorithm to determine the optimal rotation and translation between the current frame and next frame.

4.1.2. Semi-Global Block Matching

The semi-global matching approach was described briefly in Chapter 2, but due to its importance in the local dense bundling algorithm, it is more fully explained here. The local dense bundling algorithm uses SGBM, an open-source implementation available from OpenCV. The main distinctions between SGBM and the original semi-global matching algorithm are a reduced number of directions for the search (eight instead of sixteen) and the use of block matching by Birchfield and Tomasi's sub-pixel measure, instead of the mutual information measure used by Hirschmuller. The SGBM algorithm functions, as noted above, by dynamic programming from multiple directions. The SGBM algorithm comprises two steps: cost calculation and cost aggregation. Given two undistorted and rectified images, I_L and I_R , the matching pixels p in the left image and q in the right im-

age are related by the disparity map D , and the disparity $d = D(x, y)$, where $p = I_L(x, y)$ and $q = I_R(x - d, y)$. The cost for matching pixel p to a potential candidate q is based on the local intensity function surrounding the two pixels as shown in Figure 4-1. Pixel intensity is the numerical measure of the amount of light that falls on each cell of the CCD array during image exposure.

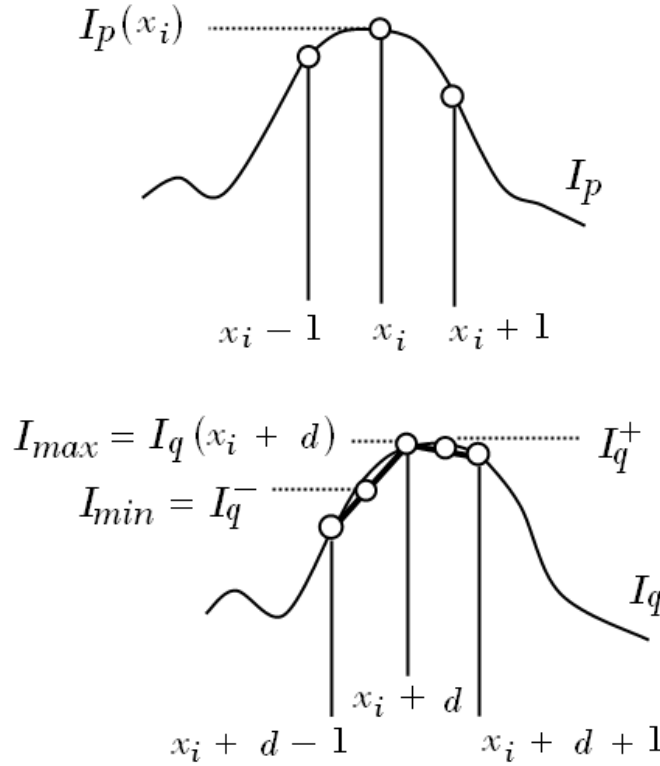


Figure 4-1. Cost function for dissimilarity in SGBM.

The cost is a direct function of the dissimilarity between the two functions, as computed by the most extreme change in intensity between I_p , the intensity at pixel p , and the local intensities around pixel q . This local function is composed of I_q , the pixel intensity at q ; I_q^+ , the intensity one half-step after; and I_q^- , the intensity one half-step before. From these values the basic cost function for a single pixel is formed as

$$C(p, q) = \max(0, I_p - I_{max}, I_{min} - I_p), \quad (4-1)$$

where $I_{max} = \max(I_q, I_q^-, I_q^+)$ and $I_{min} = \min(I_q, I_q^-, I_q^+)$. In SGBM, this cost is summed over a local window, or block, around pixel p , to form the actual cost. To aggregate this cost function over the entire image, a multi-path (Figure 4-2) dynamic

programming method is applied. Given a single linear dynamic path through an image, movement from pixel to pixel can be described as movement along a ray r . A horizontal ray, for example, would have the function $p = I_L(x + r, y)$, while a diagonal ray would be $p = I_L(x + r, y + r)$. The cost along a ray is calculated by combining block pixel difference cost with a surface continuity constraint to create the recursively defined cost L_r

$$L_r(p, d) = C(p, d) + \min(L_r(p - r, d), L_r(p - r, d \pm 1) + P_1, \min(L_r(p - r, d) + P_2). \quad (4-2)$$

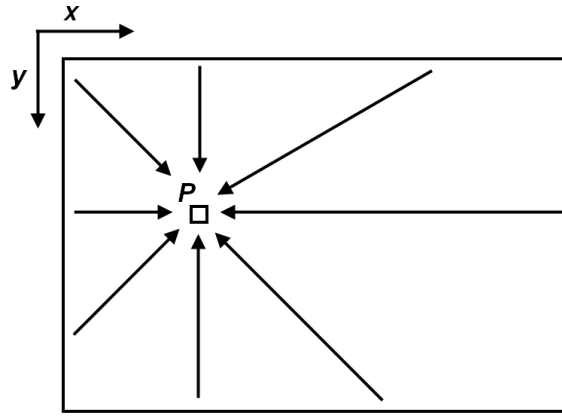


Figure 4-2. SGBM uses multiple dynamic programming paths.

The surface continuity constraint is based on the assumption that most objects in a scene are piecewise smooth and continuous, so changes in disparity (which correspond to depth) are penalized as discontinuities. This constraint is applied by the two parameters P_1 and P_2 , where $P_1 < P_2$, P_1 penalizes small changes in disparity of ± 1 , and P_2 penalizes large changes in disparity of greater than 1. For each pixel p and possible disparity d , the total cost is defined as $S(p, d) = \sum_r (L_r(p, d))$, which is the summation of costs over all directions of r . The SGBM method applies this cost over eight directions, evenly distributed across 360 degrees.

The SGBM algorithm implements sub-pixel matching of disparities by fitting a local quadratic to each disparity and its neighbors. The SGBM algorithm also uses a “uniqueness” measure as a quality constraint and removes speckles, or very small “islands” of disparity with little support. The uniqueness constraint checks to make sure that the highest scoring disparity for pixel p is some percentage higher than the next best disparity.

This constraint enforces distinctiveness of matches to reduce bad data due to repetitive structures or low texture, and is very similar to Zhang's quality measurement for SNCC. The current implementation of SGBM can perform at a speed of 10 seconds per frame with 1600x1200 size images on an Intel® Core™ 2 Duo. The SGBM algorithm performs very well on single frames, providing a relative fast method for dense reconstruction from a local camera reference. However, since the algorithm operates only in image space within a single frame, it is limited by the physical setup of the stereo system. Pixel disparity, even sub-pixel disparity, is still highly dependent on the baseline of the stereo cameras. Also, SGBM still faces errors with repetitive and low texture regions, even after applying a uniqueness constraint. This thesis presents a local dense bundling algorithm that overcomes these issues.

4.2. Local Dense Bundling Algorithm

The local dense bundling algorithm follows the general sparse to dense approach to 3D visual reconstruction by first locating and matching sparse points, performing dense matching, optimizing dense points, and detecting errors. This chapter provides an overview of the operational framework and describes algorithm operation for sparse feature detection and matching, dense feature matching, local bundle adjustment, and error removal.

4.2.1. Operational Framework

The operational framework of the overall stereo vision code is shown in Figure 4-3.

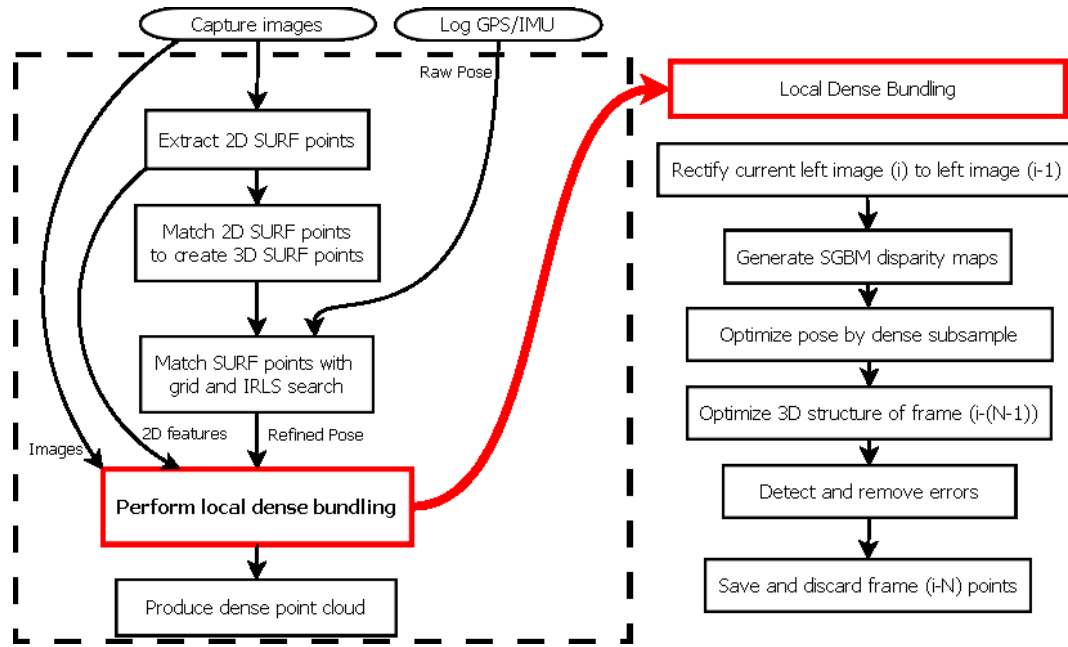


Figure 4-3. Operational framework of the local dense bundling algorithm.

The entire framework operates on sequential image pairs and GPS/IMU logs. As the helicopter follows the prescribed flight path, the stereo boom captures the three images—left, right, and color—at a rate of 1 Hz, along with GPS and IMU data for these images. Together this information forms a single frame f . The grid and IRLS search operates on the current frame f_i and the previous frame f_{i-1} , while the local dense bundling code operates on a local time window W containing N frames, where $W = (f_i, f_{i-1}, f_{i-2}, \dots, f_{i-N})$. The implementation of the local dense bundling algorithm in this thesis uses the window size of $N = 2$ to reduce memory requirements and computational time. The following sections present the local dense bundling algorithm in a generalized form for arbitrary window size.

4.2.2. Sparse Feature Detection and Matching

Given a new frame f_i , SURF feature points are first extracted from the raw, distorted right and left images, I'_{iL} and I'_{iR} . The SURF extraction is performed with an adaptive Hessian threshold to maintain a sufficient number of points for matching, usually between 1500 and 3000 points. If an insufficient number of points is located, the Hessian threshold is reduced and the SURF algorithm is rerun. These points are rectified using the stereo calibration and matched between left and right images using the epipolar con-

straint. The grid and IRLS search uses the resulting rectified points, along with GPS and IMU data, to refine the pose estimate. The local dense bundling algorithm uses the generated SURF points and refined pose estimate as a basis for dense matching and optimization of results.

Currently, the IRLS pose refinement algorithm matches SURF points *within* a frame—between left and right images of a single frame using the epipolar constraint and prior stereo calibration—but not *between* frames. The local dense bundle approach requires SURF matches between frames to perform proper rectification. To match sparse points between frames, the dense algorithm loads the SURF points for the left image from the current and previous frame and places them into a kD neighbor tree. Initial matches are computed by similarity of feature descriptor, using a threshold of 0.6, heuristically determined through experimentation. The descriptors are normalized, so the maximum dissimilarity occurs when the unit-vector descriptors point in opposite directions, producing a maximum value of 2.0. The fundamental matrix is computed from these matched points using a robust RANSAC method. After the fundamental matrix is computed, the matched points are refined by rejecting all outliers to the original RANSAC algorithm as false matches. From the successful matches and estimated fundamental matrix, the dense stereo algorithm computes the homographies H_1 and H_2 to align the previous left image, $I_{(i-1)L}$ with the current left image I_{iL} . The homographies rotate the two left cameras in image space such that the central axes are parallel and the epipolar lines run along the x -axis. Both the fundamental matrix and the homographies are calculated using OpenCV functions. Application of H_1 and H_2 to the left images will act as a second rectification process such that $H_1 * I_{iL} = I_{L1(i)}$ and $H_2 * I_{(i-1)L} = I_{iL2}$. For each frame, three image pairs now exist: I'_{iL} and I'_{iR} are the raw left and right images; I_{iL} and I_{iR} are the left and right current images after initial calibrated rectification; and I_{iL1} and I_{iL2} are the current left image and previous left image after secondary SURF-based calibration for frame i . This process is illustrated in Figure 4-4.

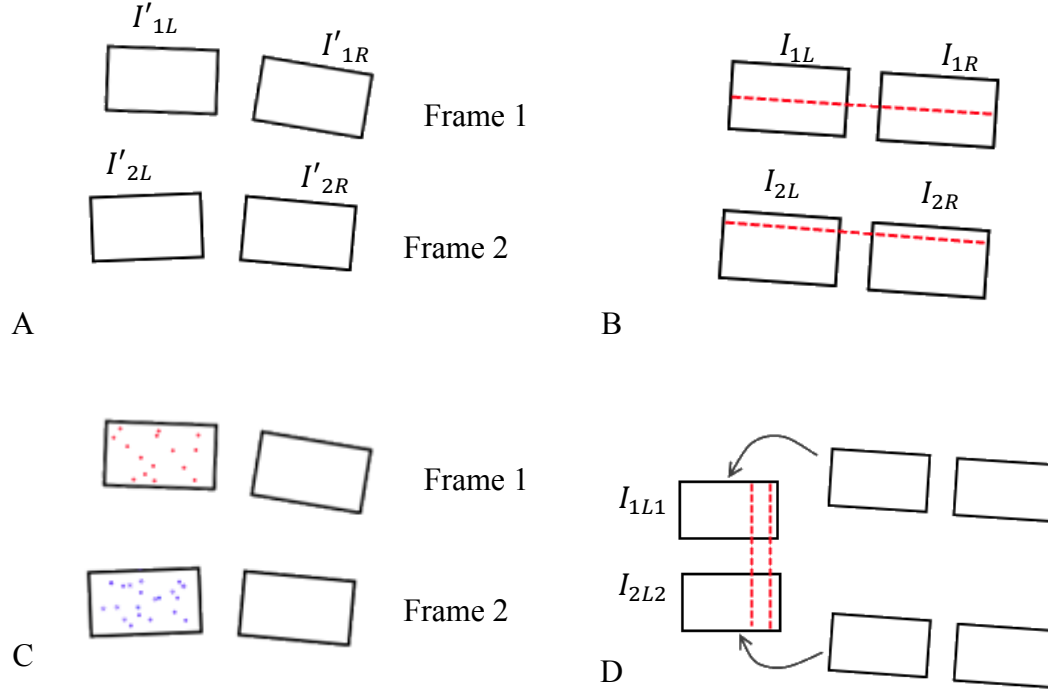


Figure 4-4. Raw images (A) are first rectified by stereo calibration (B). SURF points are extracted (C), matched, and used for secondary rectification of the left images (D).

4.2.3. Dense Feature Matching

After the computation of proper homographies and rectification of I_{iL1} and I_{iL2} , the dense algorithm constructs and relates dense disparity maps. The SGBM algorithm operates on I_{iL} and I_{iR} in the current frame to produce the current left-right disparity map D_{iLR} . The SGBM algorithm then operates on I_{iL1} and I_{iL2} to produce the left-left disparity map D_{iLL} . Together, these two maps relate points from the current stereo image pair I_{iL} and I_{iR} to the previous left image, which is used to construct a local list of matching points. One of the contributions of this thesis is this technique for interlinking stereo frames, using both rectification and the SGBM algorithm to create a list of linked dense points.

The local list of dense points is $G_d = \{A_1, A_2, \dots, A_j \dots A_n\}$, where

$$A_j = (\mathbf{X}_j, \mathbf{x}_{1jL}, \mathbf{x}_{1jR}, \mathbf{x}_{2jL}, \mathbf{x}_{2jR} \dots \mathbf{x}_{ijL}, \mathbf{x}_{ijR}) \quad (4-3)$$

is a single point with 3D world coordinates $\mathbf{X}_j = [X \ Y \ Z]$ and $\mathbf{x}_{ijL}, \mathbf{x}_{ijR}$ are the 2D coordinates of the point as shown in the images I_{iL}, I_{iR} . G_d is first constructed in frame $i = 1$, using only D_{1LR} , since no prior left images exist to create D_{iLL} . From D_{1LR} each

valid disparity is converted into a dense point and added to G_d . A dense point A_j is created from $D_{iLR}(x, y)$ by calculating the 3D world coordinate $\mathbf{X}_j$ and then the matching 2D image coordinates $\mathbf{x}_{1jL}, \mathbf{x}_{1jR}$. The 3D point is determined by applying the stereo back-projection matrix Q to convert from image coordinates to 3D camera coordinates, and then converting to world coordinates using the refined pose estimate. The back-projection matrix Q , valid for horizontally aligned stereo, is

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = Q * \begin{bmatrix} x \\ y \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -C_x \\ 0 & 1 & 0 & -C_y \\ 0 & 0 & 0 & f \\ 0 & 1 & -1/B & (C_{xR} - C_x)/B \end{bmatrix} \begin{bmatrix} x \\ y \\ d \\ 1 \end{bmatrix}, \quad (4-4)$$

where $\mathbf{X}_{jc} = [X_c \ Y_c \ Z_c]^T$ is the 3D point in camera coordinates, C_x, C_y is the center point of the rectified left image, C_{xR} is the center point of the rectified right image, B is the baseline of the stereo system and d is the disparity value at $D_{iLR}(x, y)$. $\mathbf{X}_c$ is converted to world coordinates using the refined pose estimate $RT_i = [R \ T]$, where R is a 3x3 orthogonal rotation matrix and T is a 3x1 translation matrix for frame i . R and T act as the homogeneous inverse of the projection matrix P , such that $P = R^{-1}[I, -T]$. Both RT_i and Q are referenced to the left camera. The 2D points $\mathbf{x}_{1jL}$ and $\mathbf{x}_{1jR}$ are computed as $\mathbf{x}_{1jL} = (x, y), \mathbf{x}_{1jR} = (x - d, y)$.

After the initial list G_d is created, an index map is also created, relating the pixel coordinates of image $I_{L(i)}(x, y)$ to $G_d(A_j)$ by

$$j = D_{ind}(x, y). \quad (4-5)$$

All invalid disparities $D_{ind}(x_{invalid}, y_{invalid})$ are set to $j = -1$. This index map allows for fast updating of the list during the next iteration of the algorithm.

After initial creation of the list in frame $i = 1$, all subsequent iterations follow the same pattern. For the next frame i the dense algorithm creates both D_{iLR} and D_{iLL} . A new local dense list G'_d is created by using Q , RT_i , and D_{iLR} . Using the index map D_{ind} , the new dense list is merged with the previous dense list G_d . For a given point A'_k in the new list G'_d , if the index $j = D_{ind}(x'_{ikL}, y'_{ikL})$ is valid, the camera coordinates $\mathbf{x}'_{ikL}, \mathbf{x}'_{ikR}$ of A'_k

for the current frame are added to point A_j . If j is invalid, A'_k is added as a new point to G_d . After list merging, a new index map is created from D_{LiR} , which corresponds to the updated list G_d .

If the updated list G_d has exceeded the full size of N frames, the dense algorithm extracts the set of 3D optimized points visible in only the trailing two frames $f_{i-(N-1)}$ and f_{i-N} . The algorithm saves this subset of points and removes the points from G_d . This reduces redundancy by preventing multiple optimizations or repeated saving of the same point.

4.2.4. Local Bundle Adjustment

Another contribution of this thesis is the use of a stereo-constrained local dense bundle adjustment to optimize the list G_d , which results in a higher accuracy 3D terrain model but only requires N frames out of the total dataset, instead of a global bundle adjustment which requires all frames in the dataset *a priori*. G_d maintains a local window of N frames, which consist of data from frames $f_i, f_{i-1}, f_{i-2}, \dots, f_{i-n}$. Current implementation limits the local window to two frames to make storing and optimizing data more tractable and efficient. Each frame can hold over one million points, which must be optimized across every single camera in which a point is visible. The optimization is focused on frame $f_{i-(N-1)}$, and all other frames act as support frames to increase 3D structure accuracy.

The algorithm implements the local bundle adjustment with the sparse bundle adjustment (SBA) C++ package from Lourakis and Argyros. The SBA code requires three main inputs: a parameter vector p , which contains the 3D point positions and camera pose; a measurement vector x , which contains the original 2D image coordinates of the 3D points as seen from each camera; and a functional relation fn that relates the two. The dense algorithm supplies SBA with the parameter vector

$$p = (RT_i, RT_{i-1}, RT_{i-2}, \dots, RT_{i-N}, X_1, X_2, \dots, X_j), \quad (4-6)$$

for all points j in list G_d .

The measurement vector x_m consists of the camera measurements

$$x_m = \begin{bmatrix} \mathbf{x}_{(i)1L}, & \mathbf{x}_{(i)1R} & \mathbf{x}_{(i-1)1L} & \mathbf{x}_{(i-1)1R} & \mathbf{x}_{(i-2)1L} & \mathbf{x}_{(i-2)1R} & \cdots & \mathbf{x}_{(i-N)1L} & \mathbf{x}_{(i-N)1R} \\ \mathbf{x}_{(i)2L} & \mathbf{x}_{(i)2R} & \mathbf{x}_{(i-1)2L} & \mathbf{x}_{(i-1)2R} & \mathbf{x}_{(i-2)2L} & \mathbf{x}_{(i-2)2R} & \cdots & \mathbf{x}_{(i-N)2L} & \mathbf{x}_{(i-N)2R} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \ddots & \cdots & \cdots \\ \mathbf{x}_{(i)jL} & \mathbf{x}_{(i)jR} & \mathbf{x}_{(i-1)jL} & \mathbf{x}_{(i-1)jR} & \mathbf{x}_{(i-2)jL} & \mathbf{x}_{(i-2)jR} & \cdots & \mathbf{x}_{(i-N)jL} & \mathbf{x}_{(i-N)jR} \end{bmatrix}. \quad (4-7)$$

Since not all 3D points are visible in every camera, the 2D coordinates for invisible points are supplied as null values and not used by the SBA software. The functional relationship fn is defined as a projection of each 3D point onto a stereo camera system,

$$fn = [fn_{11} \quad fn_{21} \quad \cdots \quad fn_{i1} \quad fn_{12} \quad fn_{22} \quad \cdots \quad fn_{ij}] \quad (4-8)$$

$$fn_{ij} = [\hat{\mathbf{x}}_{ijL} \quad \hat{\mathbf{x}}_{ijR}] \quad (4-9)$$

$$fn_{ij} = [K * P_i * \mathbf{X}_j \quad K * P_{iR} * \mathbf{X}_j], \quad (4-10)$$

where P_i is the projection matrix for the left camera of frame f_i , obtained from inverting the camera pose RT_i so that $P_i = R_i^{-1}[I, -T_i]$. P_{iR} is the projection matrix for the right camera, which is found by inverting camera pose RT_i and adding the stereo camera horizontal baseline B to produce

$$P_{iR} = R_i^{-1}[I, (-T_i + \begin{bmatrix} B \\ 0 \\ 0 \end{bmatrix})]. \quad (4-11)$$

Stereo-constrained sparse bundle adjustment functions by minimizing the distance of the residual error $\epsilon^T \epsilon$, where the residual error is given by $\epsilon = [\mathbf{x}_{ijL} - \hat{\mathbf{x}}_{ijL} \quad \mathbf{x}_{ijR} - \hat{\mathbf{x}}_{ijR}]$. The stereo-constrained SBA operates thrice to optimize the local dense results. Initially, the SBA optimizes the pose refinement from IRLS and GPS/IMU across a smaller subset of the dense point list. Due to the time consuming nature of optimizing hundreds of thousands of points, and since only a fraction of the points are needed for good pose estimate, a small subset is sufficient to optimize the camera pose. The algorithm extracts a random subset points G_r , arbitrarily set at a level over 9000, from the dense list G_d . Both pose and

3D points are optimized across G_r using stereo-constrained SBA for frames $f_i, f_{i-1}, \dots, f_{i-(N-1)}$, while f_{i-N} is held fixed. The algorithm passes the results through the error-checking function, described below, to remove outliers via reprojection error, and repeats the pose and structure SBA with the refined points. After optimizing camera pose, the dense algorithm fixes all camera poses and optimizes the structure of points within G_d that are visible only in the last frame, f_{i-N} , to produce the optimized list $\hat{G}_d$.

4.2.5. Error checking

After optimizing the structure of G_d to form $\hat{G}_d$, the dense algorithm then performs error checking to detect and remove mistakes from the SGBM procedure. The SBA optimization process minimizes a stereo constraint with projection error; however, the local dense bundling algorithm can also use the known distance accuracy of the stereo system, given by (3-10). The local dense bundling algorithm uses both projection error and the vertical accuracy to reject points. Before optimization, a copy of the un-optimized 3D point positions $\mathbf{X}_{est} = \{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_j\}$ is created. After local bundling, the optimized results are extracted from $\hat{G}_d$ in the form $\hat{\mathbf{X}} = (\hat{\mathbf{X}}_1, \hat{\mathbf{X}}_2, \dots, \hat{\mathbf{X}}_j)$. The local dense bundling algorithm calculates the distance each point moved during optimization and compares this distance to the expected stereo accuracy with respect to the left camera in frame f_{i-N} . Expected stereo accuracy, derived from (3-10), acts as a maximum acceptable translation in 3D space for each point $\mathbf{X}_j$, and is calculated by the following

$$\Delta \mathbf{X}_{max,j} = \frac{T_{z,i-N} \Delta d}{Bf} \|\mathbf{T}_{i-N} - \mathbf{X}_j\|, \quad (4-12)$$

where $\Delta \mathbf{X}_{max,j}$ is the maximum acceptable movement for a 3D point $\mathbf{X}_j$, $\mathbf{T}_{i-N}$ is the 3D location of the left camera for frame f_{i-1} as stored in RT_{i-1} , $T_{z,i-N}$ is the z component of $\mathbf{T}_{i-N}$, B is the baseline distance between cameras, f is the focal length, and Δd is the minimum value of one pixel of disparity change. If the optimized movement of $\mathbf{X}_j$ is greater than $\Delta \mathbf{X}_{max,j}$, the point is rejected as a mismatch. Also, if the reprojection error in any of the cameras exceeds 2 pixels, heuristically determined by experimentation, the point is rejected as a mismatch. After error checking, the local dense bundling algorithm replaces

G_d with the optimized, error-corrected list $\hat{G}_d$ and moves to the next frame $i + 1$ to continue iteration until all frames are optimized.

4.3. Blender Synthetic Image Creator

For testing and evaluation of the local dense bundling algorithm, synthetic imagery was constructed using the open-source modeling and rendering software Blender [2]. Blender provides the ability to construct a 3D model using a variety of tools for structure, shaping, and modeling as shown in Figure 4-5. Various surface properties, such as reflectance, color, texture, and specularities may be applied to the model. Textures are constructed and overlaid by mapping a pre-created image to the 3D structure, called texture mapping. Illumination of the model comes from different user-selected lighting conditions, including point sources, ambient light, and large unidirectional sources, such as the sun. To view the scene, Blender can define and place a camera model with properties such as focal length and resolution. Together, a 3D model with texture, lighting conditions, and a camera for viewing is called a scene. Blender then *renders* the entire scene, using geometry, surface properties, and lighting to calculate the light intensities falling on each pixel of the camera, which produces a final realistic image of the scene, such as the one in Figure 4-6.

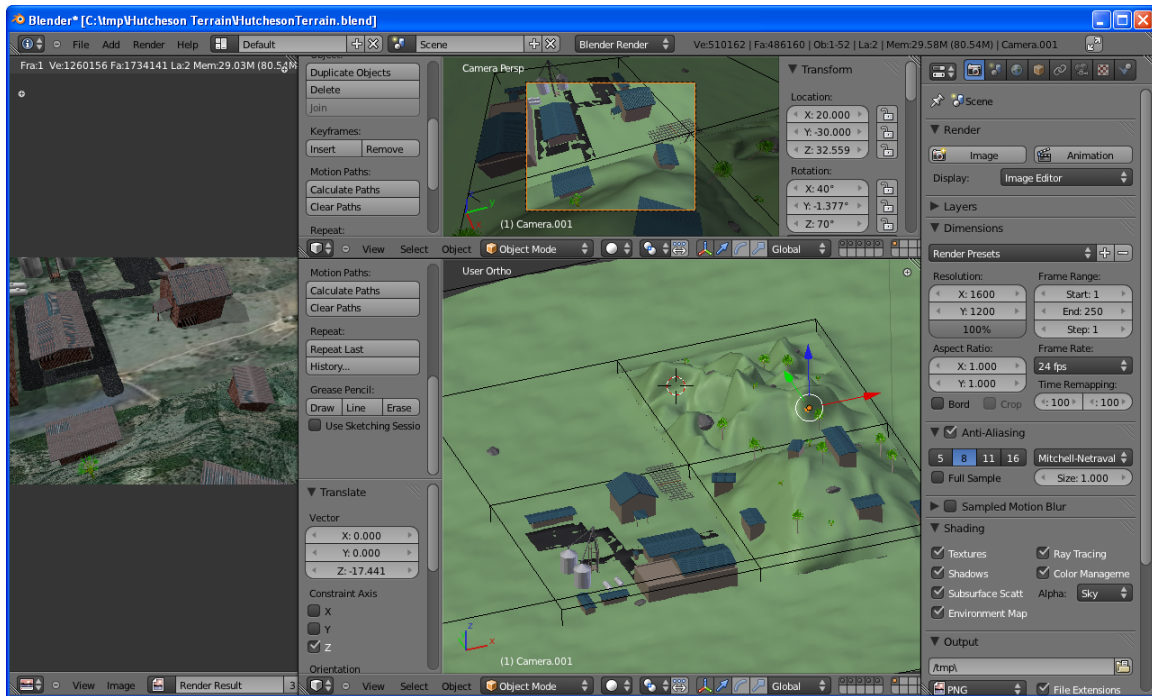


Figure 4-5. Blender 3D modeling and rendering software.

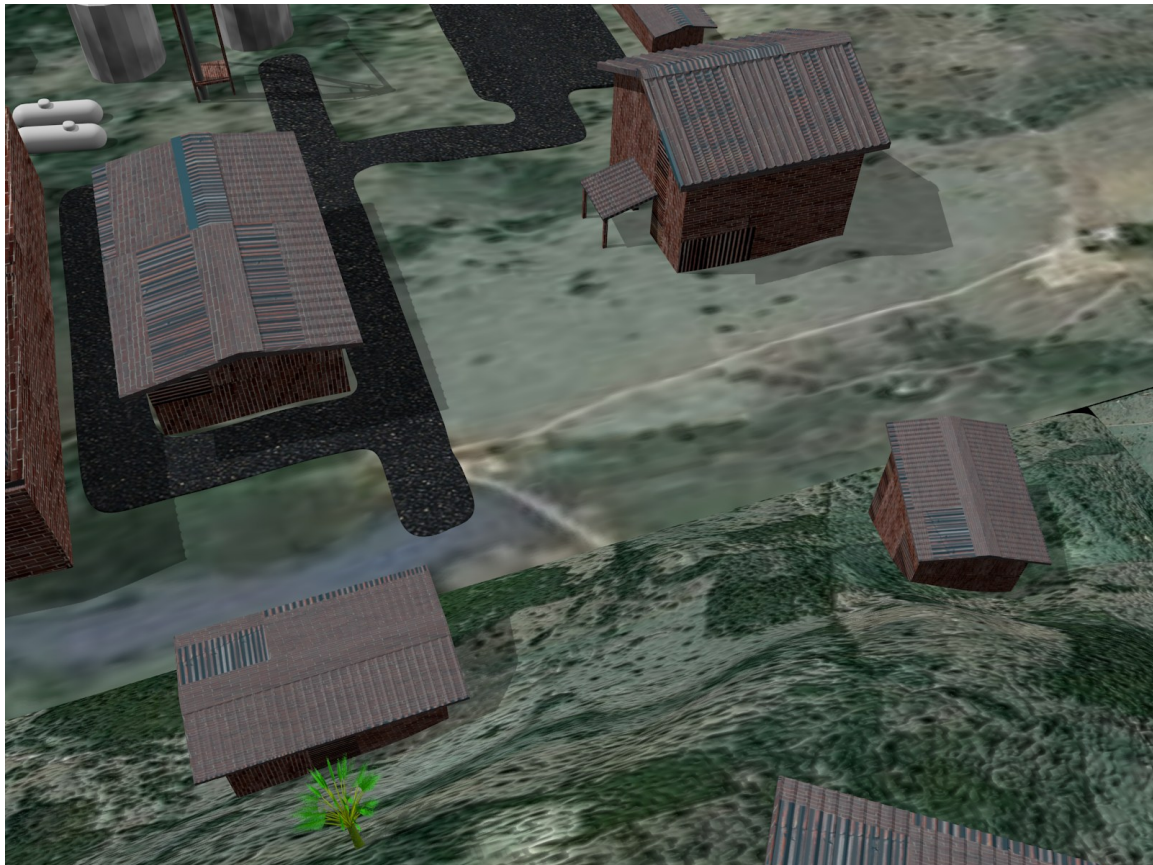


Figure 4-6. Example rendered image.

Blender also offers a Python API for interfacing with the main program. The Python API was used to create a script for automated rendering of synthetic stereo imagery, which simulates a helicopter flight. The script loads a stereo GPS/IMU log file, creates a rigid stereo boom of left and right cameras, and moves the stereo boom along the flight path, defined by GPS and IMU data, while rendering images at each location and orientation in the log file. The dataset created for evaluation of the local bundling algorithm consisted of imagery taken at three altitudes: 40 m, 60 m, and 80 m AGL. The left and right cameras had identical intrinsics, with a focal length of 30 mm (1500 pixels), no radial or tangential distortion, and extrinsics with no rotation between cameras and a baseline of 1.5 m along the left camera x -axis.

5. Results

The local dense bundling stereo algorithm was evaluated using a combination of real and synthetic imagery. This chapter examines algorithm performance on synthetic data through numerical error analysis, for both ideal and noisy cases, and evaluates performance on real data qualitatively through visual inspection.

5.1. Synthetic Imagery

Synthetic imagery provides a rich environment to test and evaluate computer vision algorithms. Images can be created with varying levels of realism, incorporating any number of surface properties, textures, lighting conditions and camera locations. For visual 3D reconstruction, synthetic imagery provides the key benefit of precisely known camera pose and exact 3D model structure. Precise *a priori* pose and structure allow analysis of both quality and robustness of algorithms. This section explores the performance of the local dense bundling algorithm on ideal imagery, algorithm performance in noise, details of the reconstructed 3D models, and computation time.

5.1.1. Ideal Performance

The local dense bundling algorithm was first run over the ideal data set, consisting of 11 images for each altitude 40 m, 60 m and 80 m AGL. The spacing between images, or virtual baseline, was 8 meters. All resulting data is compared against the previous SGBM algorithm. A side-view of the center section of the ground truth model is shown in Figure 5-1. For each altitude, the average error (in meters) is compared against distance from the left camera (in meters) as shown in Figure 5-2.

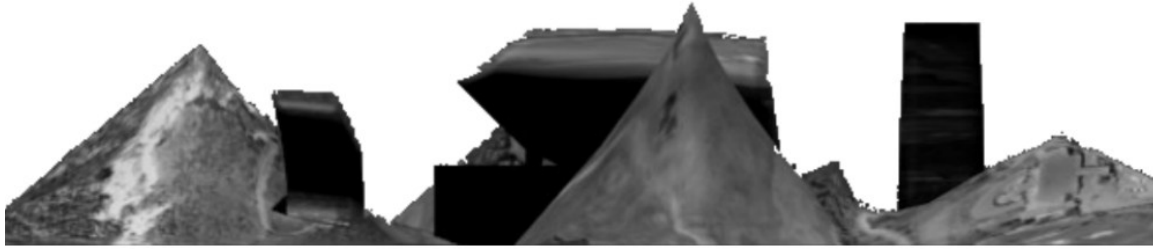


Figure 5-1. Side-view of ground truth blender model.

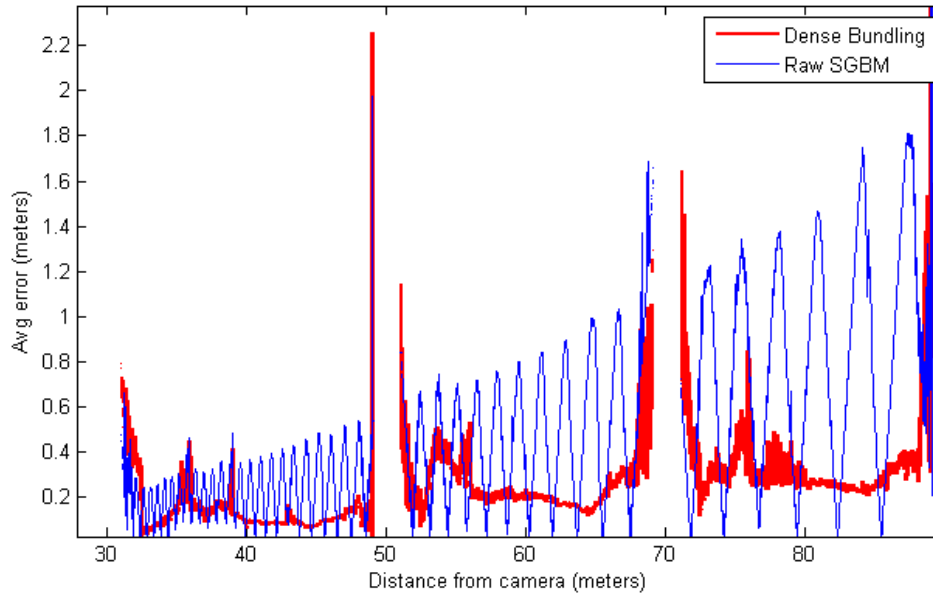


Figure 5-2. Average error over 5 frames (in meters) relative to distance (in meters) from left camera as produced by Raw SGBM and local dense bundling algorithms.

Figure 5-2 shows that the dense bundling algorithm significantly decreases average error across the range of distances, as compared to SGBM. The zig-zag pattern observed in the raw SGBM data is caused by the combination of a planar surface constraint in the SGBM algorithm and pixel discretization from the stereo cameras. Two-frame, rectified stereo algorithms, such as SGBM, are always limited due to the discretization of a continuous real-world scene by the charge-coupled device (CCD) array on a camera. To increase resolution, SGBM performs subpixel sampling by fitting a quadratic to nearby disparities; however, this technique only works near rapid changes in height. If a surface changes height gradually at a rate below the vertical resolution of the boom, the benefit of subpixel sampling is eliminated. One of the primary contributions of this thesis is that, by the

addition of multiple frames, the local dense bundling adjustment algorithm overcomes this limitation.

The local dense bundling algorithm can operate at a variety of virtual baselines. The algorithm was run at three virtual baselines—4 m, 8 m, and 12 m—for each of the three altitudes. Table 5-1 presents the average errors for successful, inlier matches. Points were considered outliers if the error exceeded the expected stereo error for the point, calculated from ground truth point location and camera pose using equation (4-12).

Table 5-1. Average error (in meters) for 3D reconstructed model constructed from ideal imagery across varying altitudes and virtual baselines.

Altitude (m)	Basic SGBM	Local dense bundling		
		4 m baseline	8 m baseline	12 m baseline
40	0.4095	0.2167	0.0969	0.0693
60	0.8244	0.4268	0.2764	0.5212
80	1.2894	1.0009	0.4503	0.3653

As evidenced in Table 5-1, the local dense bundling algorithm shows improved accuracy with increasing virtual baseline distance. This performance is expected, since each increase in virtual baseline acts as an increase in resolution.

Table 5-2. Factor of reduction in average error relative to basic SGBM for different virtual baselines at varying altitude.

Altitude (m)	Virtual baseline distance (m)		
	4	8	12
40	1.89	4.23	5.91
60	1.93	2.98	1.58
80	1.29	2.86	3.53

As evidenced in Table 5-2, the factor of reduction in average error ranges from 1.29 to 5.91 of the basic SGBM accuracy, following a generally increasing trend, except at the 12 m baseline at 60 m AGL. It is unknown why the reduction in errors drops at this altitude. Of note is the fact that accuracy gain per increase in virtual baseline drops off for higher altitudes. This is evidenced in the large accuracy increase of a factor of 5.91 for 40

m AGL scan with 12 m virtual baseline, and a smaller 3.53 factor increase in accuracy for the 80 m AGL and 12 m virtual baseline. This reduced effectiveness results from the quadratic loss of stereo boom vertical resolution with increased altitude. Despite this reduced effectiveness, the dense bundle improvement can still produce equivalent or better accuracy than SGBM at higher than normal altitudes, such as the accuracy of the 12 m baseline at 80 m AGL, which is slightly higher than the 40 m AGL SGBM accuracy. This improvement is another key contribution of this thesis, since local dense bundling allows the helicopter to perform higher altitude scans without a loss of vertical accuracy. Higher-altitude scans can be flown at higher speeds and still maintain equivalent overlap to a lower-altitude flight at a lower speed. Also, at higher altitudes the stereo boom views a much greater area per frame. Together, faster flights and greater viewing area per frame allow the helicopter to map terrain more efficiently, thus extending the range and the capabilities of the mapping mission.

Theoretically, the helicopter could ascend to any height and reconstruct the terrain with high vertical accuracy, so long as the virtual baseline distance is increased accordingly; however, this increase in vertical accuracy comes at a cost of point density and horizontal resolution. As either altitude or virtual baseline distance increases, the horizontal point density suffers. Higher altitudes increase the area imaged by a single pixel, which decreases horizontal spatial resolution, while large virtual baselines decrease overlap. Sufficient overlap is required for both image rectification and point density. The local dense bundling algorithm was run over the 60 m AGL data at several longer baselines to illustrate the point. The results in Table 5-3 reveal that the number of points matched roughly parallels the percent overlap; however, as evidenced by the ratio of change in percent overlap as compared to change in matched points, the change in density is not equivalent to percent overlap. At the low end, from 1.5 m to 12 m meters, a greater percentage of points are matched than predicted by percent overlap, while the higher virtual baselines of 16 m to 20 m suffer loss in point density higher than expected. Consequently, the dense bundling algorithm provides the most benefit with respect to point density for flight data with overlap of 97.7% to 81.25%.

Table 5-3. Loss of points and percent overlap for imagery at 60 m AGL and variable virtual baselines.

Virtual Baseline (m)	1.5	4	8	12	16	20
Percent overlap	97.656	93.750	87.500	81.250	75.000	68.750
Percent overlap/ percent overlap at 1.5 m	1.000	0.960	0.896	0.832	0.768	0.704
Number of points matched	974,357	965,906	951,171	810,946	632,845	546,726
Number of points matched/ number of points matched at 1.5 m	1.000	0.991	0.976	0.832	0.650	0.561

5.1.2. Performance with Noise

Results from the ideal data set show the local dense bundling algorithm performs as expected, providing increased resolution and lower average error for stereo imagery, based on the virtual baseline. These results depend on both a good helicopter flight path to provide proper images and perfect knowledge of pose to guarantee correct optimization. Many other dense visual 3D reconstruction algorithms rely on a similar assumption of highly accurate *a priori* pose knowledge, such as Furukawa et al. [38] and Zhang [13]. Unfortunately, the current grid and IRLS algorithm cannot achieve such precise results and even after refinement, pose errors of 0.4 meters to 0.7 meters exist. Likewise, the helicopter flight path is not guaranteed to be straight. Wind gusts or GPS drift could cause the flight path to deviate from a proper course. The local dense bundling algorithm should overcome as many of these challenges as possible.

To test the algorithm's performance in the presence of noise—from either poor pose refinement or an improper flight path—the algorithm was passed through two tests. The tests use the same imagery dataset as the ideal case; however, varying levels of noise were added to the supplied pose information. The pose information was corrupted with zero-mean Gaussian noise and a standard deviation for position and rotation of (1 m, 5°). The resulting average error (in meters) relative to distance from the left camera (in meters) is shown in Figure 5-3. The pose information was then corrupted by greater zero-mean Gaussian noise with standard deviation of (2 m, 10°) for position and rotation. For

both sets of noise, the algorithm operated on 60 m AGL imagery with a 4 m virtual baseline.

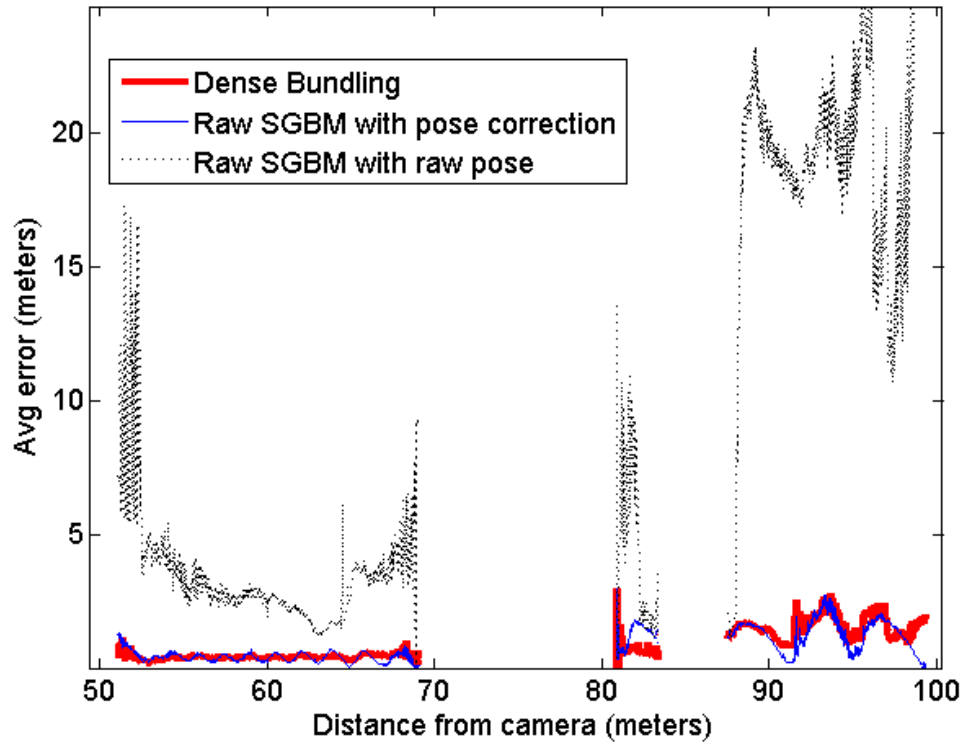


Figure 5-3. Average error across 5 frames with respect to distance from camera in the presence of 1 m 5-degree noise. Local bundling and SGBM were run at 60 m AGL with a virtual baseline of 4 m.

As evidenced by the graph, the raw uncorrected SGBM results have very high error. This error is dramatically improved by the local dense bundling pose correction step. By using the local bundling pose correction, the total average error decreases from 6.824 m to 0.598 m. The average pose offset (from ground truth) for the corrupted data and corrected pose are shown in Table 5-4. The local bundling pose correction reduces pose offset to less than one tenth the original error for both position and rotation. The pose correction itself is only slightly affected by increasing noise from 1 m to 2 m standard deviation, with a 15%, or 0.027 meter, average increase in pose error.

Table 5-4. Average offset errors in camera pose before and after correction by local bundling.

		Average positional errors (m)			Average rotational errors (degrees)		
		X	Y	Z	r	p	h
1 m, 5°	raw pose	1.645	1.142	1.015	4.205	5.297	3.589
	corrected pose	0.106	0.237	0.115	0.049	0.096	0.140
2 m, 10°	raw pose	1.588	1.172	1.148	5.977	6.052	2.449
	corrected pose	0.092	0.210	0.149	0.051	0.100	0.118

Despite the successful reduction in pose offset, local bundling algorithm 3D reconstruction is still hampered by the noise. For (1 m, 5°) noise, local bundling reduces error from 0.598 m to 0.445 m, a reduced improvement from the ideal case, where a 4 m virtual baseline increased resolution from 0.82442 m to 0.4268 m. For (2 m, 10°) noise the local bundling actually decreased the overall accuracy from 0.597 m to 0.622 m. This indicates that the supplied initial pose estimates must have sufficiently low error for the local dense bundling algorithm to successfully reduce error and increase resolution

5.1.3. 3D Reconstruction Analysis

Visual analysis of the model allows for a detailed assessment of the quality of the reconstructed models and observation of the differences between the SGBM method and the local dense bundling algorithm. The ground truth 3D model with rendered texture is shown in Figure 5-4. The texture is an aerial image, provided with permission by DigitalGlobe, and is uncorrelated to actual 3D model geometry. The image is simply used to provide texture for the SURF point extraction and SGBM matching. The models were constructed from 80 m AGL altitude imagery for both ideal and noisy data are shown in Figure 5-5.

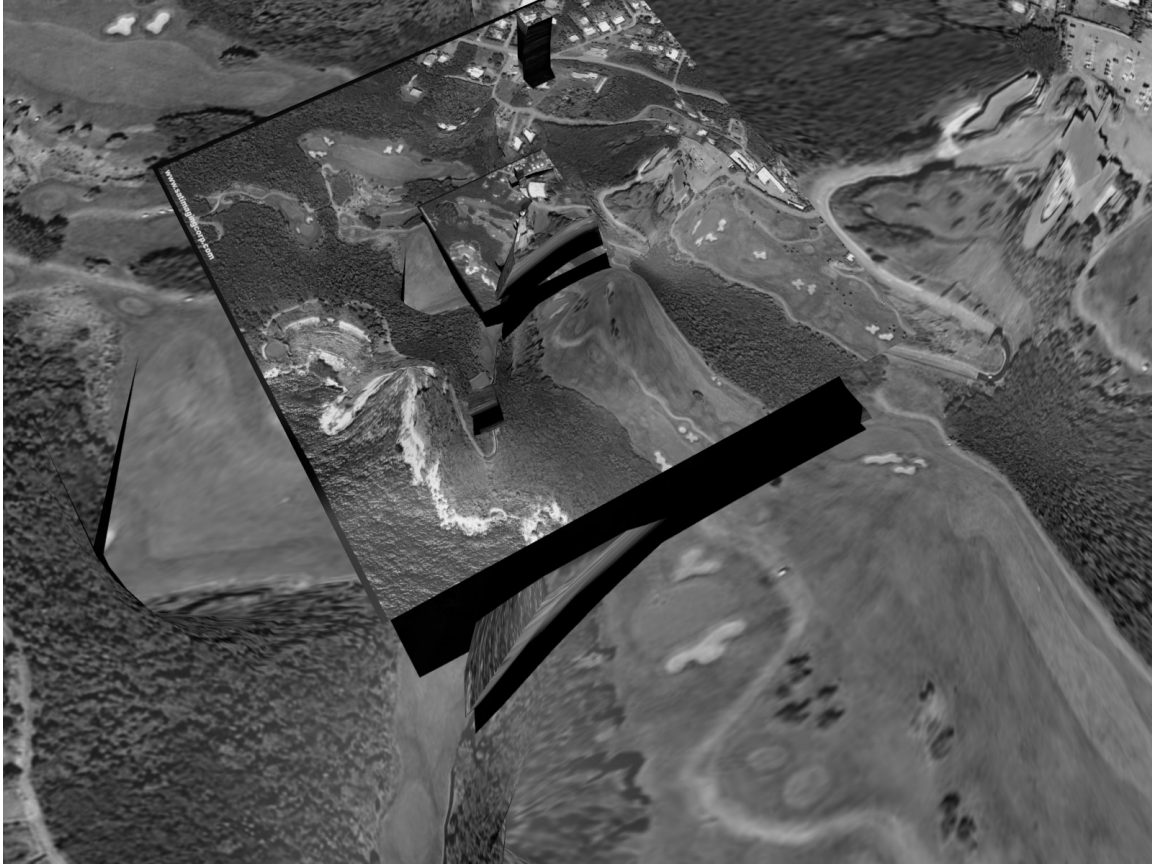


Figure 5-4. Ground truth 3D model with texture overlay.

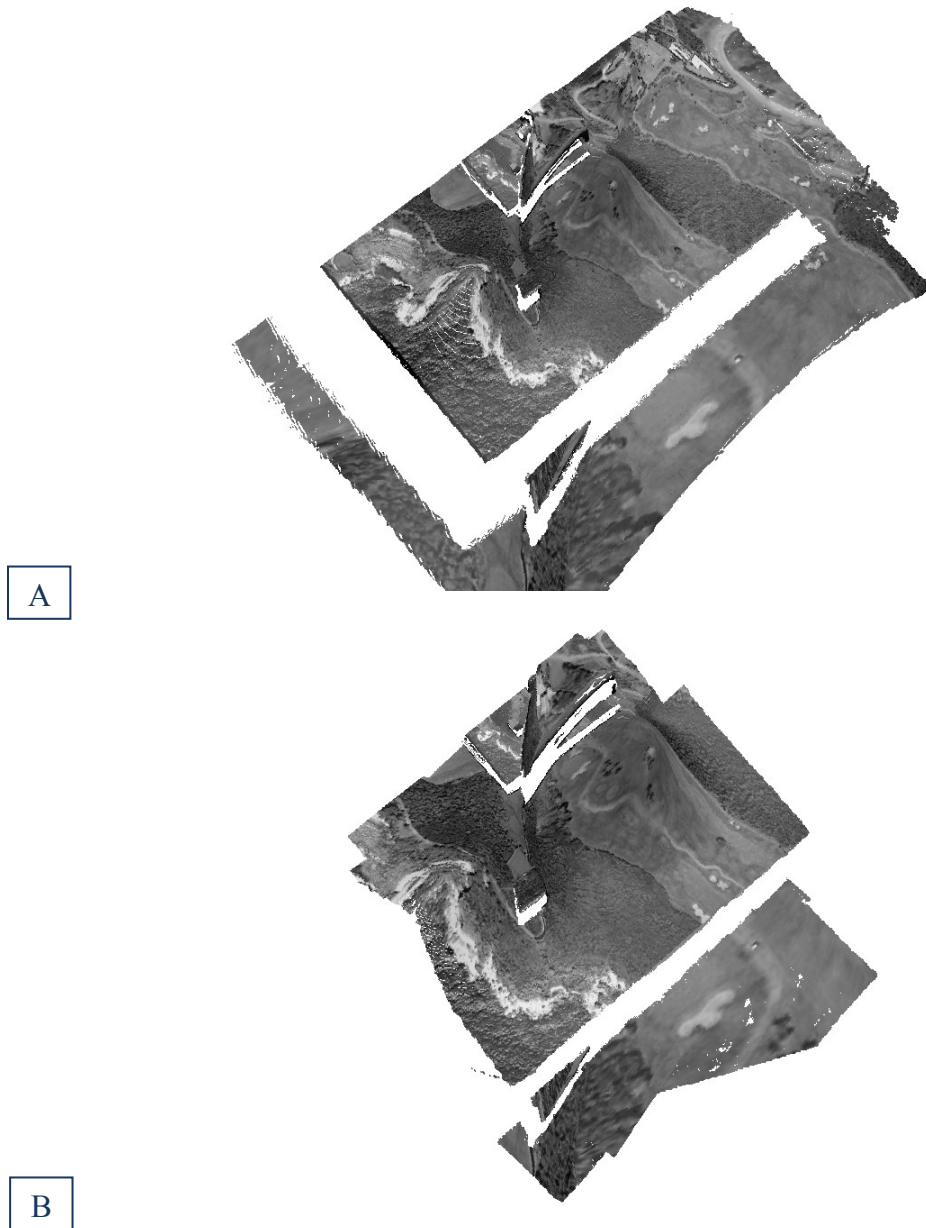


Figure 5-5. 3D reconstruction using ideal pose with 8 m virtual baseline at 80 m AGL (A) and noisy pose with 8 m virtual baseline at 60 m AGL(B).

Improved Resolution

The primary visible sign of the improved resolution resulting from dense bundling is the reduction of terracing. Terracing—which appears as sharp vertical steps—results from pixel discretization and occurs at disparity changes near the maximum resolution of stereo boom, where the terrain “jumps” from one disparity to another. The effect is shown in Figure 5-7, which is a close-up of the terrain, as shown in Figure 5-6. The effect is reduced by subpixel disparity estimation, as mentioned previously, which appears in the

rounded corners of the steps. However, subpixel estimation is limited to regions near disparity changes.

In comparison, the local dense bundling model provides a better reconstruction of the entire slope, with higher resolution as shown in Figure 5-8. Terracing still occurs, as a natural result of discretization; however the resolution is greatly increased, resulting in a smaller set of steps to describe the rapidly changing slope. Another example of terracing reduction is supplied in Figure 5-9.

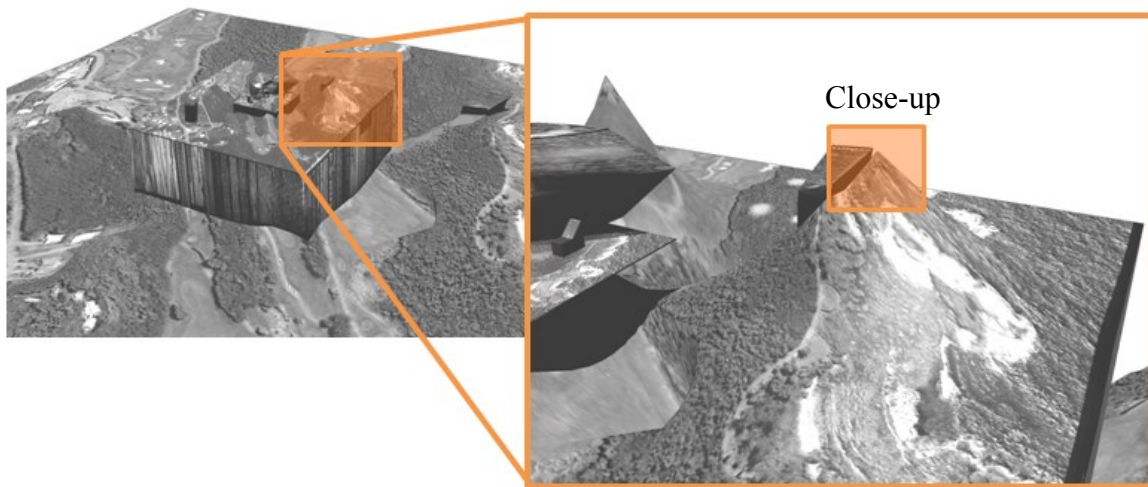


Figure 5-6. Close-up of peak (ground truth model shown) used for comparing 3D reconstruction.

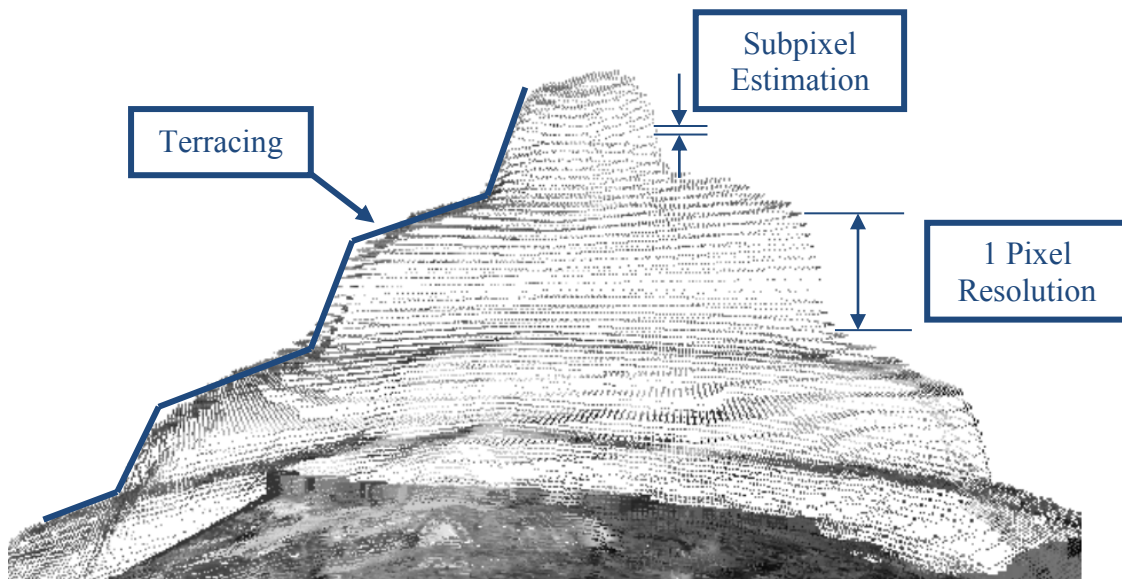


Figure 5-7. Close up where terracing occurs over a peak in the raw SGBM point cloud from 60 m AGL imagery.

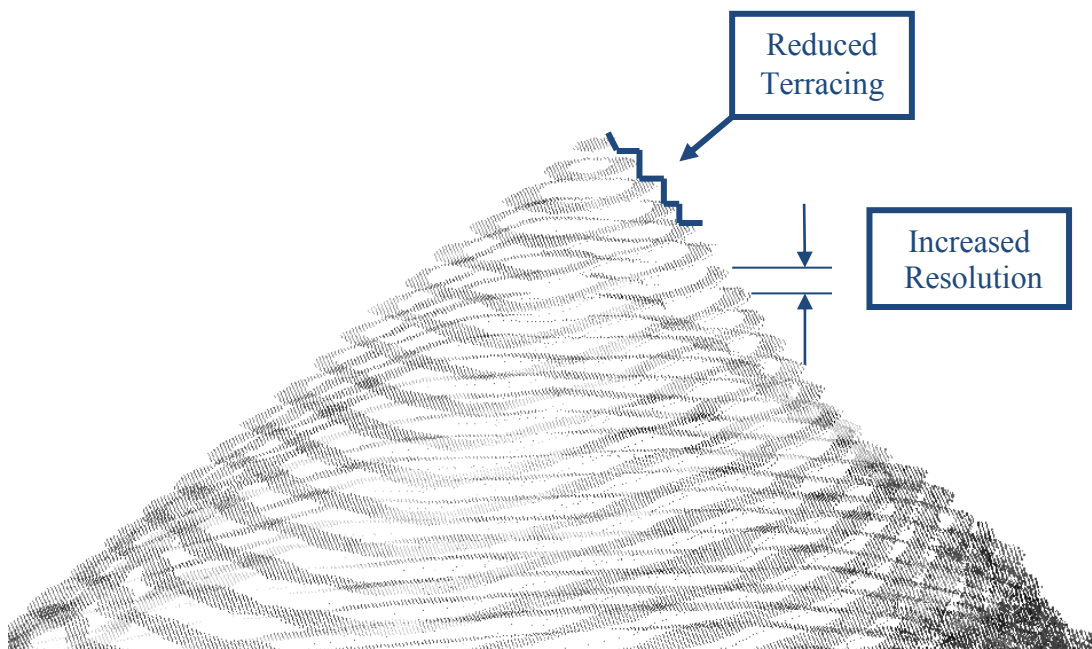


Figure 5-8. Close up where terracing over a peak is reduced by local dense bundling.

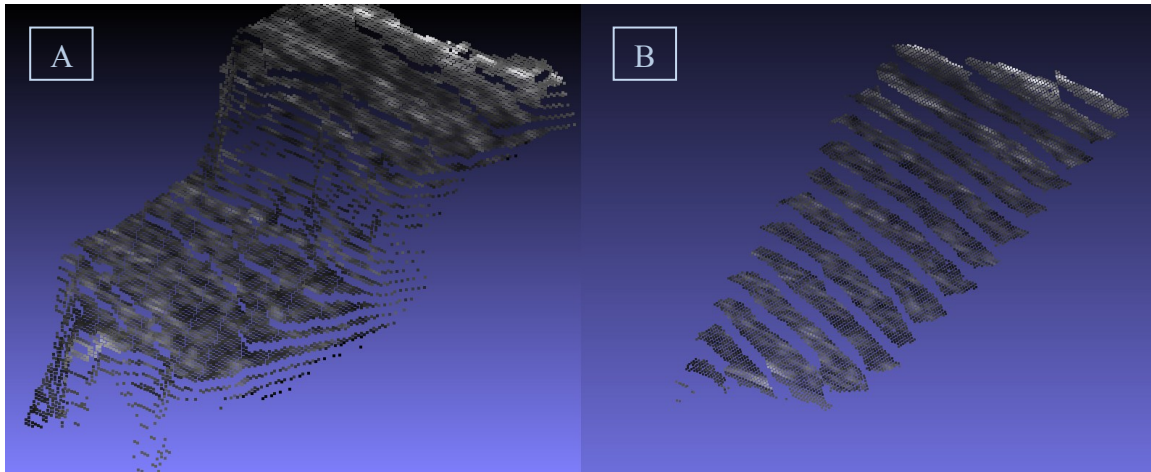


Figure 5-9. Terracing in SGBM cloud (A) is reduced by local dense bundling (B) with a virtual baseline of 8 m at 60 m AGL

Improved Pose Correction

The improved pose correction is best illustrated by comparing the 3D reconstructed models for raw SGBM with no pose correction and the local dense bundling results, as shown in Figure 5-10.

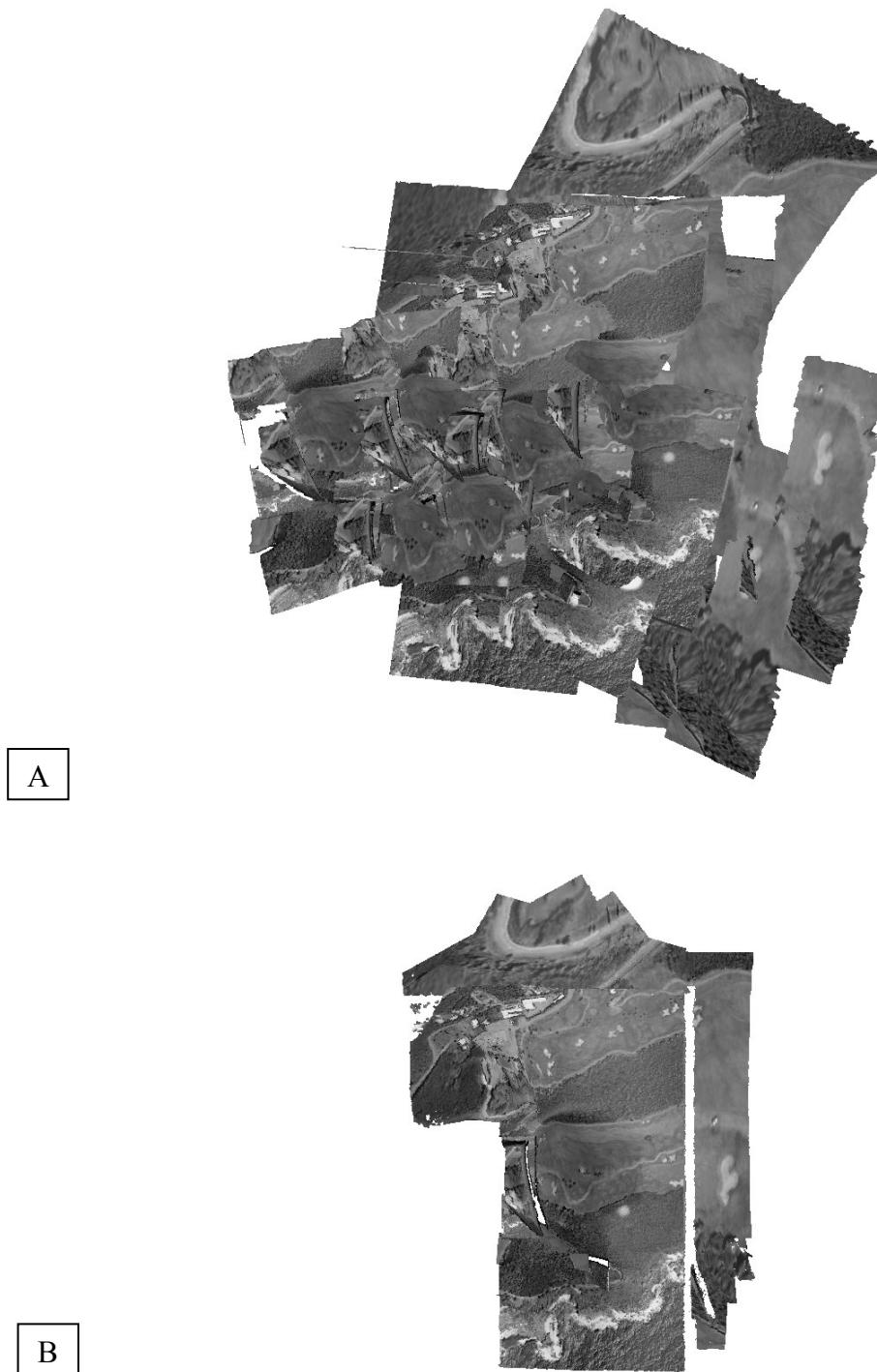


Figure 5-10. 3D reconstructed point clouds from Gaussian corrupted imagery/pose. (A) Raw SGBM with no pose correction. (B) Pose corrected local dense bundling. All data at 60 m AGL

Despite Gaussian noise added to both position and orientation, the local dense bundling algorithm reconstructed a cohesive 3D model. The data is most accurate in the x and y directions with higher error in the z direction. From closer inspection it was discovered

that the angle of discretization shifts from image to image in the noisy data. The angle of discretization is defined as the normal vector along which depth changes occur, shown in Figure 5-11. For normal SGBM and most two-frame rectified stereo algorithms, these changes occur parallel to the central axis projecting out of the camera. However, because the dense bundling algorithm combines data from multiple stereo images, the angle is not guaranteed to parallel any of the central camera axes. If motion occurs within a plane and rotation occurs solely around the camera's projection axis, such as in nadir imagery, the angle of discretization for local dense bundling will be normal to the imaging plane. However, if the images are taken at differing angles, the angle of discretization becomes the vector sum of the two camera principle axes. The angle results from the secondary image rectification and the SGBM algorithm used to link the stereo images. The image rectification aligns the two left stereo images with one another, rotating each camera a half rotation to bring the left cameras into parallel alignment, causing the second SGBM step to link points between the two stereo booms at an angle. The result is a planar linking of points orthogonal to both stereo cameras, which corresponds to the angle of discretization.

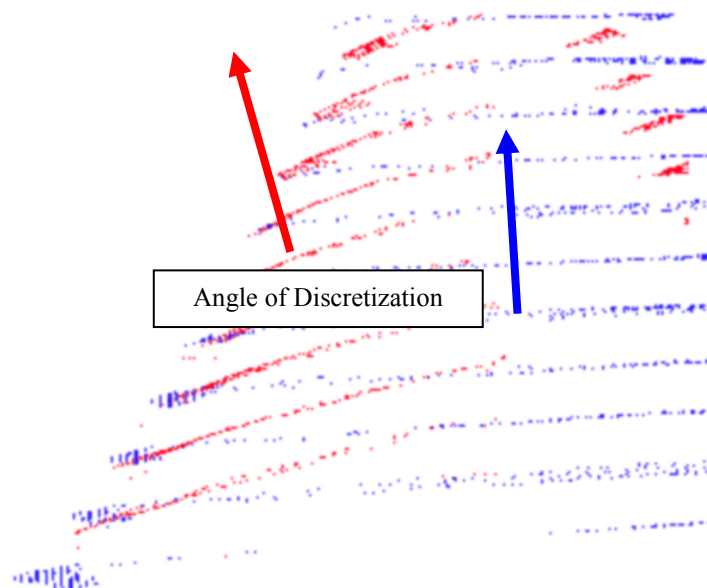


Figure 5-11. Angle of discretization for two dense bundled point clouds with different camera orientations.

This angled discretization causes the local dense bundling algorithm models to appear noisier than the raw SGBM models in the case of nadir flight with noisy IMU measure-

ments. The varying angles of discretization generate multiple planes of points at different angles, effectively “filling up” the region of spatial uncertainty for a single pixel. Visually, this is unappealing, since the model loses the appearance of surface continuity at close ranges; however, the average distance accuracies for the two models, both SGBM and local dense bundling, is the same. As a result, the local dense bundling data should be spatially averaged to provide a more accurate and operator-friendly terrain map.

Overall, the synthetic data shows both the promise and potential pitfalls of the local dense bundling algorithm. Given perfect accuracy, the algorithm can increase resolution from by 28% to 491%, depending on virtual baseline. The algorithm can also enable the stereo boom to function effectively at higher altitudes. The local dense bundling algorithm can also correct pose within an average error of 27 cm in position and 0.12 degrees in rotation, despite the addition of Gaussian noise. Even with pose correction, however, Gaussian error above (2 m, 10°) for position and rotation reduces dense bundling accuracy to normal SGBM levels. Therefore, the local dense bundling algorithm requires flights with nadir imagery and pose error of less than (1 m, 5°) .

5.1.4. Computational Time

The computational time of the various components of the algorithm are summarized in Table 5-5. All processing was performed on an Intel® Core™ 2 Duo with 3.45 GB RAM. As expected, sparse matching is very fast, while dense matching takes considerably longer. Sparse bundle adjustment, when implemented with feature points, is very fast; however, the sheer number of points involved in local dense bundling, usually close to one million, slows down optimization for both pose and structure. Currently, the stereo-constrained SBA routine analytically calculates the Jacobian of the residual at each step, which requires repeated trials for estimation. The non-stereo-constrained SBA uses a mathematically derived Jacobian function, which greatly speeds up calculations, allowing the algorithm to operate nearly twice as fast. Local dense bundling speed could be significantly increased by calculating the Jacobian of a stereo-constrained system and supplying it to the SBA routine. The current algorithm also expends a significant amount of time in searching and storing lists of dense points, which could be drastically reduced by a more efficient list class and structure.

Table 5-5. Computational time for various functions in the local dense bundling algorithm.

Function	Number of points	Time to process (seconds)
Sparse feature matching	2,358	5.53
Dense point generation(total)	974,357	93.10
SGBM	—	10.58
Match dense list	—	49.79
Dense optimization (total)	—	438.26
Pose SBA (stereo-constrained)	10,000	17.16
Structure SBA (stereo-constrained)	974,357	297.59
Error checking	974,357	28.31
Total time for 1 frame		536.88
For comparison:		
Structure SBA (small point number)	210,948	86.91
Structure SBA (no stereo constraint)	974,357	174.9

5.2. Real Flight Imagery

Synthetic imagery provides many benefits for evaluating accuracy, robustness, and developing visual 3D reconstruction algorithms. However, synthetic images remain only an approximation of the real physical world, even after corrupting synthetic data with noise. The truest test of an algorithm is its performance on real world data. To evaluate the local dense bundling algorithm, the algorithm was run over low-altitude aerial imagery, captured by the RMAX helicopter. This section examines the details of the aerial imagery dataset, the final 3D reconstruction, and an analysis of errors.

5.2.1. Datasets

The dataset consists of aerial imagery taken by the RMAX helicopter over the Kentland Farms area. The Kentland Farms area is a large set of fields and miscellaneous buildings located in Blacksburg, VA. Terrain consists of rolling hills, crop fields, forested areas, and small collections of buildings, including silos, sheds, and barns. Three datasets were created from the Kentland farms area on different dates: August 17, 2010; October 26, 2010; and April 14, 2011. This thesis uses the results from the April 14 flight, which imaged mostly buildings, gravel, and farm equipment. During flight, imagery was captured using the auto-exposure software while the stereo boom logged GPS/IMU measurements.

The move from synthetic data to aerial imagery removes several key assumptions. Calibration of the stereo cameras is no longer precisely known, potentially resulting in uncorrected lens distortions and image misalignment. The GPS and IMU data can be noisy, as noted previously, and the cameras are now subject to motion blurring, illumination changes, and unknown quality of texture. No ground truth topographical data for the Kentland Farms area was available for error analysis, so quality of reconstruction was performed manually by visual verification.

5.2.2. 3D Reconstruction

The local dense bundling algorithm was run over the April 14 dataset. The final 3D reconstruction was built mostly from raw SGBM results, since the majority of imagery proved unreliable for local bundle matching. The unreliability stems from an inconsistent

helicopter flight path and a very low image overlap of 75%. For regions meeting proper flight path and overlap requirements, the local dense bundling algorithm was successfully able to increase resolution and improve pose. One example of the improvement is the 3D model of the shed, shown in Figure 5-12A. Figure 5-12B shows a 3D reconstruction using SGBM and the raw GPS/IMU pose information for comparison. In normal operation, the grid and IRLS pose refinement would supply higher-accuracy initial data; however, the code was unavailable during the testing of the local dense bundling code, so all pose correction is performed by the local dense bundling algorithm on raw GPS and IMU data. The overhead view in Figure 5-12 shows that the local dense bundling algorithm corrects position effectively along the x and y directions, significantly reducing horizontal offset. The dense bundling algorithm successfully increases the resolution of the shed scene, especially the ground plane and the shed roof. Figure 5-13 and Figure 5-14 illustrate the improved discretization of both the ground and roof planes.



Figure 5-12. 3D reconstructed model of the shed in April 14 dataset using local dense bundling (A) and raw, uncorrected SGBM (B).

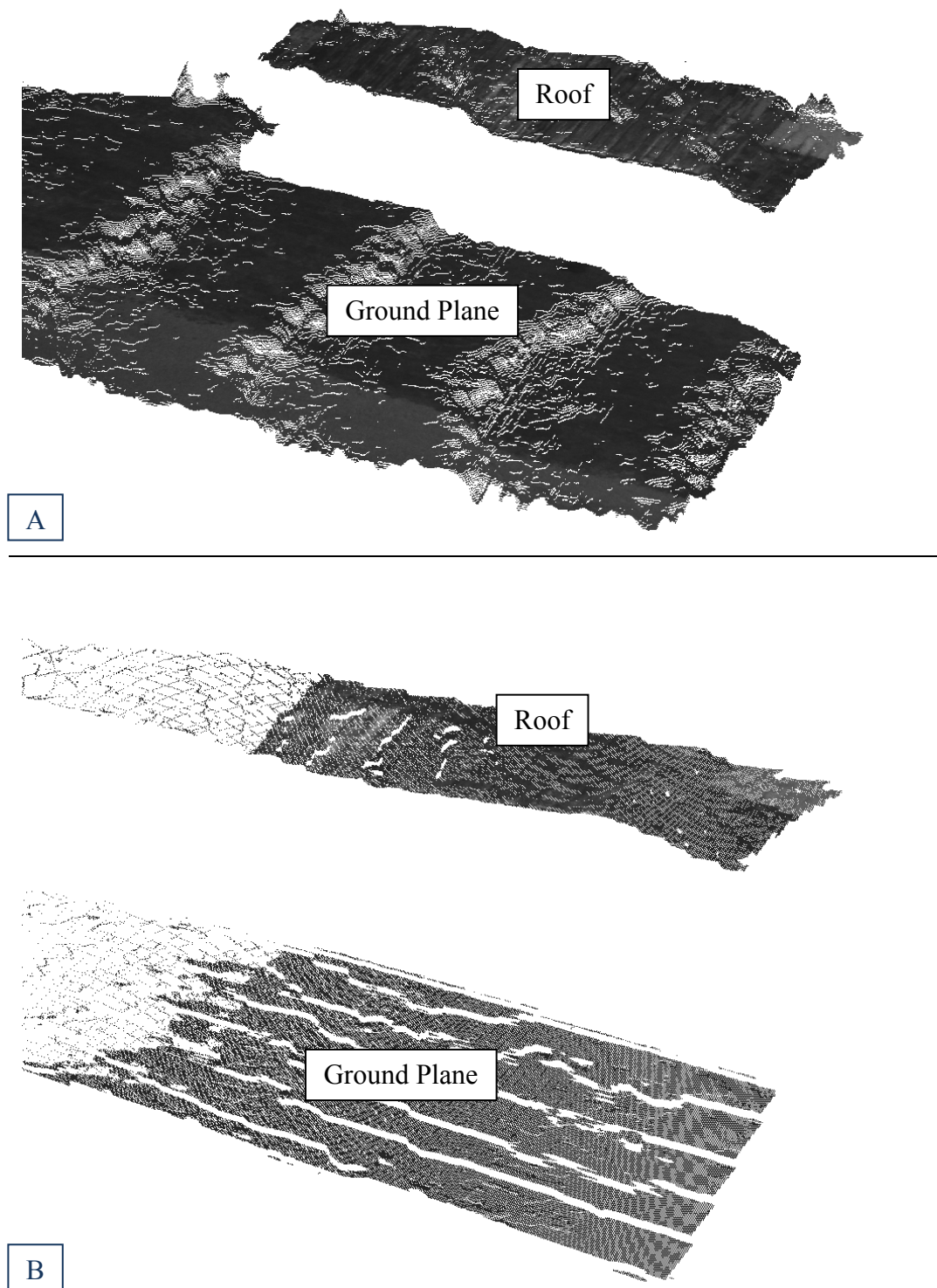


Figure 5-13. Close-up of 3D reconstructed shed from April 14 flight. SGBM (A) and local dense bundling (B).

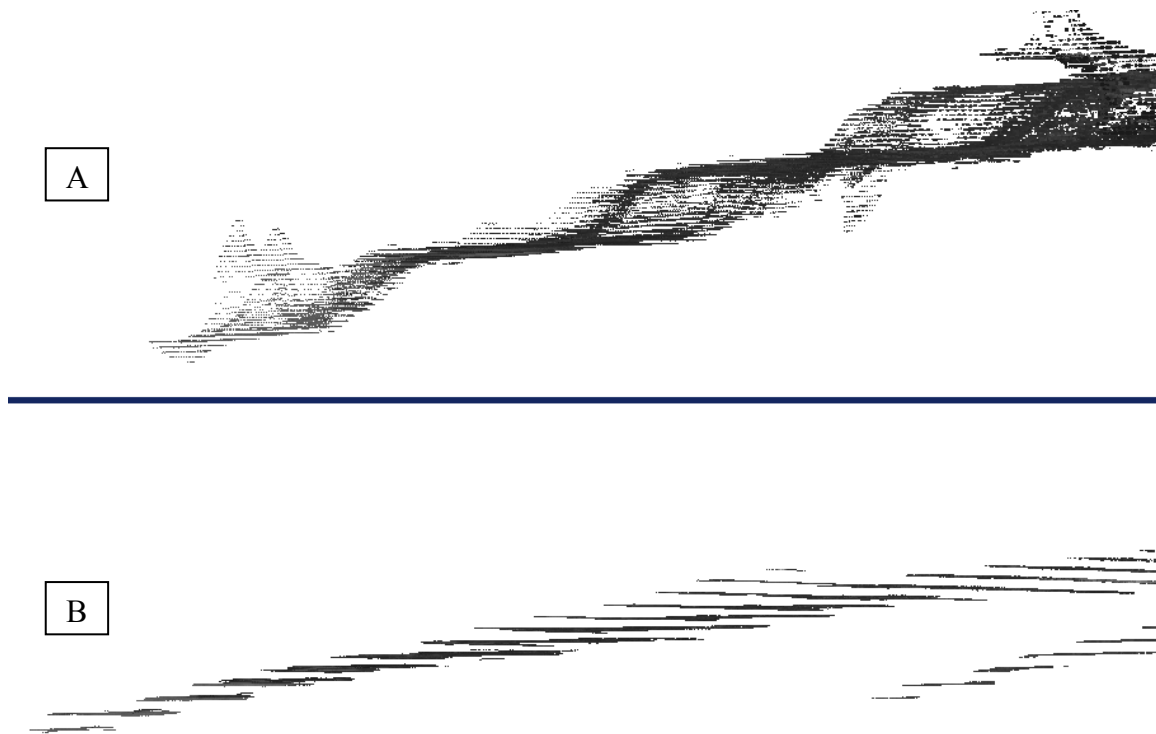


Figure 5-14. Close-up detail of increased resolution between raw SGBM result (A) and local dense bundling result (B).

5.2.3. Failure Analysis

Analysis of the results, both from real and synthetic imagery, revealed two key areas of failure for the local dense bundling algorithm. The algorithm breaks down in the face of poor secondary rectification and divergence in the local bundle adjustment.

Poor Secondary Rectification

The local dense bundling algorithm relies heavily upon the secondary rectification to align left images from the current and previous frames. The secondary rectification depends on locating sufficient features in the images, proper matching of these features, and calculation of the fundamental matrix with rectifying homographies. If overlap is too small between the images or if feature matching fails due to low/repetitive texture, the rectification fails, preventing the local dense bundling algorithm from matching points between frames to provide improved resolution. Even with sufficient matches to generate a fundamental matrix, the rectification can severely distort the images, which breaks down the operational assumptions necessary for SGBM. The dense bundling algorithm applies rectification as a planar homography change to the current and previous left im-

ages. Rotations in the pitch and roll of the helicopter can cause large projective distortions in rectification, as shown in Figure 5-15. Poor feature point matching can also produce improperly rectified images.

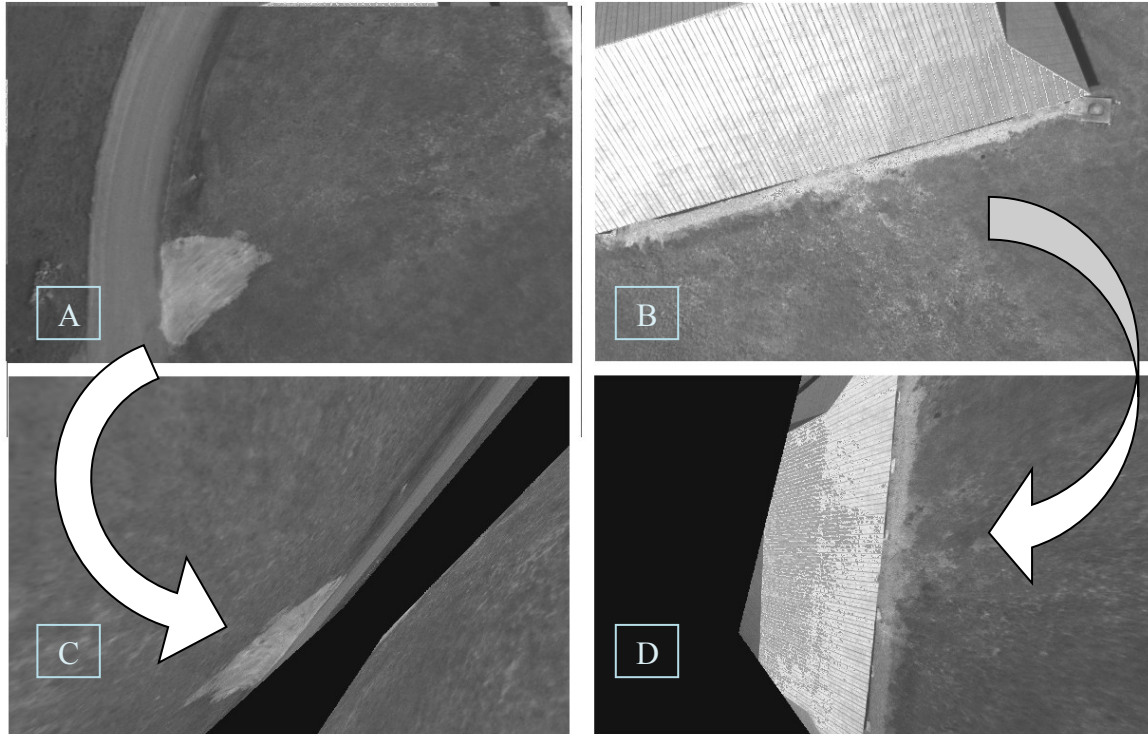


Figure 5-15. Improper rectification occurs when original the previous (A) and current (B) left images undergo projective distortion (C),(D) due to incorrect fundamental matrix calculation. Distortion in this example results from low texture and poor overlap.

The SGBM uses a block matching score as a base cost function, relying on the assumption that the scene can be approximated as a set of piecewise linear patches. If the image is distorted, this assumption breaks down, since projective distortions cause a rectangular window to cover nonrectangular Euclidean space. This results in poor SGBM quality and poor frame-to-frame matches. If the distortion is severe enough to cause full rectification failure, the local dense bundling algorithm reverts back to basic SGBM for 3D reconstruction, losing any improved resolution for the frame, but still producing 3D data.

Divergence and Non-robustness in Local Bundling

The local dense bundling algorithm also faces difficulties in divergence and non-robustness. If the initial estimates for stereo boom position or 3D point structure are significantly off—such as from poor left-to-left matching due to improper rectification—the

optimization routine for stereo boom pose correction will fail to converge. Such divergence is handled by limiting optimized pose results with GPS/IMU measurements. If the optimized pose exceeds the uncertainty of the GPS/IMU values, the optimized pose is rejected and raw GPS/IMU data are used instead. The local dense bundling algorithm verifies optimized structure by computing outliers, defined as points with reprojection error greater than two pixels. The algorithm rejects the optimization if the outlier percentage exceeds some threshold, set arbitrarily at 85%, and replaces the structure with the original SGBM 3D reconstruction.

Another challenge to local bundling is the non-robustness of SBA to outliers. SBA operates by minimizing the squared distance $(x_m - \hat{x})^T(x_m - \hat{x})$, where $\hat{x}$ is the collection of 2D image points as reprojected using current camera pose and 3D structure, and x_m is the original measurements of the 2D image points. SBA is therefore a maximum likelihood estimator; however, it performs no rejection or weighting of data, so it will seek to minimize the error for all data, including outliers. Due to the surface continuity constraint, SGBM can potentially propagate high scoring disparities incorrectly into regions of low texture. This propagation generates mismatched points in the dense point list. The non-robustness is partially counteracted in the local dense bundling by performing two pose optimizations and removing reprojection errors after the initial pose correction; however, if the outliers constitute a significant percentage of the dense point population, the effect can significantly warp the results. The local dense bundling algorithm uses percentage of reprojection error outliers as a metric to detect large warps in the optimization results and replaces the erroneous structure with the basic SGBM structure.

The local dense bundling algorithm faces several potential failures; however, most of the failures result from an improper flight path or insufficient overlap. A “good” flight path is defined as nadir camera orientation with strictly forward flight, pitch and roll movements of less than 5 degrees, and GPS measurements with less than 1-meter standard deviation. Given a proper flight plan, 81.25% or greater overlap, and sufficient scene texture for feature point matching, the local dense bundling algorithm provides enhanced resolution and better pose accuracy in constructing a 3D terrain model.

6. Conclusion and Future Recommendations

This thesis presents a local dense bundling algorithm with improved resolution and error reduction for terrain mapping from an autonomous aerial vehicle (UAV) as a part of the Unmanned Systems Lab (USL) nuclear project. The thesis surveys the existing literature on visual 3D reconstruction, exploring detection and matching for both sparse and dense features, as well as algorithm frameworks. The thesis explains the fundamentals of projective vision, explores the challenges of visual 3D reconstruction, and establishes the assumptions and constraints for the local dense bundling algorithm. The thesis presents the operational framework and design of the local dense bundling algorithm and evaluates the algorithm's performance on both synthetic and real imagery. This chapter presents final conclusions on the local dense bundling algorithm as well as recommendations for future work.

6.1. Local Dense Bundling

This thesis presents the local dense bundling algorithm for improved stereo vision 3D reconstruction of terrain from aerial imagery. The algorithm operates on a local window of stereo frames by detecting and matching sparse SURF feature points to rectify images between frames, followed by semi-global block matching (SGBM) within and between frames to create disparity maps. The algorithm links the disparity maps together to form a dense point list and optimizes camera pose and 3D point structure using local bundling. The algorithm was tested using both real and synthetic imagery. Synthetic imagery was generated by a new tool comprised of Python scripting within an open-source 3D modeling and rendering program called Blender. Ideal synthetic data trials revealed the capability of the local dense bundling algorithm to increase vertical resolution for 3D reconstruction and reduce overall error of the final model. At 40 meters above ground level, the local dense bundling algorithm reduced average height error in the reconstructed synthetic model from 0.409 meters for basic SGBM to 0.217 meters for a 4-meter virtual baseline and 0.069 meters for a 12 meter virtual baseline. The ideal imagery also revealed that, given a sufficiently large virtual baseline, the algorithm can increase the effective altitude for a stereo vision mapping flight. Noisy synthetic imagery showed the algorithm

corrects stereo camera pose effectively; however, noise above (1 m, 5 degree) standard deviation for position and rotation results in a loss of optimized accuracy, eliminating the improvement from local dense bundling.

Local dense bundling on real data produced mixed results. Due to low percent overlap and a poor helicopter flight path for the April 14 dataset, the algorithm provides improved resolution for only a subset of the total imagery. The failure to produce results stems from several causes, most notably poor rectification and divergent/non-robust bundle adjustment. Despite these possible failures, for the higher overlap regions of the April 14 flight, such as the “shed” scene, the algorithm successfully increases vertical resolution and reduces the “terracing” or discretization errors, as observed through visual inspection of ground and roof planes. In conclusion, given a proper helicopter flight path with sufficient overlap between images, the local dense bundling algorithm can increase vertical resolution and increase accuracy between 28% and 491% as compared to the basic SGBM algorithm, depending on virtual baseline. The local dense bundling algorithm also reduces the restrictions on altitude for aerial stereo-boom mapping, allowing for higher and faster flights that cover more area. Higher-resolution mapping or greater area coverage will provide higher-quality results to the operator during the helicopter mission, providing greater situational awareness for decision-making, guidance, and analysis, in the wake of a nuclear disaster.

6.2. Future Works

This section explores potential improvements and extensions to the local dense bundling algorithm, including more robust rectification, optimized code, global bundle adjustment, non-sequential matching, multi-frame SGBM, and 3D hypothesis modeling.

6.2.1. Robust Rectification Between Frames

Rectification between frames is one of the most crucial components of the local dense bundling algorithm; however, rectification breaks down when the helicopter flight path and imaged scene fail to meet assumptions. Such failure can result from poor overlap and textureless scenes, which generate mismatched features and cause incorrect calculation of

the fundamental matrix. This failure can also result from non-nadir cameras, due to significant helicopter roll and pitch; however, in all real datasets, occurrences of poor overlap and low texture far exceed improper helicopter rotations.

To improve rectification, the fundamental matrix and homography calculation could be enhanced by refining SURF matches and optimizing results. Currently, the local dense bundling algorithm uses a two-stage RANSAC method to calculate the fundamental matrix. The fundamental matrix is calculated initially with low precision, results are pruned by rejecting outliers, and a more precise fundamental matrix is calculated. SURF feature matching initially occurs within a kD tree, with no influence from the epipolar constraint. Consequently, rejection of outliers by the fundamental matrix fails to remove mismatches which coincidentally occurred along the correct epipolar lines. To improve results, a secondary, refining SURF match could be performed after fundamental matrix calculation. This search would use both the epipolar and the ordering constraint to limit potential candidates for each match. The ordering constraint assumes all points maintain viewing order in both images along the epipolar line, which is valid for a continuous surface. Constrained refinement would greatly reduce the search space for feature matching and therefore lower the number of potential mismatches for homogenous textures, providing higher-quality matches.

From this higher quality set of points, the fundamental matrix can be recalculated more accurately and precisely using both pose estimates and optimization. The fundamental matrix can be directly related to helicopter motion by

$$F = K_L^{-T} \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix} R * K_L, \quad (6-1)$$

where K_L is the left camera intrinsic matrix, $T = [T_x \ T_y \ T_z]$ is the translation from previous to current left camera and R is the rotation matrix from previous to current left camera. Camera intrinsics are known from the stereo boom calibration and refined pose estimates are available for both left cameras from the grid/ILRS search. A more robust and accurate rectification procedure could therefore incorporate both the additional pose

information and the higher-quality SURF matches into an optimization routine to precisely calculate the fundamental matrix and the rectifying homographies.

6.2.2. Optimized Code

Slow performance currently limits the effectiveness of local dense bundle algorithm, especially in regards to onboard helicopter performance. The computational time for many of the functions in the code could be significantly increased by optimizing the code. Such optimizations include supplying the Jacobian matrix of a stereo-constrained reprojection to the SBA function to reduce internal calls for analytic Jacobian estimation, as well as using optimized linear algebra libraries for faster calculations. The dense point list structure can also be replaced with a class designed for quick storing and searching. Finally, many components of the local dense bundling algorithm could be parallelized to run on a multi-core processor.

6.2.3. Non-sequential Matching

Currently, the local dense bundling algorithm matches dense points between only subsequent frames. To link a dense point from the last frame f_{i-N} to the most current frame f_i , the point must be successfully matched between all frames. Consequently, if a point is lost from view, due to occlusion, or fails matching at some frame between first and last, when the point returns to view it is reintroduced as a new point. This causes both a loss of potential information to the local bundle adjustment, and the creation of redundant points. The same effect occurs for flight paths that self-intersect, where previously imaged terrain is modeled again as completely new data. To overcome these issues, a more comprehensive bundling algorithm could perform matching between not only the immediately prior frame, but *all* prior frames, as limited by GPS/IMU data. The inclusion of such non-sequential matching would significantly increase computational complexity and time; however, such processing requirements may be acceptable for the increase in spatial information and reduction in redundancy.

6.2.4. Global Bundle Adjustment

In conjunction with non-sequential matching, a global bundle adjustment could be added to provide full map consistency and increased pose accuracy. A global bundle adjustment optimizes points across all cameras to create the full 3D terrain map, with no removal of points and cameras like in local dense bundling. For dense points, with over nine hundred thousand points per matched frame, such an optimization is completely infeasible. Instead, sparse SURF points can be linked between images, using non-sequential matching, and supplied to the global bundle adjustment. Depending on the desired framework, the algorithm could function either in a “real time” fashion, where dense optimization occurs after each new frame is added, or in a post-processing framework, where all data is available. In a post-processing framework, the algorithm would initially perform non-sequential matching of SURF points across the entire dataset and supply the results to the global optimization routine to obtain corrected pose. The dense optimization would then use the globally optimal pose and perform local dense bundle adjustment for each frame. The post-processing framework would provide the most accurate results but necessitates all imagery *a priori*; while the “real-time” framework could operate during a helicopter flight to provide lower accuracy 3D terrain data within five minutes of each acquired frame.

6.2.5. Multi-view SGBM

Another potential improvement to the local dense bundle adjustment is expanding the semi-global block matching algorithm from two-frame stereo to multi-view stereo matching. Given a set of two frames, each with left and right cameras, local dense bundling currently performs SGBM three times to generate a dense list: twice for matching points within frames and once to match between frames using the left cameras. Each SGBM routine operates independently from one another, and the final disparity solution, based on image intensity and surface continuity, is optimal only within each pair of images. Conversely, a multi-view SGBM algorithm would optimize across multiple images simultaneously.

The basic SGBM algorithm calculates the cost for each pixel in the base image from a 1D correspondence search along the epipolar line in the opposite image. The SGBM algorithm then optimizes this cost across multiple linear directions. The multi-view SGBM algorithm would calculate the cost for each pixel in the base image from a $(N - 1) * D$ correspondence search along the epipolar lines in $(N - 1)$ images. The multi-directional optimization would proceed exactly the same as the basic SGBM algorithm. Such a search would prove far more computationally expensive than basic SGBM; however, the multi-view SGBM algorithm would provide a far more accurate disparity map by optimizing intensity similarity and surface continuity across multiple images, instead of independent image pairs.

6.2.6. 3D Hypothesis modeling

A final method of improving the local dense bundling algorithm is 3D hypothesis modeling. After initial sparse feature matching and rectification between frames, instead of a full-scale SGBM routine, down-sampling the images could produce a fast, approximate disparity map. Projecting this disparity map into 3D space creates a rough 3D hypothesis model, which could provide a multitude of possibilities for improving point matching and optimization. The 3D model could guide the SGBM routine by geometrically limiting disparity search space or by allowing calculation of proper window shape to reduce projective camera distortion due to surface geometry.

The local dense bundling algorithm could also utilize the 3D hypothesis model to incorporate not only camera pose and 3D point structure, but also pixel intensity and surface continuity into the full dense optimization. After optimization of each frame, the 3D hypothesis model would be refined and used to update a global 3D hypothesis model, maintained across all frames. Such a global model would require tremendous memory to store in detail; consequently, a fast data storage/access scheme would be necessitated. Such schemes include partitioning space into an octree, using a univalued Z-axis model such as a digital elevation map (DEM) or digital surface model (DSM), or simply storing a lower-quality wireframe model and interpolating between points. Three-dimensional hypothesis modeling is one of the most comprehensive and complex extensions to the local dense bundling algorithm; however, such a framework could provide a drastic in-

crease in accuracy, greater global cohesiveness, large reductions in redundancy, and higher-quality error checking. Finally, whether from 3D model hypothesis or a more robust rectification, each improvement to the local dense bundling algorithm will boost quality of the final 3D terrain model, taking the USL nuclear response project to new heights and greater dimensions.

Bibliography

1. Stefanik, K., J. Gassaway, K. Kochersberger, and A. Abbott, *UAV-Based Stereo Vision for Rapid Aerial Terrain Mapping*. GIScience & Remote Sensing, 2011. **48**(1): p. 24-49.
2. Blender. *blender.org - Development*. 2011 [cited 2011 June 22]; Available from: <http://www.blender.org/development/>.
3. Lucas, B.D. and T. Kanade, *Iterative Image Registration Technique with an Application to Stereo Vision*, in *Proceedings of the 7th International Joint Conference on Artificial Intelligence*. 1981: Vancouver, BC, Can. p. 674-679.
4. Shi, J. and C. Tomasi, *Good features to track*, in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*. 1994. p. 593-600.
5. Lowe, D.G., *Distinctive image features from scale-invariant keypoints*. International Journal of Computer Vision, 2004. **60**(Copyright 2005, IEE): p. 91-110.
6. Bay, H., T. Tuytelaars, and L.V. Gool, *SURF: speeded up robust features*, in *Computer Vision - ECCV 2006. 9th European Conference on Computer Vision. Proceedings, Part I, 7-13 May 2006*. 2006, Springer-Verlag: Berlin, Germany. p. 404-17.
7. Viola, P. and M. Jones, *Rapid Object Detection using a Boosted Cascade of Simple Features*, in *Conference on Computer Vision and Pattern Recognition*. 2001: Kauai, HI, USA. p. 511-518.
8. Scharstein, D. and R. Szeliski, *A taxonomy and evaluation of dense two-frame stereo correspondence algorithms*. International Journal of Computer Vision, 2002. **47**(Copyright 2002, IEE): p. 7-42.
9. Brown, M.Z., D. Burschka, and G.D. Hager, *Advances in computational stereo*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2003. **25**(Compendex): p. 993-1008.
10. OpenCV. *Camera Calibration and 3D Reconstruction—Opencv v2.1 Documentation*. 2010 [cited 2011 June 22]; Available from: http://opencv.willowgarage.com/documentation/cpp/calib3d_camera_calibration_pose_estimation_and_stereo.html.

11. SangUn, Y., M. Dongbo, and S. Kwanghoon, *Fast dense stereo matching using adaptive window in hierarchical framework*, in *Advances in Visual Computing. Second International Symposium, ISVC 2006. Proceedings, Part II, 6-8 Nov. 2006*. 2006, Springer-Verlag: Berlin, Germany. p. 316-25.
12. Kanade, T. and M. Okutomi, *A stereo matching algorithm with an adaptive window: theory and experiment*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1994. **16**(Copyright 1994, IEE): p. 920-32.
13. Zhang, L., *Automatic Digital Surface Model (DSM) generation from linear array images*, Geodesy, Geomatics, Cartography, Remote Sensing in 2005, Swiss Federal Institute of Technology (ETH) Zürich: Zürich
14. Zabih, R. and J. Woodfill, *Non-parametric Local Transforms for Computing Visual Correspondence*, in *Proceedings of the Third European Conference-Volume II on Computer Vision - Volume II*. 1994, Springer-Verlag. p. 151-158.
15. Birchfield, S. and C. Tomasi, *Depth discontinuities by pixel-to-pixel stereo*, in *Proceedings of IEEE 6th International Conference on Computer Vision, 4-7 Jan. 1998*. 1998, Narosa Publishing House: New Delhi, India. p. 1073-80.
16. Hirschmuller, H., *Accurate and efficient stereo processing by semi-global matching and mutual information*, in *Proceedings. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 20-25 June 2005*. 2005, IEEE Comput. Soc: Los Alamitos, CA, USA. p. 807-14.
17. Hirschmuller, H., *Stereo processing by semiglobal matching and mutual information*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008. **30**(Compendex): p. 328-341.
18. Haller, I., C. Pantilie, F. Oniga, and S. Nedevschi, *Real-time semi-global dense stereo solution with improved sub-pixel accuracy*, in *2010 IEEE Intelligent Vehicles Symposium (IV), 21-24 June 2010*. 2010, IEEE: Piscataway, NJ, USA. p. 369-76.
19. Baillard, C. and A. Zisserman, *Automatic reconstruction of piecewise planar models from multiple views*, in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 23-25 June 1999*. 1999, IEEE Comput. Soc: Los Alamitos, CA, USA. p. 559-65.
20. Gallup, D., J.M. Frahm, and M. Pollefeys, *Piecewise planar and non-planar stereo for urban scene reconstruction*, in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. 2010. p. 1418-1425.
21. Miled, W., B. Pesquet-Popescu, and W. Cherif, *A variational framework for simultaneous motion and disparity estimation in a sequence of stereo images*, in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. 2009. p. 741-744.

22. Miled, W. and J.C. Pesquet, *Disparity Map Estimation Using A Total Variation Bound*, in *Computer and Robot Vision, 2006. The 3rd Canadian Conference on*. 2006. p. 48-48.
23. Gong, M. and Y.-H. Yang, *Disparity Flow Estimation using Orthogonal Reliability-based Dynamic Programming*, in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*. 2006. p. 70-73.
24. Koch, R., *3-D surface reconstruction from stereoscopic image sequences*, in *Computer Vision, 1995. Proceedings., Fifth International Conference on*. 1995. p. 109-114.
25. Triggs, B., P.F. McLauchlan, R.I. Hartley, and A.W. Fitzgibbon, *Bundle Adjustment - A Modern Synthesis*, in *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*. 2000, Springer-Verlag. p. 298-372.
26. Dickscheid, T., T. Labe, and W. Förstner, *Benchmarking Automatic Bundle Adjustment Results*. 21st Congress of the International Society for Photogrammetry and Remote Sensing Part B3a, 2008: p. 1-6.
27. Lourakis, M.I.A. and A.A. Argyros, *SBA: A software package for generic sparse bundle adjustment*. ACM Trans. Math. Softw., 2009. **36**(1): p. 1-30.
28. El-Hakim, S.F., J.A. Beraldin, M. Picard, and G. Godin, *Detailed 3D reconstruction of large-scale heritage sites with integrated techniques*. Computer Graphics and Applications, IEEE, 2004. **24**(3): p. 21-29.
29. Remondino, F., S.F. El-Hakim, A. Gruen, and Z. Li, *Turning images into 3D models*. IEEE Signal Processing Magazine, 2008. **25**(Copyright 2008, The Institution of Engineering and Technology): p. 55-64.
30. Golparvar-Fard, M., F. Pena-Mora, and S. Savarese, *D4ar- a 4 dimensional augmented reality model for automation construction progress monitoring data collection, processing and communication*. Journal of Information Technology in Construction, 2009. **14**: p. 129-153.
31. Snavely, N., S.M. Seitz, and R. Szeliski, *Photo tourism: exploring photo collections in 3D*. ACM Trans. Graph., 2006. **25**(3): p. 835-846.
32. Li, R., J. Hwangbo, Y. Chen, and K. Di, *Rigorous Photogrammetric Processing of HiRISE Stereo Imagery for Mars Topographic Mapping*. 2008(Compendex).
33. Akbarzadeh, A., J.M. Frahm, P. Mordohai, B. Clipp, C. Engels, D. Gallup, P. Merrell, M. Phelps, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, H. Towles, D. Nister, and M. Pollefeys, *Towards Urban 3D Reconstruction from Video*, in *3D Data Processing, Visualization, and Transmission, Third International Symposium on*. 2006. p. 1-8.

34. Mills, S.J., M.P.G. Castro, L. Zhengrong, C. Jinhai, R. Hayward, L. Mejias, and R.A. Walker, *Evaluation Of Aerial Remote Sensing Techniques For Vegetation Management In Power-line Corridors*. IEEE Transactions on Geoscience and Remote Sensing, 2010. **48**(Copyright 2010, The Institution of Engineering and Technology): p. 3379-90.
35. Moghadam, P., W.S. Wijesoma, and F. Dong Jun, *Improving path planning and mapping based on stereo vision and lidar*, in *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on*. 2008. p. 384-389.
36. Eisenbeiss, H., *Applications of Photogrammetric Processing using an Autonomous Model Helicopter*, in *ISPRS Commission I Symposium*. 2006, International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences: Paris, France.
37. Rusu, R.B., A. Sundaresan, B. Morisset, K. Hauser, M. Agrawal, J.C. Latombe, and M. Beetz, *Leaving Flatland: efficient real-time three-dimensional perception and motion planning*. Journal of Field Robotics, 2009. **26**(Copyright 2010, The Institution of Engineering and Technology): p. 841-62.
38. Furukawa, Y. and J. Ponce, *Accurate, Dense, and Robust Multi-View Stereopsis*, in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*. 2007. p. 1-8.
39. Li, J., E. Li, Y. Chen, L. Xu, and Y. Zhang, *Bundled Depth-Map Merging for Multi-View Stereo*, in *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 13-18 June 2010*. 2010, IEEE: Piscataway, NJ, USA. p. 2769-76.
40. Tola, E., V. Lepetit, and P. Fua, *A fast local descriptor for dense matching*, in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. 2008. p. 1-8.
41. Jung, I.K. and S. Lacroix, *High resolution terrain mapping using low attitude aerial stereo imagery*, in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. 2003. p. 946-951 vol.2.
42. BAE-Systems. *SOCET SET (R) BAE Systems*. 2011 [cited 2011 June 27]; Available from: http://www.baesystems.com/ProductsServices/bae_prod_nss_socetset.html.
43. Terzopoulos, D., *Regularization of Inverse Visual Problems Involving Discontinuities*. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 1986. **PAMI-8**(4): p. 413-424.
44. Stefanik, K., *Sequential Motion Estimation and Refinement for Applications of Real-time Reconstruction from Stereo Vision*, M.S. in *Bradley Department of Electrical and Computer Engineering*. 2011, Virginia Tech: Blacksburg

45. Short, N., *3-D Point Cloud Generation from Rigid and Flexible Stereo Vision Systems*, M.S. in *Mechanical Engineering*. 2009, Virginia Tech: Blacksburg