A Framework for Human Body Tracking Using an Agent-based Architecture

Bing Fang

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science and Applications

February 23, 2011

Blacksburg, Virginia

Keywords: Computer Vision, Agent-based, Human Tracking

A Framework for Human Body Tracking Using an Agent-based Architecture

Bing Fang

Abstract

The purpose of this dissertation is to present our agent-based human tracking framework, and to evaluate the results of our work in light of the previous research in the same field.

Our agent-based approach departs from a process-centric model where the agents are bound to specific processes, and introduces a novel model by which agents are bound to the objects or sub-objects being recognized or tracked. The hierarchical agent-based model allows the system to handle a variety of cases, such as single people or multiple people in front of single or stereo cameras. We employ the job-market model for agents' communication. In this dissertation, we will present several experiments in detail, which demonstrate the effectiveness of the agent-based tracking system.

Per our research, the agents are designed to be autonomous, self-aware entities that are capable of communicating with other agents to perform tracking within agent coalitions. Each agent with high-level abstracted knowledge seeks evidence for its existence from the low-level features (e.g. motion vector fields, color blobs) and its peers (other agents representing body-parts with which it is compatible). The power of the agent-based approach is its flexibility by which the domain information may be encoded within each agent to produce an overall tracking solution.

Acknowledgments

I am heartily thankful to my committee members, especially to Dr. Quek (my advisor) and Dr. Ehrich, whose encouragement, supervision and support from the preliminary to the concluding level enabled me to develop an understanding of this research. I also offer my regards and blessings to all of those who supported me in any respect during the completion of this dissertation.

This research has been partially supported by NSF grants "Embodied Communication: Vivid Interaction with History and Literature", IIS-0624701, "Interacting with the Embodied Mind", CRI-0551610, and Embodiment Awareness, Mathematics Discourse and the Blind, NSF-IIS- 0451843.

All photos in this dissertation are captured by the author, 2011.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Human-body tracking has a wide range of applications, such as surveillance, motion-capture for animation, video analysis, game interaction and human-computer interaction (HCI) [12, 35, 75]. Generally, there are two ways in which human-body tracking can be performed: one is using a motion capture system, and the other is performing the vision-based analysis [74]. This disertation is focused on the latter (the vision-based analysis) in human body tracking.

In vision-based tracking, problems arise from variations in the environment, the appearance of other human beings and the unintended purposes of different tracking tasks. These problems require vision-tracking to be the target specific. In other words, most vision-tracking algorithms have to work under pre-defined conditions. Otherwise, the reusability of the system is rather limited, or it is very difficult to adapt for use in new settings. One key component of computer vision and machine learning systems is targeted at combining various features, which are extracted with image processing techniques (such as skin-color), for different tracking problems, but it remains an open question to find these combinations. This dissertation is an attempt trying to address the problems like these by creating a more general framework for vision-based human tracking tasks.

1.1 Human tracking

The analysis of sequential images involving humans draws more and more attention in the domains of computer vision and behavior analysis. Various applications, such as Microsoft Kinect, are known to us in our daily lives. With the ever rapid growth of high-performance computers, the evolution of high quality digital cameras, and the increasing need for automatic video analysis and real-time tracking, there has been considerable interest in human tracking. Human tracking research may be targeted at the following applications:

- *automatic surveillance* - monitoring a scene (or multiple scenes) to detect suspicious activities or unexpected events [22, 23, 25, 52, 83];
- *vision-based motion recognition* - human gesture identification based on poses, automatic data registration, etc. [8, 23, 34, 52, 83];
- *video analysis* - automatic annotation and retrieval of the videos [2, 23, 52, 58, 83];
- *human-computer and game interaction* - gesture recognition, eye gaze tracking for information inputs to computers or game stations, etc. [72, 85];

There are three approaches used in human tracking according to the existing research [23, 51, 52]: 2D approach without explicit shape models; 2D approach with explicit shape models; and 3D approach. By and large, there are three key steps in analyzing human tracking in 2D or 3D approaches [52, 83]: detecting the likely object candidates, estimating the properties of the objects in frame sequences, and analyzing the behaviors of the objects.

Since the degree of accuracy of detecting likely objects can affect the efficiency of registering and tracking the human body (or body parts), researchers have developed various approaches to increase detection accuracy, and they also seek a more general feature-detection model to be easily adjusted to various situations for human tracking. However, the feature-detection can be complex for the following reasons:

- noise in images, or the insufficiency of the image processing algorithms,
- complex shape of the target object, or articulated nature of the object,
- partial and full object occlusions, disappearance and re-appearance in the camera view,
- scene illumination changes,
- loss of information caused by projection of the 3D world on a 2D image,
- complex object motion,
- camera motion, and
- real-time processing requirements.

These issues make it very challenging to create a general framework to handle all vision problems, however, the efforts to develop a general approach for the human tracking have never stopped in the research world. Typically, a kinematic model of the human body is employed for human tracking. For example, Ju et al. [34] defined a "cardboard person model" in 1996, and Wren et al. [79] introduced a real-time human tracking model as "PFfinder" in 1997. In these different models, researchers have developed a variety of hypothesis. For example, PFinder hypothesized that the upper arm, lower arm and hand can be treated as one object, which is represented with a ellipse in the full body tracking to simplify the tracking problem. However, it is still impossible to avoid the following questions in human tracking:

- *Which representation of the object is suitable for tracking human body (part)?*
- *Which features (in both low level and high level) should be used for tracking analysis?*
- *Which model is more suitable to represent the motion, appearance, and silhouette of the target?*

The variations in context and environment of different tracking algorithms make it difficult to answer these questions. Moreover, different goals of the tracking problems (such as tracking the full body gesture or the hand gesture only) lead the researchers to face different problems and thus call for different approaches to resolve the problems. For example, we are more concerned about the location of the human in most surveillance video analysis as they do not require very accurate gesture tracking. But, for human computer interaction analysis, people are more concerned with the accuracy of the gesture of each interactive subject.

1.2 Agent-based approach and its motivation

Having introduced the complexity in vision-based human tracking, we come back to discuss the question, “Can we find any solution so that we can use a more general framework to deal with protean vision-tracking problems?” The answer to this question is “Yes and No”. The reason for ‘No’ is that it is fairly hard to find/build a general model to cover every case in the complex world. However, it is possible to find an alternative solution to this problem, which is to build a flexible model to allow users to switch between different cases. The agent-based approach is one of these solutions, which provides a means of modeling that distributes the complexity of a particular problem domain across a set of agents, thereby limiting the complexity within each intelligent agent [2, 14, 47]. These will be explained in greater details in the following chapters.

One of the obvious advantages to employ a generic agent-based architecture for human tracking for the business, game simulation, remote military control and etc is that it helps decompose the complex problem into simpler sub-problems.

Agent-based approaches have been extensively studied and the typical approach is to model the processing components as agents (e.g., edge detection agent, feature matching agent, color segmentation agent, labeling) [6, 30, 62, 64, 87], to assign different areas in the visual field to specific agents [63], or to model the entire user interface as an ‘interface agent’

[64]. The main thrust of this research is to focus on building such a framework using an agent-based architecture. In a human tracking application, researchers normally treat the complex human skeleton in a hierarchical structure [13, 30, 34, 79]. Generally, the focus is on extracting features, such as detecting edges and color blobs, from video frames first. Then, they address higher level questions like "how to construct a hand from the collections of lower level features", "which kinematic model is proper to construct a human being", etc. This approach leads us to build a hierarchical structure for our system, which employs low-level agents and high-level agents. The low-level agents are assigned specific processing tasks to extract low level image features. Thus, these agents can be implemented independently, and parallel processing strategies may be applied to improve the performance of the system. The high-level agents are designed to produce higher level processes, such as creating predictions for body parts, seeking detection results from body parts, finding relationships between these body parts and building full body models etc. A flexible framework of high-level agents considers individual processing, communication, information sharing and decision making. Our main contribution to this research is to develop the framework which provides a basic agent framework that users can extend easily for different objectives.

An accurate feature extraction is not always available for various reasons, such as noise and low resolution images. Based on our research [18], we have shown that it is possible to obtain more accurate tracking results by using our framework, which provides useful feedback among different hierarchies, even if we can only obtain low-quality features. Thus, one of the goals for this research is to improve the accuracy of the tracking results for human body (parts).

The prediction from the high-level agents provide feedback to help improve the quality of low-level features, and good quality features support a better tracking result of the high-level agents. We also evaluate our framework by applying it to various data (the meeting-room video data and the sports game video data). The evaluation considers the accuracy of the tracking results and the flexibility in development-time and run-time.

1.3 Contribution of this research

In computer systems, a framework is often defined as a layered structure indicating what kind of programs can or should be built and how they would interrelate. In this research, our designed framework is embedded with agent-based architecture, and it is a hierarchical structure indicating what kind of agents should be built and how these agents would communicate with each other. The detailed description of this framework is introduced in Chapter 3. Within this framework, we contribute the following:

- We develop an active agent-based architecture for human tracking. Generally, an agent based system confining the complexity across a set of agents, thereby limiting the complexity within each intelligent agent. However, existing approaches either assign different agents to different image processing areas or serve as computing resource schedulers [30, 62, 87]. A more comprehensive review of such agent-based approaches will be provided in Chapter 2. Our agent is designed to be an intelligent unit to represent a high-level abstraction of real objects, such as a hand, instead of just a processing unit. This architecture actually helps a developer implement a meaningful agent, and increases the reusability of developed agents.

- We also employ the job-market model to form the coalition of agents. Typically, there are two ways of tracking objects: top-down (model-based) and bottom-up (feature based). However, either approach can be broken easily because of the single direction searching. The job-market model works from both sides (employer and employee), and this approach provides a possibility that both sides can 'correct' their prior prediction during updates. Details will be discussed in Chapter 3.

1.4 Research questions

For the purpose of this research, we have investigated some important issues regarding human tracking systems. They are 1) the range of application of this agent-based tracking system, 2) the design of the agent-based system, 3) the vision tracking approaches being implemented in the system, 4) the vision-based tracking problems being resolved under the system and 5) evaluation of our tracking system.

1. *What range of applications is amenable to this solution?* Surveillance, video-based motion analysis and human computer interaction are three main applications for vision based human tracking analysis. As the starting point, we focus more on video-based motion analysis, and meeting-room type video data, which will be used to examine the efficiency of our system. To evaluate the effectiveness of our system, we will compare

the visual results. Another valuable question to answer is "How much adjustment of the architecture is needed to adapt to each problem?" which will provide the evaluation for the effectiveness and efficiency of this research product. We also apply our system to track players in a sports game video to demonstrate the use of our system.

2. *What is the model of the proposed agent-based architecture? How does one design a flexible framework for variety of human tracking problems?* The second and the most important question for this research is how to design our agent-based human tracking system. As we mentioned above, our design employs a hierarchical structure with low-level agents and higher level agents. Each agent processes individually and co-operates with one another to accomplish the common goal. The detailed design will be shown in Chapter 3. Generally speaking, our agent-based system considers several issues, including individual processing, communication, information sharing and decision making. Low-level and high-level agents are different because each has its own characteristics as follows:

   (a) *low-level agent*. In our system, a low-level agent is designed to extract certain features within the current environment, and generate information for later use. In human tracking literature, a variety of feature extraction approaches is employed. In this dissertation, we need to make clear which feature extraction approaches are going to be used for our low-level agent, and what information should they provide for high-level agents. These agents do not communicate with each other. Instead, they are just processing individually. The extracted features are stored in shared memory that all high-level agents are able to access.

(b) *high-level agent.* A high-level agent represents a high-level abstraction of a real object. For example, when a human agent employs a hand agent, the human agent only knows that it needs a coherent hand agent to prove its existence, and it does not need to know whether this hand agent uses skin-color blob or other features. In our system, these high-level agents seek low-level features to match their predictions, and also provide feedback to low-level agents to ask them to improve their results (if necessary). The high-level agent also builds coalitions with other high-level agents to combine their information as part of the final analysis results. In this dissertation, we provide a framework to guide people develop different agents for different tracking purposes (such as the hand tracking and the head tracking) and organizing these agents together to accomplish a common tracking goal (such as the full body tracking).

3. *What vision algorithms are employed in the system for this flexible use?* In our system, low-level agents are embedded with vision/image processing algorithms to extract various features from raw image sequences. We introduce several approaches to extract features, including adaptive skin-color model and vector coherence mapping (VCM). Our framework also allows developers to create other low-level agents using different vision/image processing techniques.

4. *What kind of vision-based tracking issues can be solved?* The ultimate goal of this research is to build a general framework for vision-based tracking, so we need to exam-

ine the following issues, which are not all resolved properly by any individual existing tracking system:

- switching tracking model for different tasks or different tracking targets,
- partial and full body-part occlusion,
- change of the environment (e.g., illumination, moving camera), and
- disappearance and re-appearance of the same body (or body part) from the video.

1.5 Dissertation organization

We first review the literature of vision-based human tracking in Chapter 2. In this chapter, we also review the literature of the agent-based approaches and existing applications for computer vision and human tracking. We will also discuss our earlier design of a hierarchical architecture for agent-based tracking [18, 19].

In Chapter 3, we introduce our agent-based framework for human tracking problems. Using an agent-based architecture for human tracking is a new topic in computer vision, and we will discuss our agent-based system design for human tracking.

In Chapter 4, we demonstrate the use of our agent-based framework by implementing a system to track human subjects in meeting-room videos.

In Chapter 5, we discuss about the general use of our system and demonstrate a tracking of players on a soccer field.

In the final chapter, Chapter 6, we will discuss the framework and possible future directions beyond this research.

Chapter 2

Review of vision-based human tracking literature

A common approach for human tracking is to detect the features of targeted objects through foreground segmentation and to track and recognize these objects by establishing correspondences in a sequence [23, 52, 84] of consecutive frames. In this chapter, we will review some commonly used vision approaches for human tracking and review the history of using agent-based approaches for human tracking.

2.1 Human tracking literature

Generally, human tracking can be categorized by the functional taxonomy presented in the survey papers by Moeslund and Granum [51, 52] as follows:

- *Model initialization*, which provides an initial interpretation of the current scene for subsequent processing.
- *Tracking*, which extracts features, segments and tracks humans in consecutive frames.
- *Pose estimation*, which predicts the human pose based on the extracted features in consecutive frames.
- *Recognition*, which recognizes the identity of a human and his behaviors in consecutive frames.

In this dissertation, we focus on the procedures for initialization, tracking and pose estimation.

2.1.1 Model initialization

Vision-based human motion analysis requires the definition of a humanoid model approximating the shape, appearance, kinematic structure, and initial pose of the individual to be analyzed [52]. The initialization determines the prior information about the individual or individuals being tracked. It also requires information be used to constrain the tracking. In earlier research, a typical approach is to use manual initialization for the length, shape, color, and kinematic structure of limbs [1, 40, 54]. A few researchers have also investigated algorithms for automatic initialization (two notable examples can be seen from [36, 37]). The approach to employ model initialization has a major limitation when the model changes during processing. Only limited research has addressed this problem, and a fully automatic initialization of kinematic structure, shape and appearance remains to be developed by new research [52]. In this section, we will review the work on the kinematic structure, shape and appearance initialization, which is directly relevant to our work.

Kinematic structure

A humanoid kinematic structure, which is comprised of a fixed number of joints with specific degrees-of-freedom (DOF), is widely employed in most human tracking systems. Hard-coded constraints, such as skeletal symmetry and anthropometric constraints, are commonly used in a variety of applications. Some approaches, such as [56, 73], have addressed the initialization of body pose and limb length by manually identifying the joint locations. Krahnstoever et al. introduced an automatic initializing method for upper-body kinematic structure based on motion segmentation in [36, 37]. Another recent trajectory of research, such as [7, 50], uses motion capture data from commercial marker-based systems to train the prior kinematic model, including limb length, DOF of each joint, and so on. Motion capture databases have been used to reconstruct three-dimensional poses from image sequences using a priori learned correspondence between image and pose in databases [73].

Shape

Representations of the humanoid model have employed simple shape primitives, such as cylinder, cone and ellipsoid, using the kinematic skeleton [51, 52]. Ju et al. [34] presented a cardboard person model in 1996 to track human bodies. In this approach, a set of connected planar patches were used to represent the limbs of the human body. This method performed vision-based recognition of human activities using optical flow. The PFinder was another famous human tracking approach introduced by Wren et al. in 1997 [79]. This approach employes a Maximum A Posteriori Probability (MAP) to detect and track the human body with simple two-dimensional models. This two-dimensional model uses ellipsoidal blobs for the kinematic human body skeleton.

Since the two-dimensional skeleton changes because of the perspective of the camera view, stereo calibration has been used to achieve more accurate and stable shape and appearance in three-dimensional space. Carranza et al. [4] presented a generic mesh model to detect multiple view silhouette images of a person with a set of pre-fixed poses to track the whole body motion. Plänker et al. [57] employed an implicit ellipsoidal meta-ball representation to fit the upper-body prior to tracking. These model fitting approaches have provided a more accurate prediction for human body initialization within three-dimensional space.

Appearance

Variations in human body appearance, such as different clothing, call for variability in vision-based tracking. Statistical color models are commonly used in human tracking [52]. The color models for skin and cloth texture are also discussed in existing research, such as [65, 66, 80, 82].

Sidenbladh et al. [65, 66] presented an approach that modeled the likelihood of image observations for different body parts. The statistical model of the appearance and motion is learnt based on a set of training images. A Gaussian color model, such as skin-color [82] and other specific colors [19], is typically used to generate various color blobs. A *color blob* is a cluster of pixels that are masked by the same color model. The pixels in the same blob have to be spatially coherent (typically, connected component labeling approach is applied to the masked pixels, and a circle or an ellipse is used to represent the component). A multivariate Gaussian distribution model [82] was also employed to extract modeled color pixels. However, this statistical model is limited because the model is pre-trained under known lighting conditions. Thus, lighting changes may potentially break the model. An extended research of an adaptive skin color model was introduced by Xiong et al. [80]. Although this model was introduced to produce skin-color blobs, we proved that it can be used for other simple color blobs as well [18, 19].

2.1.2 Tracking

Tracking in visual analysis has been discussed in the literature of computer vision research [9, 17, 58, 80]. In this dissertation, as per the discussion by Moeslund et al. in 2006 [52], we consider tracking as a two-step process: *foreground feature extraction* and *temporal correspondence*. Feature detection methods focus on separating the objects of interest, which normally target individuals or focus on objects, from the rest of the image, which consists of background or motionless objects. These methods also construct the representation of these foreground interests using existing models, such as the skin-color model. The temporal correspondence approaches focus on finding the coherence between features in the current frame and the processing results from previous frames and providing temporal trajectories.

Background subtraction

Background subtraction is a powerful and commonly used processing approach in controlled indoor spaces [52]. Stauffer et al. [69] presented a *mixture of Gaussian* (MoG) model which represents the distribution of pixels by a MoG and updates each pixel with a new Gaussian at run-time. This approach allows us to employ background subtraction with outdoor environments, and it has become a standard method for background subtraction. However, researchers, such as [17], have improved the original approach by developing new method for background initialization, representation, updating and classification.

The MoG representation can be applied in RGB, normalized RGB, YUV and HSV space [10, 38, 46, 79]. In earlier background subtraction approaches, learning the background model assumed that there is no moving object in the initial frames, and moving objects were only admitted after learning the background model. In this model, each foreground object is described by an individual distribution to be distinguished from the real background. Other approaches, such as [17, 27], also try to find 'true background' pixels by applying a *temporal median filter* to the frame. Eng et al. [17] employed a skin-color detector to remove the human when initializing the background. By measuring the overall changes to the expected background in the current frame, the changes of the background can be calculated, and the

MoG model for the background can be updated continuously and automatically at run-time. When classifying the pixels in each frame, research, such as [10, 24], has demonstrated a direct classifier to separate each pixel into *unchanged background* and *changed background*, which is caused by moving objects, shadows, and so on [52]. This classifier is mostly built with color, gradients and thresholding techniques [27, 80, 81].

Motion-based segmentation

Motion-based segmentation approaches are built based on finding the difference in consecutive images to detect the moving objects in the scene. Typically, the difference is measured using optical flow, motion vector extraction, and image differencing [29, 32, 58].

Research, such as [29, 32, 34], observes the flow vectors in consecutive frames. Since the optical flow can be noisy, the detection results are then smoothed with temporal and spatial coherence. The optical flow algorithm is robust by handling image sequences that are quantized coarsely in space and time domains.

Quek et al. [2, 58] presented a *vector coherence mapping* (VCM) algorithm, which extracts motion vectors from the source video and clusters them into spatially and directionally coherent vector clusters. This approach incorporates various local smoothness, spatial and temporal coherence constraints transparently by the application of fuzzy image processing techniques to compute an optical flow field (vector field) from a video image sequence.

Image differencing approaches are simple and can adapt quickly to changes in a scene. However, it is hard to detect the pixels of the human body, which are similar to their neighbors. Some researchers [76] have reported that comparing a rectangular area of pixels in consecutive images and calculating the shift of the rectangle can improve the results.

Shape-based segmentation

Shape-based segmentation approaches are commonly used because of the uniqueness of the silhouette of the human body with respect to other objects. The advantages of shape-based segmentation for human tracking are that it supports human detection and tracking in uncontrolled environments, and it also allows detection of the human body in still images [52]. However, similarities of the silhouettes of humans makes this approach difficult to use to distinguish different individuals.

Zhao et al. [86] presented a method for extracting the silhouette of the human body by employing a neural network to train the model of humans' silhouettes. This model represents the extracted silhouette of the human body, and it is used to determine whether a silhouette belongs to the original individual or not at run-time. Other research also employs other statistical models, such as Markov models or Bayesian networks, to detect silhouettes for human tracking [51, 52]. By considering the temporal context in processing, shape-based segmentation can be applied to track people over time.

Appearance-based segmentation

Appearance-based segmentation approaches are powerful and benefit from the fact that the appearance of humans is very different from the background, and the appearances of different individuals are probably unique with respect to each other. Some approaches, such as [80, 81], using appearance-based segmentation employ appearance models and examine each pixel in frames to decide whether the pixel belongs to the model or not. Other approaches, such as [49, 55], have built appearance models both for foreground subjects (human being) and the background, then register each pixel in consecutive frames into each model. However, appearance models suffer from similarities between foreground and background. As we discussed in the previous section, the appearance representation might change if the lighting condition changes.

Xiong et al. [80] presented an adaptive skin-color model for meeting room activity analysis. In this approach, the authors trained a Gaussian model to represent skin colors with a set of training samples. The model was used to classify the pixels in the current frame into skin-color or non-skin-color pixels. The skin-color pixels were then used to update the model for the subsequent processing. In our research [18, 19], we extended the model to extract other color blobs.

Depth-based segmentation

Shape-based and appearance-based segmentation have limitations when the models of foreground objects are similar to the background (appearance-based segmentation) or are similar to each other (shape-based model). Depth information can also help build background models to reduce the effect of lighting changes [52]. Research, such as [26, 41], used depth information from multiple cameras, which are driven by high-performance computers. This approach, using multiple cameras, calculates the features with each single camera view and registers these features to the shared coordinates (2-D to 3-D space). The registered features identify the tracked object.

In our research [19], we employed Tsai's calibration model [74] and captured moving subjects with two stereo cameras. Features extracted from each individual camera view are registered into a shared 3-D space. The 3-D space analysis was then used for tracking body parts and other analyses.

Temporal correspondences

Temporal correspondences are mainly used in case of occlusion. Tracking algorithms are used to establish the connection between prediction and measurement. Moeslund et al. emphasized in [52] that "One of the primary tasks of a tracking algorithm is to find the temporal correspondences. That is, given the state of N persons in the previous frame(s) and the current input frame(s), what are the states of the same persons in the current frame(s).”

Occlusion problems occur when new objects appear, existing objects disappear, objects merge together, or merged objects split. Previous research has been focused on developing approaches for solving these problems [52]. Quek et al. [58] analyzed the motion vectors in the current frame and fused them using temporal coherence with previous frames. The Kalman Filter [77] and the Particle Filter [13, 44] are employed to track multiple objects in the same scene.

2.1.3 Pose estimation

Pose estimation is used to estimate the kinematic structure of a human body during tracking. Pose estimation approaches can be classified as *model-free* and *model-based*.

Model-free approaches

Model-free approaches have no a priori model in processing. These approaches are frequently used for pose estimation in two-dimensional space, because the kinematic structure in 2-D space is unpredictable due to perspective effects. We can categorize model-free approaches for human tracking into two kinds of approaches [52]: *probabilistic assemblies*, and *example-based approaches*.

Probabilistic assemblies methods perform a direct bottom-up 2-D pose estimation. First, they detect all the possible body parts and assemble them. Then, the best matches between these assemblies and observations are kept. Felzenszwalb et al. [20] presented an efficient matching approach using pictorial structures to estimate the 2-D body poses in image sequences. Ioffe et al. [31] also introduced a pictorial structure, which they called a *tree* to track human body poses in 2-D image space.

Example-based approaches learn the mapping from 2-D to 3-D space by training with existing databases. Rosales et al. [59] estimate 3-D poses by learning a mapping from visual features of a segmented human body to a static pose using neural networks. This approach will not be affected by the speed and direction of individual motion.

Model-based approaches

An explicit model of kinematics, shape and appearance in the analysis-by-synthesis framework is widely used for human pose estimation in video processing [13, 57]. Based on the survey by Moeslund in 2006 [52], the model-based approach is a dominant methodology for human pose estimation. The main novel research directions include using stochastic sampling techniques with sequential Monte Carlo methods and applying constraints, such as the physical model, to form a particular learned human motion model.

As described in [57], the extended Kalman filter was applied to human tracking with an extended deterministic gradient descent-based approach for complex human motion. However, the gradient descent-based approach suffers from using single pose or state for estimation, because the pose or state is changing rapidly over time. The Particle filter [13, 44] was introduced as one of the stochastic tracking techniques for robust visual tracking, where sudden motion or a cluttered scene might cause tracking failure. As multiple camera views are widely used in the model-based algorithms to estimate the 3-D pose of human body [3, 19, 35, 68], a combination of deterministic and stochastic approaches have been used for sampling and searching in 3D pose estimation from multi-camera views.

2.2 Existing agent-based approaches

Agent-based approaches have had a long history, dating back to the 1970's when they were first introduced within the context of distributed artificial intelligence [16]. Since then, agent-based approaches have been applied in a wide variety of domains, from business to game simulations to computer vision architectures [2, 14, 47]. An agent is an intelligent unit, which uses embedded knowledge to reach its own goal and has the ability to react with the changing environments [47]. An agent-based system is constructed by these agents and should support communication between them. Key aspects of agent-based systems are the agent system architecture, inter-agent communication, and agent design [21, 47, 70, 71, 78].

One way to think about agent-based approaches is that they provide a means of modeling the distribution of the complexity of a particular problem domain across a set of agents, by which it limits the complexity of each agent. Thus, the solution is then divided across the agents and the communication among them. Agent-based systems have often been compared with standard object-oriented approaches, which also decompose complexity by a process of encapsulation within objects and object interfaces. The key difference between object-based and agent-based systems is that the agents are active and autonomous entities, which distribute both the process and the representation. However, object-based systems distribute representation and maintain centralized processing.

2.2.1 Agent-based system for computer vision

Agent-based approaches have been applied in computer vision and human-body tracking for the past ten years [2, 6, 30, 43, 62, 63, 64, 87]. The typical approach is to model the processing components as agents (e.g., edge detection agent, feature matching agent, color segmentation agent, labeling agent) [6, 30, 62, 64, 87], to assign different areas in the visual field to specific agents [63], or to model the entire user interface as an 'interface agent' [64]. Lukenhaus and Eckstein [43] describe a novel application of agents to support parallel processing by which the generic agents apply a set of operators to solve the image processing task. The agents essentially serve as computing resource schedulers. Hence, in a more general sense, the agent represents the processing operators that they schedule.

The Schema System was introduced in early research [15]. In this approach, a low-level schema was used to extract low-level image processing features. An intermediate schema was defined to track an object, such as the sky and the road, in each frame. In this research, the authors discussed that assigning an intermediate schema to track objects provides a more general use of the system. Bryll et al. introduced the design in [2], and we improved on the model in [18, 19]. Our approach departs from this process-centric model, where the agents are bound to specific processes, and introduces a model, by which agents are bound to the objects or sub-objects being recognized or tracked. Our approach provides greater flexibility using this decomposition by objects, since the agents can apply very specific operations, and features can be adapted to a variety of sensing environments. In this dissertation, we follow the work by Bryll et al. [2]. The difference is that [2] takes a rigid bottom-up approach, where features (blobs) become labeled as specific body parts, and the combination of body parts are constructed that satisfy certain relational constraints. The problem with this approach is that if the low-level features are erroneous (e.g, blobs are merged or fragmented), the agents are unable to recover from the error [19].

Minsky [48] introduced the frame systems that are constructed with related frames, where a frame is treated as a network of nodes and relations. In this system, the top-level of a frame is fixed to represent things, and the low-level of a frame has many terminals to represent data. Based on the earlier discussed schema concept, the frame system specifies the make-up of a scene in terms of it's constituent elements, and it then applies a top-down labeling and constraint satisfaction to match scene features to the frame schema. Our approach is similar in that it is hierarchical, but it is not the same.

Rosenfeld et al. [60] introduced a relaxation approach for maximizing global consistancy by applying iterative parallel operations (i.e., relaxation operations) to the global area (the whole image). A fuzzy model was used to decrease the weights on each object, thereby eliminating the ambiguities. Research, such as [11, 61, 88], extended this approach to find a local maxima for applications. In the relaxation approach, Rosenfeld et al. developed a non-linear method, which increases the confidence in a target and decreases the possibility of the ambiguities by using predefined correlations among objects. More details of this approach are discussed in Chapter 3. In the approach described in this dissertation, the body-part agents are abstractly defined, and they seek evidence for their existence and location by examining the low-level features with their predictions. Hence, specific knowledge is embedded within each agent, and the knowledge about the entire body being tracked is embedded within the human agent that models the relationships among the body parts. In our research, we introduce a job-market model to not only decrease the weights of incoherent objects, but also increase the weight of coherent objects.

2.2.2 Preliminary research on agent-based human tracking system

In our previous work [18], we first introduced our hierarchical agent-based framework to track a subject's hands and head in a two-dimensional space (with a single camera). The framework employed several simple agents and a three-layered hierarchy. The focus of the study was the agent-as-tracked-object framework itself, and the results represent a proof-of-concept of our hierarchical agent-based approach. However, since the system operated in a two-dimensional space, we did not employ 3-D constraints of the human body in the tracking, which would definitely improve the efficiency of each agent. We further supported these findings in [19], where we extended the system to three-dimensions by introducing a 3-D skeleton-based model and employing a two-camera stereo system. Multiple-camera techniques are frequently employed to solve the problem of self-occlusion and ambiguous image segmentation [2, 5, 33]. The system tracked a set of joint locations in the stereo data, assigning an agent to track each individual joint. Each of the joint agents knew how to seek evidence for their location in the stereo video, and it also understood its physical constraints with respect to its neighboring joint-agents. For example, elbow-joints can only bend in one direction (one degree of freedom). To simplify the *evidence-collection process*, we used a special suit with markers attached to each joint, such as wrists, elbows and shoulders. By applying our system, we were able to estimate full-body motion in 3D space by using two standard digital cameras.

2.3 Summary of vision-based human tracking with agent-based approach

Vision-based tracking can be discussed as a processing structure: initialization, detection/registration and tracking/estimation. Thus, each agent should be capable of these processes.

Initialization in vision-based tracking systems usually provides a correct interpretation of the scene when the system is initialized. Normally, to initialize the analysis requires the definition of a human body model, including features such as shape, appearance, kinematic structure and initial pose to track [52]. Ideally, an agent should be able to self-start automatically.

In feature detection processing, pre-modeled features are extracted with certain extraction techniques as we discussed in the previous sections, and objects (or candidate models) are formed with these features. These candidates are registered to existing objects, new objects may be created, and objects that disappear from the view may be deleted at this stage. In existing agent-based systems, sharing information between agents, self-updating, and even re-processing issues are not fully resolved. This provides us motivation to build such a system to enable these functionalities.

Normally, human tracking does not only rely on the feature detection because of noise, or lose-detection for certain reasons, such as occlusion. Moreover, using the detection result directly might also cause jitter (noise from image processing), which is a common problem in vision-based processing. Using tracking and estimation approaches, such as trajectory-based estimation, can help smooth the result, and it can also solve the problem when the system fails to find any detections in the current frame. However, the tracking/estimation may also cause problems in that the error may be cumulative. In agent-based systems, creating new agents and updating existing agents must be addressed when building an intelligent system for human tracking.

Chapter 3

Agent-based framework for human tracking

As discussed in the previous chapters, human tracking is a complex task. There are various approaches that have been developed and used for human tracking. Moreover, we also discussed the existing use of agent-based architectures for various vision-based human tracking tasks. In this chapter, we introduce our novel agent-based framework, which simplifies this complexity and facilitates the development of systems for human tracking.

3.1 A framework for agent-based human tracking

Generally, a *framework* is defined as a real or conceptual structure intended to serve as a support or guide for the building of something that expands the structure into something useful (*The American Heritage Dictionary*). In agent-based systems, a framework is often being used as a layered structure indicating what kind of agents can or should be built and how they would interrelate [39, 45, 53]. Our framework provides such a structure supporting the development of different intelligent object agents and their coalitions. We also provide a recruiter/ advertiser agent to assist with communication among agents. Thus, we introduce our framework on what kind of object agent should be included and how these agents would collaborate with each other.

In our agent-based framework, we distribute the complexity of human tracking into agents through multiple layers, and an agent will only function in its layer and collaborate with agents in connected layers. For instance, *low-level agents* carry out basic vision-feature extraction while *intermediate agents* request a proper combination of the low-level agents to track a sub-object of a human body. The intermediate agents collaborate with each other to form a high-level coalition, which combines these correlated intermediate agents into one body or a part of a body. In this framework, a coalition is defined as a collection of agents where an agent looks for another coherent agent (other agents) to join together for the same purpose, and the confidence level of these agents increases as they are grouped by strong coherence among themselves. More details of these components in this framework will be discussed in this section and the following sections. In vision-based tracking, this coherence can be temporal coherence or spatial coherence. The *high-level agent* uses a coalition or coalitions of intermediate agents as evidence of its existence to track the targeted human subject.

Before introducing the design of our agent-based framework, we review the key properties of an agent-based system. In any agent-based system, an agent is designed to be an intelligent unit that uses embedded knowledge to reach its own goal, and this agent is capable of reacting to the changing environments. An agent is a computational entity which processes the following properties [47]:

- *Autonomy*: agents are capable of self-starting, independent processing and making decisions without any intervention;
- *Adaptation*: agents are able to change their behavior with the changing of conditions of the environment;
- *Sociability*: agents have the ability to interact and communicate with each other, which provides a method to communicate with different agents.

In this section, we discuss our agent-based approach, including the *Existentialist model*, which drives the agent's main processing, and *Job-Market model*, which helps agents build coalitions with each other. The existentialist model focuses on building an autonomous and adaptive agent, and the job-market model intends to control the sociability of agents.

3.1.1 Existentialist Model

In our framework, an object agent represents an object being tracked instead of the processes that track things. Object agents inhabit a hierarchy from low-level features, to tracked objects, to coalitions of other agents. The features can be low-level processed results, such as skin-color blobs. In this dissertation, a *color blob*, which is represented as a rectangle, means a group of spatially coherent pixels. The pixels in the same blob must have the same label by applying connected component labeling approach. The color blob has two features: the position of this cluster, which is the geometric center of pixels in this cluster; and the size of the cluster, which is the total number of pixels covered in this cluster. The features can also be motion vectors, the tracked objects can be objects in the real world, such as a moving hand or a static head, and the correlated object agents combine together in representing the tracked objects. The coalition (or a combination of coalitions) of object agents representing required body parts represents a targeted human body. We call this an 'existentialist' model because each object agent is constantly seeking evidence to justify or prove its own existence from the feature space, and the high-level agent is also constantly seeking evidence to re-organize the combination of the coalitions formed by intermediate agents. In the process of initialization, each agent seeks such evidence by identifying feature groups that support its presence. The evidence might come from:

- low-level features. e.g., a skin-color blob and the coherent motion vectors around it becomes an apparently strong evidence for the existence of a hand agent;
- peers that are consistent with the agent. e.g., a hand having a strong coherence with a head becomes a strong evidence that they belong to the same person, and it is also an evidence that the person exists. Meanwhile, both the hand agent and the head agent also find evidence for their own existence through this peer coalition;

- higher-level coalitions in which it participates. e.g. an unproven hand candidate can prove its existence by finding a strong coalition with an existing human agent.

Individual agent

Since object agents represent objects being tracked, each agent has to be embedded with prior knowledge of the pre-defined characteristics of the target object that it represents. For example, a hand agent represents a real hand in the world, and it knows that a spatial skin-color blob and coherent motion vectors around this blob are necessary features, while a human agent knows that it has one head and two hands.

When positive evidence is found, the agent employs the evidence to support its existence and starts to communicate with its peers and higher-level agents to search for support. (We will discuss our communication model in Section .) For example, an existing hand agent (we will discuss about the creation of agents when we describe the agent at each layer) looks for a skin-color blob with coherent motion vectors around it to provide evidence that it is possibly a hand. Then, this hand agent starts to look for supports that include:

- *spatial proximity.* E.g., a hand agent receives support from a human agent close by, but it won't receive any support from any human agent far way.
- *temporal coherence.* E.g., a hand agent receives more support if it has higher temporal coherence with a proved hand in previous frames.

If the agent receives enough support from its peer agents, the likelihood of its existence

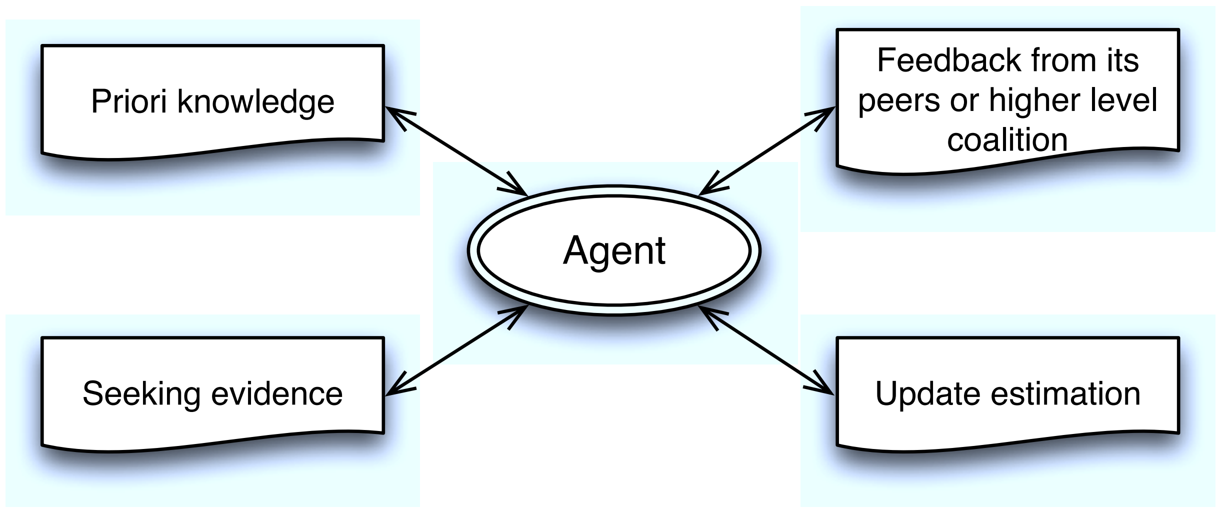

Figure 3.1: Individual intelligent agent.

increases, and an inter-connection is built between this agent and the agents that provide supports. However, an agent might compete with other agents for limited positions. For example, a human agent normally only allows two coherent hands in each frame, but there are possibly more than two hand agents that want to build a coalition to take the positions from this human agent. In this case, these hand agents have to compete for the two positions. If an agent is beaten by another agent, the winner will take the position, and the others have to search for other opportunities or look for more evidence to win.

An object agent might also receive feedback from higher level agent. For example, a hand agent employs a skin-color blob as the evidence of its existence, but the spatial or temporal prediction of the hand agent and the detection result of the skin-color blob might conflict. Thus, the hand agent will ask the skin-color blob agent to update its detection results. Theoretically, the interpretation of a real hand and the updated detection result from the skin-color blob should converge after several feedback-update iterations by removing those ambiguities that constantly have conflicts. But, noise or other ambiguities in vision processing may cause problems. Figure 3.1 shows the capability of an individual agent in our framework.

Coalition

A coalition can be treated as an action or process of joining agents with others for a common purpose. Thus, the coalition in our framework is defined as a process that object agents, such as a hand agent and an arm agent, actively join together to track a part of a human body. By joining together, the evidence of existence of each object agent in this coalition is strengthened, thereby increasing the possibility of being used by a higher-level agent. However, the confidence of existence of an object agent will decrease when leaving the coalition, which is similar to the negative correlation in Rosenfeld's relaxation process [61].

All the object agents in a coalition should have obligations that:

- each agent should constantly seek evidence of its existence thereby increasing the confidence of the whole group;
- agents should be able to take feedback from their confederates and reprocess to enhance the support of other members;
- agents should be responsible to protect their confederates by preventing any ambiguous object agent from joining the coalition;

- agents should be responsible to remove any member that cannot contribute to the coalition and find other agents for reinforcement.

In this framework, an object agent forms a coalition with other object agents:

- when there is no coalition or no proper coalition to join:
  - it seeks for another temporally and spatially coherent agent (other agents) that does not belong to any coalition to join together;
  - if no such agent exists, it will keep seeking and might be used without a coalition.
- if there is a coalition that is suitable for this agent:
  - it applies to join the coalition;
  - all the members already in the coalition vote to decide whether to accept or to decline this application;
  - the decision is positive if : (1) there is no conflict with members in the coalition; and (2) the confidence of members in the coalition increases by accepting this agent.

An agent might leave its current coalition when the others are not being used by the higher level agents, because it cannot receive any positive support from its peers in the coalition. An agent will stay in the coalition when it is beneficial to have a high priority to be used by a higher level agent, or to increase the possibility of being used by preventing ambiguities joining the coalition. A stable coalition will not have its members leaving or new members joining into frequently. However, members in a coalition may replace one member by employing a better applicant, when:

- the member in the coalition is not being employed by the higher level agent;
- the applicant with the same type was employed by the higher level agent;
- the coalition might break if no one is employed;
- an individual agent does not have a consistent tracking result (normally noise).

The communication between agents is assisted by the job-market model that we will discuss in Section 3.1.3.

Low-level agent's processing

A low-level agent is designed to extract features directly from a video frame. For example, a skin-color agent clusters all spatially coherent skin-color as its evidence of existence. Typically, low-level agents are initialized or pre-built before processing starts. The number of agents and their processing algorithms are predetermined. The low-level agents do not communicate with their peers in the same layer except during initialization. Figure 3.2 shows the processing component of a low-level agent. Generally, a low-level agent seeks results from image processing (such as color filtering) as its evidence and use its embedded a priori knowledge to provide detection results (such as a color blob). This agent waits to be used by an intermediate agent. If it is selected by an intermediate agent, the intermediate agent provides feedback that consists the spatial and temporal prediction from this intermediate agent. The selected low-level agent uses the feedback as a new constraint to reprocess in the local area, which includes its detected area in previous iterations and the predicted area provided by the intermediate agent. The reproduced detection result is updated as the new detection result of this low-level agent, and the new detection result is sent to the intermediate agent that hires it.

Rosenfeld et al. [60] introduced an iterative processing approach that maximized the global confidences in assigning labels to image features. In this relaxation approach, a non-linear operator was defined as:

$$p_i^{k+1}(\lambda) = p_i^k(\lambda) + q_i^k(\lambda) \tag{3.1}$$

where $p_i^k(\lambda)$ represents the probability of labeling the i^{th} object with label λ at the k^{th} iteration. q_i^k represents an update based upon confidences in neighboring objects and their correlations. In our approach, q_i^k is the feedback. In Rosenfeld's work, a pre-fixed pairwise correlation between objects is defined, but estimating these correlations and initial probabilities is a known problem with Rosenfeld's work, as is the enormity of the relaxation computation. The correlations between object agents in our approach are binary. For example, a hand and arm may be correlated, and an arm and torso, but not a hand and a head. The low-level agent can only be used by an intermediate agent with which it is correlated. Being employed by an intermediate agent provides a positive value of q_i, otherwise, negative. Positive feedback enhances the agent to keep updating based on current evidence. We will show an example in which a skin-color agent updates its result when it receives feedback from a hand agent in next Chapter. When an agent receives negative feedback, it seeks other available evidence locally to update.

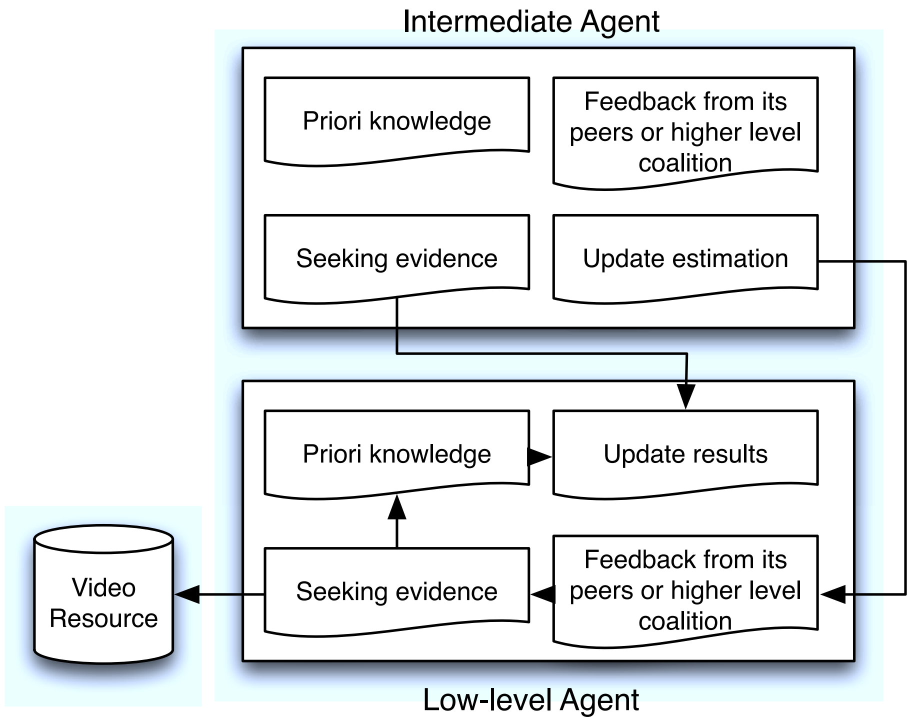

Figure 3.2: Processing of a low-level agent.

It is hard to manually assign the exact number of agents and their initial positions. In our design, each low-level agent is assigned automatically to an extracted feature at the beginning of processing. For example, a Gaussian color model is applied to classify the first original frame into skin-color pixels and non-skin-color pixels. A skin-color blob agent clusters all spatially coherent skin-color pixels and builds a blob. If this blob has been used by another skin-color agent, the agent will go to check the next available blob in this frame. If the skin-color blob agent finds an available blob, it uses this blob to initialize itself. Otherwise, this agent will keep looking for an available blob in the next frame.

After a low-level agent is initialized, it will search features in a constrained area instead of the whole image. This constrained area is updated by the temporal prediction of the low-level agent plus a certain offset allowance. If an initialized low-level agent cannot find any feature in the current frame, it uses its trajectory to predict a new position. Otherwise, the agent uses the real detection result and labels the result as 'detection'. This approach allows us to handle the merging, splitting and disappearance of features in the frame sequence.

Intermediate agent's processing

An intermediate agent is designed to associate low-level agents with an object/sub-object being tracked. In the same way as a low-level agent, an intermediate agent automatically initializes itself by finding possible associations among candidates provided by the low-level agents, in the starting frame. For example, we have an intermediate agent to track a hand of a human. This hand agent looks for an available skin-color blob (with certain size) as its initial information, and it may also take a set of coherent motion vector agents around this blob, which indicates that this hand is in motion.

After an intermediate agent is initialized, it starts looking for evidence of its existence in consecutive frames. Based on the trajectory of the object represented by an agent, the agent will predict the position and velocity of its object in the current frame, which we call an *estimation*. In the rest of this dissertation, when we use the term 'estimation', it means a spatial prediction combined with a temporal prediction. An intermediate agent will only consider the coherent candidates (e.g. color blob agents and motion vector agents), and it will use one or a combination of these candidates. The low-level agents labeled as 'guess' will only be used when there is not a coherent and available one labeled as 'detection'. When an intermediate agent finds evidence from low-level agents for its existence, it starts looking for its peers in the same layer to form a coalition to enhance the evidence of their existence. For example, a head agent will form a coalition with a torso agent, which is connected or close enough to this head agent. And a head normally appears on the top of a torso. But it will never form a coalition with any torso agent that is far away. If an intermediate agent can build such a coalition, this agent will be labeled as 'grouped', otherwise, 'up-grouped'. All the agents in this group will be maintained by each agent in this group.

When an intermediate agent uses the processing results of certain low-level agents, it can send these agents feedback and ask them to update. It also waits to be employed by a high-level agent. If it is employed, this intermediate agent is going to wait for the feedback from the high-level agent who hires it. We will discuss this employment-update process further in a later section. Figure 3.3 shows how an intermediate agent functions.

High-level agent's processing

A high-level agent is designed to represent a human being tracked. Typically, a high-level agent uses a coalition or a combination of a set of coalitions to represent a human body. For example, there is a coalition of a torso agent and a head agent, and this coalition is

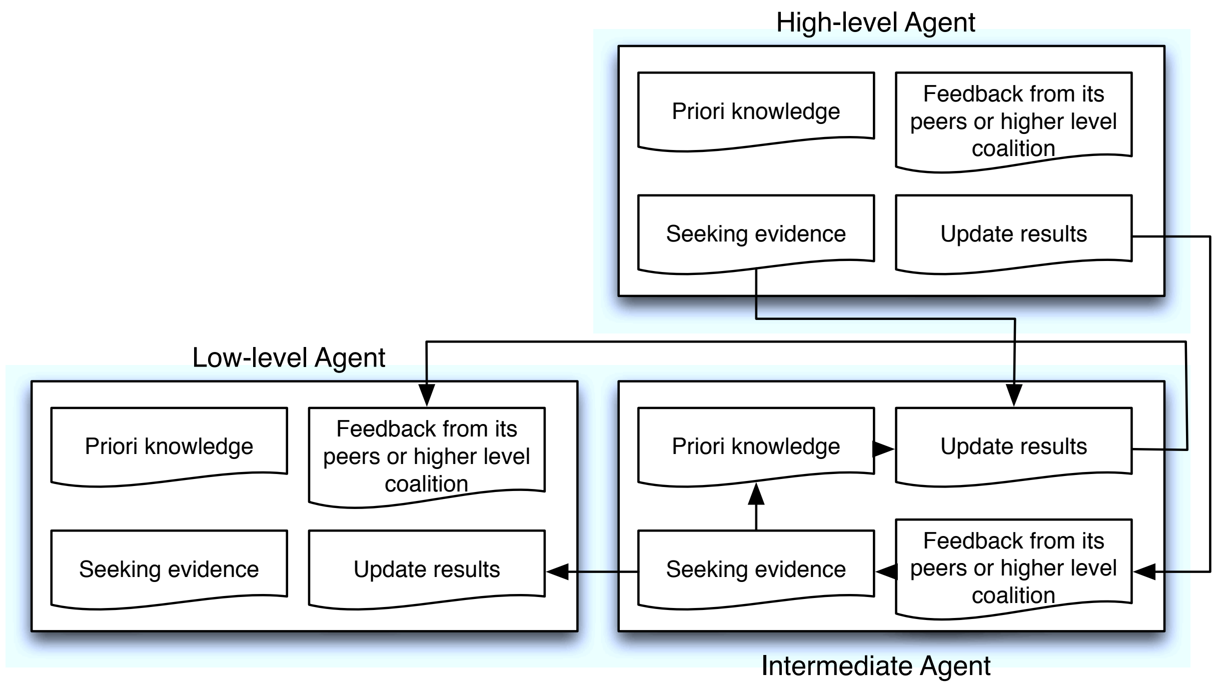

Figure 3.3: Processing of an intermediate agent.

temporally and spatially coherent with the human agent. In this case, the human agent will use this coalition, which can be evidence of an upper body.

In our approach, a high-level agent is manually initialized at the beginning of processing, including the skeleton and the expected position of each body part. Each high-level agent is assigned to one targeted person to track. Figure 3.4 shows how a high-level agent functions. A high-level agent seeks coalitions of intermediate agents as evidence of existence. However, it may also need to employ an individual agent that does not form a coalition with any other agent. When the high-level agent employs any intermediate agent, it can send a request to this agent to ask for an update.

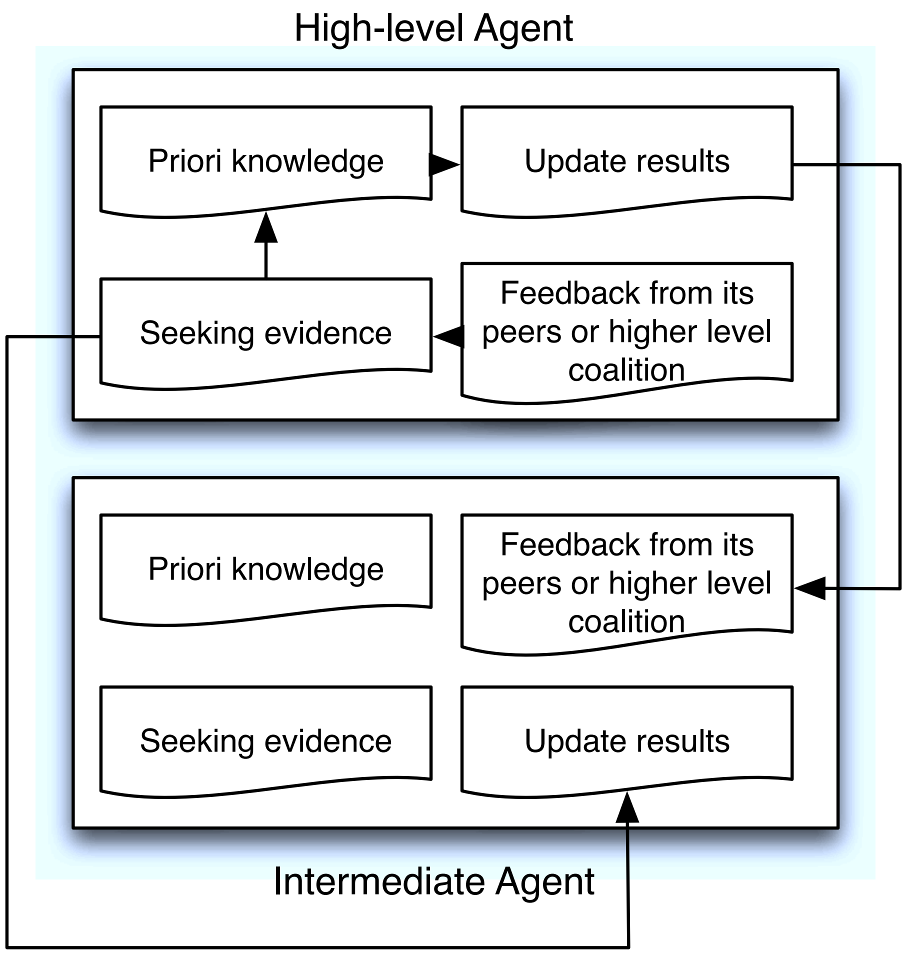

Figure 3.4: Processing of a high-level agent.

3.1.2 Our framework

Our framework is constructed with various intelligent agents, which we have introduced in the previous section. In this section, we introduce our agent-based framework, which is embedded with three-level agents: *low-level agents*, *intermediate agents* and *high-level agents*. Figure 3.5 illustrates the design of our framework. In this framework:

- *low-level agents* represent the basic features extracted from original frames in video

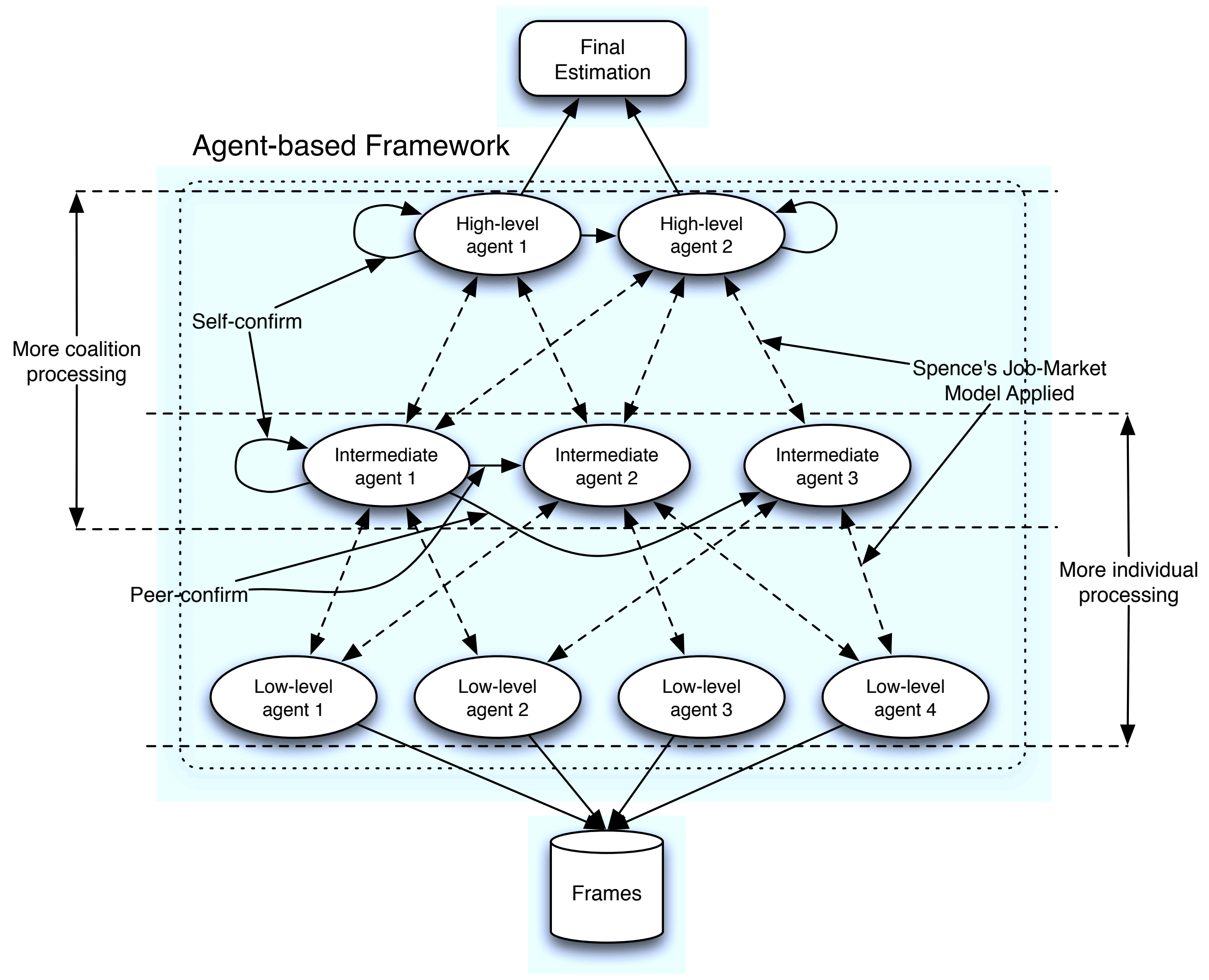

Figure 3.5: Agent-based framework for vision-based analysis.

analysis. Required processing, such as clustering, is also embedded within these agents. For example, we might have a skin-color agent to provide skin-color blobs in each frame. The skin-color agents grab frames and label all possible skin-color pixels in the current frame, and cluster these pixels into a set of skin-color blobs. The low-level agents provide the lowest level interpretations, and they do not communicate with their peers. In the same example, a skin-color agent only knows there is a skin-color blob, but it doesn't know whether this blob is a hand, a head or any other object that has a similar color. A low-level agent also takes feedback from an intermediate agent to re-do analysis to improve the results to meet the requirement from a certain intermediate agent. This feedback between different levels of agents will be discussed in next section.

- *intermediate agents* represent objects in the scene, such as hands or heads. These agents have a higher abstraction than low-level agents. The intermediate agent seeks evidence from low-level agents to support its own existence (dashed arrow in Figure 3.5). Meanwhile, it also looks for strong coalitions with its peers (peer confirm arrow in Figure 3.5). For example, a hand agent can seek to validate its existence by finding evidence of a skin-color blob and coherent motion vectors around it, which are provided by low-level agents. It can also prove its existence by building a strong coalition with an arm agent. Additionally, an intermediate agent also looks for coalitions with its own historical evidence (self confirm arrow in Figure 3.5). For example, with the estimation from previous frames, a hand agent knows that the hand should appear at a certain position in the current frame. If this estimation proves true, then it is important evidence for a hand agent to help validate its own existence.

- *high-level agents* represent high-level coalitions of objects, such as humans which can be simply constructed with two hands and a head. The high-level agent seeks evidence among objects with strong evidence of existence (which are represented by intermediate agents), and builds coalitions among these objects. A stronger coalition among objects provides a larger likelihood of the existence of a human body. As with intermediate agents, the high-level agent can support its existence by self-confirmation and peer-confirmation (shown in Figure 3.5).

Within this framework, an agent is not only an intelligent processing unit, but also an active individual that seeks evidence locally (with its prediction) from lower or higher level existence and cooperates with its peers to provide strong evidence for its own existence. Although the relaxation labeling approach provides an iterative algorithm that uses contextual information to reduce local ambiguities [11, 61, 88], it computes in a global area (the whole image) that requires much unnecessary computation. In our approach, each agent only searches locally, instead of in the whole image, which avoids unnecessary computational costs, and the job-market model, which is introduced in the next sub-section, provides a method for an agent to select the best candidate. However, it is not guaranteed that the agent representing the real targeted object will be employed every time because of the similarity between the targeted object and the object(s) around. We will show these cases with our experiments in next chapter.

The existing evidence of such an agent also helps higher level units to construct stronger coalitions. The lower level agents in the bottom layers (Figure 3.5) of this framework focus more on local processing to provide basic feature evidence. The higher level agents in the upper layers (Figure 3.5) employ more coalition processing to construct higher abstraction relationships among coherent objects.

3.1.3 Communication Model

As we have described in the previous section, our agents are intelligent objects representing meaningful objects in vision tracking, and the agents are capable of exchanging information with each other to build coalitions that support their own processing. Then the question becomes: how should these agents cooperate with each other to accomplish their goal?

Within the existentialist model, when an agent wants to derive evidence form low-level objects, it generally acts in two ways: (a) the agent is looking around to observe if there is any existing and coherent group to join to gain higher confidence in its existence, and (b) an existing group is also looking around to find certain agents to further support the group's existence. These activities are quite similar to someone looking for a job and some companies looking for certain employees to fill open positions. So, we decide to investigate a 'Job-Market' model in social economics to establish the agents' communication model of our tracking system.

Job Market Model in Social Economics

There are generally two groups [28, 67] in a job market: (1) individuals (applicants/employees) with certain productive capabilities, who want to optimize among different offered wages and investment costs, and (2) employers (firms) with certain conditional probabilistic beliefs, who want to have qualified employees to fill a set of open positions. In Spence's model [67], an individual takes further education as his investment costs, and the offered wage is what he will gain. However, more education might not bring more improvement of the productive capability of individuals. On the other hand, because an employer does not know the productive capability of any applicant before observing him after hiring, the hiring decision is treated as his investment cost. The employer benefits from the productivity of hired individuals. And both individuals and employers want to maximize their difference between the investments and benefits [67].

Typically, a job-market model defines *employer* and *employee* as following:

- An *employer* has one or multiple open positions that need to be filled by employing qualified individuals, and he also wants to have these open positions to be filled quickly. Moreover, the employer is able to fire current employees if he finds that there are better applicants existing in the employment market and the current position cannot be filled with those applicants without releasing the position.

- An *employee* (applicant) is looking for available positions and applies for what is currently the best position to get himself employed. And an employed worker is also capable of changing his job to improve his career development. Moreover, an employee is able to improve his own ability (such as pursuing higher education) to develop himself for better positions [28].

The key in a job market is to build an equilibrium between the investment costs and the benefits for both employer and employee. Spence described this relationship as informational feedback in a job market over time, as shown in Figure 3.6. In this feedback loop,

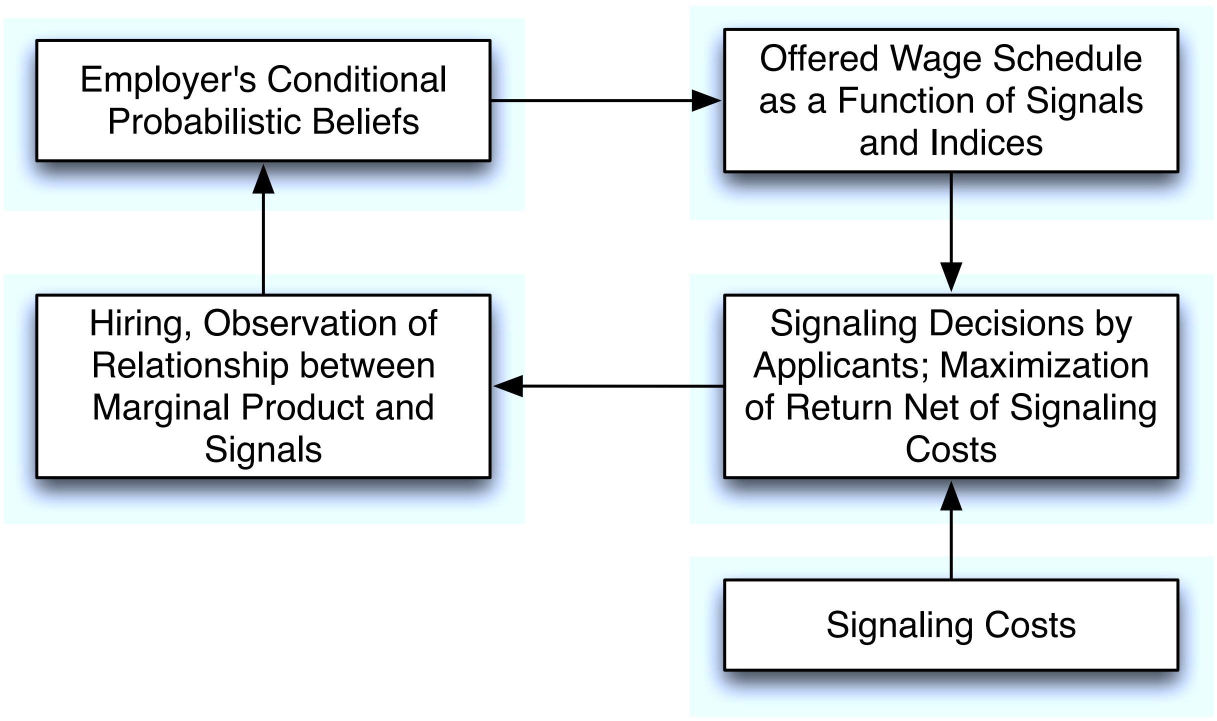
Employer's Conditional Probabilistic Beliefs
Offered Wage Schedule as a Function of Signals and Indices
Hiring, Observation of Relationship between Marginal Product and Signals
Signaling Decisions by Applicants; Maximization of Return Net of Signaling Costs
Signaling Costs

Figure 3.6: Informational Feedback in Job Market, by Spence, 1973 [67]. Used under fair use guidelines, 2011.

the employer has a prior estimation of the quality of the applicants for the open positions (conditional probabilistic beliefs), and he posts the descriptions of those positions to attract qualified applicants (offered wage schedule). When an applicant notices the updated posting, he evaluates the position by the following equation:

$$f = f_1(requirement, resume) + f_2(description, interest_rate) + f_3(wage, cost) \qquad (3.2)$$

The applicant will decide to apply for a certain position when he believes he can maximize the benefits from qualification (f_1), interest (f_2) and salary (f_3). An applicant can also choose to improve his competitiveness for a position, which he has more interest in. The employer gets new market information by hiring and observing the productive capabilities, and his conditional probabilistic beliefs are adjusted. Meanwhile, a new round of adjustment starts. The employer is self-confirmed when his conditional probabilistic beliefs are not challenged by the latest updates after certain iterations.

Job Market in Agent-Based Framework

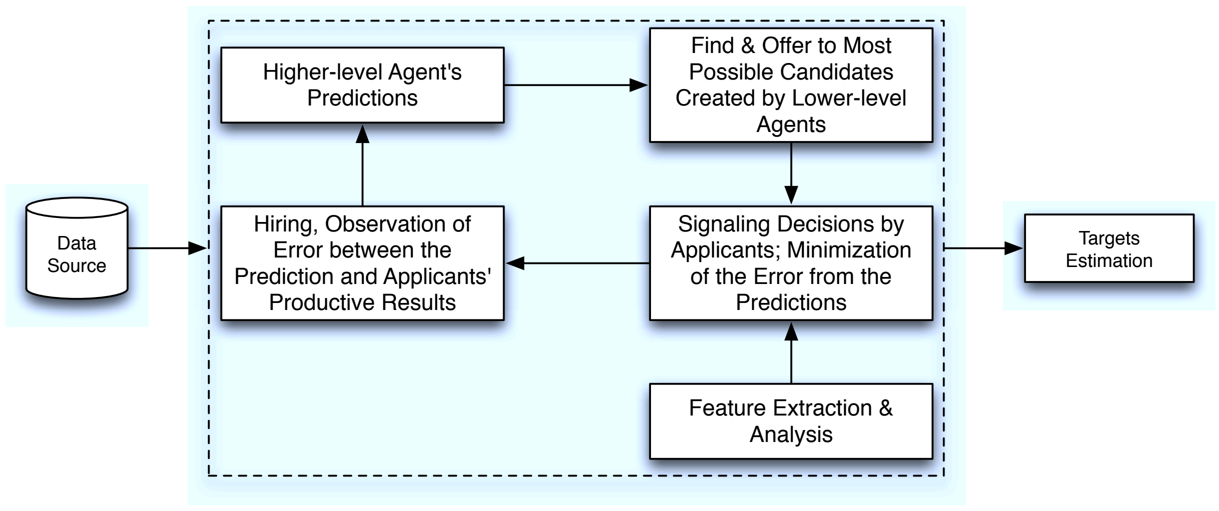
Data Source
Higher-level Agent's Predictions
Find & Offer to Most Possible Candidates Created by Lower-level Agents
Hiring, Observation of Error between the Prediction and Applicants' Productive Results
Signaling Decisions by Applicants; Minimization of the Error from the Predictions
Feature Extraction & Analysis
Targets Estimation

Figure 3.7: Informational Feedback in Agent-based Framework.

After studying Spence's job market model, we notice the correspondence between this model and the coalition we mentioned in Section 3.1.2. In this section, we are going to apply this model to our agent-based framework for human tracking. Obviously, following the job market model as described above, we also need to define two kinds of basic coalition agents in our tracking system, and they perform similar informational feedback shown in Figure 3.7:

- *Employer-agent.* This agent, which functions in the role of an employer, exists all the time during the processing to help individual agents build proper relationships with each other when they need to have other objects filled in to form up a higher-level coalition. For example, a human agent has two symmetric arms, two legs and one head connected to the same torso. When this human agent needs to prove its existence, it needs to employ two arms, two legs, one head and one torso, which are qualified with the temporal and spatial correlation. At this point, the human agent assumes the role of an employer agent to 'employ' two coherent arm agents, two leg agents, a head agent and a torso agent. This agent posts a requirement (such as requesting a head), and waits for the best fit to this request. When the position is filled, this agent updates its prediction and performs the next iteration. Moreover, the body part agents can also perform as employer agents. For example, a hand agent employs a certain connected skin-color blob to represent a possible hand candidate. An employer agent always has open positions with special requirements and looks for qualified employees to have himself best fit to its prediction. When the employer agent is not satisfied with any current employed agent, it is possible to fire the agent when it finds other more qualified candidates to fill the position.

- *Employee-agent.* This is another agent, that functions in the role of an employee and exists all the time during the processing to help individual agents build proper relationship with each other when they need to find a job (open position to take) to form up a higher-level coalition. Different from the employer agent, the employee agent plays the role of being employed by the employer agent. When this agent is informed that there is an open position request, it evaluates the requests and the capability of itself. If it fits the requirements, it applies for the position. Otherwise, it may perform more processing to improve the feature of itself to be able to fit the position request. As in the example above, the hand agent performs as an employee agent and contacts the the proper human agent when the hand agent believes that it is suitable to be the hand of the human being. Generally, each employee agent has various abilities/knowledge and looks for a suitable position to get itself filled into a position, and the employment will support the existence of the employee agent. When the employee agent believes that it is not suitable with current position, or that if it receive a better offer from other employer-agents, it is possible for the employee-agent to switch jobs and joins a better high-level coalition.

However, we might notice that each agent is changing its role between employer and employee during the processing. For example, a hand agent is an employer agent when it employs skin-color blobs to support its existence, but it becomes an employee agent when it is employed by a human agent to form a high-level coalition. So, we need other job-signaling agents to help with the role switching. In this job signaling analysis, the recruiter/advertiser agent is used to assist with hiring activities, and it will not make any decision on hiring or not hiring. Thus, we define following agents to assist with this processing:

- *Recruiter agent.* The recruiter agent is an agent that helps employer agents fill the open positions with qualified employee agents quickly (*head hunt*), and also helps employee agents, who are requesting to be hired, to fill certain open positions (*job hunt*) quickly. Although this recruiter agent helps in processing, it incurs a performance overhead over just having the employer and employee agents communicate directly. Actually, recruiter agent is not a necessary part during the processing, if there are not too many idle agents.

- *Advertiser agent*. The advertiser agent provides a place for employer and employee agents to post requesting information, especially when they do not know who will be the proper candidates: an employer agent posts his open positions with specific requirements; an employee agent submits his 'resume' which describes his capability; and, a recruiter agent monitors the resume pool to help build a connection between the employer agent and the employee agent, unless they have already created this connection.

3.2 Implementation of Our Agent-based Framework

In this section, we introduce the implementation of the framework with the agents discussed in previous sections. Figure 3.8 describes how we implement the framework, which we introduced in Section 3.1. An intermediate agent seeks evidence from low-level agents and provides its estimation for its own existence. Then, it checks with the recruiter/advertiser (R/A) agent to find if there is any open position that it fits. If it fits the requirement, it applies for an interview (evaluation) from the high-level agent who provides this position.

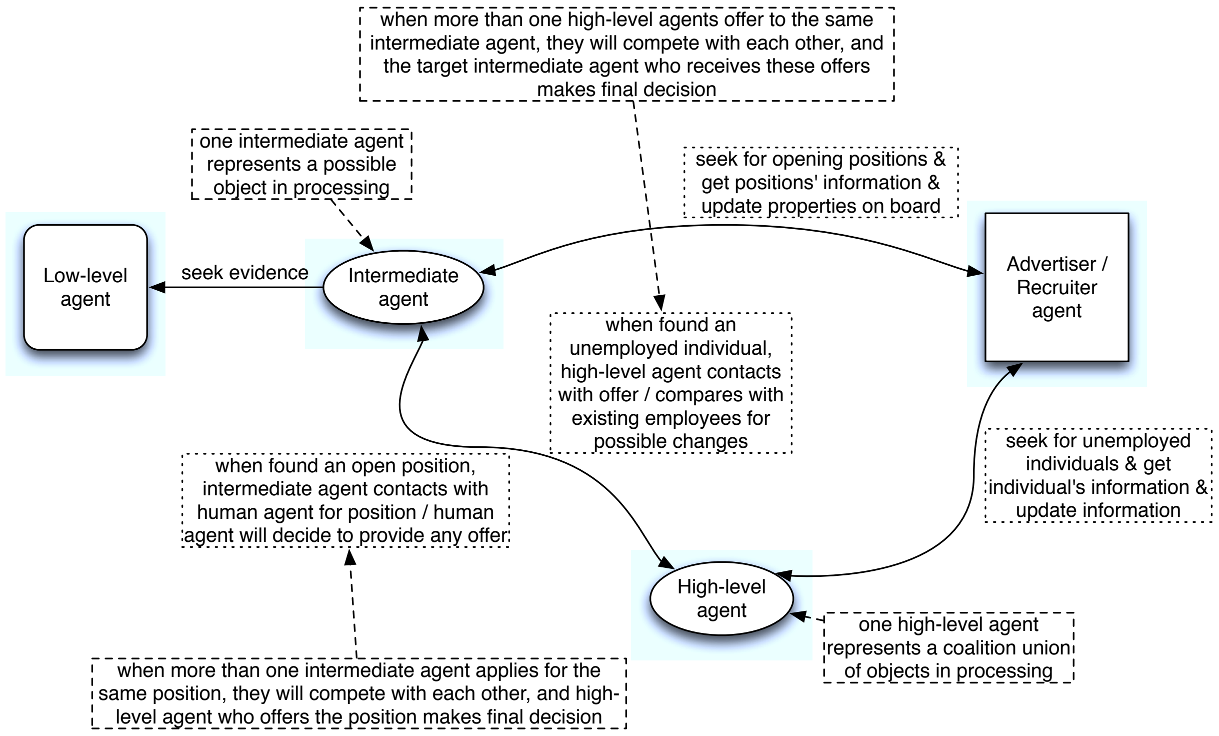
when more than one high-level agents offer to the same intermediate agent, they will compete with each other, and the target intermediate agent who receives these offers makes final decision
one intermediate agent represents a possible object in processing
seek for opening positions & get positions' information & update properties on board
Low-level agent
seek evidence
Intermediate agent
Advertiser / Recruiter agent
when found an unemployed individual, high-level agent contacts with offer / compares with existing employees for possible changes
seek for unemployed individuals & get individual's information & update information
when found an open position, intermediate agent contacts with human agent for position / human agent will decide to provide any offer
High-level agent
one high-level agent represents a coalition union of objects in processing
when more than one intermediate agent applies for the same position, they will compete with each other, and high-level agent who offers the position makes final decision

Figure 3.8: Implementation of agent-based framework for vision processing.

When it receives the positive confirmation (offer) for the position, it evaluates and decides whether to take the offer or not. This processing is shown as the flowchart in Figure 3.9. If there is no open position in the R/A agent, this intermediate agent will post its properties to the R/A agent and wait for a high-level agent to release a position. If an intermediate agent receives more than one offer, it will compare them and take the best offer. If the intermediate agent is rejected after the evaluation from the employer agent, it goes to seek more evidence to improve its qualification for the position and wait for the next round of evaluation.

On the other hand, when a high-level agent needs to find an object to fill its open position, it goes to check with the advertiser agent to see if there are any unemployed individuals that fit the position. If so, the high-level agent provides an interview opportunity to this individual and evaluates its properties. After the evaluation, the high-level agent decides whether to provide an offer or not. When the applicants accept the offer, the high-level agent updates its prediction and starts another informational feedback loop (shown as flowchart in Figure 3.10).

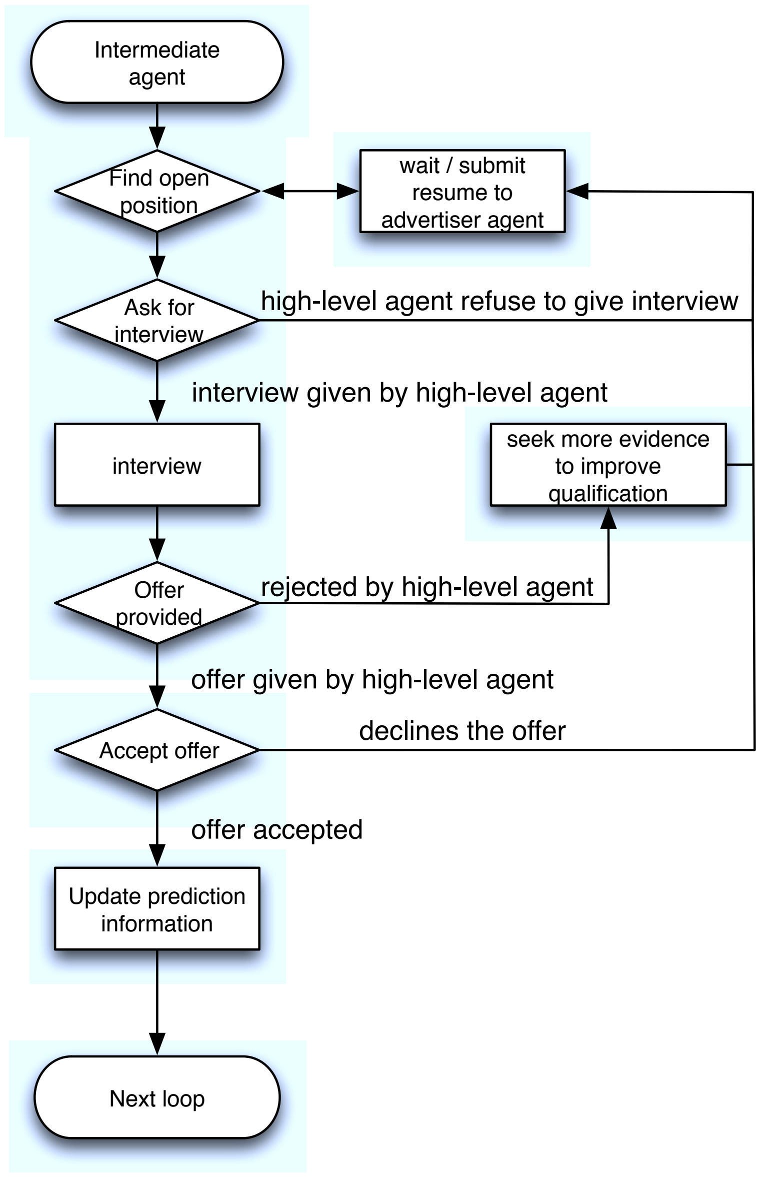
Intermediate agent
Find open position
wait / submit resume to advertiser agent
Ask for interview
high-level agent refuse to give interview
interview given by high-level agent
interview
seek more evidence to improve qualification
Offer provided
rejected by high-level agent
offer given by high-level agent
Accept offer
declines the offer
offer accepted
Update prediction information
Next loop

Figure 3.9: Flowchart of the intermediate agent's processing.

3.2.1 The agent's processing

In our framework, knowledge is pre-built in each agent, including the type of object it represents, the format of results it generates and the processing functions it uses. As an agent knows what type of evidence (such as a color blob) it needs to support its existence, the agent has a set of built-in functions to compute this evidence and to provide its own analysis result. The format of the result produced by an agent can be described by the ordered pair:

$$< agt\_id, < detection >>$$

$$< detection > = < frm\_id, obj\_type, dat\_type, < data >, < coa\_list >>$$

where *agt_id* identifies the agent that produced this record. frm_id indicates the indices of frames of the video. *obj_type* specifies the type of object that this agent represents. The value of *obj_type* is an integer, and the value table is pre-defined before processing is initialized. The pre-defined value table contains all the values used for *obj_type*. *dat_type* is the type of the result data, e.g. a blob or a vector. This value is also stored as a table of integers. $< data >$ indicates the processing result, including the position in the current frame and the vector of the velocity calculated based on the previous frame. $< coa\_list >$ is a list of all the *agt_id* of agents that forms the same coalition.

A higher level agent (intermediate or high-level agent) also produces a prediction result that can be described by the ordered pair:

$$< agt\_id, < estimation >>$$

$$< estimation >=< frm\_id, obj\_type, dat\_type, < data >, err >$$

where agt_id, frm_id and dat_type are the same as before. $< data >$ represents the prediction results, and err specifies the maximum permissible error. In this approach, we use the Euclidean distance to compute the spatial difference between the estimation and detection results, and we use the angle between the vectors to compute the temporal difference.

By applying the job-market model, it is possible that multiple applicants apply for the same position, and multiple positions may be offered to the same applicant. If more than one applicant applies for the same position, the provider of this position will evaluate these candidates with the difference between the estimation of itself and the provided result from each applicant. Similarly, an applicant evaluates the error to the estimation from the job position provider, when it receives more than one offer (interest and salary, which are f_2 and f_3 in Equation 3.2).

3.2.2 The agent's communication through R/A agent

As we apply the job-market model to our framework, we have to build a format for agents to exchange their prediction and results, including posting an open position and sending request for a position. All the agents are registered to the R/A agent before processing, including the ID and the layer information of each agent. Stacks (LIFO) are used in advertiser/recruiter agents to store the posting requests. We use stacks (LIFO) because we assume that the more qualified candidates always come late, because they require more processing, which is the same as Spence's theory that higher education normally takes extra time (cost). When an object agent is looking for a vacant position, it sends its current processing result to the R/A agent:

$$< agt\_id, < detection >, < rqt\_list >>$$

where $< rqt\_list >$ is a list that indicates the suitable types of positions for which it can apply. The rest is the same as before. When a higher-level agent has an open position and wishes to hire an agent at the lower layer, it sends its prediction and a list of suitable candidates to the R/A agent, as follows:

$$< agt\_id, < prediction >, < rqt\_list >>$$

The applications for a position are stored in $resume\_stack$, and the requests for an employee are stored in job_stack, both of which are maintained by R/A agent. If neither of these two stacks is empty, the R/A agent starts to find possible applicants in $resume\_stack$ based on the requirement in job_stack. For example, a hand agent has posted its resume to the R/A agent ($resume\_stack$) as $< agt\_1, < frm\_30, hand, circle, x_0, y_0, r_0 >, < null >>$, which indicates that a hand agent (agt_1) finds evidence in the 30^{th} frame and generates a hand object (as a circle $< x_0, y_0, r_0 >$). It does not have any coalition yet. A human agent also posts a position to the R/A agent (job_stack) as $< agt\_10, < frm\_30, han, circle, x_1, y_1, r_1 >, < err >>$), which indicates that a human agent (agt_10) has an open position for a hand at the 30^{th} frame, and its prediction is a circle ($< x_1, y_1, r_1 >$) with an error of err. Then, the R/A agent evaluates these two posts by: (1) if the frm_id and obj_type matches; and (2) if the distance between $< x_0, y_0, r_0 >$ and $< x_1, y_1, r_1 >$ is less or equal to err. When both (1) and (2) pass the evaluation, the R/A agent will send the ID of these qualified agents (qualification, which is f_1 in Equation 3.2) to each other to build a direct connection between them.

When an agent wants to construct a coalition with other agents at the same layer, it sends request to the R/A agent, and the R/A agent will send correlated information from agents at the same layer back. The only difference is, to find an applicant or a position, the R/A agent will help building a connection between agents at different layers, and to form a coalition, the R/A agent will help build a connection between agents at the same layer.

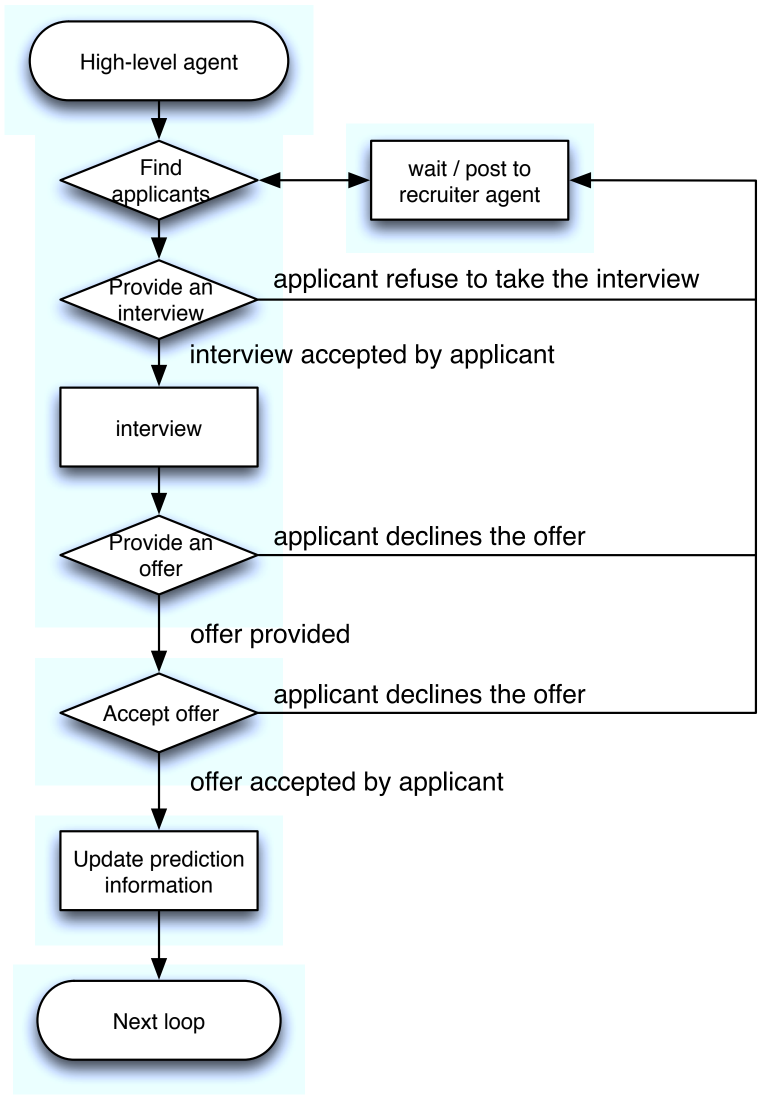
High-level agent
Find applicants
wait / post to recruiter agent
Provide an interview
applicant refuse to take the interview
interview accepted by applicant
interview
Provide an offer
applicant declines the offer
offer provided
Accept offer
applicant declines the offer
offer accepted by applicant
Update prediction information
Next loop

Figure 3.10: Flowchart of the high-level agent's processing.

Chapter 4

Application of an agent-based framework for human tracking

In the previous chapter (Chapter 3), we introduced the theoretical model of our agent-based framework. In this chapter, we introduce our application of this framework to real processing problems. First, we will use one of the AFIT (Air Force Institute of Technology) datasets (AFIT-06012004) to demonstrate the use of our framework. We will then discuss more results with other data sets.

4.1 AFIT-06012004 data description

The AFIT dataset contains meeting room-type data, typically involving 5 to 7 individuals in discussion around a table. Figure 4.1 shows the 5-people configuration of the meeting room. The five participants are labeled as C, D, E, F, G in the meeting video. Ten cameras, labeled C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9 and C_{10}, are permanently installed in the ceiling to record the meeting events simultaneously. This setting is guaranteed to have each participant captured by at least two cameras at any time in the meeting video data. Eight Vicon infrared cameras are installed in fixed positions on the ceiling in the meeting room, which are not shown in the figure. These infrared cameras track reflective markers worn on targets to provide 3D motion data as ground truth for vision analysis. T_1 and T_2 are two table microphones to record speech. In this example, our goal is to track torso, hands and head of target E.

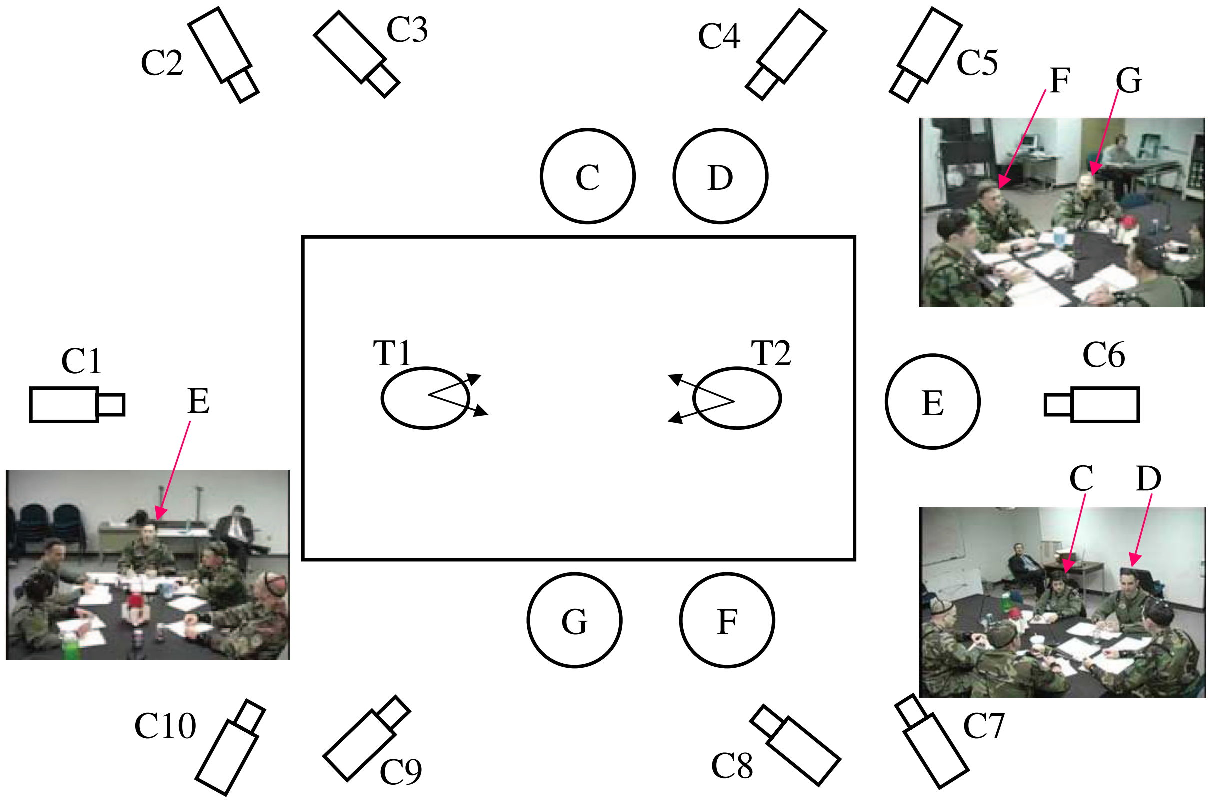

Figure 4.1: Meeting room configuration of AFIT data capture.

The AFIT data was captured for the Video Analysis and Content Extraction (VACE) R&D Program. The data was captured synchronized multimedia data (10 pair-wise stereo calibrated cameras, wireless fixed-distance directional microphones, table-mounted microphones, and a system of Vicon infrared 3D motion trackers) of the proceedings in the meeting room. The subjects in the meeting were instrumented non-obtrusively with infrared-reflective markers so that we have truth data for upper-body motion, including head orientation, shoulder orientation, torso orientation and hand motion.

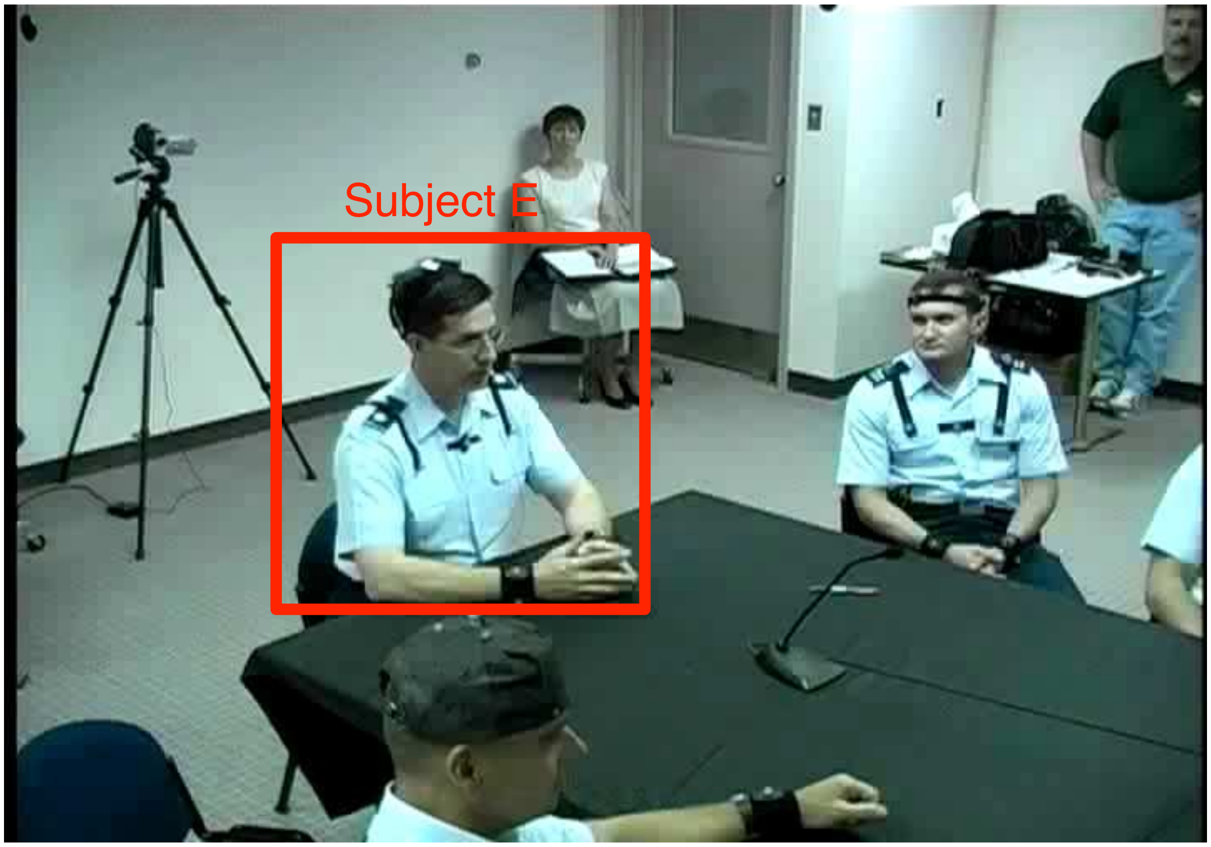

Figure 4.2: Frame from camera 3 (C_3) of AFIT data 06012004.

4.2 Use of our agent-based framework

We use the video captured from camera 3 (C_3, shown as Figure 4.2) to illustrate the framework implementation. We analyze the motion of subject E in this video. In this section, we explain the design of the low-level agents, intermediate agents and high-level agents for this video.

4.2.1 Low-level agents

We introduced various feature extraction approaches in Chapter 2. In this demonstration, we use three kinds of features: skin-color blobs, blue-shirt blobs and motion vectors. In this part, we introduce these three low-level feature agents.

Skin-color agent / Shirt-color agent

Gaussian color models are commonly used to generate various color blobs, such as skin-color [82], and other specific colors [18, 19]. A multivariate Gaussian distribution model [82] was applied to extract skin-color for facial recognition. However, this model is limited because the model is pre-trained under fixed lighting conditions. Lighting changes can break the model in the course of a long data capture session vision processing. An adaptive modification of this color model was introduced by Xiong et al. [80]. Although this model was introduced to produce skin-color blobs, we showed that it can be used for other color blobs as well [18, 19].

The adaptive color model adds an automatic adaptive strategy that uses the sampling, filtering and automatic updates to solve the lighting-change problem. Typically, a Gaussian color model ($N(\mu, \sigma)$) must be trained with a sample set of required color pixels. As discussed in [80], signal noise can affect the modeling results. Furthermore, since we use the Gaussian distribution to process image frames, some areas, which have strong correlation with the color model, may also be treated as skin color even with careful sampling.

Based on the skin color model presented by Yang et al. [82], the color model is built in normalized RGB space, where $r = \frac{R}{R+G+B}$, $g = \frac{G}{R+G+B}$, and $r + g + b = 1$. Thus, the model takes the form of a two-dimensional Gaussian distribution as Equation 4.1 in normalized $r - g$ space presented by $N(\mu, \Sigma)$.

$$p(x) = \frac{1}{2\pi|\Sigma|^{1/2}} exp[-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)] \tag{4.1}$$

where x represents $(r, g)^T$ in sample sets, μ is the mean vector, and Σ is the covariance matrix of the color model. The modeling follows Equation 4.2 to learn the initial model using *maximum likelihood estimation* (MLE).

$$\begin{cases} \mu = \frac{1}{n}\sum_{k=1}^{n} x_k \\ \Sigma = \frac{1}{n}\sum_{k=1}^{n}(x_k - \mu)(x_k - \mu)^T \end{cases} \tag{4.2}$$

When we learn the color model, we separate each pixel into a pixel contained or not contained in the color model in the current frame in the normalized color space according to the

Gaussian probability filtering:

$$P(x) = exp[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)] \tag{4.3}$$

where x is a $(r, g)^T$ value of each pixel, and $P(x)$ represents the likelihood that this pixel fits the model. We use the Monte Carlo method based on these probabilities to gain the interest points in each frame. After this filtering, we separate the pixels into two groups depending on whether they belong to the color model or not. Then, we use the pixels that belong to the color model to update the previous color model. In our experiment, we use all the pixels that fall within two standard deviations.

The skin-color agent and shirt-color agent are built upon this adaptive Gaussian model. The agent takes the consecutive frames as input, and filters and clusters the pixels into blobs. The skin-color agents are shown in Figure 4.3 (a), and the shirt-color agents are shown in Figure 4.3 (b).

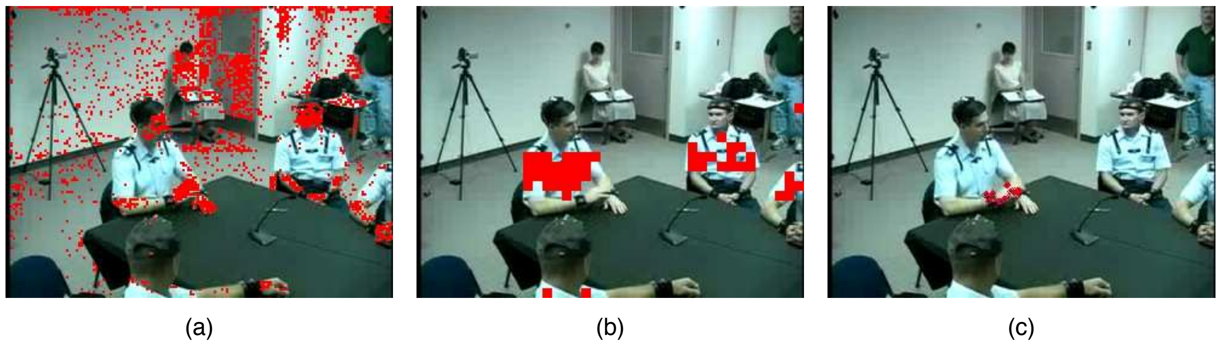

Figure 4.3: Low-level feature agents. (a) The skin-color agents; (b) The shirt-color agents; (c) The VCM agents.

In the processing, a skin-color blob agent and a shirt-color agent process their result and save them as:

$$< agt\_id, frm\_id, skin, circle, < x, y, r >, null >$$

$$< agt\_id, frm\_id, shirt, circle, < x, y, r >, null >$$

Motion vector agent

The *vector coherence mapping* (VCM) algorithm extracts motion vectors from the source video and clusters them into spatially and directionally coherent vector clusters [2, 18, 58]. VCM is a parallel algorithm for motion tracking, which can extract spatially and temporally coherent vector fields from raw video. VCM falls into a category of vector field computation that are deemed to be region-based [2, 58]. This approach transparently incorporates the various local smoothness, spatial and temporal coherence constraints by the application of fuzzy image processing techniques to compute an optical flow field (vector field) from a video image sequence. A weighted voting process in local vector space is applied to obtain the vector field. The fuzzy voting process permits the addition of a variety of constraints to the flow field by applying a set of biasing-templates to influence the vote. Such templates may be derived from motion history (temporal templates) or color distribution in the video frame. VCM applies an agglomerative clustering procedure that groups vectors into sets that are spatially close and directionally coherent.

Figure 4.4 illustrates how our VCM agent applies a spatial-coherence constraint to minimize the directional variance. Suppose we have three interest points detected across two con-

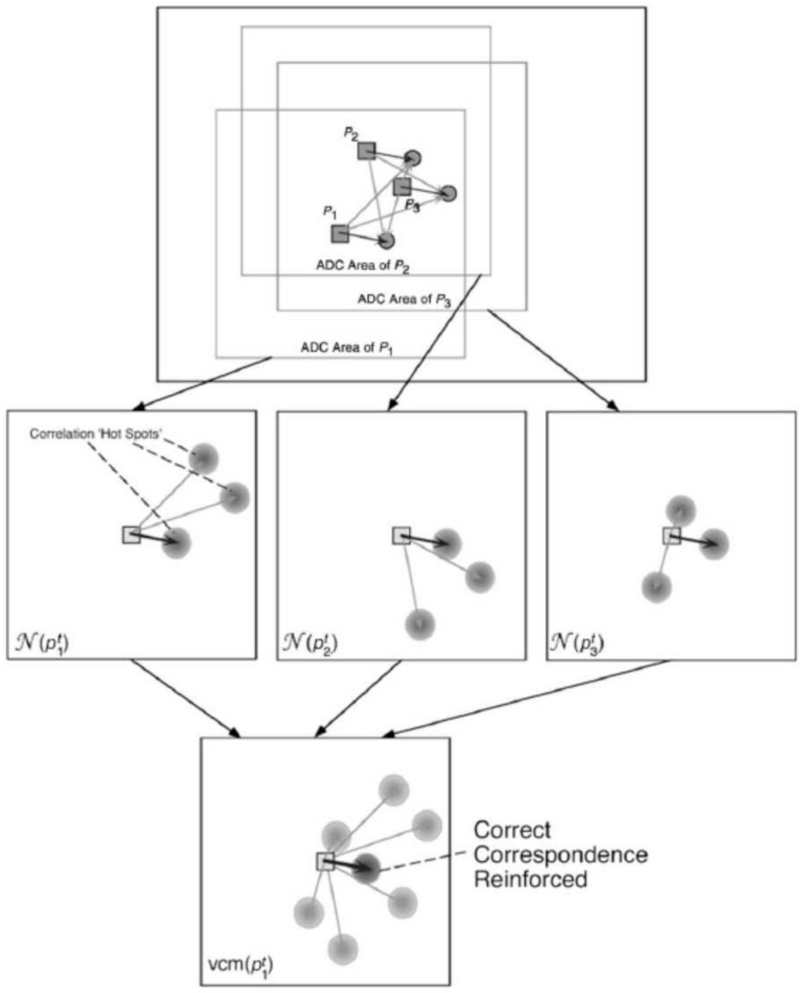

Figure 4.4: Vector Coherence Mapping [2].

secutive frames, shown as P_1, P_2 and P_3 in Figure 4.4. The shaded squares at the top of the figure represent the locations of three feature points at time t, and the shaded circles represent the corresponding position of these feature points at $t+1$. If we assume that these three interest points have equal correspondence in the frame $t+1$, the matching based on correlation of these interest points would provide three possible candidates, shown as the middle frames in Figure 4.4, which are marked as $N(p'_1)$, $N(p'_2)$, and $N(p'_3)$ (each frame, called a *normal correlation map* (NCM) is centered on the corresponding interest point). By using this weighted summation of neighboring NCMs, the vector that minimizes the local variance of the vector field can be obtained [58], shown as the bottom frame in Figure 4.4. Meanwhile, by maintaining a history of vectors extracted from previous frames, VCM is able to estimate the termination point of the next vector by applying a constant acceleration assumption [58]. Quek and Bryll used this temporal estimate to bias the vote to pick a vector that maximizes both spatial and temporal coherence from the resulting vector field. The final step in the VCM approach is an agglomerative process, by which vectors computed in each video frame are grouped into spatially clustered and directionally similar groups.

Our motion vector agent employs the VCM algorithm, and uses the grouped vectors to represent the candidates for the motion vectors in each frame, shown as Figure 4.3 (c). In the processing, a VCM agent processes its result and save them as:

$$< agt\_id, frm\_id, vcm, vector, < sx, sy, ex, ey >, null >$$

4.2.2 Intermediate agents

We have torso agents, hand agents and head agents as our intermediate agents. As we explained in the previous chapter, these agents represent possible objects in the real world. In this section, we explain how we define these agents and their coalitions.

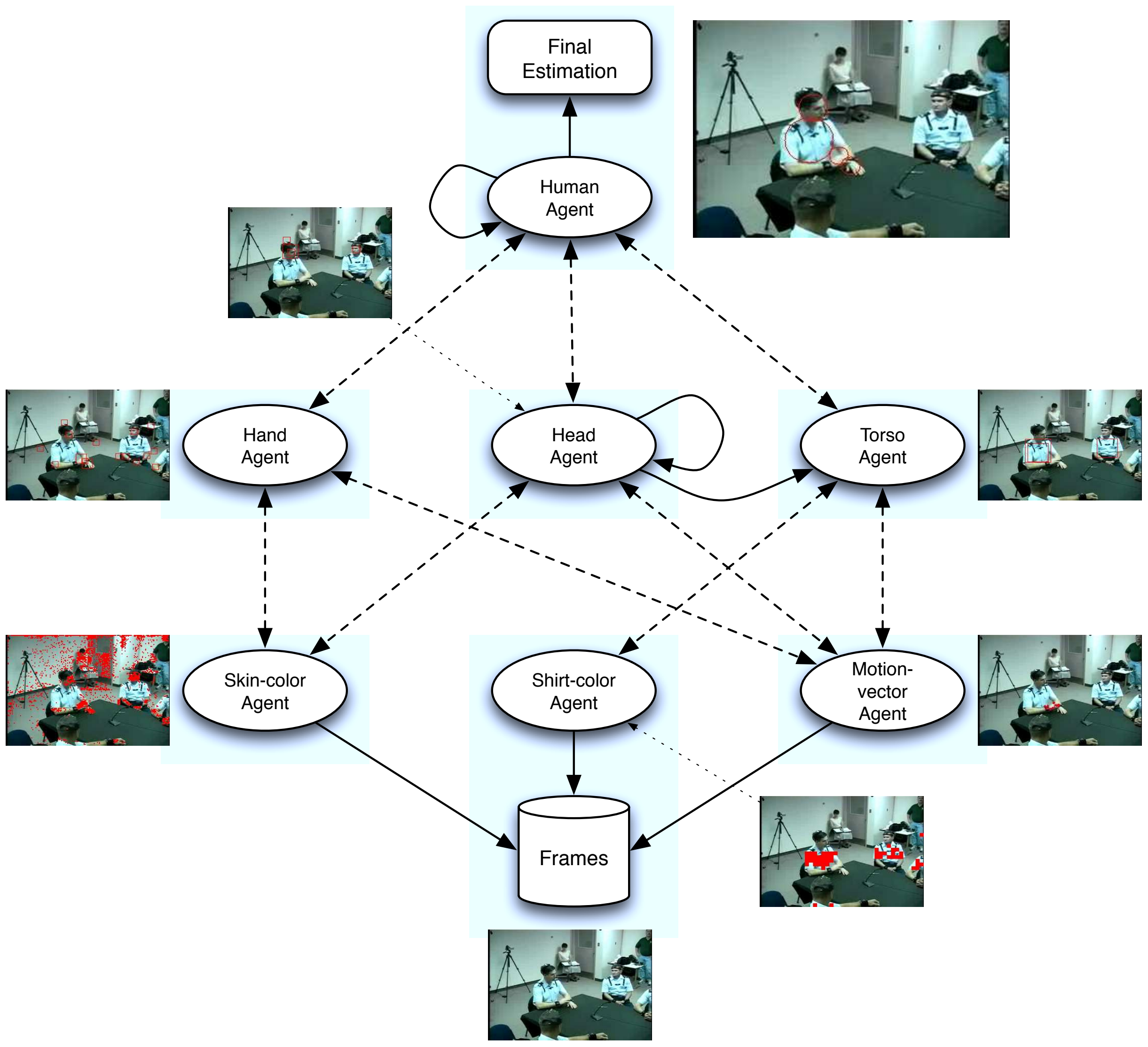

Figure 4.5: Applying agent-based framework for human tracking.

Hand agent

The human's hand can be described as a skin-color object. When it starts moving, a set of coherent motion vectors should be detected around this object. A hand is a part of a human being, so it should have certain coherence with the human body. Thus, we define our hand agent as an object, which has a pre-defined size, covered by a skin-color blob and probably a set of motion vectors around this blob. Meanwhile, this object should have a temporal and spatial coherence with a torso or a head.

In the processing, a hand agent evaluates all of the skin-color blob candidates, which are provided by the R/A agent (as we introduced in the previous chapter), for its evidence of existence. Before the evaluation, the hand agent predicts its position and velocity vector for the current frame as:

$$< agt\_id, frm\_id, < circle, vector >, < x, y, r, sx, sy, ex, ey >, err >$$

The hand agent compares the difference between its prediction and the results from all the candidates. During the evaluation, the job-market model is applied. The hand agent searches the list of all the proper blob candidates and decides whether to hire or not to hire. The candidate with the minimum error will be employed. If there is no candidate fit for the requirement, the hand agent waits for a proper one to be provided from the R/A agent. When a hand agent finds sufficient evidence to suggest its presence, it confirms with the hand agents from previous frames to find temporal coalition evidence (self-confirm), and it also confirms with the torso agents to get spatial coalition evidence (peer-confirm), which is shown in Figure 4.5. When a hand agent finishes finding this feature evidence and coalition evidence, it completes the first round building of itself. Figure 4.6 (a) illustrates the hand agents after the first round estimation. Then, the hand agent looks in the R/A agent to find a torso agent to form up a coalition. To form up this coalition, the hand agent and the torso agent should be temporally and spatially coherent with a head or torso agent. If it builds the coalition, the IDs of these agents are added to the coalition list. Otherwise, the coalition list is empty.

At this point, the hand agent starts looking for the R/A agent to see if there is any open position from some human agent to which it can apply. If there are no open positions, it will post its processing results to the R/A agent and wait for an open position. The hand agent posts its possessing result as:

$$< agt\_id, frm\_id, hand, < circle, vector >, < x, y, r, sx, sy, ex, ey >, < agt\_id_0, agt\_id_1, ... >>$$

If there is a proper position, but the hand agent does not fully meet the requirement, it will seek more evidence from its peers or the lower level feature agents to improve its estimation. For example, a human agent requires a moving hand with certain spatial and motion direction estimation. If a hand agent has enough evidence for the position requirement, it needs to seek evidence from motion-vector agent for support to fully meet the requirement for this open position requirement. However, it is still possible for this hand agent to be employed even though it did not find motion vector support. We will discuss this case in the high-level agent's section.

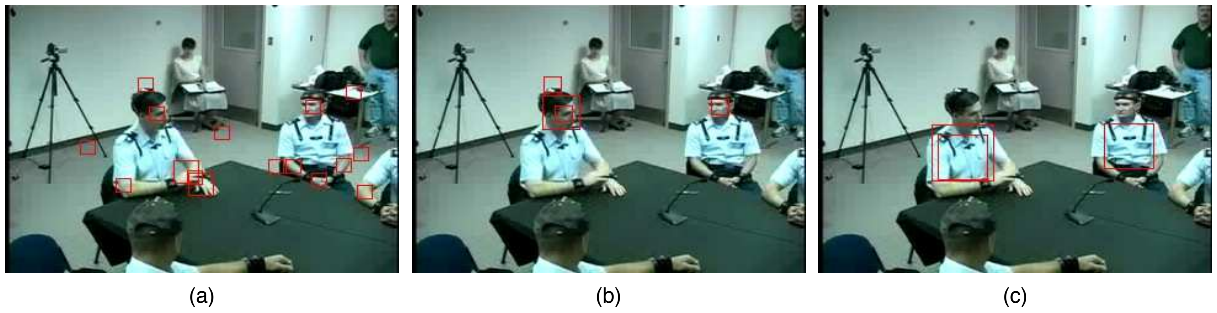

Figure 4.6: Intermediate agents. (a) The hand agents; (b) The head agents; (c) The torso agents.

Informational feedback

In Spence's job market model, an important issue is informational feedback. Generally speaking, there are two kinds of feedback in this process. By observing the capabilities of the employee, the employer changes its prediction. The other is the requirement from employer for certain positions that probably require the applicant to improve his qualification status. Figure 4.7 illustrates one use of this informational feedback.

In this example, a hand agent has an open position that needs a skin-color blob to cover most of this area, shown as Figure 4.7 (a). A skin-color blob agent has its original detection result shown as Figure 4.7 (b). At this moment, this skin-color blob agent does not fully meet the requirement for the position, so it takes more analysis (such as re-processing with the same Gaussian model) to see if it can improve the strength of its evidence. When the skin-color blob agent finds enough evidence, shown as Figure 4.7 (c), it fully meets the position requirement and applies the position. This provides a higher probability for this skin-color agent to win the competition for this job.

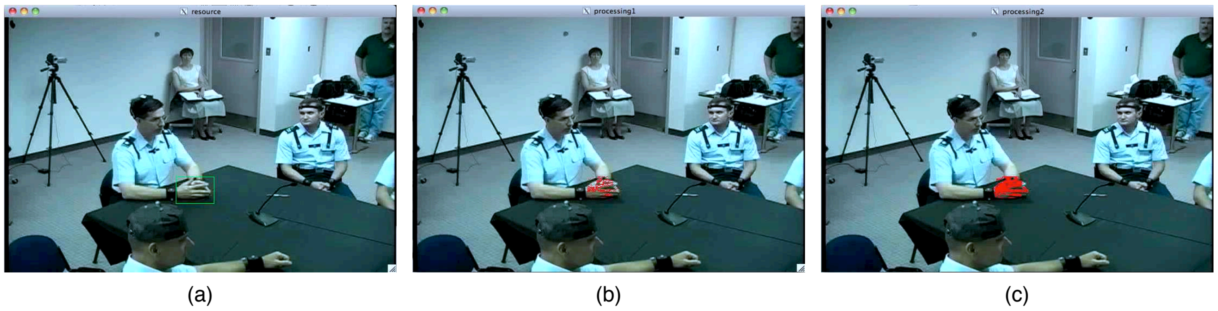

Figure 4.7: Informational feedback of agent's re-processing. (a) The open position for hand; (b) The evidence of a color blob agent; (c) The re-processing result of the color blob agent.

Head agent

As with the hand agent, a head agent also seeks all the skin-color blobs, which are spatially and temporally coherent with its history, for its feature evidence of existence. However, unlike hands, a head normally will only appear on the top of the torso agent, and this spatial condition normally does not change (unless there is perspective effect). Thus, the head agent has to find enough peer-evidence provided by torso existence. Another difference in this meeting analysis between hand agent and head agent is that a hand can move rapidly, but a head will not.

When the job-market model is applied, the head agent searches the job wanted postings for proper blob candidates and decides whether to hire or not to hire. If no candidate fits the requirement, it posts its prediction for the required skin-color blob to the board, and waits for a proper one to apply. Meanwhile, this head agent also requests a correlated torso to form up a tight coalition. When the head agent provides supporting evidence for its existence after the first round estimation, it posts its information to the job-wanted postings, and waits for a high-level agent to release an open position. Figure 4.6 (b) illustrates the head agents after the first round. In the processing, a head agent provides its prediction result and processing result to the R/A as:

$$< agt\_id, frm\_id, < circle, vector >, < x, y, r, sx, sy, ex, ey >, err >$$

$$< agt\_id, frm\_id, head, < circle, vector >, < x, y, r, sx, sy, ex, ey >, < agt\_id_0, agt\_id_1, ... >>$$

Torso agent

We will just briefly introduce the torso agent because it is similar to the head agent except for the feature part. A torso agent seeks the shirt-color blob for its feature evidence and at least an adjacent hand and a head as its coalition evidence. Figure 4.6 (c) illustrates the torso agents after the first round processing. When it finishes its first round processing, it also posts its information onto the job-wanted postings and waits for employment. It posts the prediction and processing result to the R/A agent as:

$$< agt\_id, frm\_id, < circle, vector >, < x, y, r, sx, sy, ex, ey >, err >$$

$$< agt\_id, frm\_id, torso, < circle, vector >, < x, y, r, sx, sy, ex, ey >, < agt\_id_0, agt\_id_1, ... >>$$

4.2.3 High-level agents

The high-level agents should know how to combine the information for body part tracking results and use the coalitions among them. For example, a head agent and a torso agent may form up a coalition. A human agent knows to use this coalition and combines two hands to build a simple skeleton for the upper body of a human, shown as Figure 4.5. Generally, the high-level agents are initialized at the beginning of the process. In the AFIT data, participants are normally not starting with any pre-designated pose, so we have to manually initialize the targeted body parts. In the view from camera 3 of AFIT-06012004 (shown as Figure 4.2), the high-level agent is defined as a human being (subject E) which has a head, a torso, a left hand and a right hand. After the initialization, this agent has four open positions to be filled in each frame. Each position with prediction is presented as:

$$< agt\_id, frm\_id, < circle, vector >, < x, y, r, sx, sy, ex, ey >, err >$$

When the human agent starts looking for applicants on the job-postings, if there are no qualified candidates, the human agent will wait for proper applicants to apply. When there is more than one applicant for the same position, the human agent uses the following equation to evaluate each applicant:

$$Q_{app} = w_1 \times E_{spatial}(P_{obj} - P_{app}) + w_2 \times E_{temporal}(P_{obj} - P_{app}) \tag{4.4}$$

where Q_{app} is the qualification of the applicant, $E_{spatial}$ is the spatial error and $E_{temporal}$ is the temporal error between the prediction of the object (P_{obj}) and the tracking result of the applicant (P_{app}). w_1 and w_2 are the weights of the two errors, which represent the beliefs of the employer.

Under certain circumstances, the human agent might not find any proper applicant. For example, the illumination changes might cause the failure of Gaussian color model, so that there is no applicant to apply for the job, which posted by the human agent. In this case, the human agent will use its prediction as a 'virtual applicant' to fill this position. When the human agent gets all the positions filled, it finishes the first round of processing. The next loop starts, and when there is no significant improvement of the qualification of applicants, the process ends.

Processing results

In this part, we present our tracking results for the view of camera 3 of AFIT-06012004. The total length of the video is 17,990 frames (10 minutes). We are tracking the head, torso and both hands of subject E. In this video, the person put his hands together in the beginning, and during the meeting, he made a series of gestures, including pointing, putting his right hand onto his face, etc. In this video analysis, a tracking loss is defined as the wrong tracking result of any body part of subject E. If the tracking analysis corrects itself in 5 frames, we record this wrong process as a tracking loss and mark the *recover* (2^{nd} column in Table 4.1) as 'yes'. Otherwise, we manually correct the tracking result and mark the *recover* as 'no'. In

Table 4.1: Tracking failure reports of AFIT-0601 (camera 3).

Frame	**Recover**	**Description**
2510th	yes	The right hand of subject E was blocked by the head of subject C (the person on the bottom of the view).
2600th	no	The left hand of subject E started pointing to the front, and the right hand took this evidence of its own.
5643rd	no	Both hands of subject E were blocked by the subject D (when the person moved back to her seat).
8740th	yes	The right hand of subject E took the wrong blob when the hand moving downward (the ground color was similar to skin-color).
10165th	no	The left and right hand of subject E were together, and doing gestures, which caused wrong prediction.
12143rd	yes	A person moved in to the view, and sit behind subject E, his head was recognized as the head of subject E for a few frames.
19331st	yes	The right hand of subject E took the wrong blob when the hand moving downward (the bear arm was taken as evidence of hand).

the processing of this video, we have a total of 7 tracking losses, and the result is shown in Table 4.1. In this table, the first column shows the first frame number when the tracking loss occurs, the second column shows whether it automatically recovers to the correct tracking position or not, and the third column shows the reason that causes this processing error.

4.3 More results for meeting-room videos

We also applied this framework to other data sets: AFIT data, NIST (National Institute of Standard Technology) data, and a set of interview data. Figures 4.8 (a) and (b) show the first frame of AFIT-06012004 (camera 6 and camera 8), in which the targeted subject was presenting, and he moved around and made various gestures; Figure 4.8 (c) shows the first frame of AFIT-06012004 (camera 4), in which the targeted subject behaved similarly to the subject E; Figure 4.8 (d) shows the first frame of NIST data, in which, the targeted subject moved around; and Figure 4.8 (e) shows the first frame of the interview data, in which the targeted subject was interviewed and performed some rapid gestures. In Figure 4.8 (a - d), the subject with the red circular marker is our analysis target.

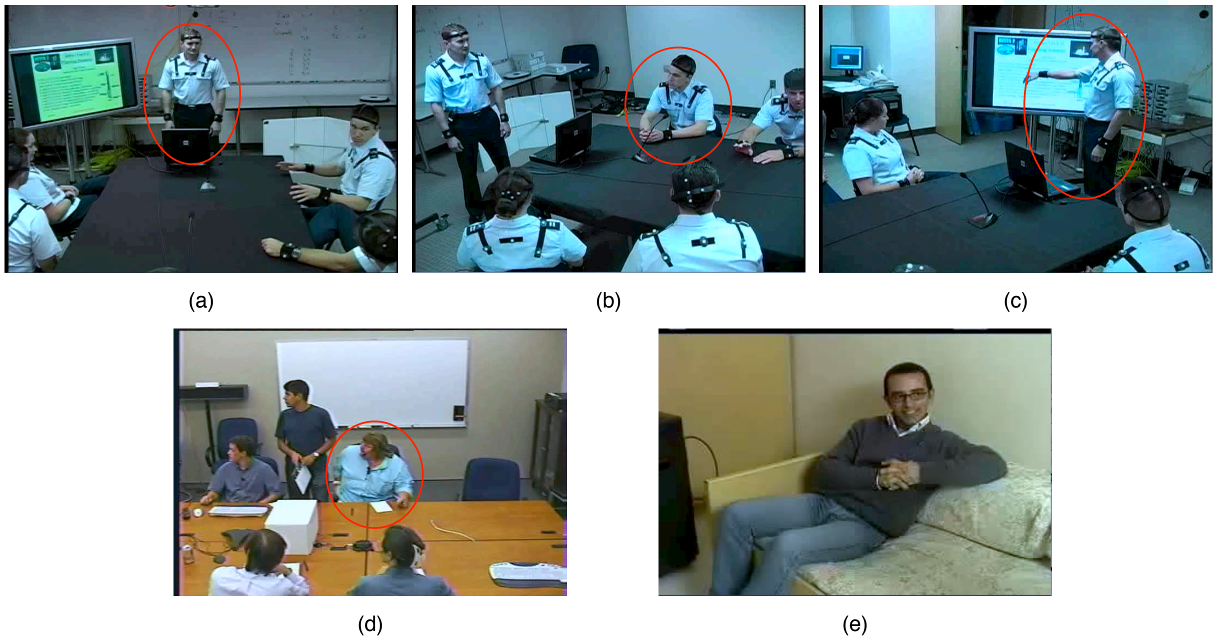

Figure 4.8: More experiments using our agent-based framework. (a) AFIT-06012004, cam 6; (b) AFIT-06012004, cam 8; (c) AFIT-06012004, cam 4; (d) NIST; (e) The interview data.

Table 4.2 shows the tracking failure reports of these five videos. The first column shows the name of the dataset, the second column shows the total number of frames of the video, the third column shows the total number of tracking failures (a tracking failure is a tracking loss that cannot automatically recover itself), and the fourth column describes the changes we made to our tracking system.

Table 4.2: Tracking failure reports of more experiments.

Data	**Total Frames**	**Recover Failures**	**Un-recover Failures**	**Changes of agents**
AFIT-06012004 (C_6)	18080 (10min 2s)	10	22	Low-level feature agents (color model and VCM parameters changed).
AFIT-06012004 (C_8)	13444 (7min 28s)	3	17	No changes.
AFIT-06012004 (C_4)	18071 (10min 2s)	5	15	Low-level feature agents (color model and VCM parameters changed).
NIST data	8465 (4min 42s)	4	41	Low-level feature agents (color model and VCM parameters changed).
Interview data	10194 (5min 39s)	3	18	Low-level feature agents (color model and VCM parameters changed). And the knowledge of hand agent changed to accept fast motion.

When switching cases, we did not change the implementation of our system, where the framework was kept the same. However, the implementations of some individual agents were changed. For example, we needed to re-sample the skin-color and shirt-color for low-level color feature agents. We also needed to change some knowledge of the agents, such as the hand agent, which knows it is far away from the camera, will accept much smaller skin-color blobs, and a hand agent, which knows it is close to the camera, will accept larger skin-color blobs. In the interview video, the person always moves much faster than the persons in the other video. Thus, we have to change the evaluation formula to trust more on the detection results on the current frame than the predictions based on previous frames (w_2 increases and w_1 drops in Equation 4.4).

From all these results, we have a failure rate of 113 unrecoverable failures over 68,245 frames (almost 38 minutes) in total. Generally, there are two reasons for these tracking failures: 1) the ambiguities of the targeted body part and the similar nearby body parts, such as the hand and the bare arm, or the similarities between a body part and an object close by, such as the hand and the yellow table (Figure 4.8 (d)); and 2) the disappearance of the targeted body part. In our system, when the targeted body part is occluded, the agent of this body part will assume it will reappear close to where it disappears. This is the reason that the tracking loss cannot always recover automatically.

Chapter 5

General use of our agent-based framework

In the previous chapter (Chapter 4), we introduced our agent-based framework to track humans in meeting-room videos. In this chapter, we explain another use of this framework and demonstrate an application of tracking players on a soccer field.

5.1 Use of our framework

Our agent-based framework is designed with a three-layer hierarchy, and agents at different layers are designed to target different objects. In the experiments that we showed in the previous chapter, a low-level agent at the bottom layer built an object (such as a skin-color blob) from extracted features (such as skin-color pixels) directly, an intermediate agent at the middle layer built an object (such as a hand) form low-level agents and form coalitions with other intermediate agents, and a high-level agent at the top layer tracked targeted subject by employing intermediate agents. We also have a R/A agent to assist with communication between agents.

5.1.1 Object agents

When applying our framework, the first question to answer is "what is the targeted object in this case". Taking the example that we introduced in Chapter 4, we know that we are going to track the upper body movement of the subject E. We assign a high-level agent (a human agent) to track the target. Then, we need to decompose this targeted object into sub-objects. For example, the subject E can be represented as a combination of a targeted head, a targeted torso and two targeted hands. The intermediate agents are assigned to track these sub-objects. Then, we need to go to each of these intermediate agents to define what kind of features can be used to build this object. For example, the feature can be a skin-color blob, an extracted edge, a motion vector, etc. The low-level agents are then assigned to detect the features that are necessary. By this step, we know what object agents at each layer need to be developed.

Each object agent should be developed with following components (an object-oriented design can increase the reusability of agents):

- *The pre-defined objects.* The agent should be able to recognize the pre-defined objects, such as *skinblob*, *hand* and *circle*, that are used in processing. All the pre-defined objects (such as) are maintained by the agent in the data-type-table.

- *The input information.* For example, a hand agent will take a skin-color blob agent that has the format as $< agt\_id, frm\_id, skinblob, circle, < x, y, r >, null >$. The agent should be able to recognize the format of the object that it takes.

- The output information. For example, a hand agent will process its tracking result as $< agt\_id, frm\_id, hand, < x, y, r, sx, sy, ex, ey >, < agt_0, agt_1, ... >>$.

- *The initial status.* If an object agent is designed to start processing automatically, it should be able to initialize its starting information (such as position) itself when the system starts. If an object agent is designed to start processing with user's input, it should be able to take the user's input and then start processing. When an object agent starts processing, it should register with the R/A agent.

- *The communication with R/A agent.* An object agent should be able to send requests to and to receive messages from the R/A agent. A request can be a message to ask for applicants from the lower layer to fill the open position, or applicants from the same layer to form coalition.

- *The communication with an object agent.* An object agent should be able to receive or send feedback (which is its new tracking result) to the object agent that hires him or is hired by him.

- *The prediction function.* An object agent should be able to predict its information by analyzing its historical information.

- *The processing function and updating status.* An object agent should be able to fuse the information, which is provided by the hired applicant, and its own prediction to provide final tracking result. If an object agent forms a coalition with other agents, it should record the IDs of these agents in its *coalition-list*. If an object agent hires or is hired by another object agent, it should be able to communicate with it.

- *The evaluation function.* An object agent should be able to select hires from applicants, and it should also be able to remove an employee that is not suited for the position from the list.

These components might be slightly changed for different cases. For example, if we apply the model of the 'CardBoard' [34] for human tracking, a body part is represented by a rectangle. A forearm agent will form a coalition with an upper-arm agent, and it normally does not form a coalition with any torso agent. But, if we use the model of the 'PFinder' [79], a body part is represented by a ellipse, and there is no forearm or upper arm agent. Instead, an arm agent is used to represent the whole arm, and it can form a coalition with a torso agent. Since the feature extraction may vary case by case, the processing function of an object agent normally has to be developed for the specific application.

5.1.2 The R/A agent

Typically, the R/A agent maintains all the object agents' information, including the IDs, the type of each agent (such as hand), the current request (open position), and the current post (resume). The R/A agent should be able to receive requests from and send information to object agents. It should be able to filter the information based on different requests before sending. For example, suppose there is a request from a hand agent asking only for a skin-color blob. The R/A agent will only send skin-color agents' information to the hand agent, even if it has other types of agents, such as motion vector agents. An evaluation function can be developed in the R/A agent as part of the filtering function. For example, an allowed error can be part of the request message. Instead of sending all the skin-color agents' information, the R/A agent may remove the ones that have an error larger than the error limit from the the sending list.

5.2 Tracking players on a soccer field

To demonstrate the use and the flexibility of our framework, we apply it to track players on a soccer field. We use the video having 345 frames and track the six players (in two teams ZAR and FCB as shown in Figure 5.1 (c)) appearing in most of these frames, as shown in Figure 5.2. We change our system described in Chapter 4 to track these targeted players in this video.

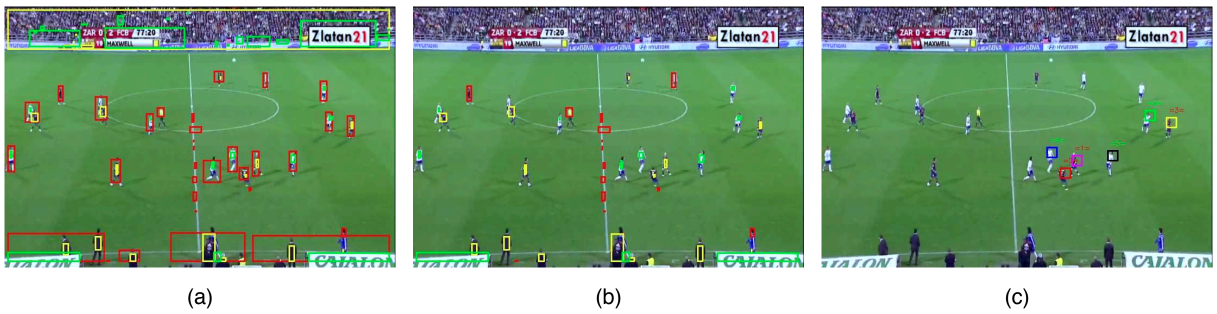
Figure 5.1: Agents for soccer-field video processing.

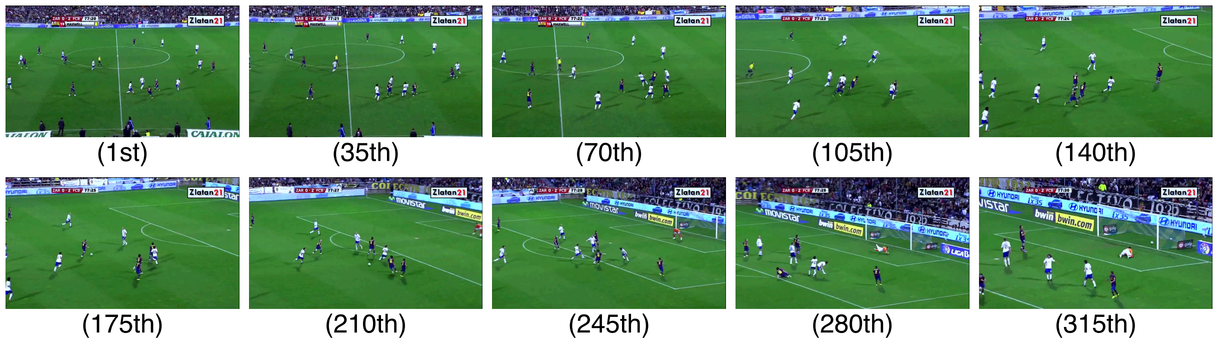
Figure 5.2: Soccer-field video frames.

In this case, a targeted player can be decomposed into an upper-body and an lower-body, both of which can take different color blobs as its evidence of existence. Thus, in this experiment, we can re-use the color-blob agents to extract color blobs in each frame. The color-blob agent takes the image that is processed with the Gaussian model (the Gaussian model has to be trained for this case). Three kinds of color blobs are extracted: the white color blob, which is the shirt color of team ZAR (green rectangles in Figure 5.1 (a)); the blue blob, which is the color of the short pants and parts of the shirt of team FCB (yellow rectangles in Figure 5.1 (a)); and color blobs that are not grass field (red rectangles in Figure

5.1 (a)). For instance, a white color blob agent processes a result as:

$$< agt\_id, frm\_id, white, rect, < x, y, w, h >, null >$$

Since the camera is moving in this case, the motion vector agents are not used.

We define three kinds of intermediate agents: players of team ZAR (ZAR player), players of team FCB (FCB player) and unknown players. The agents that are used to track head, hand and torso can be reused. A player agent is a combination of related color blobs provided by the low-level agents. Obviously, a ZAR player agent is a combination of a white color blob and a color blob that are not grass field (green rectangle in Figure 5.1 (b)). A FCB player is a combination of a blue blob and a color blob that are not grass field (yellow rectangle in Figure 5.1 (b)). An unknown player is a blob with certain size that is neither a ZAR player nor a FCB player (red rectangle in Figure 5.1 (c)).

The high-level agents are defined to track the six targeted players (shown as Figure 5.1 (c)). Each of these agents is initialized in the first frame, and it seeks the spatially and temporally coherent play agent as its evidence. When a high-level agent overlaps with others (the overlap checking is performed in each frame), it uses the evidence of the existence of the agent that blocks it, if it cannot find the evidence of its existence. If the player is out of the camera view, the agent will seek a player agent, which enters into the camera view where it disappears, with the same coalition of color blobs.

Table 5.1 shows the changes made from the system we used in Chapter 4 to track these

Table 5.1: Changes made to track players on a soccer field.

Components	**Low-level**	**Intermediate**	**High-level**
pre-defined obj	no changes	add player object	add player object
input	no changes	no changes	targets player agent
output	no changes	provides results as player agent	no changes
initialization	no changes	no changes	no change
communicate R/A	no changes	no changes	sends request to peers for evidence of existence when occlusion is recognized
communicate obj agent	no changes	a player agent does not form coalition with another agent; instead, it uses the evidence of existence of other agent when occlusion occurs	it uses the evidence of existence of other agent when occlusion occurs
prediction	does not predict with temporal information	does not predict with temporal information	does not predict with temporal information
process and update	no changes	no changes	no changes
evaluation	does not evaluate with temporal information	does not evaluate with temporal information	does not evaluate with temporal information

players on the soccer field. In this table, if an agent can be reused, we mark it with 'no change'. Otherwise, we describe what changes have been made. In this experiment, the result shows that our system can track the six targeted players with no tracking loss that we observed.

Chapter 6

Discussion, conclusion and future directions

In this final chapter, we summarize our framework using an agent-based architecture for human tracking, and we discuss the results of the experiments that we have introduced in the previous chapter, which demonstrates the use of our framework. Finally, we propose some possible research directions and open issues.

6.1 Discussion and conclusion about our framework

Within this research, we introduced an agent-based framework for human tracking in video sequences. This framework provides a structure for developers to build a hierarchical architecture of agents. In this framework, each agent employes the existentialist model, by which it seeks required features to prove its own existence. We apply the job market model to build coalitions between agents. Different from existing agent-based approaches, the job market model allows the tracking process to start from both high-level object estimation and low-level feature extraction. The informational feedback loop also provides a way for lower level agents to improve their tracking results through processing.

In our meeting-room video experiment, we tested our system with 6 sets of meeting-room videos and one interview video. There is a total of 120 failures over 86,235 frames (around 50 minutes) in these videos. These results demonstrate the effectiveness of our agent-based framework. In these frames where tracking is lost, we notice that the main reason is that our agents are not intelligent enough to handle some ambiguities (e.g. an agent cannot distinguish the object and the similar objects around). For example, at the $2,510^{th}$ frame of AFIT-06012004-Camera3, the right hand of subject E was occluded, but his bare right arm was so close to the prediction of the human agent for the right hand. And the hand agent took this similar feature (the same skin-color blob) as its evidence, and this agent was qualified by the evaluation of the human agent that represents subject E. Thus, it took this open position originally for the 'real' right hand. At the $5,481^{st}$ frame of AFIT-06012004-Camera8, there is a yellow table, which can be mistaken for skin-color (shown as Figure 4.8 (a)), behind the subject. Thus, this similar color appearance also causes ambiguities when the hand of the subject is moving close to this table.

We also notice that an agent may over-improve its qualification for a open position. As shown in Figure 4.8 (d), the skin-color is very close to the color of the table and the wall.

The informational feedback from the hand agent asks the skin-color agent to provide more evidence when the original evidence is not strong enough. This 'improved' evidence might be even worse than the original tracking result because of the noise from similar objects. Under this circumstance, when the evaluation from the new loop becomes worse, the higher-level agent in our system will ignore the updates, and takes the previous tracking result. This approach solves some problems, but it will also accumulate the error as well.

As we proposed for the framework design, our architecture takes the advantages of an agent-based system, which distributes the complexity of the entire application. This property is very powerful for complex vision processing problems. Obviously, one reason is that different feature extraction techniques might cause significantly different results. Thus, this framework provides a flexible architecture to guide developers to build their own systems. For example, a facial detection agent can provide strong evidence of the existence of a human's head, when the resolution is good (such as TV news reports), but it will not provide any help to the head agent in a very poor resolution frame (such as surveillance images). The other reason is that when applying various feature extraction algorithms, our framework provides a method for establishing a higher-level coalition of these features, which helps developers fuse them easily. Moreover, in our architecture, we track the target by finding coalitions of its sub-objects, which also helps developers solve their problems hierarchically.

Although we proposed this framework for vision-based human tracking, we can apply it to other tracking problems too. For example, we can apply this framework to help auto-vehicle driving analysis. In this case, edge detection, line detection, sign detection approaches, etc., can be embedded into low-level feature agents. A road agent, line agent, close-car agent and sign agent can be employed as intermediate agents. The high-level agent represents the coalition of these objects on the road that provides the current status around the driven vehicle, which helps driving systems to make decisions. We can also apply it to a tracking task, in which the feature does not only come from image processing. We still use human tracking as an example. Instead of using an image processing approach, we can attach devices (accelerator sensor, such as a Wii remote) onto humans' body parts. Then the low-level agents will use the data from these sensors to compute their results, and the higher-level coalition hierarchy generally remains the same if no new body parts are added.

6.2 Future directions of this research

In an agent-based system, an agent is designed to have properties of an autonomous and adaptive processing unit, and it is able to communicate with others to share information. In our system, we emphasize the adaption and sociability by using the job market model and an informational feedback loop. It will be interesting to research if an agent is able to choose various analysis approaches autonomously. For example, the VCM algorithm, as we described in Chapter 4, handles faster motion better, and the Lucan-Kanade method [42] employs the image-flow equations that handle smooth-and-slow motion well. It will be challenging for an agent to automatically choose the VCM for faster motion and the Lucas-Kanade method for slower motion.

Another possible direction to follow this research is to optimize the job market model in this agent-based architecture. Although the current design helps each agent improve its estimation through the informational feedback loop, it is not guaranteed that this 'improvement' correctly estimates the real-world situation. Thus, we might need a better optimization strategy that keeps the improvement on the correct trajectory.

In this dissertation, we mainly apply our agent-based framework to the meeting-room data, and we also apply our framework to a soccer-field human tracking problem. But, it is necessary to keep researching with other kinds of data, such as surveillance or human interaction data. Since our framework employs the informational feedback loop to improve the estimation results, it is also necessary to optimize this loop for the requirement of real-time processing.

Bibliography

[1] J. Berclaz, F. Fleuret, and P. Fua. Robust people tracking with global trajectory optimization. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 744–750, June 2006.

[2] R. Bryll, R. Rose, and F. Quek. Agent-based gesture tracking. *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, 35(6):795–810, Nov. 2005.

[3] Q. Cai and J. K. Aggarwal. Tracking human motion in structured environments using a distributed-camera system. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(11):1241–1247, 1999.

[4] J. Carranza, C. Theobalt, M. A. Magnor, and H.-P. Seidel. Free-viewpoint video of human actors. *ACM Trans. Graph.*, 22(3):569–577, 2003.

[5] T.-J. Cham and J. Rehg. A multiple hypothesis approach to figure tracking. *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, 2(2):239–245, 1999.

[6] T. K. Cheng, L. Kitchen, T. keung Cheng, L. Kitchen, Z. qiang Liu, Z. qiang Liu, J. Cooper, and J. Cooper. An agent-based approach for robot vision system. In *proceedings of ICSC 95*, pages 489–490, 1995.

[7] CMU. Motion capture research. http://mocap.cs.cmu.edu/, March 2011.

[8] C. Colombo, A. Del, and B. A. Valli. Non intrusive full body tracking for real-time avatar animation. In *Proceedings International Workshop on Very Low Bitrate Video Coding VLBV01, Athens*, pages 119–123, 2001.

[9] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):564–577, May 2003.

[10] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts, and shadows in video streams. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(10):1337–1342, 2003.

[11] A. Danker and A. Rosenfeld. Strip detection using relaxation. *Pattern Recognition*, 12(2):97–100, 1980.

[12] D. Demirdjian, T. Ko, and T. Darrell. Constraining human body tracking. *Computer Vision, IEEE International Conference on*, 2:1071, 2003.

[13] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, 2:126–133, 2000.

[14] M. d'Inverno, M. Luck, and U. . Contributors. Multi-agent systems research into the 21st century. *Knowledge Engineering Review*, 16(3):271–275, 2001.

[15] B. Draper, R. Collins, J. Brolio, A. Hanson, and E. Riseman. The schema system. *The International Journal of Computer Vision*, 2(1):209–250, January 1989.

[16] E. H. Durfee, V. R. Lesser, and D. D. Corkill. Trends in cooperative distributed problem solving. *IEEE Transactions on Knowledge and Data Engineering*, 1:63–83, 1989.

[17] H.-L. Eng, K.-A. Toh, A. Kam, J. Wang, and W.-Y. Yau. An automatic drowning detection surveillance system for challenging outdoor pool environments. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, volume 1, pages 532–539, October 2003.

[18] B. Fang, P.-K. Chung, and F. Quek. Hand tracking using agent-based framework. In *Visual Information Engineering, 2008. VIE 2008. 5th International Conference on*, volume 29-34, August 2008.

[19] B. Fang, L. Xie, P.-K. Chung, Y. Cao, and F. Quek. Full body tracking using an agent-based architecture. *Applied Image Pattern Recognition Workshop*, 0:1–7, 2008.

[20] P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 66–73, 2000.

[21] J. Ferber. *Multi-Agent System: An Introduction to Distributed Artificial Intelligence.* Addison Wesley Longman, 1999.

[22] L. M. Fuentes and S. A. Velastin. *Tracking People for Automatic Surveillance Applications*, chapter Pattern Recognition and Image Analysis, pages 238–245. Springer Berlin / Heidelberg, 2003.

[23] D. M. Gavrila. The visual analysis of human movement: a survey. *Comput. Vis. Image Underst.*, 73(1):82–98, 1999.

[24] P. Guha, A. Mukerjee, and K. Venkatesh. Efficient occlusion handling for multiple agent tracking by resoning with surveillance envet primitives. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, pages 49–56, 2005.

[25] I. Haritaoglu, D. Harwood, and L. S. Davis. W4: Real-time surveillance of people and their activities. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22:809–830, 2000.

[26] K. Hayashi, M. Hashimoto, K. Sumi, and K. Sasakawa. Multiple-person tracker with a fixed slanting stereo camera. *International conference on Automatic Face and Gesture Recognition*, 0:681, 2004.

[27] M. Heikkila, M. Pietikaimen, and J. Heikkila. A texture-based method for detecting moving objects. In *British Machine Vision Conference*, pages 187–196, 2004.

[28] E. Hopkins. Job market signaling of relative position, or becker married to spence. *Discussion paper, University of Edinburgh*, 2005.

[29] B. K. P. Horn and B. G. Schunck. Determining optical flow. *ARTIFICAL INTELLIGENCE*, 17:185–203, 1981.

[30] I. Infantino, M. Cossentino, and A. Chella. An agent based multi-level architecture for robotics vision systems. In *Artificial Intelligence and Information Analysis*, pages 386–390, 2002.

[31] S. Ioffe and D. Forsyth. Human tracking with mixtures of trees. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 690–695, 2001.

[32] A. Jepson, D. Fleet, and T. El-Maraghi. Robust online appearance models for visual tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(10):1296–1311, Oct 2003.

[33] N. Jojic, M. Turk, and T. Huang. Tracking self-occluding articulated objects in dense disparity maps. *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, 1:123–130, 1999.

[34] S. Ju, M. Black, and Y. Yacoob. Cardboard people: a parameterized model of articulated image motion. *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on*, pages 38–44, Oct 1996.

[35] R. Kehl, M. Bray, and L. Van Gool. Full body tracking from multiple views using stochastic sampling. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2:129–136 vol. 2, June 2005.

[36] N. Krahnstoever and R. Sharma. Articulated models from video. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages 894–901, June 27 - July 2 2004.

[37] N. Krahnstoever, M. Yeasin, and R. Sharma. Automatic acquisition and initialization of articulated models. In *Machine Vision and Applications*, pages 218–228, 2002.

[38] F. Kristensen, P. Nilsson, and V. Owall. Background segmentation beyond rgb. In P. Narayanan, S. Nayar, and H.-Y. Shum, editors, *Computer Vision in ACCV 2006*, volume 3852 of *Lecture Notes in Computer Science*, pages 602–612. Springer Berlin / Heidelberg, 2006.

[39] H. Li, F. Karray, O. Basir, and I. Song. A framework for coordinated conrol of multiagent systems and its applications. *IEEE Trans. on Syst., Man, Cybern. A., Syst., Humans*, 38(3):534–548, May 2008.

[40] H. Lim, V. Morariu, O. Camps, and M. Sznaier. Dynamic apperance modeling for human tracking. *Computer vision and pattern recognition*, 2006.

[41] S. Lim, A. Mittal, L. Davis, and N. Paragios. Fast illumination-invariant background subtraction using two views: error analysis, sensor placement and applications. *Com-*

puter Vision and Pattern Recognition, IEEE Computer Society Conference on, 1:1071–1078, 2005.

[42] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of Imaging Understanding Workshop*, pages 121–130, 1981.

[43] M. Luckenhaus and W. Eckstein. A multi-agent based system for parallel image processing. In *in Proceedings of the International Conference on Parallel and Distributed Methods for Image Processing at SPIE's Annual Meeting, Proc. SPIE 3166*, pages 21–30, 1997.

[44] S. Maskell and N. Gordon. A tutorial on particle filters for on-line nonlinear/non-gaussian bayesian tracking. In *Signal Processing Workshop, IEEE Transactions on*, volume 2, pages 1–15, Oct. 2001.

[45] D. McArthur, R. Steeb, and S. Cammarata. A framework for distributed problem solving. *the Proceedings of AAAI-82*, 1982.

[46] S. McKenna, S. Jabri, Z. Duric, and H. Wechsler. Tracking interacting people. *International conference on Automatic Face and Gesture Recognition*, 0:348–353, 2000.

[47] R. A. Michael Luck and M. d'Inverno. *Agent-Based Software Development*. Artech House, London, 2004.

[48] M. Minsky. Minsky's frame system theory. In *Proceedings of the 1975 workshop on Theoretical issues in natural language processing*, TINLAP '75, pages 104–116, Stroudsburg, PA, USA, 1975. Association for Computational Linguistics.

[49] A. Mittal and L. Davis. M2tracker: a multi-view apporach to segmenting and tracking people in a cluttered scene. *International Journal of Computer Vision*, 51(3):189–203, 2003.

[50] T. Moeslund and E. Granum. Pose estimation of a human arm using kinematic constraints. *The 12th scandinavian conference on image analysis*, 2001.

[51] T. B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 8(3):231–268, 2001.

[52] T. B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Comput. Vis. Image Underst.*, 104(2):90–126, 2006.

[53] S. Nestinger, B. Chen, and H. Cheng. A mobile agent-based framework for flexible automation systems. *IEEE/ASME Transactions on Mechatronics*, 15(6):942–951, 2009.

[54] P. Nillius, J. Sullivan, and S. Carlsson. Multi-target tracking - linking identities using baysian network inference. *Computer vision and pattern recognition*, 2:2187–2194, 2006.

[55] K. Okuma, A. Taleghani, N. Freitas, J. Little, and D. Lowe. A boosted particle filter: multitarget detection and tracking. In T. Pajdla and J. Matas, editors, *Computer Vision*

- *ECCV 2004*, volume 3021 of *Lecture Notes in Computer Science*, pages 28–39. Springer Berlin / Heidelberg, 2004.

[56] V. Parameswaran and R. Chellappa. View independent human body pose estimation from a single perspective image. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages 16–22, June-2 July 2004.

[57] R. Plankers and P. Fua. Articulated soft objects for multiview shape and motion capture. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(9):1182–1187, Sept. 2003.

[58] F. Quek and R. Bryll. Vector coherence mapping: A parallelizable approach to image flow computation. In *ACCV '98: Proceedings of the Third Asian Conference on Computer Vision-Volume II*, pages 591–598, London, UK, 1997. Springer-Verlag.

[59] R. Rosales, M. Siddiqui, J. Alon, and S. Sclaroff. Estimating 3d body pose using uncalibrated cameras. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages 821–827, 2001.

[60] A. Rosenfeld, R. Hummel, and S. Zucker. Scene labeling by relaxation operations. *Systems, Man and Cybernetics, IEEE Transactions on*, 6(6):420–433, June 1976.

[61] W. S. Rutkowski, S. Peleg, and A. Rosenfeld. Shape segmentation using relaxation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-3(4):368 – 375, 1981.

[62] M. Saptharishi, J. Hampshire, II, and P. Khosla. Agent-based moving object correspondence using differential discriminative diagnosis. In *Computer Vision and Pattern Recognition, Proceedings, IEEE Conference on*, pages 652–658, 2000.

[63] E. Shakshuki and Y. Wang. Using agent-based approach to tracking moving objects. In *AINA '03: Proceedings of the 17th International Conference on Advanced Information Networking and Applications*, pages 578–581, Washington, DC, USA, 2003. IEEE Computer Society.

[64] Y. Shang and H. Shi. A web-based multi-agent system for interpreting medical images. *World Wide Web*, 2(4):209–218, 1999.

[65] H. Sidenbladh and M. J. Black. Learning image statistics for bayesian tracking. In *In IEEE International Conference on Computer Vision*, pages 709–716, 2001.

[66] H. Sidenbladh and M. J. Black. Learning the statistics of people in images and video. *International Journal of Computer Vision*, 54:183–209, 2003.

[67] M. Spence. Job market signaling. *The quarterly journal of economics*, 87:355–374, Aug. 1973.

[68] J. Starck and A. Hilton. Model-based multiple view reconstruction of people. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, volume 2, pages 915–922, Oct. 2003.

[69] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, pages 246–252, 1999.

[70] L. Sterling and T. Juan. The software engineering of agent-based intelligent adaptive systems. In *ICSE '05: Proceedings of the 27th international conference on Software engineering*, pages 704–705, New York, NY, USA, 2005. ACM.

[71] P. Stone and M. M. Veloso. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3):345–383, 2000.

[72] E. Sudderth, M. Mandel, W. Freeman, and A. Willsky. Visual hand tracking using nonparametric belief propagation. *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*, pages 189–189, June 2004.

[73] C. Taylor. Reconstruction of articulated objects from point correspondences in a single uncalibrated image. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 677–684, 2000.

[74] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, RA-3(4):323–344, 1987.

[75] R. Urtasun and P. Fua. 3d human body tracking using deterministic temporal motion models. In *Lecture Notes in Computer Science*, pages 92–106. Springer Berlin / Heidelberg, 2004.

[76] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, 63(2):153–161, 2005.

[77] G. Welch and G. Bishop. An introduction to the kalman filter. In *in SIGGRAPH 2001*, volume Course 8, 2001.

[78] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152, 1995.

[79] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentl. Pfinder: Real-time tracking of the human body. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19:780–785, 1997.

[80] Y. Xiong, B. Fang, and F. Quek. Extraction of hand gestures with adaptive skin color models and its applications to meeting analysis. *Multimedia, 2006. ISM'06. Eighth IEEE International Symposium on*, pages 647–651, Dec. 2006.

[81] Y. Xiong and F. Quek. Meeting room configuration and multiple camera calibration in meeting analysis. In *ICMI '05: Proceedings of the 7th international conference on Multimodal interfaces*, pages 37–44, New York, NY, USA, 2005. ACM.

[82] J. Yang and A. Waibel. A real-time face tracker. In *Applications of Computer Vision, 1996. WACV '96., Proceedings 3rd IEEE Workshop on*, pages 142–147, 1996.

[83] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13, 2006.

[84] A. Yilmaz, X. Li, and M. Shah. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(11):1531–1536, Nov 2004.

[85] Q. Yuan, S. Sclaroff, and V. Athitsos. Automatic 2d hand tracking in video sequences. *Application of Computer Vision, 2005. WACV/MOTIONS '05 Volume 1. Seventh IEEE Workshops on*, 1:250–256, Jan. 2005.

[86] L. Zhao and C. Thorpe. Stereo- and neural network-based pedestrian detection. In *Intelligent Transportation Systems, 1999. Proceedings. 1999 IEEE/IEEJ/JSAI International Conference on*, pages 298–303, 1999.

[87] Q. Zhou, D. Parrott, M. Gillen, D. M. Chelberg, and L. Welch. Agent-based computer vision in a dynamic, real-time environment. *The Journal of the Parttern Recognition Society*, 37:691–705, 2003.

[88] S. W. Zucker, Y. G. Leclerc, and J. L. Mohammed. Continuous relaxation and local maxima selection: conditions for equivalence. In *Proceedings of the 6th international joint conference on Artificial intelligence - Volume 2*, pages 1014–1016, San Francisco, CA, USA, 1979. Morgan Kaufmann Publishers Inc.