

**Tracking and Analysis of Articulated Motion with an
Application to Human Motion**

A Thesis
Submitted to the Faculty of the Graduate School
of the University of Minnesota
by

Osama Masoud

in Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy

March 2000

© Copyright 2000

by

Osama Masoud

Abstract

Articulated motion is a subset of non-rigid motion in which the object of interest is composed of several rigid components connected to each other by ball and hinge joints. The human body, many animals and insects, and machinery all exhibit such motion. This dissertation addresses the problem of vision-based tracking and analysis of this type of motion. The importance of this problem can be seen in many application domains including surveillance, traffic monitoring, entertainment, user interfaces, medicine, sports, video annotation, and image compression. This dissertation deals with two important subproblems of the general problem: whole-body tracking and motion recognition. In whole-body tracking, the body is tracked as one unit without paying attention to the details of the posture and limbs. Current solutions to this problem suffer from being too sensitive to small changes in the environment. We present a novel approach which reduces these restrictions significantly. This is achieved by separating the concepts of a blob from that of a body and by tracking each independently while maintaining a many-to-many relationship between the two. The approach makes use of the Extended Kalman Filter and outputs trajectory information in world coordinates. The method was tested by tracking pedestrians in a variety of environments and achieved real-time performance and a high degree of robustness. Motion recognition is the high level problem of classifying an action taking place in a video sequence into one of several action categories. Most of the present approaches attempt to perform three-dimensional reconstruction of the articulated shape prior to recognition, which is an inherently difficult problem made even more difficult due to the non-rigidity of the articulated object. We argue that reconstruction is not a necessary step that must precede motion recognition. We present a novel motion-based approach for motion recognition which can be generalized to recognize any articulated motion. In our approach, an action is first represented by a sequence of efficiently computed motion features which are then mapped to a manifold in eigen-space where recognition takes place. Extensive experimentation with human subjects performing different actions demonstrated the effectiveness of our approach.

Table of Contents

Chapter 1: Introduction.....	1
1.1 Articulated Motion.....	2
1.2 Application Domain.....	2
1.3 Tracking.....	4
1.4 Motion Recognition.....	6
1.5 Problem Definition.....	7
1.6 Structure of the Thesis.....	8
Chapter 2: Related Work.....	9
2.1 Whole-Body Tracking.....	10
2.2 Limb Tracking.....	11
2.2.1 2-D Limb Tracking.....	11
2.2.2 3-D Limb Tracking.....	12
2.3 Motion Recognition.....	13
2.3.1 Recognition Based on 2-D Tracking.....	13
2.3.2 Recognition Based on 3-D Tracking.....	14
2.3.3 Recognition Based on Motion Features.....	15
2.4 Hand Tracking, Hand Gesture Recognition and Lip-reading.....	16
Chapter 3: Whole-Body Tracking.....	17
3.1 Image Subtraction.....	18
3.2 Blobs Level.....	20
3.2.1 Blob Extraction.....	20
3.2.2 Blob Tracking.....	20
3.3 Pedestrians Level.....	23
3.3.1 Pedestrian Model.....	24
3.3.2 Pedestrian Tracking.....	25

3.3.2.1	Relating Pedestrians to Blobs	25
3.3.2.2	Prediction	25
3.3.2.3	Calculating Pedestrian Positions	26
3.3.2.4	Estimation	27
3.3.2.5	Refinement	28
3.4	Experimental Results	29
3.4.1	Other Applications: Monitoring Weaving Sections	34
3.5	Summary	37
Chapter 4:	Motion Recognition	38
4.1	Feature Image	39
4.2	Learning and Recognition	41
4.2.1	Magnitude and Size Normalization	42
4.2.2	Principle Component Analysis	43
4.2.2.1	Overview	43
4.2.2.2	PCA for Action Recognition	44
4.2.3	Recognition	46
4.2.3.1	Distance Measure	46
4.2.3.2	Classification	46
4.3	Action Data	47
4.3.1	Data Selection	47
4.3.2	Acquisition	50
4.4	Experimental Results	51
4.4.1	Classification Experiment	51
4.4.2	Parameter Selection	55
4.4.2.1	Resolution	55
4.4.2.2	Number of Training Images	56
4.4.3	Complexity	56
4.5	Summary	59

Chapter 5: Conclusions	60
5.1 Summary	60
5.2 Contributions	61
5.2.1 A Robust Articulated Whole-Body Tracking Method.....	61
5.2.2 An Application to Vehicle Tracking	61
5.2.3 A General-Purpose Efficient Blob Tracking Method	61
5.2.4 Limb Tracking not a Pre-requisite for Action Recognition	62
5.2.5 A General Articulated Motion Recognition Method.....	62
5.2.6 Testing Using a Large Action Data Set	62
5.3 Future Work.....	62
5.3.1 Tracking	62
5.3.2 Real-time Implementation of Action Recognition	63
5.3.3 Recognition from a General View Angle.....	63
 Bibliography.....	 64

Chapter 1

Introduction

A great deal of research in computer vision has dealt with the analysis of temporal image data. Video provides very rich information when compared with information obtained through other sensors. It is no surprise that humans obtain more than 70% of their sensory information through their eyes. Applications that require interaction with dynamic environments, such as robotic visual servoing, obstacle detection, collision avoidance and surveillance, can benefit the most from video as a source of sensory information. Video has become even more suitable in the recent years due to the availability of high quality and continually cheaper acquisition devices (CCD, CMOS, and digital cameras). In addition, the dramatic increase in computer processing power, the increase in storage devices capacity and decrease in their cost, and the increase in bandwidth for transferring media have all contributed to making video processing a more realistic solution than it used to be.

Another reason in favor of using temporal image data is that psychophysical evidence suggests that biological visual systems are well adapted to process temporal information. The human visual system is very sensitive to motion. This is also true for many animals. Stationary objects are much harder to detect than moving objects. Many animals use this fact as a camouflage strategy.

In this thesis, we deal with two related, yet different, aspects of motion: tracking and recognition. Tracking attempts to identify certain spatial and dynamic quantities of motion through time. Motion recognition is a higher level process which attempts to identify the type of motion taking place. More details will be given in the problem definition section.

1.1 Articulated Motion

In computer vision, motion can be classified into rigid and non-rigid motion. Articulated motion is considered an important subset of non-rigid motion where the object of interest is composed of several rigid components connected to each other by ball and hinge joints. The human body, many animals and insects, and machinery all exhibit articulated motion. Because of the wealth of applications that involve the human body, most of the research that dealt with articulated motion was aimed at human body motion. For the sake of usefulness, the work introduced in this thesis also deals with human body motion although it can be generalized to any articulated motion. Because of the high articulation and flexibility of the human body, its motion can be very complex. Analysis of this motion is made even more difficult because of the effect of clothing on the appearance of the body. Another source of difficulty is that occlusions of some parts and reappearance of other parts occur very frequently as the body performs even a simple action like walking. These difficulties make the problem of dealing with human body motion a challenging one.

1.2 Application Domain

Vision-based tracking and analysis of articulated motion has the advantage of being non-intrusive. This is especially important when dealing with humans. In some applications, the only alternative would be the placement of markers on many parts of the body to track motion. Other applications may require the use of active sensors such as infrared and ultra-sonic sensors. The convenience and safety of visual sensing comes at the cost of difficulty in achieving the desired goal. Many vision-based solutions do not provide a satisfactory solution or in some cases, are too sensitive to small changes in the environment. However, this field is relatively recent and many research tracks have yet to be explored.

There are many applications for visual tracking and analysis of articulated motion. In *surveillance*, a human operator has been traditionally used. Automating surveillance can be highly desirable in cases where using a human operator is not feasible. Automated surveillance can be used to detect intruders in a restricted area or find suspicious activities. Parking lots, department stores and ATM machines are a few examples where such a system can be used. Simple motion detectors suffer from the problem of giving too many false positives. A human, a dog, or a moving tree due to the wind blowing will all trigger an alarm. The system needs to be intelligent enough to distinguish between a human and other moving objects. Furthermore, the system should be able to distinguish a suspicious activity from a regular one. Surveillance can also use trajectory information of pedestrians to detect and classify events and behaviors [36,41]. Automated surveillance can also be used in factories to monitor machinery as an alternative to a more expensive manual system. Breakdowns can be detected early when one of the machines starts moving in a different way than it is supposed to or stops moving completely. Early detection is very

critical in situations like this.

Pedestrian *traffic* monitoring is another demanding application. In traffic control, tracking pedestrians at intersections can be used to both increase safety and optimize traffic timing. Safety can be increased by either providing extra crossing time for people who need extra time or by providing a warning signal to drivers indicating the presence of pedestrians in the crosswalk. Optimization is a direct outcome of dynamically adjusting the walk signal delay. When there are no pedestrians, the walk signal should not be part of the cycle. Pedestrian traffic monitoring can be used to count pedestrians and provide other quantities such as the different paths pedestrians take and most visited locations. This type of application is particularly useful for retailers and shopping centers who can use the data to improve operating efficiency, evaluate performance, and charge hourly for retail spaces.

In the field of *entertainment*, there are several interesting applications. Computer-generated movies and TV series are becoming increasingly popular. During the making of these shows, animation of synthetic actors is done using motion-capture technology, in which the animators put a suit on an actor and assign his/her movements to a computer-generated character. Vision-based tracking of the human body can eliminate the need for the suit, thus giving the actor more flexibility and allowing motion-capture to be carried out in an uncontrolled environment such as outdoors. Computer games and virtual worlds are two other applications with similar demands. In computer games, interfaces that increase the player involvement are desirable. The player can actually use his/her body to control the game by kicking in the air for example. In virtual worlds, a more realistic presence of the person can be achieved by capturing the person's motion, actions, and facial expressions. For children, a play-space with virtual characters that interactively respond to the children's actions can provide an educational and entertaining experience [7].

A closely related application domain is *user interfaces*. Vision-based lip-reading is one such application which can be used to complement a speech recognition system for more robust recognition. Another very useful application is real-time interpretation of sign language without the need to wear expensive wired gloves. This, in conjunction with a speech recognition system, can help the hearing impaired communicate with people who do not know sign language. Gesture-driven user interfaces are another potential application where the user can use gestures to control graphical objects such as performing pointing and selecting in a menu-driven application.

In the *medical field*, analysis of gait and posture helps in detecting abnormalities in patients. Vision-based systems can track the human body and provide the necessary trajectory data for this analysis. Other related applications are kinesiological analysis, ergonomic designs, and biomechanical simulations.

Sports is another application domain. Athletic training sometimes involves the comparison of the trajectory of certain body parts to a mathematical model of the optimum

motion. Retrieval of such a trajectory is usually a tedious process which involves manually locating the joint positions in every frame. Automation of this process would be desirable. Another application would be a personalized training system, such as a virtual aerobic instructor, which provides feedback to the user performing a certain skill.

Automated sports *video annotation* can benefit entertainment companies, newscasters, and sports teams. Video annotation, or context-based indexing of video, makes it possible to textually search the video database for events. In sports videos, the interesting events usually involve human actions which makes the application a suitable human action recognition application. A typical query would be “find segments where a player does a scissors kick” in a soccer video. Another use of video annotation is in choreography of ballet where a large vocabulary (about 800 names of steps) is used to describe it.

Finally, in the domain of *image compression*, several compression improvements can be achieved. For example, in teleconferencing, tracking the face can allow putting more emphasis on the quality of face region and less emphasis elsewhere. Alternatively, tracking the face in 3D can provide a very short representation in terms of pose and deformation parameters. These parameters can be transmitted to the other end and used to reconstruct the face image. A similar idea can be used in transmitting sign language gestures as opposed to the more ambitious goal of sign language interpretation mentioned earlier. Tracking parameters of the hands and fingers are sufficient to reconstruct a virtual hand model on the other side.

In each of the previous applications, a vision-based solution will require either tracking the subject (or parts of the subject) of interest or recognition of an action. These two requirements are described in more detail below along with the motivation behind the choices of the problems we addressed in this thesis and the solution methods we employed.

1.3 Tracking

Tracking implies the recovery of trajectories. In tracking humans, there are different levels at which tracking can be performed. At the highest level, the whole body is tracked without paying attention to the details of the posture and limbs. At a lower level, the posture and limbs are tracked. At an even lower level, one or two parts of the body (such as hands) are tracked. The finest level would be tracking the fingers of a hand or facial features. The type of application determines the tracking level. Table 1 summarizes the applications mentioned earlier and the required tracking level. The different levels are denoted as (1) *whole*, (2) *limb*, (3) *head and hand*, and (4) *face and finger*. *Action* denotes that action recognition is required for the corresponding application. Action recognition will be discussed in the next section.

Domain	Application	Requirement
Surveillance	Intruder detection	whole
	Suspicious action detection	action
	Machinery monitoring	action
Traffic Applications		whole
Entertainment	Synthetic actor animation	limb or face
	Games	limb or action
	Virtual worlds	limb, face, or action
	Children interactive play-space	action
User Interfaces	Lip-reading	action
	Sign language interpretation	action
	Gesture-driven control	finger, hand, or action
Medical Applications		limb
Sports	Athletic training	limb
	Personalized training	action
Video Annotation Applications		action
Image Compression	Teleconferencing	head or face
	Sign language coding	finger or hand

Table 1.1: Application Domain.

Part of the scope of this thesis deals with the highest level of tracking: tracking the whole body as one unit. There are several reasons for our choice of this problem. In addition to the fact that it is an important issue in some applications, it is a problem whose solution needs to be as general as possible. Applications that require whole body tracking, whether it is surveillance or traffic-related, involve mostly outdoor settings where very little environment control is possible. Changing weather and lighting conditions are some examples of inevitable factors. A satisfactory solution to this problem needs to handle these changes. In addition, controlling the subjects (e.g., having them wear a certain color or behave in a certain way) is not applicable in this case. Controlled environments may be applicable to other areas that require different tracking levels to reduce the complexity of

the problem. For example, in sign language interpretation, the subject can be required to have a certain pose and wear colored gloves. The lighting conditions can also be controlled. Many solutions have been presented for tracking the whole body as a single unit but the majority imposed impractical restrictions. We chose this problem because of the lack of a general solution, in an attempt to present a solution with as few restrictions as possible. Another motivation for this choice is that solving this problem can provide a useful tool to deal with the second problem we address in this thesis, namely, action recognition. The usefulness stems from the fact that locating the subject of interest will help in confining the recognition effort to that location.

In tracking humans as single units, the drawback in many of the solutions that have been presented originates from an implied assumption that there is a clean one-to-one correspondence between extracted features, or blobs, and pedestrians in the scene. Arbitrary images may not satisfy this assumption due to clutter, color of clothes, and weather conditions among other factors. In our solution, we allow maximum flexibility by allowing this relation to be many-to-many. Our method tracks features and pedestrians at different levels and dynamically updates this relation.

1.4 Motion Recognition

Given a number of predefined actions, recognition of an articulated object motion is the problem of classifying the object motion into one of these actions. Normally, the set of actions has a meaning in a certain domain. In sign language for example, the set of actions corresponds to the set of possible words and letters that can be produced. In ballet, the actions are the step names in one of the ballet notation languages.

The study of human body motion perception by the human visual system was made possible by the use of the so-called moving light displays (MLDs) first introduced by Johansson in 1973 [39,40]. Johansson devised a method to isolate the motion cue by constructing an image sequence where the only visible features are a set of moving lights corresponding to joints of the human body. Figure 1.1 shows an example.

He found that when a subject was presented an MLD corresponding to an actor performing an activity such as walking, running, or stair climbing, the subject had no problem recognizing the activity. It took less than 200 milliseconds to identify the activity. The subjects were not able to identify humans when the lights were stationary. Cutting and Kozlowski [15,47] demonstrated that the gender of the walking person and the gait of a friend can be identified from MLDs. It was later shown that subjects can identify more complex movements such as hammering, box lifting, ball bouncing, dancing, greeting, and boxing [19]. Two theories on how people recognize actions from MLDs have been suggested. In the first theory, people perform three-dimensional reconstruction of the

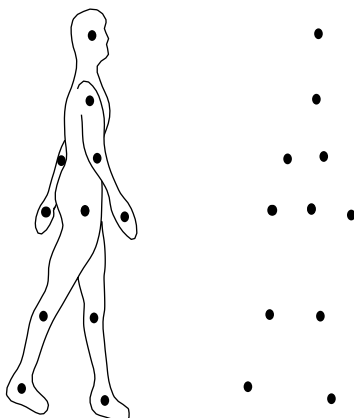


Figure 1.1 A moving light display with and without the human body outline.

object and then use that to recognize the action. In the second theory, people utilize motion information directly without performing reconstruction.

Consequently, in computer vision, the problem of motion recognition has been attempted using two approaches. The first is a reconstruction-based approach that attempts to do structure-from-motion first and then use the recovered three-dimensional information in recognition. The second is a motion-based approach that uses motion parameters directly without performing any reconstruction. The shape-from-motion problem in its general sense is a classic computer vision problem. It involves the recovery of three-dimensional motion parameters and the relative depth map. When dealing with an articulated object, this process becomes more complex due to the non-rigidity of the object. To date, the problem of articulated object shape-from-motion, which is the same as the problem of limb tracking and posture recovery described in the previous section, has not been satisfactorily solved. The vast majority of the work related to articulated motion, however, falls under the reconstruction-based category, probably more motivated by the applications that make use of the reconstruction phase alone than by psychophysics. In this thesis, we argue that reconstruction is not a necessary step that must precede motion recognition. We present a novel motion-based approach for motion recognition which unlike other motion-based approaches, can be generalized to recognize any articulated motion.

1.5 Problem Definition

Tracking and analysis of articulated objects has many practical applications, most of which involve the human body. The desired tracking and analysis outcome depends on the specific applications. Two of these outcomes are dealt with in this thesis: tracking the body as a whole and recognizing actions. Therefore, we define our problem as the problem of

tracking and action recognition of the human body. The problem consists of the following two sub-problems:

1. Tracking the human body as a whole: It is the problem of real-time recovery of motion trajectories in world coordinates of moving pedestrians in a scene bounded by the view field of a single stationary camera. To alleviate the recovery of world coordinates, camera parameters such as resolution, focal length, height and tilt angle are assumed to be known. No restrictions are made on the camera position, the number of pedestrians, pose, motion direction, speed, or occlusions. There is also no restriction regarding weather conditions or the existence of other possibly moving objects such as oscillating trees.
2. Action recognition: It is the problem of classifying the action performed by a human in a video sequence. No other sensory input such as three-dimensional joint locations is allowed. The domain of possible actions is provided along with samples of each action. The technique is required to be capable of generalization to any domain with any set of actions. The actions performed may have variable durations. The same action is also allowed to have different speeds. Moreover, temporal alignment of actions is not required. Recognition should not be influenced by the actor, his/her height, shape or style in performing the actions.

1.6 Structure of the Thesis

The chapters of the thesis are organized as follows:

Chapter 2 provides a review of the related work that has been done in this field. The review uses our own classification of related work, a classification that shows where our techniques stand with respect to other work. The shortcomings in other techniques, which we attempt to resolve, are pointed out.

Chapter 3 presents our tracking method and results. The goal is to track pedestrians as single units (without tracking the limbs). The chapter describes the different processing levels which are image-, blob-, and pedestrian-level, and the interactions among them. Results of tracking under various conditions are presented along with an application to a different domain which shows the applicability of this method to rigid motion tracking as well.

In Chapter 4, the action recognition approach is explained. After explaining the extraction of features which are used in recognition, the training and matching techniques are explained. Results and analysis of testing on a large data set of actions are also presented in the chapter.

Finally, summary, contributions, and future work are given in Chapter 5.

Chapter 2

Related Work

Articulated motion has been a subject of interest in many different fields. In robotics, researchers utilize control techniques in an attempt to build legged robots which can perform a variety of tasks such as walking and running. Biomechanics deals with the problem of providing data that describes as close as possible human motion. In psychophysics, the perception of articulated motion by the human visual system has been studied. In computer graphics, the emphasis is on motion synthesis, or animation. Finally, in computer vision, the focus is on computational techniques for tracking and perception of articulated motion.

This chapter provides a review of the body of work that has been done in computer vision in relation to articulated motion. We classify the work into three categories, which reflect the organization of this chapter:

1. Whole-body tracking: This deals with tracking the articulated body (i.e., a pedestrian) as a single unit, without the identification or tracking of the individual parts of the body.
2. Limb tracking: Here, the emphasis is on the individual body parts and the recovery of the structure of the body. Limb tracking techniques can be either two-dimensional or three-dimensional, depending on whether the recovered parameters are in image coordinates or in world coordinates.
3. Motion recognition: Methods belonging to this category attempt to identify the action as one of the actions in a database. Motion recognition can rely on two- or three-dimensional tracking information. Alternatively, it can utilize motion features directly.
4. Hand Tracking, Hand Gesture Recognition and Lip-reading.

The work described in this thesis belongs to whole-body tracking (Section 2.1) and motion recognition using motion features (Section 2.3.3).

2.1 Whole-Body Tracking

Several attempts have been made to track pedestrians as single units. Baumberg and Hogg [5] used deformable templates to track the silhouette of a walking pedestrian. The advantage of their system is that it is able to identify the pose of the pedestrian. Tracking results were shown for one pedestrian in the scene and the system assumed that overlap and occlusions are minimal [4]. Another use of the silhouette was made by Segen and Pingali [68]. In their case, features on the pedestrian silhouette were tracked and their paths were clustered. The system ran in real-time but was not able to deal well with temporary occlusions. Occlusions and overlaps seem to be a primary source of instability for many systems. Rossi and Bozzoli [66] avoided the problem by mounting the camera vertically in their system which aimed to mainly count passing pedestrians in a corridor. Such a camera configuration, however, may not be feasible in some cases. Occlusions and overlaps occur very commonly in pedestrian scenes; and cannot be ignored by a pedestrian tracking system. The use of multiple cameras can alleviate the occlusion problem. Cai and Aggarwal [9] tracked pedestrians with multiple cameras. The system, however, did not address the occlusion problem in particular but rather how to match the pedestrian across different camera views. The switching among cameras was done manually.

Smith *et al.* [72] performed pedestrian detection in real-time. The system used several simplistic criteria to judge whether the detected object is a pedestrian or not but did not actually track pedestrians. Sullivan *et al.* [74] used deformable models to track pedestrians. Initialization and tracking was performed using segmented pedestrians from the background. In general, handling occlusions is difficult using deformable models. Shio and Sklansky [70] presented a method for segmenting people in motion with the use of correlation techniques over successive frames to recover the motion field. Iterative merging was then used to recover regions with similar motion. The method was able to deal with partial occlusions. The assumption was that pedestrians do not change direction as they move. A disadvantage of this method is the high computational cost of the correlation and the iterative merging steps. An interesting approach which was presented by Heisele *et al.* [32] is based on their earlier work on color cluster flow [31]. An image is clustered into regions of similar color. In the subsequent images, the clusters are updated using a k-means clustering algorithm. Assuming that the pedestrian legs form one cluster, a step to recognize legs enables the system to recognize and track pedestrians. This was done by checking the periodicity of the cluster shape and by feeding the gray scale images of the legs into a time delay neural network. The advantage of this approach is that it works in the case of a moving camera. Unfortunately, due to several costly steps, real-time implementation was not possible.

2.2 Limb Tracking

2.2.1 2-D Limb Tracking

Very early work by Akita [1] attempted to track body parts using a model-based approach. Body parts were recognized in the order: legs, head, arms, and then trunk. The author assumes that the legs are the most stable to detect and the trunk is the least stable. Correlation was used to estimate the body parts position. When correlation fails, a set of predefined key frames are used to give a rough estimate of the position of the person. There are many assumptions in this method in addition to many simplifications such as foreground-background straightforward segmentation. Also, the motion was assumed to be known beforehand.

Leung and Yang [49] used a model that consisted of five ribbons and a trunk to label the different body parts. The features used were image edges and moving edges. Labeling was done according to some constraints such as ratios of length to width of various body parts. The choice of thresholds which may vary from one person to another is critical to this method. Chang and Huang [11] also used ribbons to track arms and legs. Labeling was simplified by imposing some assumptions. For example, the arms are assumed to be located above the legs in the scene.

Ju *et al.* [42] proposed a 2-D model-based approach to track the leg closer to the camera using connected planar patches. Their model was parameterized to accommodate for the deformations the patches undergo as the leg moves.

Niyogi and Adelson [54] tracked the legs of walking people by analyzing the X-T plane at different Y values. For a Y value around the legs area, a braided pattern appears on the X-T plane corresponding to the legs motion. Deformable contours (snakes) are used to fit the pattern. A simple model of a 5-stick figure (two sticks for each leg and one for torso) was used by simple line fitting. Gait recognition was performed by comparing the contour data of these snakes to distinguish different walkers. In their work, they assumed a rough knowledge of the heights of the head and feet and that the person is walking parallel to the image plane at a constant speed. They were able to achieve an 81% rate of classifying the walking action as belonging to one of five people.

Wren *et al.* [80] developed a real-time limb tracking system called “Pfinder”. It is based on blob tracking where each blob corresponds to a different body part (head, hands, feet, shirt, and pants) and has certain statistical properties. Initial labeling is done by having the person face the camera and stand in a specific pose. The system ran in real-time and handled occlusions robustly. However, the authors did not mention how sensitive the system is when body parts have the same color. The system was limited to tracking one person in the view of the camera with a static background.

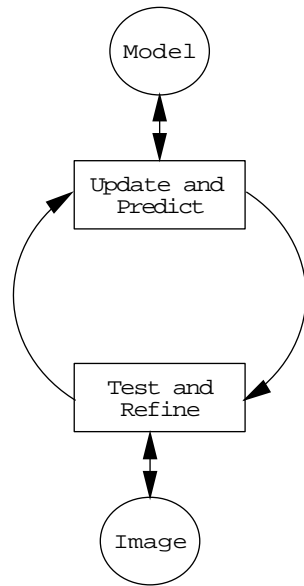


Figure 2.1 General interpretation cycle.

2.2.2 3-D Limb Tracking

Much of the research in human body and articulated motion tracking has been in the area of 3-D tracking. The results in this area, however, are still limited. Most approaches have made use of a predefined 3-D model. A model consists of primitives such as generalized cylinders [21,26,34,65] or superquadrics [24,43]. Superquadrics are generalizations of ellipsoids which have squareness parameters along the axes. The shape parameters of the 3-D model (such as limb lengths, etc.) were almost always assumed to be constant and the emphasis has been on pose recovery. The models were mainly specified by articulation points. Use of further constraints such as joint angle limits, collision, or dynamic constraints such as balance and gravity has been limited. Some methods used general articulated motion constraints such as in-plane rotation [33] and fixed-axis rotation [79]

Some methods [21,24,34,55,56,58,65] adopted the general scheme for model-based approaches proposed by Kanade [44]. The general idea of this scheme is to have a feedback loop as depicted in Figure 2.1. The image and the model are treated as two different domains. The model supplies a set of hypothesis by predicting the future. These hypotheses are tested for validity, refined and sent back to the model domain so that the model is updated. Due to the complexity of the human body motion, a prediction process may generate a huge number of hypotheses. Testing therefore involves performing a search in a high dimensional space. If testing is not sufficiently reliable, the wrong hypothesis may be

chosen. This problem was sometimes handled by either reducing the number of predicted hypotheses by restricting the movement freedom of the model [65] or making testing more reliable by assuming a easy segmentation or a clean input such as a synthesized image sequence [56]. Other approaches did not use the model feedback loop [13]. The flow of information was therefore in one direction: from the image domain to the model domain.

An alternative to this scheme using inverse kinematics was used by other methods [26,78,82,84]. This technique is often used in robot control theory. There is a nonlinear mapping which maps the state space comprised of pose configuration and joint angles, etc. to the image space. It involves several coordinate transformations at articulation points and perspective projection. Inverse kinematics attempts to invert this mapping by linearization and gradient-based optimization to recover the changes in state parameters. Generally speaking, techniques based on inverse kinematics are computationally more efficient because of their use of gradient-based optimization. However, they are more likely to get stuck in a local minimum due to the highly nonlinear nature of the mapping. In addition, measurements may not be smooth for fast motions using standard video sampling rate.

Multiple cameras have been used by some methods [24,43,2] to reduce the effect of occlusions and much less often to utilize triangulation in 3-D information recovery. However, the majority still cannot handle significant occlusions. Azarbajani and Pentland [2] proposed a technique to recover 3-D coordinates of the head and the two hands using a stereo setup. The features used were blobs which were obtained using the same technique as in [80]. Blobs can be considered as intermediate-level features. Other intermediate- and low-level features that were used include edges [26,24,65] or regions [78,82]. Some methods used higher-level features such as joints. The assumption was that the location of joints is provided or easily segmented [13,35,56,69,84]. This is a strong assumption since the detection of joints is generally a difficult task. Color and texture features were rarely used. Use of these features may be necessary if tracking people with loose-fitting clothes is desired.

Finally, there are other issues that still need to be addressed in 3-D tracking. One such issue is model initialization. Most methods do not handle initialization of the model and deal only with incremental pose estimation. Another issue is tracking of more than one person.

2.3 Motion Recognition

2.3.1 Recognition Based on 2-D Tracking

Goddard [25] used tracked point from Moving Light Displays (MLDs). The tracking information was assumed to be provided. The features used were relative angles and rela-

tive angular velocities of joints. Low level features were integrated to form higher level features. A connectionist model was used for recognition. Successful classification of walk, run, and skip action was achieved using this approach.

Guo *et al.* [27] classified actions into three categories: walking, running and everything else. They performed two-dimensional tracking of the silhouette of the body which is extracted using background subtraction. A stick figure is fitted to the silhouette to track the motion. Parameters extracted from the stick figure are then used for action recognition. This is done by performing Fourier transform on one period of these parameters to obtain the first few frequency components. The assumption is that one cycle of the action can be reliably segmented. Recognition is done by feeding the frequency components into a neural network. They used a number of samples, both real and synthetic, to achieve a 97.3% recognition rate.

Yacoob and Black [81] used 2-D tracking data in the form of parameterized models of the tracked legs. The recovered parameters over the duration of the action were then compressed using principle component analysis. Recognition was done by finding the nearest neighbor in eigenspace taking into consideration temporal shifts and differences in duration. Four actions were classified with an 82% accuracy.

Finally, Rangarajan *et al.* [63] matched motion trajectories using scale space. They used speed and direction parameters rather than locations to achieve translation and rotation invariance. The input was manually tracked points on several parts of the body performing the action. Given two speed signals, matching is performed by differencing the scale space images of the signals. Direction matching is done in the same way. In the experiments, a good match score was given for two walking actions performed by the same person, while a lower score was given when the person was different.

2.3.2 Recognition Based on 3-D Tracking

Upon successful 3-D tracking, motion recognition can make use of any or the recovered parameters such as joint coordinates and joint angles. Work done in this area has been limited to inputs of the form of MLDs obtained by placing markers on various body joints which are tracked in 3-D. Ideally, the 3-D tracking data would come from one of the 3-D limb tracking methods explained above.

Campbell and Bobick [10] used phase-space representation to model actions. In this representation, parameters are plotted without the time axis forming a series of points in space. Recognition was performed by considering two-dimensional subspaces of the phase space. This representation is useful because it provides invariance to variation in speed. Another way of dealing with differences in speed variations is by using the Dynamic Time Warping (DTW) technique [52]. In this technique, given two temporally aligned patterns of lengths M and N elements, an optimal pairing of the element of the

two patterns that preserves order is generated. Gavrilu and Davis [24] used DTW to distinguish three actions.

2.3.3 Recognition Based on Motion Features

Yamato *et al.* [83] used Hidden Markov Models (HMMs) to distinguish different tennis strokes. A segmentation step through background subtraction was first performed. The resulting image is then binarized with an appropriate threshold and then normalized to obtain a binary foreground image. This image is then divided into a grid of equal rectangular regions and the number of pixels in each region is counted. A feature vector is formed for every frame based on these numbers. For every action, a new HMM was learned. Recognition is done simply by selecting the HMM that was most likely to generate the given sequence of feature vectors (output symbols). The main advantage of such an approach is that it is easy to add a new action to the database. This is done by training an HMM and adding it to the set of HMMs. The approach, however, was sensitive to the shape of the person performing the stroke. Use of motion features rather than spatial features may have reduced this sensitivity.

Davis and Bobick [17] used what they called *motion-history* images (MHIs). An MHI represents motion recency where locations of more recent motions are brighter than older motions. A single MHI is used to represent an action. A pattern classification technique using seven Hu moments of the image was then used for recognition. They presented results of recognizing aerobic exercises performed by two actors, one for training and one for testing. The choice of an appropriate duration parameter used in the MHI calculation is critical. Temporal segmentation was done by trying all possible parameters. The system was able to successfully classify three different actions: sitting, arm waving and crouching. It remains to be seen how well this representation can capture long cyclic actions. For example, if two motions at different times take place in the same area, the latter will destroy the history information of the former.

Motion information extracted directly from the image sequence was also utilized by Polana and Nelson [61]. In their work, they used normal flow (the component of the flow field which is parallel to the gradient). The feature vector in their case was computed by temporally dividing the action into six divisions and finding the normal flow in each. Furthermore, each division is spatially partitioned into 4 by 4 cells. The summation of the magnitude of the normal flow at each cell was used to make up the feature vector. Thus, the vector was composed of 96 values. Recognition was done by finding the most similar vector in the training set using nearest centroid algorithm. The duration of the action was determined by calculating a periodicity measure [60]. This helps in correcting for temporal scale but not temporal translation. To overcome this problem, their technique matched the feature vector at every possible phase shift (six in this case). They tested their method using six different activities each performed several times by the same person and one activity performed by a toy frog. The activities were: walking, running, swinging (on a

swing), skiing on a skiing machine, exercising on a machine, jumping jacks. Half the performances were used for reference and the other half was used for testing. A different person was used to supply samples for one of the activities (walking). The method worked fairly well which shows the discriminatory power of the motion features used. More testing, especially using different actors, would be needed to demonstrate the usefulness of this method. Moreover, the actions used, with the exception of walking and running, were very different in terms of spatial scale which may have contributed to the recognition success. Most of the computation time was spent in normal flow computation.

2.4 Hand Tracking, Hand Gesture Recognition and Lip-reading

The problems of hand tracking and hand gesture recognition had a particular interest motivated by demanding applications in gesture-driven control and sign language coding and interpretation. Consequently, there was an emphasis on the tracking problem [14,20,30,48,64] as well as on the recognition problem [8,12,16,18,23,37,62,73,76]. Many of the ideas mentioned in the previous sections were used here. In addition, various constraints were imposed to simplify the problem. But unlike the constraints on tracking and recognition of human body motion, these are more acceptable due to the nature of the applications. For example, it is easier to require someone using a gesture-driven control application to put on colored gloves than to require people to wear a certain color in a surveillance application.

In lip-reading, the goal is to classify spoken words into one of the classes specified in a database. This problem is fundamentally different than human body motion analysis because the motion of the mouth is closer to deformable motion than to articulated motion. A few approaches addressed lip-reading. The features used were mouth opening shape [59], mouth elongation [51], tracked dots placed around the mouth [22], and optical flow [50]. Recognition was done by measuring distances among representations derived from these features. Yacoob and Black [81] used their human action recognition method (see Section 2.3.1) to perform lip-reading using optical flow features. Kirby *et al.* [45] did not perform recognition. Instead, they used principle component analysis to compress images from the sequence of the spoken word down to three coefficients per image to achieve a low transmission bandwidth.

Chapter 3

Whole-Body Tracking

Whole-body tracking is an important subproblem of articulated motion tracking with many applications in the field of surveillance and traffic (see Section 1.2 for a description of applications). In developing our method, robustness in arbitrary input scenes with arbitrary environmental conditions without compromising real-time performance was the primary motivation. Our approach does not have a restriction on the camera position. More importantly, we do not make any assumptions about occlusions and overlaps. Our system uses a single fixed camera mounted at an arbitrary position. We use simple rectangular patches with a certain dynamic behavior to model pedestrians. Overlaps and occlusions are dealt with by allowing pedestrian models to overlap in the image space and by maintaining their existence in spite of the disappearance of some cues. The cues that we use are blobs obtained by thresholding the result of subtracting the image from the background.

Our choice of using blobs obtained after background subtraction is motivated by the efficiency of this preprocessing step even though some information is permanently lost. In a typical scene, a blob obtained this way does not always correspond to a single pedestrian. An example is shown in Figure 3.1. This is the main source of weakness in many of the existing approaches which assume a clean one-to-one correspondence between blobs and pedestrians (See “Whole-Body Tracking” on page 10.) In our system, we allow maximum flexibility by allowing this relation to be many-to-many. This relation is updated iteratively depending on the observed blobs’ behavior and predictions of pedestrians’ behavior. Figure 3.2 gives an overview of the system. Three levels of abstractions are used. The lowest level deals with raw images. It receives a sequence of images and performs background subtraction producing *difference images*. In the second level, which deals with blobs, difference images are segmented to obtain blobs which are subsequently tracked. Tracked blobs are passed on to the pedestrians level where relations between pedestrians and blobs as well as information about pedestrians is inferred using previous information about pedestrians in that level. The advantage of this paradigm stems from its modularity. Changes and improvements can be made at any level independent of the other levels.

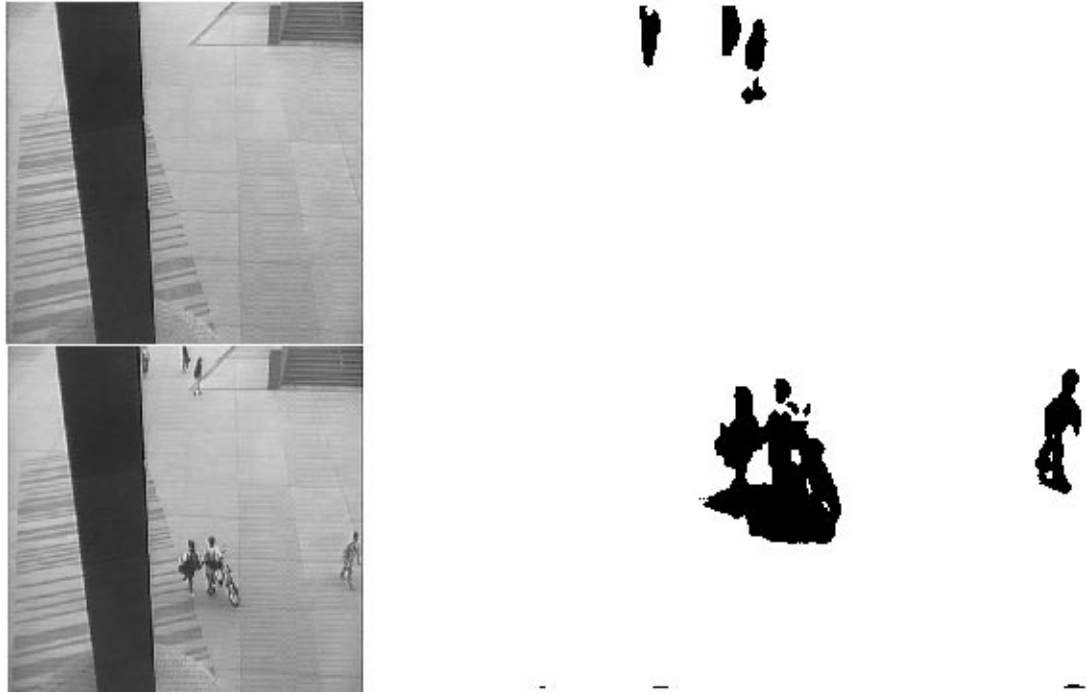


Figure 3.1 Top left: background image. Bottom left: foreground. Right: difference image showing that a blob does not always correspond to one pedestrian.

The next section describes image subtraction which is all that is done at the image level. Section 3.2 describes the processing done at the blobs level. The pedestrians level is presented in Section 3.3. Experimental results follow in Section 3.4. Finally, conclusions are presented in Section 3.5.

3.1 Image Subtraction

Background subtraction has been used by many researchers to extract moving objects in the scene [4,9,68,72]. Change detection algorithms that compare successive frames [38,71] can also be used to extract motion. However, these algorithms also output regions in the background that get uncovered by the moving object as well which is undesirable in our case. In our system, images are first low pass filtered to reduce the effect of noise. Then, the background and the new image are subtracted and the resulting image is thresholded using an appropriate thresholding value. The choice of the threshold is critical. Many thresholding techniques [57,67] work very well when there is an object in the scene. This is because these techniques assume that the image to be thresholded contains two categories of pixel values and they try to separate the two. However, when there is only one category, the results become unpredictable. In our case, this happens often since the scene

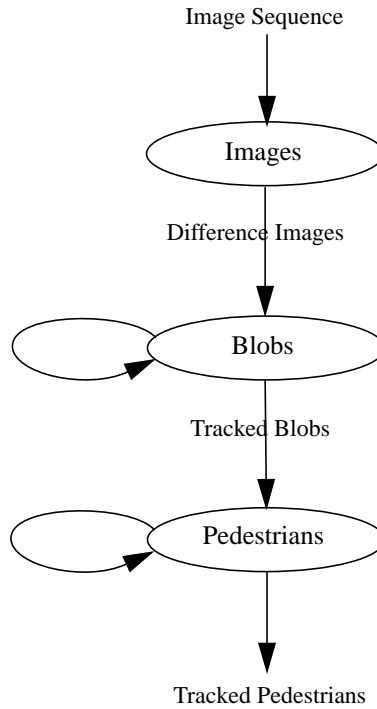


Figure 3.2 The three levels of abstraction and data flows among them.

may not have any objects at one point in time. Instead, we used a fixed threshold value. The value was obtained by examining an empty background for a short while and measuring the maximum fluctuation of pixel values during this training period. The threshold is set to be slightly above that value. This technique worked sufficiently well for our purpose. Several measures were taken to further reduce the effect of noise. A single step of erosion followed by a step of dilation is performed on the resulting image and small clusters are totally removed. Also, the background image is updated using a very slow recursive function to capture slow changes in the background (e.g., changes in lighting conditions due to a passing cloud). We call the resulting image, the *difference image*.

It should be noted that even with these precautions, in a real world sequence, the difference image may still capture some undesirable features (e.g., shadows, excessive noise, sudden change in lighting conditions, and trees moved by the wind (see Figure 3.7 frame 104), etc.). It also may miss parts of moving pedestrians due to occlusions (see Figure 3.8 frame 32) or color similarity between the pedestrian clothes and the background (see Figure 3.8 frame 44). Our system handles the majority of these situations as will be explained in the subsequent sections.

3.2 Blobs Level

In this level, we present a novel approach to track blobs regardless of what they represent. The tracking scheme attempts to describe changes in the difference image in terms of motion of blobs and by allowing blobs to merge, split, appear, and vanish. Robust blob tracking was necessary since the pedestrians level relies solely on information passed from this level.

3.2.1 Blob Extraction

Once a difference image is computed, connected segments are extracted efficiently using border following. Another way to extract connected components is to use the raster scan algorithm. The advantage of the latter method is that it extracts holes inside the blobs while border following does not. However, for the purpose of our system, holes do not constitute a major issue. Moving pedestrians usually form solid blobs in the difference image and if these blobs have holes, they may be still considered part of the pedestrian. Border following has the extra advantage of being more efficient. While raster scan algorithm has to traverse every pixel in the image, with border following, the interior of blobs does not need to be considered. Thus, the larger the total area of all the blobs in the image, the faster the segmentation process becomes. The following parameters are computed for each blob b :

1. Area, denoted by $A(b)$: the number of pixels inside the blob,
2. Bounding box: The smallest rectangle surrounding the blob,
3. Density, denoted by $D(b)$: $A(b)$ / Bounding box area,
4. Velocity, denoted by $\mathbf{V}(b)$, calculated in pixels per second in horizontal and vertical directions.
5. Age: The total time the blob has been present.

3.2.2 Blob Tracking

When a new set of blobs is computed for frame i , an association with frame $(i - 1)$'s set of blobs is sought. Ideally, this association can be an unrestricted relation. With each new frame, blobs can split, merge, appear or disappear. Examples of blob behavior can be seen in the blob images in Figures 3.7 and 3.8. The relation among blobs can be represented by an undirected bipartite graph, $G_i(V_i, E_i)$, where $V_i = B_i \cup B_{i-1}$. B_i and B_{i-1} are the sets of vertices associated with the blobs in frames i and $i - 1$, respectively. We will refer to this graph as a *blob graph*. Since there is a one-to-one correspondence between the blobs in frame i and the elements of B_i , we will use the terms blob and vertex interchangeably. Figure 3.3 shows how the blobs in two consecutive frames are associated. The blob graph in the figure expresses the fact that blob 1 split into blobs 4 and 5,

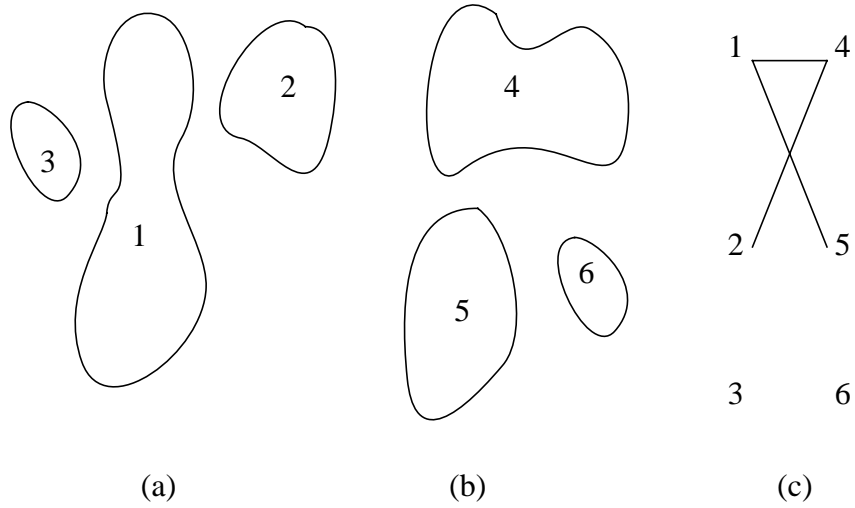


Figure 3.3 (a) Blobs in frame $(i - 1)$. (b) Blobs in frame i . (c) Relationship among blobs.

blob 2 and part of blob 1 merged to form blob 4, blob 3 disappeared, and blob 6 appeared.

The process of blob tracking is equivalent to computing G_i for $i = 1, 2, \dots, n$, where n is the total number of frames. Let $N_i(u)$ denote the set of neighbors of vertex $u \in V_i$, $N_i(u) = \{v | (u, v) \in E_i\}$. To simplify graph computation, we will restrict the generality of the graph to those graphs which do not have more than one vertex of degree more than one in every connected component of the graph. Mathematically, this can be expressed as:

$$\forall (u, v) \in E_i, |N_i(u)| > 1 \Rightarrow |N_i(v)| = 1. \quad (3.1)$$

This is equivalent to saying that from one frame to the next, a blob may not participate in a splitting and a merging at the same time. We refer to this as the *parent structure constraint*. According to this constraint, the graph in Figure 3.3(c) is invalid. If, however, we eliminate the arc between 1 and 5 or the arc between 2 and 4, it will be a valid graph. This restriction is reasonable assuming a high frame rate where such simultaneous split and merge occurrences are rare.

There are exponentially many ways a general bipartite graph can be constructed. In fact, given two sets of vertices of sizes m and n , there are 2^{mn} different possible graphs. This number is reduced with the parent structure constraint but still remains exponential. To further reduce this number, we use another constraint which we call the *locality constraint*. With this constraint, vertices can be connected only if their corresponding blobs have a bounding box overlap area which is at least half the size of the bounding box of the smaller blob. This constraint, which significantly reduces possible graphs, relies on the

assumption that a blob is not expected to be too far from where it is predicted to be, taking into consideration its speed and location in the previous frame. This is also reasonable to assume if we have a relatively high frame rate. We refer to a graph which satisfies both the parent structure and locality constraints as a *valid* graph.

To find the optimum G_i , a cost function, $C(G_i)$, is defined to compare different graphs. A graph with no edges, i.e. $E_i = \emptyset$, is one extreme solution in which all blobs in V_{i-1} disappear and all blobs in V_i appear. This solution has no association among blobs and should therefore have a high cost. In order to proceed with our formulation of the cost function, we define two disjoint sets, which we call *parents*, P_i , and *descendents*, D_i , whose union is V_i such that $D_i = \bigcup_{u \in P_i} N_i(u)$. P can be easily constructed by selecting from V_i all vertices of degree more than one, all vertices of degree zero, and all vertices of degree one which are only in B_i . Furthermore, let $S_i(u) = \sum_{v \in N_i(u)} A(v)$ be the total area occupied by the neighbors of u . The cost function that we use penalizes graphs in which blobs change significantly in size. A perfect match would be one in which blob sizes remain constant (e.g., the size of a blob that splits equals to the sum of the sizes of blobs it split into). We now write the formula for the cost function as

$$C(G_i) = \sum_{u \in P_i} \frac{|A(u) - S_i(u)|}{\max(A(u), S_i(u))}. \quad (3.2)$$

This function is a summation of ratios of size change over all parent blobs.

Using this cost function, we can proceed to compute the optimum graph. First, we notice that given a valid graph $G(V, E)$ and two vertices $u, v \in V$, such that $(u, v) \notin E$, the graph $G'(V, E \cup \{(u, v), (v, u)\})$ has a lower cost than G provided that G' is a valid graph. If it is not possible to find such a G' , we call G *dense*. Using this property, we can avoid some useless enumeration of graphs which are not dense. In fact, this observation is the basis of our algorithm to compute the optimum G .

Our algorithm to compute the optimum graph works as follows: A graph G is constructed such that the addition of any edge to G makes it violate the locality constraint. There can be only one such graph. Note that G may violate the parent structure constraints at this moment. The next step in our algorithm systematically eliminates just enough edges from G to make it satisfy the parent structure constraint. The resulting graph is valid and also dense. The process is repeated so that all possible dense graphs are generated. The optimum graph is the one with the minimum cost. By systematically eliminating edges, we are effectively enumerating valid graphs. The computational complexity

of this step is highly dependent on the graph being considered. If the graph already satisfies the parent structure constraint, it is $O(1)$. On the other hand, if we have a fully connected graph, the complexity is exponential in the number of vertices (bounded by 2^{mn}). Fortunately, because of the locality constraint and the high frame rate, the majority of graphs considered already satisfy the parent structure constraint. Occasionally, a small cluster of the graph may not satisfy the parent structure constraint and the algorithm will need to enumerate a few graphs. In practice, the algorithm never took more than a few milliseconds to execute even in the most cluttered scenes. Other techniques to find the optimum (or near optimum) graph (e.g., stochastic relaxation using simulated annealing) can also be used. The main concern, however, would be their efficiency which may not be appropriate for this real-time application due to their iterative nature.

At the end of this stage, we use a simple method to calculate the velocity of each blob, v , based on the velocities of the blobs at the previous stage and the computed blob graph. The blob velocity will be used to initialize pedestrian models as described later. If v is the outcome of a splitting operation, it will be assigned the same velocity as the parent blob. If v is the outcome of a merging operation, it will be assigned the velocity of the largest child blob. If v is a new blob, it will be assigned zero velocity. Finally, if there is only one blob, u , related to v , the velocity is computed as

$$\mathbf{V}(v) = \beta \frac{(\mathbf{b}_v - \mathbf{b}_u)}{\delta t} + (1 - \beta)\mathbf{V}(u) \quad (3.3)$$

where \mathbf{b}_v and \mathbf{b}_u are the centers of the bounding boxes of v and u , respectively, β is a weight factor set to 0.5 (found empirically), and δt is the sampling interval since the last stage.

3.3 Pedestrians Level

The final level in our hierarchy is the pedestrians level. The input to this level is tracked blobs and the output is the spatio-temporal coordinates of each pedestrian. The relationship between pedestrians and blobs in the image is not necessarily one-to-one. A pedestrian wearing clothes which are close in color to the background may show up as more than one blob. Partially occluded pedestrians may also result in more than one blob or even in no blobs at all if the pedestrian is fully occluded. Two or more pedestrians walking close to each other may give rise to a single blob. For this reason, it was necessary to make the pedestrians level capable of handling all the above cases. We do this by modeling the pedestrian as a rectangular patch with a certain dynamic behavior. We found that for the purpose of tracking, this simple model adequately resembles the pedestrian shape and motion dynamics. We now present this model in more detail and then describe how track-

ing is performed.

3.3.1 Pedestrian Model

Pedestrians usually walk with a constant speed. Moreover, the speed of a pedestrian usually changes gradually when the pedestrian desires to stop or start walking. Three different approaches for pedestrian modeling have been attempted. The latter two are based on the assumption that the scene has a flat ground. This assumption is necessary so that back projection from the scene to the image plane can be performed with the knowledge of the camera geometry to determine the expected dimensions and dynamic behavior of the pedestrian in the image coordinate system. Small variations in ground elevation will still be tolerated especially in distant areas. This restriction can be removed if the scene topology can be determined *a priori*.

1. 2-D dynamics and 2-D shape:

The pedestrian is modeled as a fixed size rectangular patch whose dimensions are similar to the projection of the dimensions of an average size pedestrian located somewhere near the middle of the scene. The patch is assumed to move with a constant velocity in the image coordinate system.

2. 2-D dynamics and 3-D shape:

The pedestrian is modeled as a rectangular patch whose dimensions depend on its location in the image. The dimensions are equal to the projection of the dimensions of an average size pedestrian at the corresponding location in the scene. As in the first approach, the patch is assumed to move with a constant velocity in the image coordinate system.

3. 3-D dynamics and 3-D shape:

The rectangular patch dimensions are as in the previous approach but the patch is assumed to move with constant velocity in the scene coordinate system.

In all these approaches, the patch acceleration is modeled as zero-mean, Gaussian noise to accommodate for changes in velocity. Given a sampling interval δt , the discrete-time dynamic system for the pedestrian model can be described by the following equation:

$$\mathbf{x}_{t+1} = \mathbf{F}\mathbf{x}_t + \mathbf{v}_t, \quad (3.4)$$

where $\mathbf{x} = [x \ \dot{x} \ y \ \dot{y}]^T$ is the state vector consisting of the pedestrian location, (x, y) and

velocity, (\dot{x}, \dot{y}) , \mathbf{F} is the transition matrix of the system given by $\begin{bmatrix} 1 & \delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}$, and \mathbf{v}_t is a

sequence of zero-mean, white, Gaussian process noise with covariance matrix \mathbf{Q} . In the first two approaches above, (x, y) is the image coordinates of the bottom left corner of the patch, while in the third approach, x is the distance on the ground between the center line

of the camera and the left side of the patch, and y is the distance on the ground between the camera optical center and the patch. \mathbf{Q} is computed as in [3] (p. 84) to become

$$\begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{A} \end{bmatrix} q \text{ where } \mathbf{A} = \begin{bmatrix} \frac{(\delta t)^3}{3} & \frac{(\delta t)^2}{2} \\ \frac{(\delta t)^2}{2} & \delta t \end{bmatrix} \text{ and } q \text{ represents the variance of the acceleration.}$$

3.3.2 Pedestrian Tracking

Tracking pedestrians depends on the current state of pedestrians as well as the input to the pedestrians level which is the tracked blobs. In our system, we use Kalman filtering (KF) in the case of the first two modeling approaches and extended Kalman filtering (EKF) in the case of the third approach to estimate pedestrian parameters. We maintain a many-to-many relationship between pedestrians and blobs and then use it to provide measurements to the filter. The next five sections describe one tracking cycle.

3.3.2.1 Relating Pedestrians to Blobs

We represent the relationship between pedestrians and blobs as a directed bipartite graph, $GP_i(VP_i, EP_i)$, where $VP_i = B_i \cup P$. B_i is the set of blobs computed from the i th image as in Section 3.2.1. P is the set of pedestrians. An edge (p, u) , $p \in P$ and $u \in B_i$ denotes that blob u participates in pedestrian p . We call GP_i a pedestrian graph. Given a blob graph, $G_i(V_i, E_i)$, as computed in Section 3.2.2, and a pedestrian graph, GP_{i-1} , EP_i is computed as follows:

$$EP_i = \{(p, u) | (u, v) \in E_i \wedge (p, v) \in EP_{i-1}\}. \quad (3.5)$$

In other words, if a pedestrian was related to a blob in frame $(i-1)$ and that blob is related to another blob in the i th frame (through a split, merge, etc.), then the pedestrian is also related to the latter blob.

3.3.2.2 Prediction

Given the system equation as in Section 3.3.1, the prediction phase of the Kalman filter is given by the following equations:

$$\begin{aligned} \hat{\mathbf{x}}_{t+1} &= \mathbf{F}\mathbf{x}_t, \\ \hat{\mathbf{P}}_{t+1} &= \mathbf{F}\mathbf{P}_t\mathbf{F}^T + \mathbf{Q}. \end{aligned} \quad (3.6)$$

Here, $\hat{\mathbf{x}}$ and $\hat{\mathbf{P}}$ are the predicted state vector and state error covariance matrix, respec-

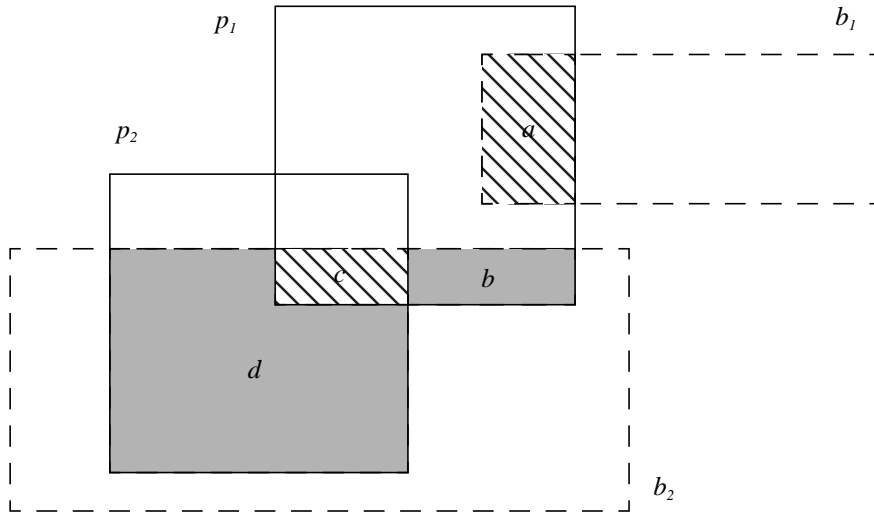


Figure 3.4 Overlap area. Pedestrians p_1 and p_2 share blob b_2 while b_1 is only part of p_1 (See Section 3.3.2.3 for overlap area computation).

tively. \mathbf{x} and \mathbf{P} are the previously estimated state vector and state error covariance matrix.

3.3.2.3 Calculating Pedestrian Positions

In this step, we use the predicted pedestrian locations as starting positions and we apply the following rule to update the locations of pedestrians: Move each pedestrian, p , as little as possible so that it covers as much as possible of its blobs, $\{u | (p, u) \in EP_i\}$; and if a number of pedestrians share some blobs, they should all participate in covering all these blobs. The amount by which a pedestrian covers a blob implies a measure of overlap area. We have already used the bounding box as a shape representation of blobs. However, since the blob bounding box area may be quite different from the actual blob area, we will include the blob density as computed in Section 3.2.1 in the computation of pedestrian-blob overlap area. Let $BB(p)$ be the bounding box of a pedestrian p , and $BB(b)$ be the bounding box of a blob, b . The intersection of $BB(p)$ and $BB(b)$ is a rectangle which we denote its area as $X(BB(p), BB(b))$. The overlap area between p and b is computed as $X(BB(p), BB(b)) \times D(b)$. When more than one pedestrian share a blob, the overlap area is computed this way for each pedestrian only if the other pedestrians do not also overlap the intersection area. If they did, that particular overlap area is divided by the number of pedestrians whose boxes overlap the area. Figure 3.4 illustrates this situation. The overlap

area for p_1 is computed as $a \times D(b_1) + b \times D(b_2) + \frac{c \times D(b_2)}{2}$. For p_2 , the overlap area is $d \times D(b_2) + \frac{c \times D(b_2)}{2}$.

The problem of finding the optimum locations of pedestrians can be stated in terms of the overlap area measure that we just defined. We would like to place pedestrians such that the total overlap area of each pedestrian is maximized. We restate the optimization problem as the problem of finding the minimum total overlap area arrangement of pedestrians which has the least distances between old and new locations of the pedestrian. We do not attempt to solve the problem optimally because of its complexity. Instead, we resort to a heuristic solution using relaxation. First, a large step size is chosen. Then, each pedestrian is moved in all possible directions by the step size and the location which minimizes the overlap area is recorded. Pedestrian locations are then updated according to the recorded locations. This completes one iteration. In each following iteration, the step size is decreased. In our implementation, we start with a step of 64 pixels and halve the step size in each iteration until a step size of 1 pixel is reached.

The resulting locations from the measurements will be fed back into the KF or the EKF to produce the new state estimates. Moreover, we use the overlap area to provide feedback about the measurement confidence by setting the measurement error standard deviation, which is described below, to be inversely proportional to the ratio of the overlap area to the pedestrian area. That is, the smaller the overlap area, the less confident the measurement is considered.

3.3.2.4 Estimation

A measurement is a location in the image coordinate system as computed in the previous section, $\mathbf{z} = [u \ v]^T$. Measurements are related to the state vector by

$$\mathbf{z}_t = \mathbf{h}(\mathbf{x}_t) + \mathbf{w}_t, \quad (3.7)$$

where \mathbf{h} is the measurement function and \mathbf{w}_t is a sequence of zero-mean, white, Gaussian

measurement noise with covariance \mathbf{R}_t given by $\begin{bmatrix} \sigma_t^2 & 0 \\ 0 & \sigma_t^2 \end{bmatrix}$. The measurement error stan-

dard deviation, σ_t , depends on the overlap area computed in the previous section. In the case of 2-D dynamics modeling approaches in which pedestrian locations are in image coordinates, \mathbf{h} is a linear function given by

$$\mathbf{h}(\mathbf{x}) = \mathbf{H}\mathbf{x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{x}. \quad (3.8)$$

In this case, the state estimation equations for the Kalman filter are

$$\begin{aligned} \mathbf{K}_{t+1} &= \hat{\mathbf{P}}_{t+1} \mathbf{H}^T (\mathbf{H} \hat{\mathbf{P}}_{t+1} \mathbf{H}^T + \mathbf{R}_t)^{-1}, \\ \mathbf{x}_{t+1} &= \hat{\mathbf{x}}_{t+1} + \mathbf{K}_{t+1} (\mathbf{z}_{t+1} - \mathbf{H} \hat{\mathbf{x}}_{t+1}), \\ \mathbf{P}_{t+1} &= (\mathbf{I} - \mathbf{K}_{t+1} \mathbf{H}) \hat{\mathbf{P}}_{t+1}, \end{aligned} \quad (3.9)$$

where \mathbf{K}_{t+1} is the Kalman gain at $t+1$. In the third modeling approach, however, pedestrian locations are expressed in world coordinates resulting in \mathbf{h} being a non-linear function which performs projection into image coordinates. In this case, the extended Kalman filter was used. We let \mathbf{H} be the Jacobian of \mathbf{h} . The EKF state estimation equations become

$$\begin{aligned} \mathbf{K}_{t+1} &= \hat{\mathbf{P}}_{t+1} \mathbf{H}^T (\mathbf{H} \hat{\mathbf{P}}_{t+1} \mathbf{H}^T + \mathbf{R}_t)^{-1}, \\ \mathbf{x}_{t+1} &= \hat{\mathbf{x}}_{t+1} + \mathbf{K}_{t+1} (\mathbf{z}_{t+1} - \mathbf{h}(\hat{\mathbf{x}}_{t+1})), \\ \mathbf{P}_{t+1} &= (\mathbf{I} - \mathbf{K}_{t+1} \mathbf{H}) \hat{\mathbf{P}}_{t+1}. \end{aligned} \quad (3.10)$$

The estimated state vector \mathbf{x}_{t+1} is the outcome of the pedestrians level. When using a 3-D shape modeling technique, we also compute the dimensions of the pedestrian patch based on the estimated pedestrian location.

3.3.2.5 Refinement

At the end of this stage, we perform some checks to refine the pedestrian-blob relationships since pedestrians have been relocated. These can be summarized as follows:

- a. If the overlap area between a pedestrian and one of its blobs becomes less than 10% of the size of both, it will no longer be considered belonging to this pedestrian. This serves as the splitting procedure when two pedestrians walk past each other.
- b. If the overlap area between a pedestrian and a blob that does not belong to any pedestrian becomes more than 10% of the size of either one, the blob will be added to the pedestrian blobs. This makes the pedestrian re-acquire some blobs that may have disappeared due to occlusion.
- c. Look for a cluster of blobs which are not related to any pedestrians and whose age is larger than a threshold (i.e., have been successfully tracked for a certain number of frames). A new pedestrian may be initialized only if it will be more than 30% covered by the blobs cluster. The pedestrian is given an initial velocity equal to the average of the blobs velocities. This serves as the initialization step. The requirement on the age helps in reducing chances of unstable blobs being used to initialize pedestrians (for example, blobs resulting from tree motion caused by wind and blobs due to noise).

- d. Select one of the blobs which is already assigned one or more pedestrians but can accommodate more pedestrian patches. Create a new pedestrian for this blob as in c. This handles cases in which a group of people form one big blob which does not split. If we do not do this step, only one pedestrian would be assigned to this blob.
- e. If a pedestrian leaves the view of the camera or has not been covered by any blobs for an extended period of time, the pedestrian is deleted.

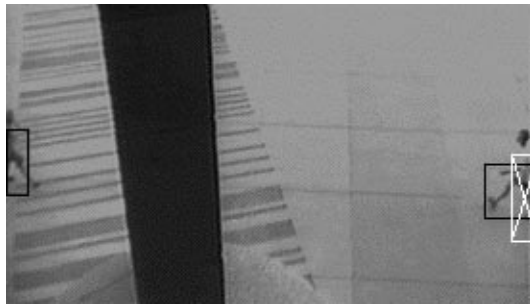
3.4 Experimental Results

The system was initially implemented on the Minnesota Vision Processing System (MVPS) which is the image processing component of the Minnesota Robotic Visual Tracker (MVRT). MVPS consists of a Motorola MVME-147 SBC running real-time operating system OS-9, a Datacube MaxVideo 20 video processor, and a Datacube Max860 vector processor. It was later ported to a 400Mhz Pentium PC equipped with a C80 Matrox Genesis vision board.

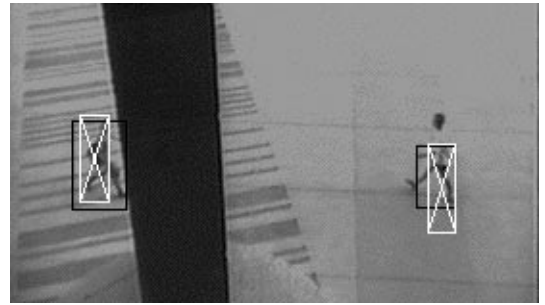
The three modeling approaches were tested. As we expected, the performance was noticeably better when moving from the 2-D shape model to the 3-D shape model and then to the 3-D dynamic model. The system was tested on several indoor and outdoor image sequences. Several outdoor sequences in different weather conditions (sunny, cloudy, snow, etc.) have been used. In most cases, pedestrians were tracked correctly throughout the period they appeared in the scene. Scenarios included pedestrians moving at slow or very high speeds, partial and full occlusions, bicycles, persons on roller-blades and several pedestrian interactions. Interactions between pedestrians included occlusion of one another, repeated merging and splitting of blobs corresponding to two or more pedestrians walking together, pedestrians walking past each other, and pedestrians meeting and then walking back in the direction they came from. The system has a peak performance of over 30 frames per second. In a relatively cluttered image with about 6 pedestrians, the frame processing rate dropped down to about 25 frames per second.

Figure 3.5 shows 16 snapshots spanning a sequence of 8.4 seconds. The sequence demonstrates the system behavior against occlusions, both partial and full. Figure 3.6 shows 12 snapshots from a scene under different weather conditions. The snapshots span a sequence of 35 seconds. A more cluttered sequence is shown in Figure 3.7. This sequence was taped during a snowstorm. One can see the effect of the wind on the tree which resulted in false blobs in the difference images (frames 36, 66, 104, 140). The system deals with this situation in several ways:

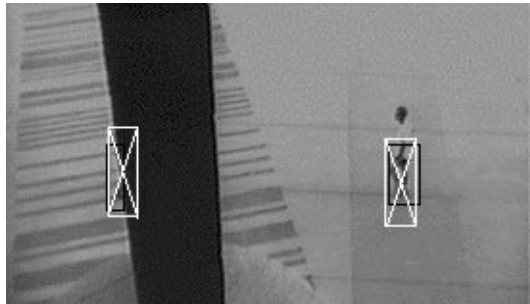
1. If the blobs generated are too small, they are automatically eliminated.
2. A blob is considered at the pedestrian level only if it is tracked successfully over a certain number of frames. This eliminates blobs that appear, then disappear momentarily (such as most blobs that appear due to tree motion back and forth).



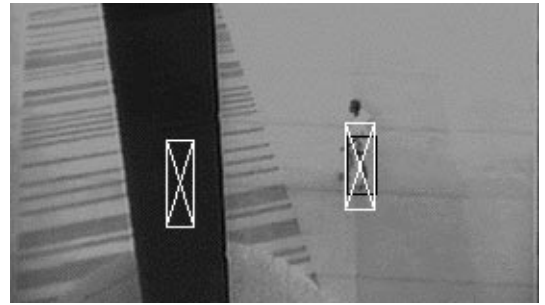
frame 10



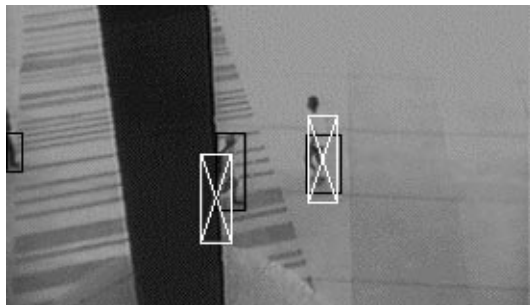
frame 41



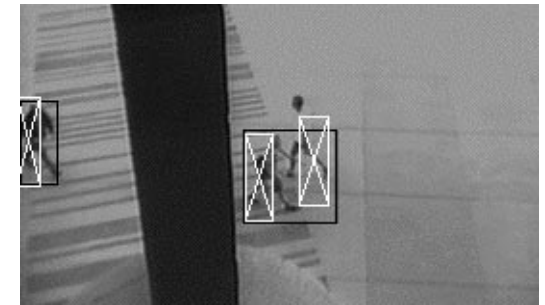
frame 52



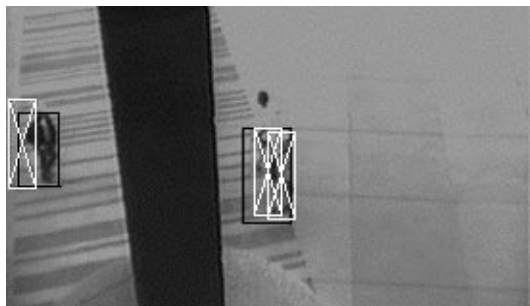
frame 69



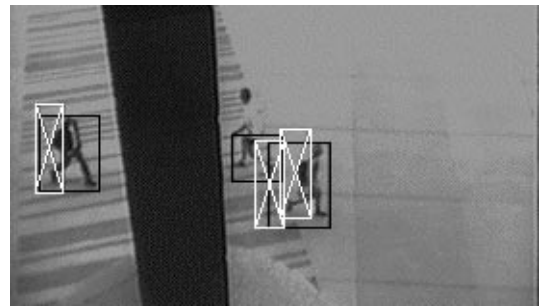
frame 82



frame 92

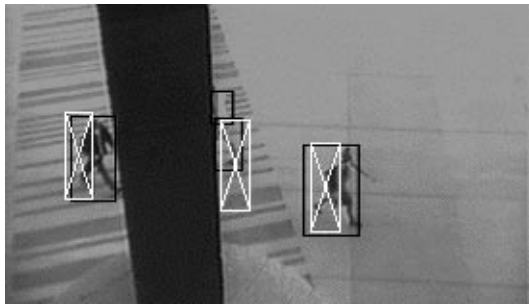


frame 100

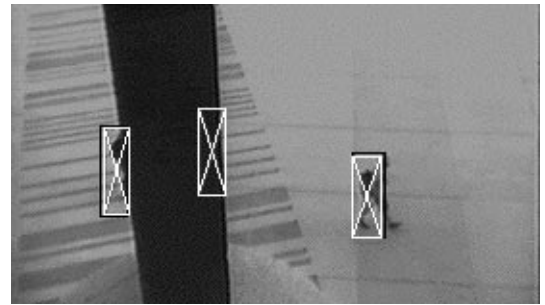


frame 109

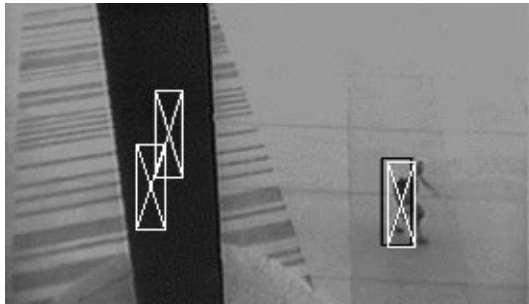
Figure 3.5 A number of snapshots from the input sequence overlaid with pedestrian boxes shown in white and blob boxes shown in black.



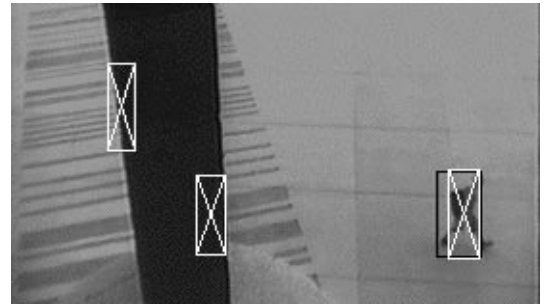
frame 120



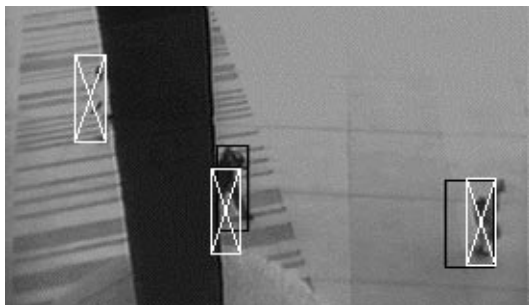
frame 129



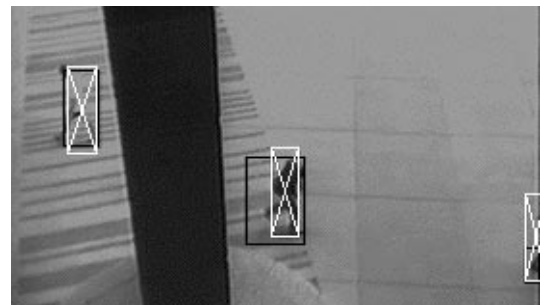
frame 141



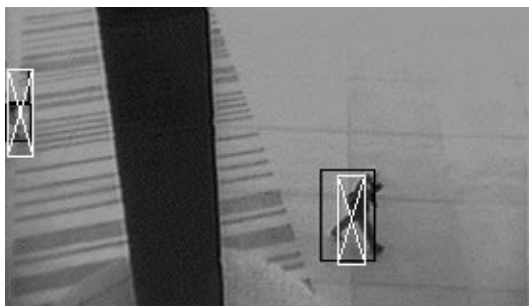
frame 156



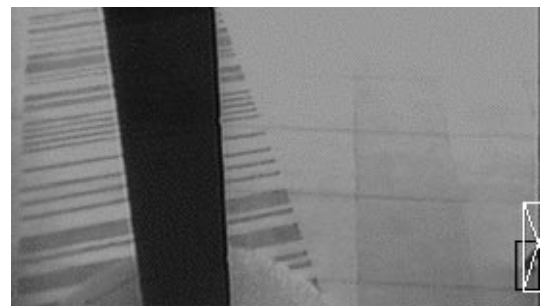
frame 164



frame 182



frame 206



frame 262

Figure 3.5 (Continued).

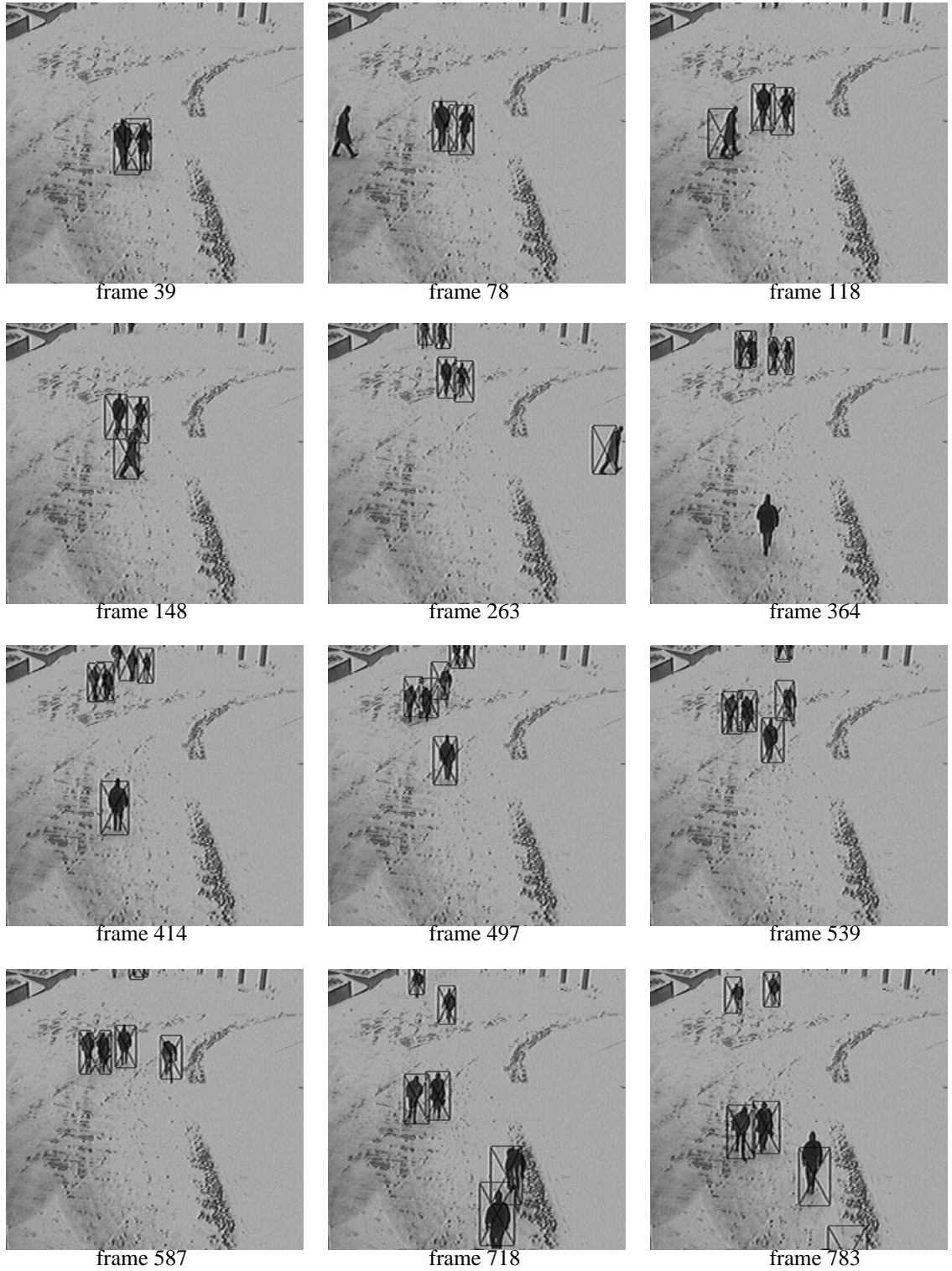


Figure 3.6 A number of snapshots from the input sequence in a snowy afternoon overlaid with pedestrian boxes shown in black.

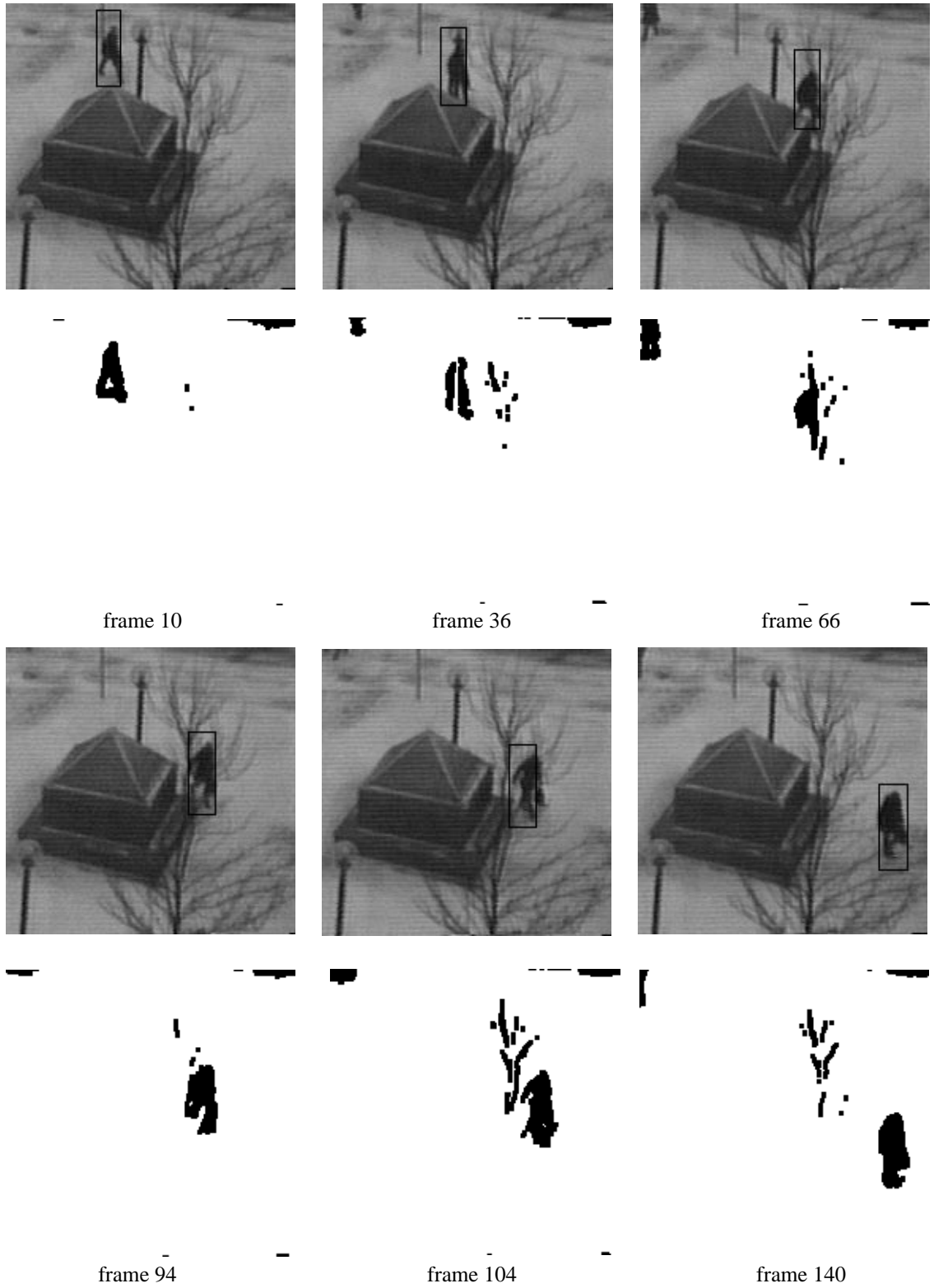


Figure 3.7 Tracking sequence of a pedestrian in different occlusion situations.

3. The system can be given information about the scene. In particular, the locations where pedestrians can be expected to appear. Thus, the system will not instantiate pedestrian boxes except at these locations.

Figure 3.8 shows another tracking sequence which involves missing pedestrian blob information due to occlusion (frame 32) and similarity of color to the background (frame 44). It also shows what happens when two pedestrian walk past each other. Kalman filtering is essential here because it provides good prediction when there is little or no data. The blob-pedestrian relationship refinement procedures guarantee that the pedestrian will be related to the correct blobs when data is available again.

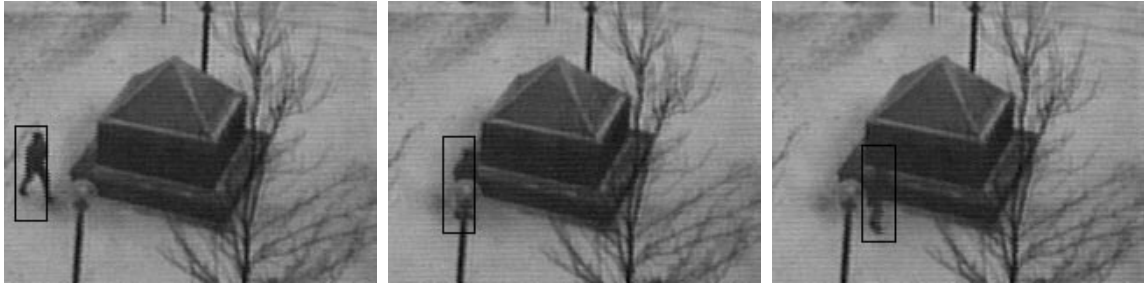
We also performed a pedestrian counting experiment for a sequence of 12 minutes in which 124 pedestrians were counted manually. The system gave a count of 130. Most of the failures were due to bicyclists who were double counted because the blob they generated was closer to the size of two pedestrians.

There are other cases when the system failed. Those include highly crowded images. Also, when a pedestrian becomes totally occluded but then reappears at an unexpected location, the pedestrian box will loose track. Shadows cast by pedestrian show as blobs in the difference image. When the shadows are not too large, the system can usually handle the situation by considering the shadow blob part of the pedestrian. However, large shadows are problematic and they can be confused as pedestrians. More work is needed to address robust handling of the effects of shadows.

Overall, our system works well in a variety of real-world conditions. We feel that this is the major contribution and what makes our approach distinguishable from other approaches to the same problem.

3.4.1 Other Applications: Monitoring Weaving Sections

To demonstrate the versatility of our approach, we used our system with slight modifications to track vehicles. The idea was to track vehicles in short weaving sections which have more than one point of entry followed by more than one point of exit. Sensors which are based on lane detection or tripline detection often fail to monitor weaving sections since they cannot track vehicles which cross lanes. Our system was able to successfully track and record the speed of each vehicle. It also gave periodic averages of speeds of vehicles that belong to one of several categories. The categories were in the form of “Vehicles that remained in lane X” and “Vehicles that changed lanes from X to Y”. Example snapshots are given in Figure 3.9.



frame 12



frame 32



frame 44



frame 92



frame 106



frame 114

Figure 3.8 Tracking sequence demonstrating occlusions and pedestrian overlap.



(a)



(b)



(c)



(d)



(e)

Figure 3.9 A number of snapshots from a weaving section tracking sequence.

3.5 Summary

We presented a real-time model-based pedestrian tracking method capable of working robustly under many difficult circumstances such as occlusions and ambiguities. For each pedestrian in the view of the camera, the system produces location and velocity information in the world coordinate system as long as the pedestrian is visible. Testing provided qualitative results as well as quantitative results in the form of pedestrian counts. The results demonstrate the robustness of this approach and its suitability for a real world applications.

Chapter 4

Motion Recognition

In [17], it was demonstrated that our visual capabilities allow us to perceive actions with ease even when presented with an extremely blurred image sequence of an action. Several snapshots from a similar sequence are shown in Figure 4.1. These experiments suggest that using motion alone to recognize actions may be favorable to reconstruction-based approaches.

Our method uses motion extracted directly from the image sequence. At each frame, motion information is represented by a *feature image*. Motion information is calculated efficiently using an Infinite Impulse Response (IIR) filter. A different method, though conceptually similar, was used by Davis and Bobick [17]. Unlike [17], an action is represented by several feature images rather than just one image. Actions can be complex and repetitive making it difficult to capture motion details in one feature image. Motion features were also used by Polana and Nelson [61]. In this case, several motion features were extracted throughout the action duration. The features were based on normal flow and were very small in size (a 4×4 matrix). Most of computation time was spent in normal flow computation. Our choice of IIR filtering is motivated by the efficiency of this approach. The feature image used is not limited to a small size. Higher representation resolution could give more discriminatory power when there is a similarity among actions. Dimensionality reduction using principle component analysis (PCA) is then utilized at the recognition stage.

To evaluate our approach we perform classification experiments involving eight different action classes, each performed by 28 different people for a total of 232 samples, 168 of which were used as test samples.

After explaining the details of the feature image representation in Section 4.1, recognition is described in Section 4.2. The action data that was used for testing and its acquisition steps are presented in Section 4.3. Finally, results and analysis follow in Section 4.4.



Figure 4.1 Snapshots from a motion sequence (of a person skipping) where the images have been blurred. Humans have no difficulty perceiving the action.

4.1 Feature Image

As mentioned earlier, an IIR filter is used to construct the feature image. In particular, we use the response of the filter as a measure of motion in the image. A slightly different formulation of this measurement has been used by Halevi and Weinshall [28]. The idea is to represent motion by its recency: recent motion will be brighter than older motion. This technique, also called *recursive filtering*, is simple, time-efficient and therefore, suitable for real-time applications. A weighted average at time i , M_i , is computed as

$$M_i = \alpha \times I_{i-1} + (1 - \alpha) \times M_{i-1}, \quad (4.1)$$

where I_i , is the image at time i , and α is a scalar in the range 0 to 1. The feature image at time i , F_i , is computed as follows: $F_i = |M_i - I_i|$. Figure 4.2 is a plot of the filter response to a step function with α set to 0.5. F can be described as an exponential decay function similar to that of a capacitor discharge. The rate of decay is controlled by the

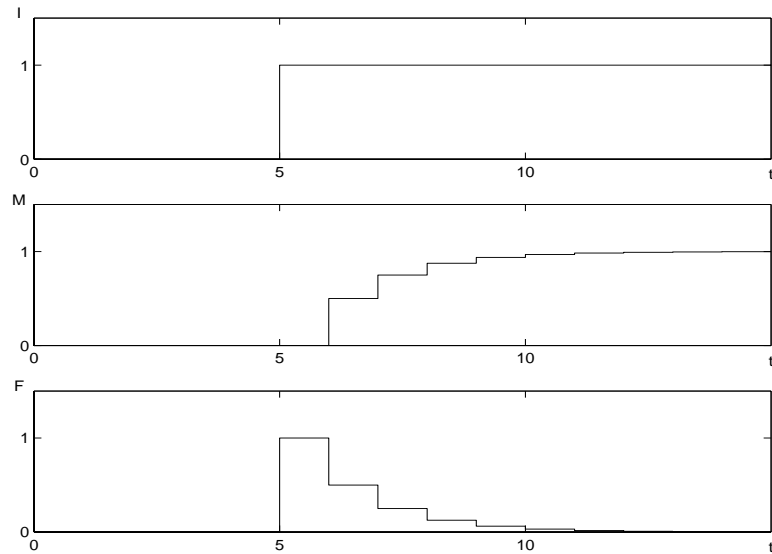


Figure 4.2 Filter response to a step signal.

causes M to be equal to the previous frame. In this case, F becomes equivalent to image differencing. Between these two extremes, the feature image captures temporal changes (features) in the sequence. Moving objects result in a fading trail behind them. In the one-dimensional case, a step edge in the image moving to the left will produce a feature image similar to that shown in Figure 1(c) (after replacing the time coordinate by a spatial coordinate). In fact, if the edge was moving one pixel at a time to the left, the feature image will be identical to Figure 1(c).

The speed and direction of motion are implicit in this representation. The spread of the trail indicates the speed while the gradient of the region indicates direction. Figure 4.3 shows several frames from a motion sequence along with the extracted motion features using this technique. Note that it is the contrast of the gray level of the moving object which controls the magnitude of F , not the actual gray level value.

With the assumption that the height, h , of the person and his/her location in the image are known, feature images are sized and located accordingly. The feature image is computed in a box of dimensions $0.9h$ by $1.1h$ whose bottom is aligned with the base line and centered around the midline of the person. This is illustrated in Figure 4.4. The extra height is needed in cases of some actions that involve jumping. The width is large enough to accommodate the legs motion and the motion trails behind them. Details about how the person's height and location can be estimated will be explained in section 4.3.2.

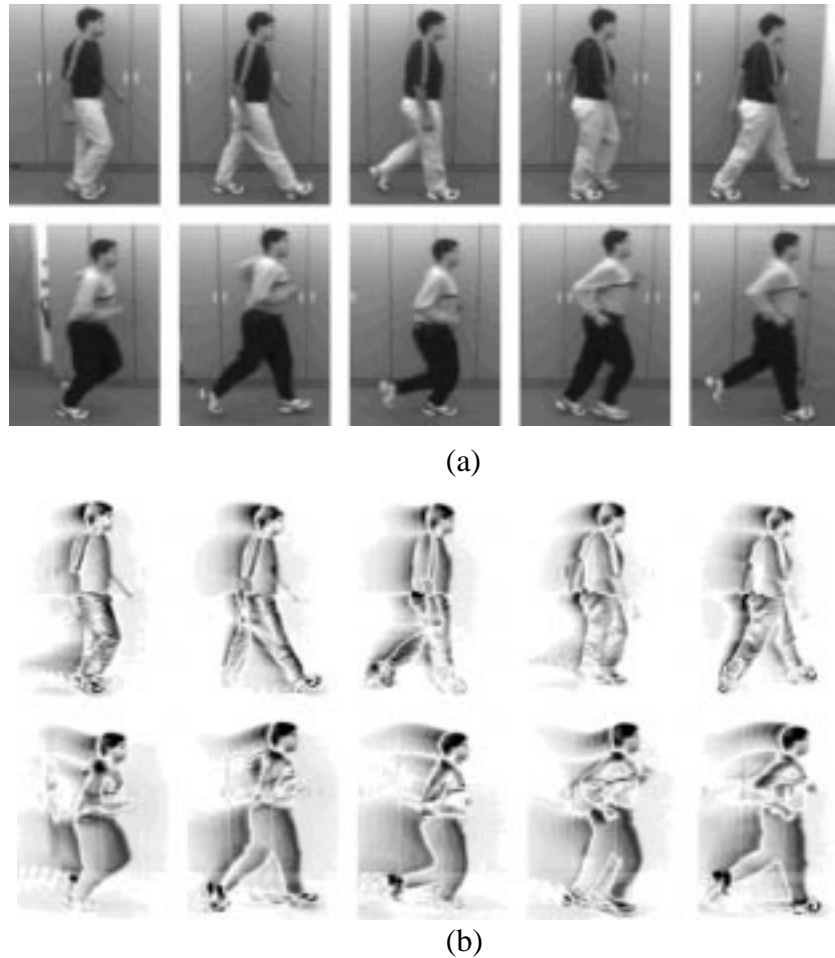


Figure 4.3 An example of a walking and a running motion sequence. (a) Original images. (b) Filtered images (feature images) with $\alpha = 0.3$.

The feature image values are normalized to be in the range $[0, 1]$. They are also thresholded to remove noise and insignificant changes (a threshold of 0.05 was found appropriate). Finally, a low-pass filter is applied to remove additional noise.

4.2 Learning and Recognition

Our goal is to classify actions into one of several categories. We use the feature image representation calculated throughout the action duration. The idea is to compare the feature images with reference feature images of different learned actions and look for the best

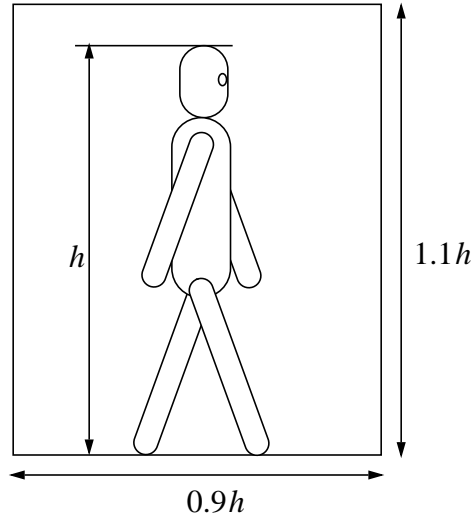


Figure 4.4 Feature image size selection.

match. There are several issues to consider using this approach. Action duration is not necessarily fixed for the same action. Also, the method should be able to handle small speed-ups or slowdowns. Even if we assume that actions are performed at the same speed, we cannot assume temporal alignment and therefore a frame-by-frame matching starting from the first frame should be avoided. The frame-to-frame matching process itself needs to be invariant to the actor's physical attributes such as height, size, color of clothing, etc. Moreover, since an action can be composed of a large number of frames, correlation-based methods for matching may not be appropriate due to their computationally intensive nature.

All these issues have been considered in the development of our recognition method. This section describes the details of our algorithm and addresses the issues mentioned above.

4.2.1 Magnitude and Size Normalization

As actions are represented as sequences of feature images, two types of normalization are performed on a feature image:

1. Magnitude normalization: Because of the way feature images are computed, a person wearing clothes similar to the background will produce low magnitude features. To adjust for this, we normalize the feature image by the 2-norm of the vector formed by concatenating all the values in all the feature images corresponding to the action. The values are then multiplied by the square root of the number of frames to provide invariance to action length (in number of frames).
2. Size normalization: The images are resized so that they are all of equal dimensions.

Not only does this type of normalization works across different people but also it corrects for changes in scale due to distance from the camera, for instance.

4.2.2 Principle Component Analysis

4.2.2.1 Overview

Principle component analysis (PCA) is a technique which is commonly used for dimensionality reduction and for classification. Given N n -dimensional samples, $\mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \dots \ \mathbf{x}_N]$, $\mathbf{x}_i \in \mathfrak{R}^n$, a measure of scatter is the expected value of the squared between-sample distance. This scatter measure can be calculated as

$$\bar{d}_{\mathbf{X}}^2 = 2 \operatorname{tr} \Sigma_{\mathbf{X}}, \quad (4.2)$$

where $\Sigma_{\mathbf{X}}$ is the covariance matrix of \mathbf{X} . $\Sigma_{\mathbf{X}}$ is defined as

$$\Sigma_{\mathbf{X}} = \sum_{i=1}^n (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T, \quad (4.3)$$

where $\boldsymbol{\mu} \in \mathfrak{R}^n$ is the mean of all samples. An orthonormal transformation $\Phi = [\phi_1 \ \phi_2 \ \dots \ \phi_m]$ consisting of m n -dimensional vectors ϕ_i 's is sought to map the original n -dimensional samples (\mathbf{x}_i 's) into a new set of m -dimensional vectors, where $m < n$ (dimensionality reduction). Let $\mathbf{Y} = [\mathbf{y}_1 \ \mathbf{y}_2 \ \dots \ \mathbf{y}_N]$ be the set of mapped vectors,

$$\mathbf{Y} = \Phi^T \mathbf{X}. \quad (4.4)$$

Hence, the scatter measure of \mathbf{Y} is

$$\begin{aligned} \bar{d}_{\mathbf{Y}}^2 &= 2 \operatorname{tr} \Sigma_{\mathbf{Y}} = 2 \operatorname{tr} (\Phi^T \Sigma_{\mathbf{X}} \Phi) \\ &= 2 \sum_{i=1}^m \phi_i^T \Sigma_{\mathbf{X}} \phi_i. \end{aligned} \quad (4.5)$$

In PCA, it is desired to find the ϕ_i 's that maximize this scatter measure. This implies that the expected value of the squared distance among the mapped vectors is as large as possible which make them appropriate for classification. It turns out that the m eigenvectors of $\Sigma_{\mathbf{X}}$ corresponding to the m largest eigenvalues are precisely those ϕ_i 's. This transformation is also known as the discrete *Karhunen-Loeve (K-L) expansion*. Since ϕ_i is an eigenvector of $\Sigma_{\mathbf{X}}$,

$$\Sigma_{\mathbf{X}} \phi_i = \lambda_i \phi_i, \quad (4.6)$$

where λ_i is the eigenvalue associated with ϕ_i . Substituting equation (4.6) into equation (4.5) yields

$$\bar{d}_{\mathbf{Y}}^2 = 2 \sum_{i=1}^m \phi_i^T \lambda_i \phi_i = 2 \sum_{i=1}^m \lambda_i. \quad (4.7)$$

Thus, the scatter is proportional to the sum of the eigenvalues used.

Modeling and recognition using PCA is usually done in a number of steps. First, a number of samples (training data) is used to calculate the eigenvectors. Each sample is then projected in eigenspace using a selected number of eigenvectors. The resulting vector can be viewed as a point in eigenspace which has a much lower dimensionality than the original space. The vector values are sometimes referred to as coefficients because the original vector can be approximated by multiplying these coefficients by the basis (the eigenvectors). When a new sample is to be recognized, it is first projected in eigenspace in a similar manner. A classifier is then used in eigenspace to find which class in the training data the new sample belongs to. There are some variants of PCA that use other techniques such as SVD decomposition.

4.2.2.2 PCA for Action Recognition

PCA has been extensively used in the field of face recognition [6,29,46,75,77]. The use of PCA in action recognition has been limited, however. Of a particular relevance to this work is the work of Yacoob and Black [81]. In their method, the features used were based on tracking five body parts using the work of Ju *et al.* [42]. Each tracked part provided eight temporal measurements. Thus, in total, 40 temporal curves are used to represent an action. Training data is composed of these curves for every example action. Each training sample is composed by concatenating all 40 curves. The training data is then compressed using a PCA technique. An action can now be represented in terms of coefficients of a few basis vectors. Given a new action, recognition is done by a search process which involves calculating the distance between the coefficients for this action and the coefficients of every example action and choosing the minimum distance. Their method handles temporal variation (temporal shift and temporal duration) by parameterizing this search process using an affine transformation.

Our method differs in that an action is not represented by a single point in eigenspace but rather a manifold whose points correspond to the different feature images the action goes through. This moves the burden of temporal alignment and duration adjustments from searching in the measurement space to searching in eigenspace. We see two main advantages for doing this:

1. Reduction in search complexity: Because the eigenspace has a much lower dimension than the measurement space, a more exhaustive search can be afforded.
2. Increased robustness: PCA is based on linear mapping. Action measurements are inherently nonlinear and this nonlinearity increases as these measurements are aggre-

gated across the whole action. PCA can provide better discrimination if the action is not considered as one entity but a sequence of entities.

Nayar *et al.* [53] used parameterized eigenspace manifolds to recognize objects under different pose and lighting conditions.

In our method, the training set consists of a actions each performed a certain number of times, s . For each of the as samples, normalized feature images are computed throughout the action duration. Let the j -th sample of action i consist of T_{ij} feature images:

$F_1^{ij}, F_2^{ij}, \dots, F_{T_{ij}}^{ij}$. A corresponding set of column vectors $\mathbf{S}_{ij} = \begin{bmatrix} \mathbf{f}_1^{ij} & \mathbf{f}_2^{ij} & \dots & \mathbf{f}_{T_{ij}}^{ij} \end{bmatrix}$ is constructed where each \mathbf{f} is formed by stacking the columns of the corresponding feature image. To avoid bias in the training process, a fixed number L of \mathbf{f} 's is used since the number of feature images T_{ij} for a particular sample depends on the action and how the action was performed. From every set of \mathbf{f} 's, we select L evenly spaced (in time) set of vectors $\mathbf{g}_1^{ij}, \mathbf{g}_2^{ij}, \dots, \mathbf{g}_L^{ij}$. L should be small enough to accommodate the shortest action. To ensure that the selected feature images for the samples of one action correspond to similar postures, the samples for each action are assumed to be temporally aligned. This restriction is removed in the testing phase. The grand mean, μ , of these vectors (\mathbf{g} 's) over all i 's and j 's is computed. The grand mean is subtracted from each one of the \mathbf{g} 's and the resultant vectors are the columns of the matrix $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_N \end{bmatrix}$, where $N = asL$ is the total number of columns. The number of rows of \mathbf{X} is equal to the size of the feature image. The first m eigenvectors $\Phi = [\phi_1 \phi_2 \dots \phi_m]$ (corresponding to the largest m eigenvalues) are then computed as described in Section 4.2.2.1. Each sample \mathbf{S}_{ij} is first updated by subtracting μ from each column vector and then projected using these eigenvectors.

Let $\tilde{\mathbf{S}}_{ij} = \begin{bmatrix} \tilde{\mathbf{f}}_1^{ij} & \tilde{\mathbf{f}}_2^{ij} & \dots & \tilde{\mathbf{f}}_{T_{ij}}^{ij} \end{bmatrix}$ be such that $\tilde{\mathbf{f}}_k^{ij} = \mathbf{f}_k^{ij} - \mu$. The projection into eigenspace is computed as

$$\begin{aligned} \mathbf{Y}_{ij} &= \Phi^T \tilde{\mathbf{S}}_{ij} \\ &= \begin{bmatrix} \mathbf{y}_1^{ij} & \mathbf{y}_2^{ij} & \dots & \mathbf{y}_{T_{ij}}^{ij} \end{bmatrix}. \end{aligned} \tag{4.8}$$

Each \mathbf{y}_k^{ij} is an m -dimensional column feature vector which represents a point in eigenspace (the values are coefficients of the eigenvectors). \mathbf{Y}_{ij} is therefore a manifold representing a sample action. We will refer to the set of all the \mathbf{Y} 's as the reference manifolds. Recognition will be performed by comparing the manifold of the new action to the refer-

ence manifolds as will be explained in the next section.

4.2.3 Recognition

As mentioned earlier, recognition is done by comparing the manifold of the test action in eigenspace to the reference manifolds. The manifold of the test action is computed in the same way as described above using the computed eigenvectors at the training stage. In this section, we describe the distance measure that was used for comparison and explain how it is used for classification.

4.2.3.1 Distance Measure

The computed manifold depends on the duration and temporal shift of the action which should not have an effect on the comparison. Our distance measure can handle changes in duration and is invariant to temporal shifts. Given two manifolds $\mathbf{A} = [\mathbf{a}_1 \mathbf{a}_2 \dots \mathbf{a}_l]$ and

$\mathbf{B} = [\mathbf{b}_1 \mathbf{b}_2 \dots \mathbf{b}_h]$, we define

$$d(\mathbf{A}, \mathbf{B}) = \frac{1}{l} \sum_{i=1}^l \min_{1 \leq j \leq h} \left\| \frac{\mathbf{a}_i}{\|\mathbf{a}_i\|} - \frac{\mathbf{b}_j}{\|\mathbf{b}_j\|} \right\| \quad (4.9)$$

as a measure of the mean minimum distance between every normalized point in \mathbf{A} and every normalized point in \mathbf{B} . To ensure symmetry, the distance measure which we use is

$$D(\mathbf{A}, \mathbf{B}) = d(\mathbf{A}, \mathbf{B}) + d(\mathbf{B}, \mathbf{A}). \quad (4.10)$$

This distance measure is a variant of the Hausdorff metric (we use the mean of minima rather than the maximum of minima) which still preserves metric properties. The invariance to shifts is clear from the expression. In fact, $d(\cdot)$ is invariant to any permutation of points since there is no consideration for order altogether. This flexibility comes at the cost of allowing actions which are not similar, but somehow have similar feature images in a different order, to be considered similar. The likelihood of this happening, however, is quite low. This approach is similar to phase space approaches where the time axis is collapsed [10]. The temporal order in our case is not completely lost, however. The feature image representation has an implicit locally temporal order specification. This measure also handles changes in the number of points as long as the points are more or less uniformly distributed on the manifold. The normalization of points in equation (4.9) is effectively an intensity normalization of feature images.

4.2.3.2 Classification

Using the distance measure equation (4.10), three different classifiers have been considered:

1. Minimum Distance (MD): The test manifold is classified as belonging to the same action class the nearest manifold belongs to, over all reference manifolds. This requires finding the distance to every reference manifold.
2. Minimum Average Distance (MAD): The mean distance to reference manifolds belonging to each action class is calculated; and the shortest distance decides classification. This also involves finding the distance to every reference manifold.
3. Minimum Distance to Average (MDA) (also called nearest centroid): For each action, the centroid of all reference manifolds belonging to that action is computed. This is also a manifold with a number of points equal to the average number of points in each reference manifold belonging to the action. We do not interpolate to compute this manifold. Instead, the nearest points (temporally) on the reference manifolds are averaged to compute the corresponding point on the centroid manifold. A test manifold is classified as belonging to the action class with the nearest centroid. Testing involves calculating a number of distances equal to the number of action classes.

4.3 Action Data

4.3.1 Data Selection

To evaluate our recognition method, we recorded video sequences of eight actions each performed by 29 different people. Several frames from one sample of each action are shown in Figures 4.5 and 4.6. The actions are named as follows: Walk, Run, Skip, Line-walk, Hop, March, Side-walk, Side-skip. There are several reasons for our choice of this particular data set:

1. Discrimination becomes more challenging when there is a high degree of similarity among actions. Many of the actions we chose are very similar in the sense that the limbs have similar motion paths.
2. Rather than having the same person perform actions several times, we chose to have different people. This provides a more realistic data since in addition to the fact that people have different physical characteristics, they also perform actions differently both in form and speed. This would be a good test for the versatility of our approach. It can be seen from Figures 4.5 and 4.6 that people sizes as well as color of clothing are different. A few samples also had more complex backgrounds. Table 4.1 shows the variation action performance speed throughout the data set. The table shows that the actions were performed at significantly varying speeds (more than double the speed in the case of Hop for instance).
3. Another consideration for a more realistic data set was that we avoided the use of a treadmill. Using a treadmill not only restricts speed but also simplifies the problem since the background is static relative to the actor.

To our knowledge, this is the largest set of action data compared to what has been



Figure 4.5 Several frames from Walk, Run, Skip, and March actions.

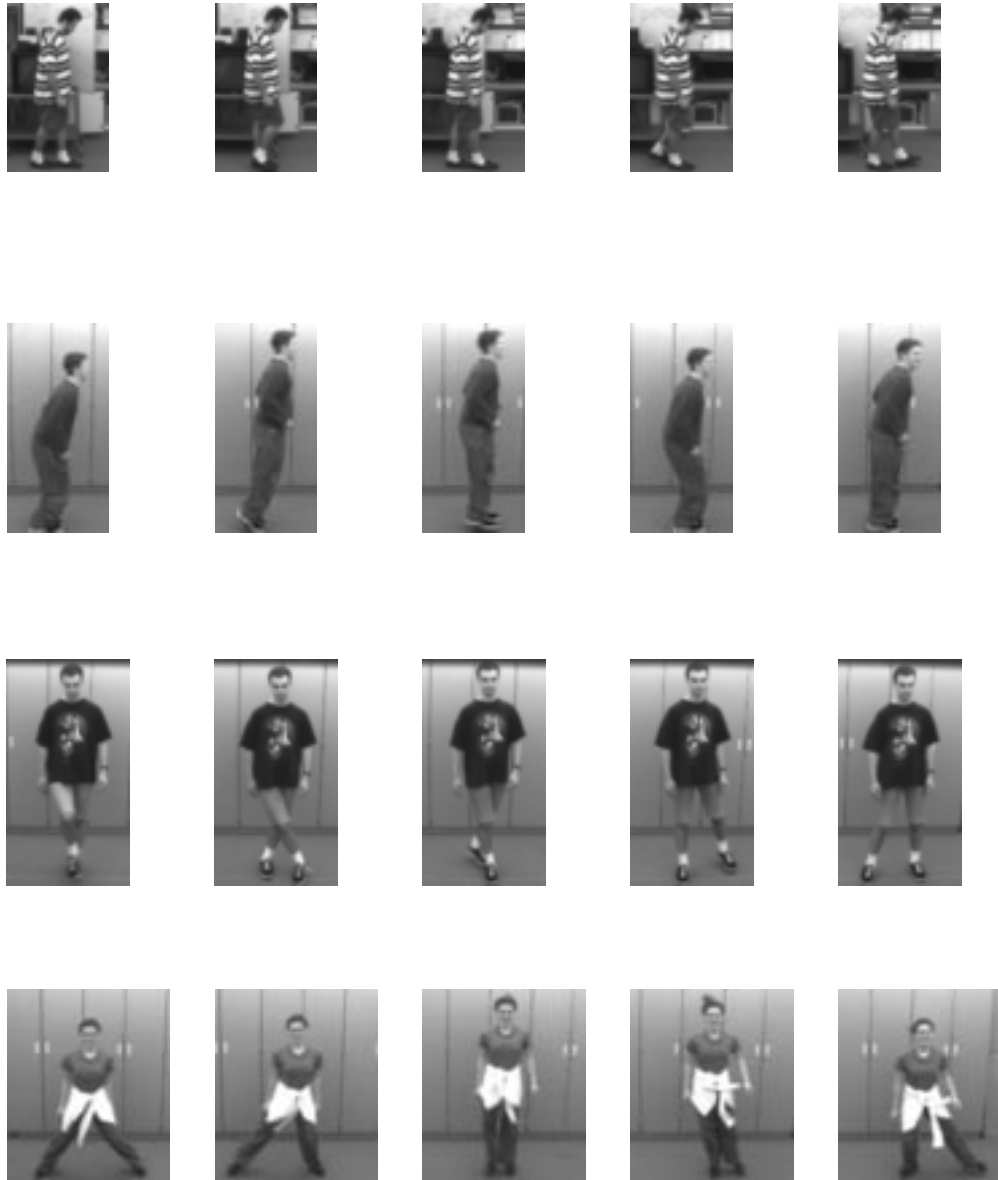


Figure 4.6 Several frames from Line-Walk, Hop, Side-walk, Side-skip actions.

Action	Minimum Duration (sec.)	Maximum Duration (sec.)
Walk	0.93	1.77
Run	0.70	0.93
Skip	1.10	1.73
March	1.13	1.93
Line-walk	1.47	2.20
Hop	0.70	1.67
Side-walk	1.06	1.80
Side-skip	0.57	0.93

Table 4.1: Variation in the duration of one cycle for the data set.

reported in related research in terms of the number of subjects performing the actions multiplied by the number of actions.

4.3.2 Acquisition

The video sequences were recorded using a single stationary monochrome CCD camera mounted in such a way that the actions are performed parallel to the image plane.

In our approach, we assumed that the height (in the image plane) and location of the person performing the action are known. Recovering location is necessary to ensure that the person is in the center of the feature images. Height is used for scaling the feature images to handle differences in people's sizes and distance from the camera. To attain the recovery of these parameters, we tracked the subjects as they performed the action. Background subtraction was used to isolate the subject. A simple frame-to-frame correlation was used to precisely locate the subject horizontally in every frame. A small template corresponding to the top third of the subject's body where little shape variation is expected was used. The height was recovered by calculating the maximum blob height across the sequence. For the general case, our tracking method in the previous chapter can be used to locate the subject boundaries. Correlation can then be applied to find the exact displacement across frames. The computation of feature images as explained in Section 4.1 deals with the raw image data without any knowledge of the background. The information provided by the acquisition step is the location of the person throughout the sequence and the person's height.

4.4 Experimental Results

4.4.1 Classification Experiment

In our experiments, we used the data for eight of the 29 subjects for training (64 video sequences). This leaves a test data set of 168 video sequences performed by the remaining 21 subjects. The training instances were used to obtain the principle components. The number of selected frames (parameter L in Section 4.2.2.2) was arbitrarily set to 12. We will later show the effect of changing this parameter in further experiments. The resolution of feature images was also arbitrarily set to 25 horizontal pixels by 31 vertical pixels. Again, the effect of changing the resolution will be shown later. Decreasing the resolution has a computational advantage but reduces the amount of detail in the captured motion.

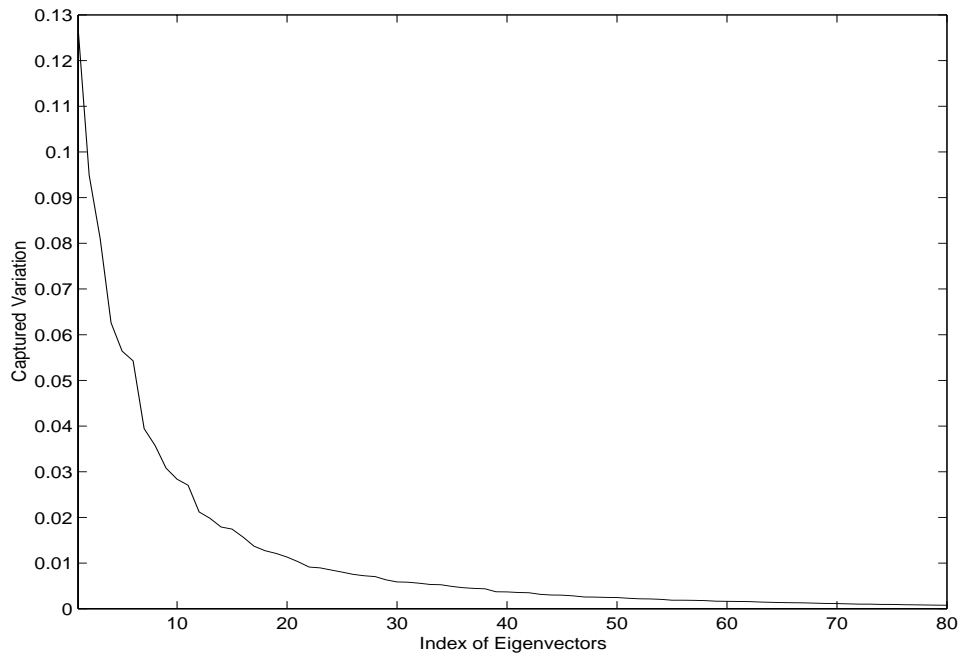
The training samples were organized in a matrix \mathbf{X} as described in Section 4.2.2.2. The number of columns is $asL = 8 \times 8 \times 12 = 768$. The number of rows is equal to the image size ($n = 25 \times 31 = 775$). The eigenvectors are then computed for the covariance matrix of \mathbf{X} . Most of the 775 resulting eigenvectors do not contribute much to the varia-

tion of the data. The plot $\lambda_i / \left(\sum_{k=1}^n \lambda_k \right)$ in Figure 4.7(a) illustrates the contribution of each eigenvector. It can be seen that from around the 50th eigenvector and on, the contribution

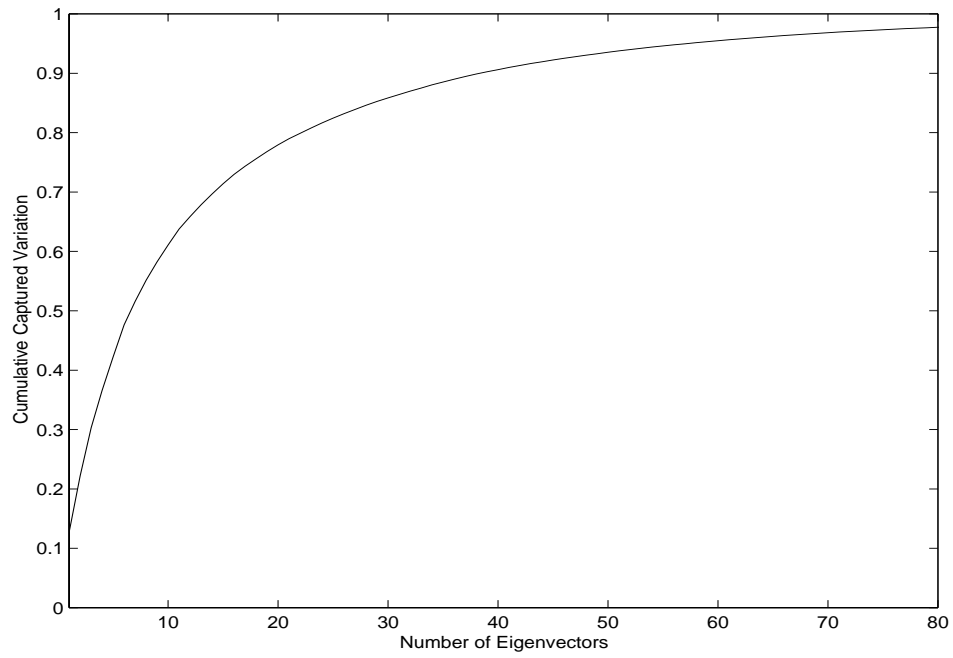
is less than 0.5%. Figure 4.7(b) shows the cumulative contribution $\left(\sum_{k=1}^i \lambda_k \right) / \left(\sum_{k=1}^n \lambda_k \right)$.

The curve increases rapidly during the first eigenvectors. The first ten eigenvectors alone capture more than 60% of the variation. The first 50 capture more than 90%. In Figure 4.8, the first ten eigenvectors are shown. The gray region corresponds to the value of 0 while the darker and brighter regions correspond to negative and positive values, respectively. It can be seen from the figure that different eigenvectors are tuned to specific regions in the feature image.

In our experiments, the choice of m (the number of eigenvectors to be used) was varied from 1 to 50. Using a small m is computationally more efficient but may result in a low recognition rate. As m increases, the recognition rate is expected to improve and approach a certain level. Recognition was done on the 168 test sequences as described in Section 4.2.3 using all three classifiers (MD, MAD, MDA). Recognition rate was computed as the ratio of number of samples classified correctly to the total number samples. Figure 4.9 displays the recognition performance for the different classifiers as a function of m . It can be seen that the recognition rate rises rapidly during the first few values of m . At $m = 14$,



(a)



(b)

Figure 4.7 Eigenvectors contribution to variation in data. (a) Individual contribution. (b) Cumulative contribution.

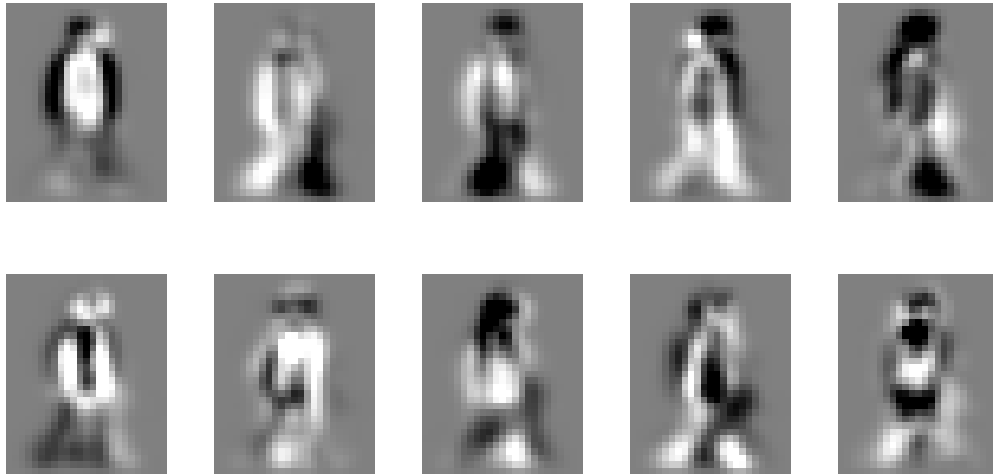


Figure 4.8 The first ten eigenvectors.

the rate using MDA reaches over 91.6%. At $m = 50$, the rate is over 92.8% for MDA. MAD performance is slightly lower while MD is about 10% below. One explanation for this behavior is that some clusters are close to each other so that a point, which may be classified correctly using MDA, can be misclassified using MD. In later experiments, only the MDA classifier will be shown.

Yacoob and Black [81] reported a recognition rate of 82% but they had only four action classes.

Table 4.2 shows the confusion matrix for $m = 50$. Most actions had a perfect or near perfect classification except for the Skip action. Although the Skip action was classified correctly about 70% of the time, it was mistaken with Walk, March, and Hop actions numerous times. The 12 misclassified actions are shown in Figure 4.10. One person (number 15) had two action misclassified while the remaining people had at most one misclassification. When the correct action class was allowed to be within the first two choices, the number of misclassified actions becomes five. All these five actions (mostly Skip actions) were either executed erroneously or had a very low color contrast.

To give an indication of the quality of classification, Figure 4.11 shows a confusion plot which represents the distance among test and reference actions averaged across all subject. The larger the box size, the smaller the distance it represents. The diagonal in the figure stands out and very few other boxes come near the sizes of the boxes at the diagonal. However, it can be seen that there is mutual closeness in matching between Walk and Skip

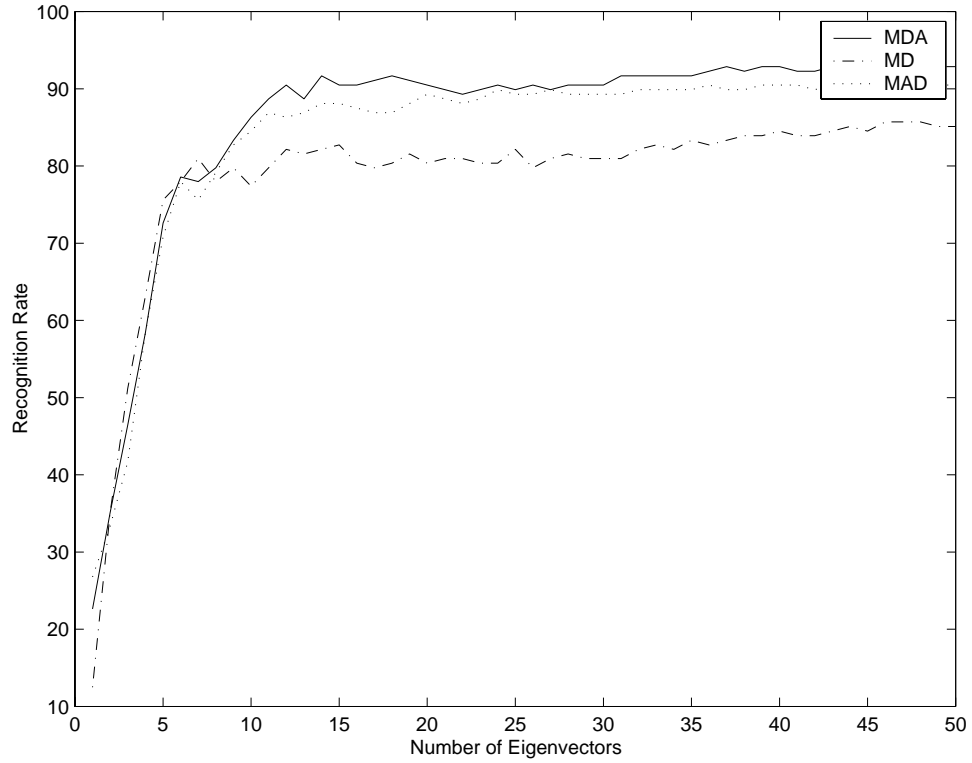


Figure 4.9 Recognition performance.

Action	Walk	Run	Skip	March	Line-walk	Hop	Side-walk	Side-skip
Walk	20	0	0	0	1	0	0	0
Run	1	20	0	0	0	0	0	0
Skip	2	0	15	2	0	2	0	0
March	1	0	1	19	0	0	0	0
Line-walk	0	0	0	0	21	0	0	0
Hop	0	0	0	0	0	21	0	0
Side-walk	0	0	0	0	1	0	19	1
Side-skip	0	0	0	0	0	0	0	21

Table 4.2: Confusion matrix.

	Walk	Run	Skip	March	Line-walk	Hop	Side-walk	Side-skip
1				1				
2							8	
3			1					
4								
5	5							
6								
7			1					
8								
9			6					
10								
11			6					
12							5	
13								
14								
15		1	4					
16								
17								
18				3				
19								
20								
21			4					

Figure 4.10 Misclassified actions for each subject. The numbers indicate the actions that were chosen incorrectly (1=Walk, 8=Side-skip).

actions (a Walk action is close to a Skip action and vice-versa). This was expected due to the high degree of similarity between these two actions.

4.4.2 Parameter Selection

4.4.2.1 Resolution

The resolution of feature images decides the amount of motion detail captured. In size normalization of feature images, a certain resolution must be chosen (See “Magnitude and Size Normalization” on page 42.) Figure 4.12 shows an example feature image and feature images normalized at different resolutions. The classification experiment was run with different resolutions to see if there is a resolution beyond which little or no improvement in performance is gained. Such a reduced resolution has computational benefits. It also give an indication of the smallest “good” resolution which can be used to decide the maximum distance from the camera the action can take place (assuming the camera parameters are known). In Figure 4.13, the classification performance is shown for different resolutions. It can be seen from the figure that increasing the resolution beyond 25×31 does not pro-

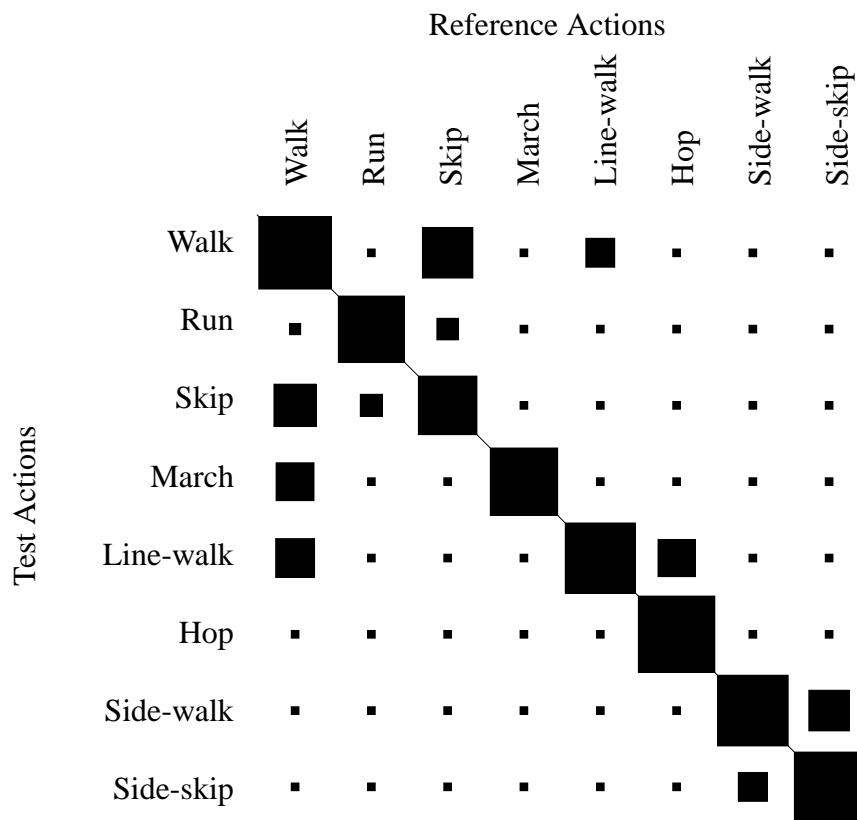


Figure 4.11 Confusion plot. The area of the squares indicates the distance using the distance measure in Section 4.2.3.1. The distances are averaged over all test samples.

duce any gain in performance.

4.4.2.2 Number of Training Images

The parameter L is used in the training process to select the same amount of feature images from every training action sequence (See “PCA for Action Recognition” on page 44.) Here we see the effect of choosing different values for L on performance. Figure 4.14 shows the classification results for the values: 1, 2, 3, 4, 6, 12, 18, and 24. Values of 3 and above seem to have identical performance. This suggests that three feature images from an action sequence capture most of the variation in the different postures.

4.4.3 Complexity

Testing an action involves computing feature images, projecting them in eigenspace,

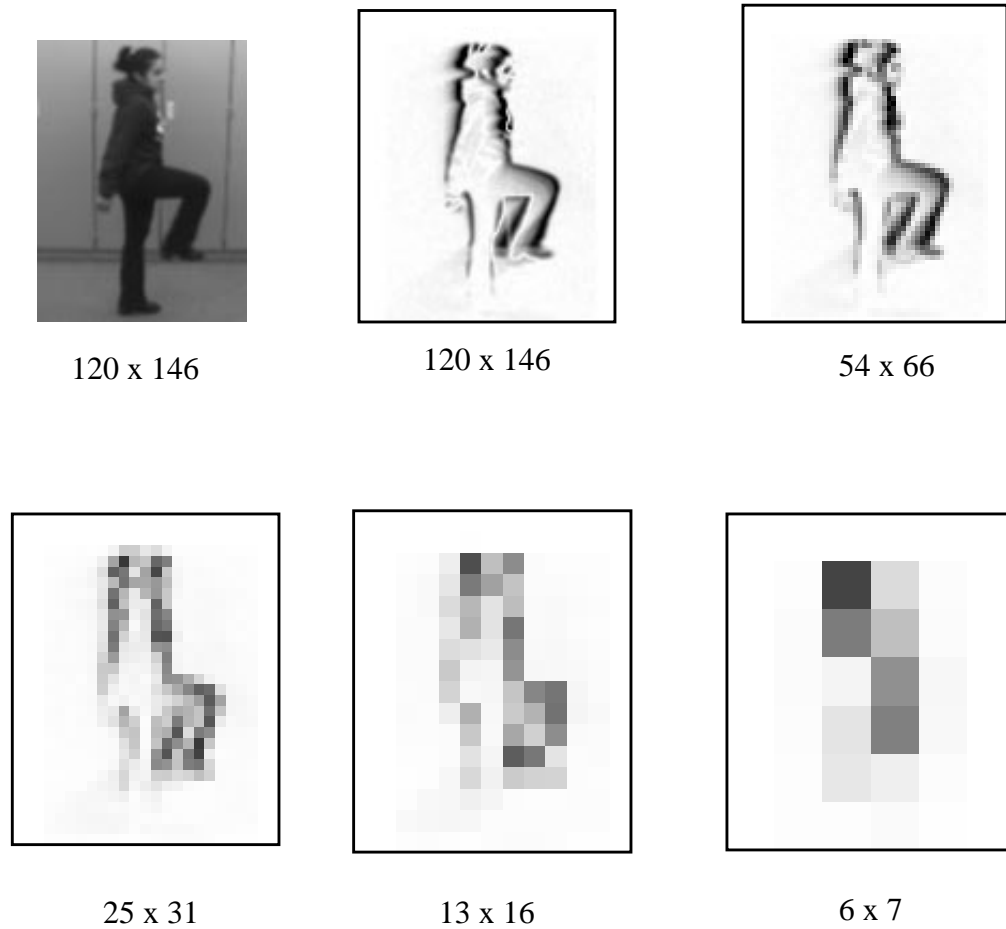


Figure 4.12 Original frame and its feature images at different resolutions.

and comparing the resulting manifold with the reference manifolds. Computing feature images require low level image processing steps (addition and scaling of images) which can be done efficiently. Let n be the number of pixels in the scaled feature image according to the selected resolution. Using m eigenvectors, projecting a feature requires an inner product operation with each eigenvector and thus, a complexity of $O(mn)$. If the action has l frames, the time needed to compute the manifold is $O(lmn)$. Manifold comparison involves calculating the distance between every point on the action manifold and every point on every reference manifold. Assuming there are a action classes with s samples of each. If the average length of the reference actions is T , there will be $asTl$ distance calcu-

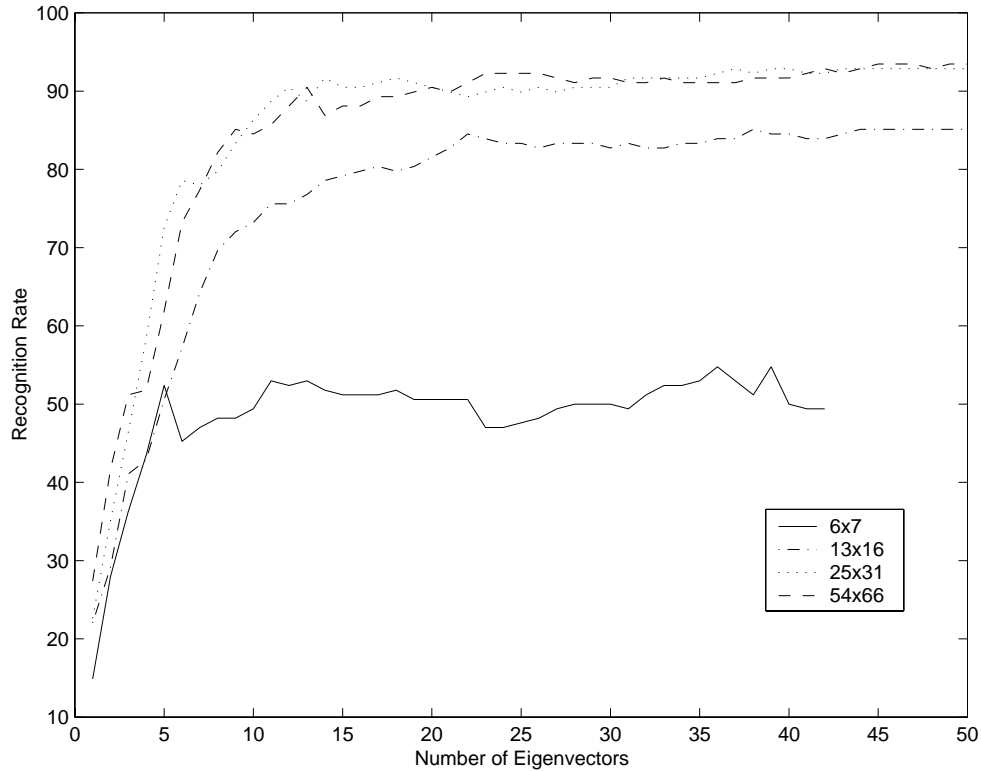


Figure 4.13 Effect of resolution on classification performance.

lations in the case of MD and MAD and aTl calculations in the case of MDA. Calculating a distance between two points in an m -dimensional eigenspace is $O(m)$. Therefore, recognizing an action using MD or MAD is $O(asTlm)$ while in the case of MDA, it is only $O(aTlm)$. In our experiments, $a = 8$, $s = 8$, $T \approx 37$, $m = 50$, and $n = 25 \times 31 = 775$.

The total complexity for MDA is therefore, $O(lmn) + O(aTlm)$, or $O(l)$ since the remaining variables are constant. This demonstrates the efficiency of this method and its suitability for a real-time implementation. On-line implementation is also possible where the distance measure is updated upon receiving new frames, requiring a small number of comparisons per frame. This allows an incremental recognition such that certainty increases as more frames are available. The choice of the implementation approach depends on the application at hand. We left the actual real-time implementation as a future work.

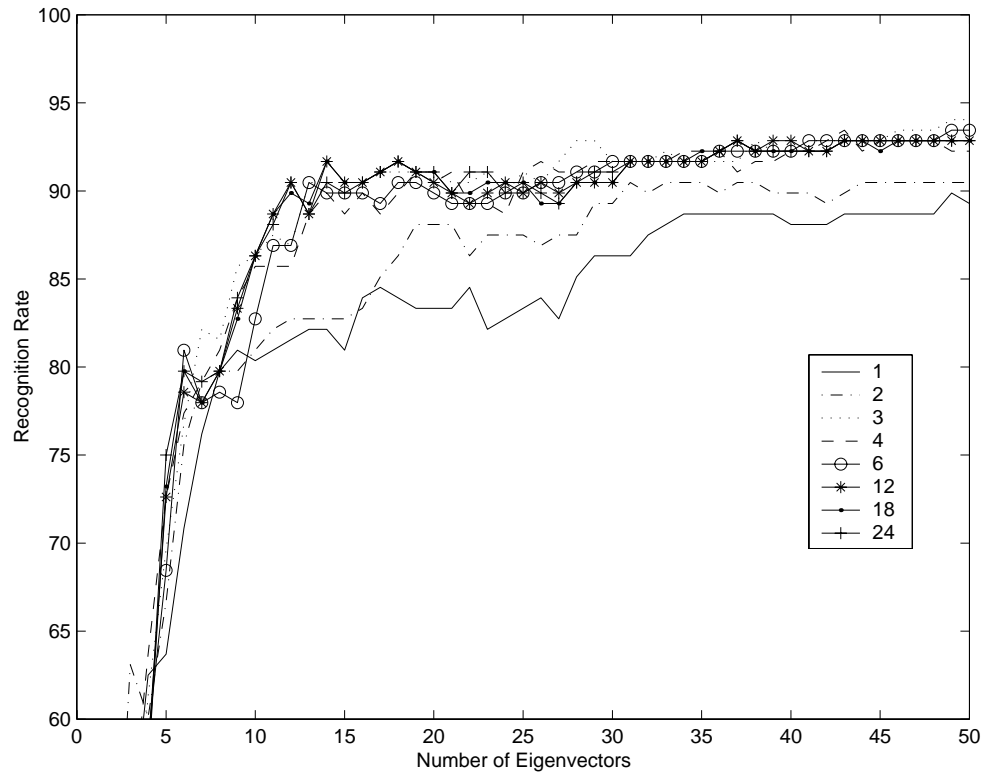


Figure 4.14 Classification performance for different L values.

4.5 Summary

This chapter describes a motion recognition approach. The approach is based on low level motion features which can be efficiently computed using an IIR filter. Once computed, motion features at every frame which we call feature images are compressed using PCA to form points in eigenspace. An action sequence is thus mapped to a manifold in eigenspace. A distance measure was defined to test the similarity between two manifolds. Recognition is performed by calculating the distances to some reference manifolds representing the learned actions. Experimental results for a large data set (168 test sequences) were presented and recognition rates of over 92.8% have been achieved. Complexity was also analyzed. The results demonstrate the promise and efficiency of the proposed approach.

Chapter 5

Conclusions

5.1 Summary

In this thesis, we presented solutions to two problems pertaining to articulated motion. The first problem is tracking of articulated motion as a whole without identifying individual limb motion. Our goal was to address certain shortcomings in previous solutions to this problem, the main shortcoming being their over-constrained nature. Some of the constraints are: restriction to indoor environments, little or no occlusions, fixed lighting and weather conditions, limits on the number of tracked objects, restriction on camera placement, using stereo cameras, and constrained motion. Our solution which was presented as a real-time pedestrian tracking system is capable of working under many difficult circumstances. The method relies on the underlying robust tracking of blobs. The tracked blobs are used to instantiate a pedestrian model. The model has both a spatial as well as temporal specifications which, although simple, were found adequate in modeling the pedestrian shape and motion. Extensive experimentation using long video sequences demonstrated the robustness of our system and its generality. The system performed well in the problem of tracking rigid motion. This was demonstrated by an application to track vehicles across lanes at a weaving section in a highway scene.

The second problem is recognition of articulated motion. The goal here was to show that the recovery of three-dimensional properties of the object or even two-dimensional tracking of the object parts are not necessary steps that must precede action recognition. The approach we devised uses motion features only. Unlike other similar approaches, the motion features are used in such a way to represent complex and long actions as well as to distinguish different actions with many similarities. Motion features are computed very efficiently and subsequently projected into a lower dimension space where the matching is performed. Each action is represented as a manifold in the lower dimension space and

matching is done by comparing these manifolds. To demonstrate the effectiveness of this approach, it was used on a large data set of similar actions each performed by many different actors. Classification results were very accurate in comparison to previously reported action classification results. The experimental results show that this approach can handle many challenges such as variations in performers physical attributes, color of clothing, and stylistic attributes. Moreover, we have shown that the method is efficient and suitable for real-time implementation.

5.2 Contributions

5.2.1 A Robust Articulated Whole-Body Tracking Method

One of the contributions of this work is the development of a real-time pedestrian tracking system that uses a single camera. The system was tested extensively in indoor and outdoor settings and proved its ability to simultaneously track multiple pedestrians. It is also robust under many different situations including partial and full occlusions and was tested in different weather conditions. The model used for pedestrians seems to capture spatial and dynamic attributes accurately. None of the pedestrian tracking systems in the past [5,9,32,66,68,70,72,74] were able to achieve robustness in different scenarios while being efficient enough for real-time performance (See “Whole-Body Tracking” on page 10.)

5.2.2 An Application to Vehicle Tracking

Although intended for tracking non-rigid motion, the pedestrian tracking system is also capable of tracking rigid motion. We have developed an application that tracks vehicles and provides real-time speed and location information for any traffic setting.

5.2.3 A General-Purpose Efficient Blob Tracking Method

Our novel method for blob tracking which is based on graph optimization is very general and can be used in a variety of applications. The generality stems from the fact that in addition to handling appearance and disappearance of blobs, blobs are also allowed to split and merge. The need for blob tracking arises in many tracking applications where the targets of interest can be modeled as blobs. In many cases, image noise, occlusions, and other conditions cause extracted blobs to split, merge, appear, and disappear. Our tracking method provides, in addition to trajectories, blob behavior information which can be used by a higher tracking layer in the application.

5.2.4 Limb Tracking not a Pre-requisite for Action Recognition

In computer vision, tracking the parts of an articulated object is often considered a pre-requisite to recognizing the action being performed. Our work shows that this is not the case for scenes containing complex articulated non-rigid motion.

5.2.5 A General Articulated Motion Recognition Method

We presented a method for action recognition whose input is grayscale image sequences. This is very important for an action recognition application to be considered vision-based. Use of markers or requiring special colored clothing to provide input is not convenient from a general application perspective, and in many cases not possible (e.g., surveillance applications). In addition, our method has several properties which make it as general as possible. It is invariant to the shape of the person performing the action, and the action length, type and complexity, and is also very efficient. Very few methods have been presented that deal with grayscale images; and even the ones that did, none had all these properties combined.

5.2.6 Testing Using a Large Action Data Set

Because of the complex nature of general actions and differences in people's stylistic and physical attributes, an action recognition method must be thoroughly tested before drawing generalized conclusions. We have gathered a large set of data that reflects the variance in the input domain. In our testing data, we used 21 different performers, differing in size, height, and clothing colors. Each performed eight different actions. The choice of actions was such that they have subtle differences yet long enough to achieve complexity. This is the largest data set in terms of the number of performers times the number of actions compared to what has been presented in the literature. The results demonstrate the robustness of the proposed approach.

5.3 Future Work

5.3.1 Tracking

There are several issues that still need to be addressed. Spatial interpretation of blobs is one such issue. In the current system, the only spatial attribute of blobs taken into consideration is the blob area. The shape of the blob can give a good clue on its contents. Although blobs obtained from difference images can be sufficient to decide the location of pedestrians in many cases, the intensity information may be useful to resolve certain ambiguities. The use of such information in the form of statistical distribution of intensities

may add to the robustness of the current system and is well worth pursuing.

Although the system is robust in many circumstances, large shadows can get tracked as pedestrians. The problem of shadows is challenging since they cannot be easily distinguished from other moving objects. Detecting shadows is one area of possible improvement. In the past, this has been attempted by considering statistical shadow properties or by using prior knowledge about the location of the light source.

In its current state, our method assumes that all the objects in the scene are pedestrians. This means that if another object (such as a vehicle) comes into the scene, it will be tracked as a pedestrian (or a group of pedestrians). The problem is currently handled by limiting pedestrian tracking to areas where there are no other large moving objects. Action recognition can be useful in this situation to classify moving objects as pedestrians based on the action being performed (most likely walking or running).

5.3.2 Real-time Implementation of Action Recognition

We have shown that the proposed method is efficient and appropriate for real-time implementation (See “Complexity” on page 56.) This can provide a convenient way of further testing as well as training for other actions. A real-time implementation can also be used in conjunction with the pedestrian tracking system in a real world surveillance application.

5.3.3 Recognition from a General View Angle

Our experiments considered actions performed parallel to the camera plane. This is an undesired restriction in some applications. Recognition performance has not been tested for other viewing angles. Although size scaling should compensate for shrinking in width, we expect it to handle only small angle variations since the motion features will change dramatically for larger angles. This has been a problem for most action recognition methods which are not based on three-dimensional tracking. It is normally handled by considering actions performed at different viewing angles as separate actions, thus increasing the size of the learned action database. This is still an option in our case which is worth trying.

Bibliography

- [1] K. Akita, "Image sequence analysis of real world human motion," *Pattern Recognition*, vol. 17, no. 1, pp. 73-83, 1984.
- [2] A. Azarbayejani and A. Pentland, "Real-time self-calibrating stereo person tracking using 3-D shape estimation from blob features," in *Proc. of International Conference on Pattern Recognition*, Vienna, 1996.
- [3] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*, Academic Press, 1988.
- [4] A. Baumberg and D. Hogg, "Learning flexible models from image sequences," in *Proc. of European Conference on Computer Vision*, vol. 1, pp. 229-308, May 1994.
- [5] A. Baumberg and D. Hogg, "An efficient method for contour tracking using active shape models," in *Proc. of IEEE Workshop on Motion of Non-rigid and Articulated Objects*, pp. 194-199, IEEE Computer Society Press, Nov. 1994.
- [6] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 19, no. 7, pp. 771-720, 1997.
- [7] A. Bobick, J. Davis, S. Intille, F. Baid, L. Campbell, Y. Ivanov, C. Pinhanez, A. Schutte, and A. Wilson, "KIDSROOM: Action recognition in an interactive story environment," MIT Media Lab Perceptual Computing Group Technical Report No. 398, MIT, December 1996.
- [8] A. Bobick and A. Wilson, "A state-based technique for the summarization and recognition of gesture," in *Proc. of International Conference on Computer Vision*, pp. 382-388, Cambridge, 1995.
- [9] Q. Cai and J. K. Aggarwal, "Tracking human motion using multiple cameras," in *Proc. of the 13th International Conference on Pattern Recognition*, pp. 68-72, 1996.
- [10] L. Campbell and A. Bobick, "Recognition of human body motion using phase space constraints," in *Proc. of International Conference on Computer Vision*, pp. 624-630, Cambridge, 1995.
- [11] I.-C. Chang and C.-L. Huang, "Ribbon-based motion analysis of human body movements," in *Proc. of International Conference on Pattern Recognition*, pp. 436-440, Vienna, 1996.
- [12] C. Charayaplan and A. Marble, "Image processing system for interpreting motion in American Sign Language," *Journal of Biomedical Engineering*, vol. 14, no. 15, pp. 419-425, 1992.
- [13] Z. Chen and H. J. Lee, "Knowledge-guided visual perception of 3-d human gait from a single image sequence," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 22, no. 2, pp. 336-342, 1992.
- [14] T. Cootes, C. Taylor, D. Cooper, and J. Graham, "Active shape models - their training and applications," *Computer Vision and Image Understanding*, vol. 61, no. 1, pp. 38-59, 1995.

- [15] J. E. Cutting and L. T. Kozlowski, "Recognizing friends by their walk: Gait perception without familiarity cues," *Bull. Psychometric Soc.*, vol. 9, no. 5, pp. 353-356, 1977.
- [16] T. Darrel and A. Pentland, "Space-time gestures," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 335-340, New York, 1993.
- [17] J. W. Davis and A. F. Bobick, "The representation and recognition of human movement using temporal templates," in *Proc. of IEEE Computer Vision and Pattern Recognition*, pp. 928-934, 1997.
- [18] J. Davis and M. Shah, "Gesture recognition," Technical Report CS-TR-93-11, University of Central Florida, 1993.
- [19] W. H. Dittrich, "Action categories and the perception of biological motion," *Perception*, vol. 22, pp. 15-22, 1993.
- [20] B. Dörner, "Hand shape identification and tracking for sign language interpretation," in *Proc. of Looking at People, International Joint Conference on Artificial Intelligence*, Chambery, 1993.
- [21] A. Downton and H. Drouet, "Model-based image analysis for unconstrained human upper-body motion," in *Proc. IEE International Conference on Image Processing and its Applications*, pp.274-277, 1992.
- [22] K. E. Finn and A. A. Montgomery, "Automatic optically-based recognition of speech," *Pattern Recognition Letters*, vol. 8, pp. 159-164, 1988.
- [23] W. Freeman, K. Tanaka, J. Ohta, and K. Kyuma, "Computer vision for computer games," in *Proc. of IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 100-105, Killington, 1996.
- [24] D. M. Gavrila and L. S. Davis, "3-D model-based tracking of humans in action: a multi-view approach," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 73-80, San Francisco, 1996.
- [25] N. Goddard, "Incremental model-based discrimination of articulated movement direct from motion features," in *Proc. of IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pp. 89-94, Austin, 1994.
- [26] L. Goncalves, E. Di Benardo, E. Ursella, and P. Perona, "Monocular tracking of the human arm in 3-D," in *Proc. of International Conference on Computer Vision*, pp. 764-770, Cambridge, 1995.
- [27] Y. Guo, G. Xu, and S. Tsuji, "Understanding human motion patterns," in *Proc. of the 12th IAPR International Conference on Pattern Recognition*, pp. 325-329, 1994.
- [28] G. Halevi and D. Weinshall, "Motion of disturbances: detection and tracking of multi-body non-rigid motion," in *Proc. of IEEE Conference Computer Vision and Pattern Recognition*, pp. 897-902, June 1997, Puerto Rico.
- [29] P. Hallinan, "A low-dimensional representation of human faces for arbitrary lighting conditions," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 995-999, 1994.
- [30] T. Heap and D. Hogg, "Towards 3-D hand tracking using a deformable model," in *Proc. of IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 140-145, Killington, 1996.

- [31] B. Heisele, U. Kressel, and W. Ritter, "Tracking non-rigid, moving objects based on color cluster flow," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 257-260, San Juan, 1997.
- [32] B. Heisele and C. Wohler, "Motion-based recognition of pedestrians," in *Proc. of the Fourteenth International Conference on Pattern Recognition*, vol. 2, pp. 1325-1330, Brisbane, Australia, Aug. 1998.
- [33] D. Hoffman and B. Flinchbaugh, "The interpretation of biological motion," *Biological Cybernetics*, vol. 42, pp. 195-204, 1982.
- [34] D. Hogg, "Model based vision: A paradigm to see a walking person," *Image and Vision Computing*, vol. 1, no. 1, pp. 5-20, 1983.
- [35] R. Holt, A. Netravali, T. Huang, and R. Qian, "Determining articulated motion from perspective views: A decomposition approach," in *Proc. of IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pp.126-137, Austin, 1994.
- [36] R. Hosie, S. Venkatesh, and G. West, "Detecting deviations from known paths and speeds in a surveillance situation," in *Proc. of the Fourth International Conference on Control, Automation, Robotics and Vision*, pp. 3-6, Singapore, Dec 1996.
- [37] E. Hunter, J. Schlenzig, and R. Jain, "Posture estimation in reduced-model gesture input systems," in *Proc. of International Workshop on Automatic Face and Gesture Recognition*, pp. 290-295, Zurich, 1995.
- [38] R. Jain, D. Militzer, and H.H. Nagel, "Separating non-stationary from stationary scene components in a sequence of real-world TV images," in *Proc. of the Int. Joint Conf. on Artificial Intelligence*, pp. 612-618, 1977.
- [39] G. Johansson, "Visual perception of biological motion and a model for its analysis," *Perception and Psychophysics*, vol. 14, no. 2, pp. 201-211, June 1973.
- [40] G. Johansson, "Visual motion perception," *Sci. Amer.*, vol. 232, pp. 75-88, June 1976.
- [41] N. Johnson, and D. Hogg, "Learning the distribution of object trajectories for event recognition," *Image and Vision Computing*, vol. 14, no. 8, pp. 609-615, Aug. 1996.
- [42] S. Ju, M. Black, and Y. Yacoob, "Cardboard people: A parameterized model of articulated image motion," in *Proc. of IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 38-44, Killington, 1996.
- [43] I. A. Kakadiaris and D. Metaxas, "3D human body model acquisition from multiple views," in *Proc. of the Fifth International Conference on Computer Vision*, pp. 618-623, Boston, MA, Jun. 1995.
- [44] T. Kanade, "Region segmentation: signal vs. semantics," *Computer Graphics and Image Processing*, vol. 13, no. 4, pp. 279-297, August 1980.
- [45] M. Kirby, F. Weisser, and G. Dangelmayr, "A model problem in the representation of digital image sequences," *Pattern Recognition*, vol. 26, no. 1, pp. 63-73, 1993.
- [46] M. Kirby and L. Sirovich, "Application of the Karhunen-Loeve procedure for the characterization of human faces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp. 103-108, 1990.
- [47] L. T. Kozlowski and J. E. Cutting, "Recognizing the sex of a walker from dynamic point-light displays," *Perception and Psychophysics*, vol. 21, no. 6, pp. 575-580,

1977.

- [48] J. Kuch and T. Huang, "Vision-based hand modeling and tracking for virtual teleconferencing and telecollaboration," in *Proc. of International Conference on Computer Vision*, pp. 666-671, Cambridge, 1995.
- [49] M. Leung and Y. Yang, "First sight: a human body outline labeling system," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 4, pp. 359-377, 1995.
- [50] G. A. Martin and M. Shah, "Lipreading using optical flow," in *Proc. National Conference on Undergraduate Research*, March 1992.
- [51] K. Mase and A. Pentland, "Lip reading: automatic visual recognition of spoken words," Technical Report 117, M.I.T. Media Lab Vision Science, 1989.
- [52] C. Myers, L. Rabinier, and A. Rosenberg, "Performance tradeoffs in dynamic time warping algorithms for isolated word recognition," *IEEE Transactions on ASSP*, vol. 28, no. 6, pp. 623-635, 1980.
- [53] S. K. Nayar, S. A. Nene, and H. Murase, "Real-time 100 object recognition system," in *Proc. of IEEE Conference on Robotics and Automation*, vol. 3, pp. 2321-2325, Minneapolis, 1996.
- [54] S. Niyogi and E. Adelson, "Analyzing and recognizing walking figures in XYT," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp.469-474, 1994.
- [55] J. Ohya and F. Kishino, "Human posture estimation from multiple images using genetic algorithm," in *Proc. of International Conference on Pattern Recognition*, pp. 750-753 (A), 1994.
- [56] J. O'Rourke and N. I. Badler, "Model-based image analysis of human motion using constraint propagation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 2, no. 6, pp. 522-536, November 1980.
- [57] J.R. Parker, "Gray level thresholding in badly illuminated images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, pp. 813-819, 1991.
- [58] F. Perales and J. Torres, "A system for human motion matching between synthetic and real images based on a biomechanic graphical model," in *Proc. of IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pp. 83-88, Austin, 1994.
- [59] E. D. Petajan, B. Bischoff, D. Bodoff, and N. M. Brooke, "An improved automatic lipreading system to enhance speech recognition," in *SIGCHI '88: Human Factors in Computing Systems*, pp. 19-25, October 1988.
- [60] R. Polana and R. Nelson, "Detecting activities," *Journal of Visual Communication and Image Representation*, vol. 5, no. 2, pp. 172-180, 1994.
- [61] R. Polana and R. Nelson, "Detection and recognition of periodic, nonrigid motion," *International Journal of Computer Vision*, vol. 23, no. 3, pp. 261-282, 1997.
- [62] F. Quek, "Eyes in the interface," *Image and Vision Computing*, vol. 13, no. 6, pp. 511-525, 1995.
- [63] K. Rangarajan, W. Allen, and M. Shah, "Matching motion trajectories using scale space," *Pattern Recognition*, vol. 26, no. 4, pp. 595-610, 1993.
- [64] J. Rehman and T. Kanade, "Visual tracking of high DOF articulated structures: an

- application to human hand tracking,” in *Proc. of European Conference on Computer Vision*, pp. 34-46, Stockholm, 1994.
- [65] K. Rohr, “Towards model-based recognition of human movements in image sequences,” *CVGIP: Image Understanding*, vol. 59, pp. 94-115, Jan. 1994.
- [66] M. Rossi and A. Bozzoli, “Tracking and counting moving people,” in *Proc. of Second IEEE International Conference on Image Processing*, pp. 212-216, 1994.
- [67] P.K. Sahoo, S. Soltani and A.K.C. Wong, “A survey of thresholding techniques,” *Computer Vision Graphics Image Processing*, vol. 41, pp. 233-260, 1988.
- [68] J. Segen and S. Pingali, “A camera-based system for tracking people in real time,” in *Proc. of the 13th International Conference on Pattern Recognition*, pp. 63-67, 1996.
- [69] T. Shakunaga, “Pose estimation of jointed structures,” in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 566-572, 1991.
- [70] A. Shio and J. Sklansky, “Segmentation of people in motion,” in *Proc. of IEEE Workshop on Visual Motion*, pp. 325-332, 1991.
- [71] K. Skiestad and R. Jain, “Illumination independent change detection for real world image sequences,” *Computer Vision Graphics and Image Processing*, vol. 46, pp. 387-399, 1989.
- [72] C. Smith, C. Richards, S. A. Brandt, and N. P. Papanikolopoulos, “Visual tracking for intelligent vehicle-highway systems,” *IEEE Trans. on Vehicular Technology*, vol. 45, no. 4, pp. 744-759, Nov. 1996.
- [73] T. Starner and A. Pentland, “Real-time American Sign Language recognition from video using hidden Markov models,” in *International Symposium on Computer Vision*, pp. 265-270, Coral Gables, 1995.
- [74] M. J. Sullivan, C. A. Richards, C. E. Smith, O. Masoud, and N. P. Papanikolopoulos, “Pedestrian tracking from a stationary camera using active deformable models,” in *Proc. of the Intelligent Vehicles '95*, pp. 90-95, September 1995.
- [75] D. L. Swets and J. Weng, “Using discriminant eigenfeatures for image retrieval,” *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 18, no. 8, pp. 831-836, 1996.
- [76] S. Tamura and S. Kawasaki, “Recognition of sign language motion images,” *Pattern Recognition*, vol. 21, no. 4, pp. 343-353, 1988.
- [77] M. Turk and A. Pentland, “Eigenfaces for recognition,” *Journal of Cognitive Neuroscience*, vol. 13, no. 1, pp. 71-86, 1991.
- [78] J. Wang, G. Lorette, and P. Bouthemy, “Analysis of human motion: A model-based approach,” in *Proc. 7th Scandinavian Conference on Image Analysis*, Aalborg, 1991.
- [79] J. A. Webb and J. K. Aggarwal, “Structure from motion of rigid and jointed objects. Artificial Intelligence,” vol. 19, pp. 107-130, 1982.
- [80] C. R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, “Pfinder: real-time tracking of the human body,” in *Proc. of the Second International Conference on Automatic Face and Gesture Recognition*, pp. 51-56, Oct 1996.
- [81] Y. Yacoob and M. J. Black, “Parameterized modeling and recognition of activities,” *Journal of Computer Vision and Image Understanding*, vol. 73, no. 2, pp. 232-247.

- [82] M. Yamamoto and K. Koshikawa, "Human motion analysis based on a robot arm model," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 664-665, Maui, 1991.
- [83] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in time sequential images using Hidden Markov Model," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 379-385, 1992.
- [84] J. Zhao, *Moving Posture Reconstruction from Perspective Projections of Jointed Figure Motion*, Ph.D. thesis, University of Pennsylvania, 1993.