ABSTRACT

Title of Dissertation: VIDEO ANALYSIS UNDER SEVERE OCCLUSIONS

Anurag Mittal, Doctor of Philosophy, 2002

Dissertation directed by: Dr. Larry Davis
Department of Computer Science

Many surveillance sites are heavily crowded. Occlusion is a major factor to be considered in building surveillance systems for these sites. To achieve a certain degree of visibility, one requires multiple cameras and collaboration between them so that an object is detected using information available from all the cameras in the scene. In this thesis, we develop methods for sensor planning that can determine the minimum number of sensors required and their configuration so that a certain minimum level of visibility is achieved. Then, we present an algorithm that integrates information from multiple widely separated cameras and obtains a globally optimum detection and tracking result, taking occlusion into consideration. Lastly, we present an algorithm for body pose estimation that uses shape analysis of the silhouettes of a person observed in multiple views. These algorithms are fast, work under partial occlusions and are fully automatic. Due to these features, they can form the basis for a fully automatic, real-time surveillance system for use in areas with high density of objects.

VIDEO ANALYSIS UNDER SEVERE OCCLUSIONS

by

Anurag Mittal

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2002

Advisory Committee:

Dr. Larry Davis, Chairman/Advisor
Dr. Samuel Goward
Dr. Dan Huttenlocher
Dr. Ramalingam Chellappa
Dr. Amitabh Varshney

DEDICATION

To my teachers and family

ACKNOWLEDGEMENTS

Gratitude is first due to all my teachers who have guided me throughout my life, for without them, it would not have been possible to reach this milestone in my life. Specifically, I would like to thank my advisor, Prof. Larry Davis, for his guidance and advice during my two years at the University of Maryland, during which time most of this work was done. He was kind enough to accept me as his student in the middle of my graduate studies, and gave me excellent advice, especially in guiding my research efforts to the right problems. I would also like to thank Prof. Dan Huttenlocher, who guided me during my studies at Cornell, and is very kind to travel from Ithaca for my defense. I would also like to thank the other committee members for their valuable time and advice.

Thanks are due to many friends and colleagues in the university with whom I have had many fruitful discusions and from whom I have learnt a lot : Motilal Agrawal, Ahmed Elgammal, Harsh Nanda, Chiraz BenAbdelKader, Kyungnam Kim and Ramani Duraiswami among others. Thanks are also due to Nikos Paragios and Viswanathan Ramesh for their helpful discussions and comments.

Last, but not the least, this dissertation would not have been possible without the understanding and support of my family. My wife, Neetu, was very patient with me during this time and has been a constant source of love and inspiration in my life. My parents were also very accommodating in allowing me to pursue higher studies.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction and Related Work

1.1 Introduction

Many surveillance sites (for e.g. airports, subway stations and railway stations) are heavily crowded. Occlusion is a major factor to be considered in building surveillance systems for these sites. Typically, multiple sensors (i.e. cameras) are used to increase visibility.

In manned systems where security personnel are looking at the video stream, visibility is the guiding factor for the selection of the sensor arrangement. In automated systems, where advanced algorithms are used to detect and track people, occlusion handling is again an important issue. In order to deal with this condition, such systems rely on the generation of motion models during visibility. Motion information is then used to "fill" in the trajectories during occlusion [58, 115]. Objects can then be found and correctly labeled when they become visible again. However, if objects are occluded for a significant amount of time, the motion models become unreliable and the tracking unstable. The use of appearance models can be considered to label these tracks so that common objects are detected. The accuracy of such a labeling depends to a great extent on the frequency and duration of the occlusion. Above a certain level of occlusion, the probability of recovering an inaccurate result is quite high. Therefore, it is important to limit the rate of occlusion by having a sufficient number of sensors and arranging them properly. Such arrangement has to guarantee that a minimum level of visibility is attained at all positions in the area under surveillance.

In chapter 2 of this thesis, we develop a theoretical framework to calculate the rate of occlusion at different locations in an area for a certain sensor configuration. This analysis can be used to determine the minimum number of cameras required and their optimal placement w.r.t. a given visibility requirement.

The next chapter describes a system that uses a single camera for "sweeping" over a wide field of view and maintains a mixture-of-Gaussians model for each pixel in the panoramic scene. We have developed an adaptive framework whereby two methods for developing the models, i.e. a constant weight updating scheme and exponential weighting scheme, are integrated such that the program automatically chooses the scheme most

appropriate for a given situation. This not only helps us obtain robust estimates of the background, but also yields good estimates for the "new" objects in the scene, which can be further tracked and used in surveillance applications. The method also provides improved results in mosaicing as the new frames are registered against a background image formed from the mixture models. This background image does not contain any moving objects in the scene and theoretically, contains only the background pixels. This helps us obtain improved results for mosaicing as traditional methods register against a mosaic formed from the most recent pixels and the pixels in the moving object get matched to pixels in the mosaic belonging to this object, thus distorting the registration.

Chapter 4 describes a distributed system that integrates information from multiple widely separated cameras and obtains a globally optimum detection and tracking result, taking occlusion into consideration. The system uses multiple synchronized cameras. A Bayesian classification method is developed that uses occlusion-encoded priors to obtain good image segmentations using information available from approximate object positions. This method combines the occlusion information with the color information available in order to obtain an optimal segmentation. Also developed is a novel region-based stereo that is capable of finding 3D points in space using only information about the projections of the objects onto the image planes of the cameras. No exact matching of pixels across views (as in traditional stereo) is required and the method yields points inside objects (as opposed to surface points). These points are used to determine the location of the objects on the ground plane. The position information from different camera pairs is combined using an occlusion analysis scheme that gives more weightage to pairs that have a clear view of a person as opposed to pairs whose views are obstructed by other people or objects. The segmentation and object position determination algorithms are iterated till convergence and the object locations are tracked using a kalman filter.

The algorithm has several novel features that distinguishes it from existing methods:
(1) We do not assume that a foreground connected component belongs to only one object; rather, we segment the views taking into account color models for the objects and the background.
(2) It is fully automatic and does not require any manual input or initializations of any kind.
(3) Instead of taking decisions about object detection and tracking from a single view or camera pair, we collect evidences from each pair and combine the evidence to obtain a decision in the end. This helps us to obtain much better detection and tracking as opposed to traditional systems.

Finally, in chapter 5, we describe a system for human body pose estimation. The requirements of such a system for surveillance tasks are that it must be able to work under occlusions and partial occlusions. A rigid 3D model of the person is not given, nor an initial pose is known. Such a system should also be reasonably fast. However, very accurate body pose values are typically not required, and an answer close to the actual body pose might be adequate. Our goal in this chapter is to make an advancement

towards the development of such a system.

We make use of recent work in decomposition of a silhouette into 2D parts. These 2D part primitives are matched across views to build assemblies in 3D. In order to search for the best assembly, we use a likelihood function that integrates information available from multiple views about body part locations. Occlusion is modeled into the likelihood function so that the algorithm is able to work in a crowded scene even when only part of the person is visible in each view.

The distinguishing feature of our work is that it is able to work in a crowded scene so that in all of the views, the person might be fully or partially occluded. This is accomplished by explicitly modeling occlusion and developing prior models for person shapes from the scene. This helps us to decouple the problems of pose estimation for multiple people so that the degrees of freedom of the problem are decreased substantially.

1.2 Previous Work

1.2.1 Sensor Planning

Sensor planning has been researched quite extensively and there are several different variations depending on the application.

One set of methods use an active camera mounted on a robot. The objective then is to move the camera to the best location in the next view based on the information in the current view. These methods are called next-view planning[86, 85, 99, 88, 81, 72, 91, 54, 63, 9, 109, 67, 111, 75, 79, 55].

Another set of methods obtain 3D reconstruction of a model or scene by moving the camera around the scene [52, 59, 56, 93, 2]. Such a reconstruction imposes certain constraints on the camera position, and satisfaction of these constraints guarantees optimum and stable reconstruction.

Methods that are similar to ours are those that determine the location of static cameras so as to obtain the best views of a scene [10, 97, 78, 103, 36, 105, 104, 102, 112, 113, 62, 61]. Existing methods determine constraints based on several factors not limited to (1) resolution, (2) focus, (3) field of view, (4) visibility, (5) view angle, and (6) prohibited regions. The set of possible sensor configurations satisfying all the constraints for all the features in the scene is then determined. There are several approaches to determining the optimal sensor parameters.

Some systems [87] take a *generate-and-test* approach, in which sensor configurations are generated and then evaluated with respect to the task constraints. In order to limit the number of sensor configurations that are considered, the domain of sensor configurations is discretized.

The sensor planning methods described in [1, 10, 104, 105] take a *synthesis* approach. In this approach, the task requirements are characterized analytically and the sensor parameter values that satisfy the task constraints are directly determined from

these analytical relationships.

There has also been work related to sensor planning in the area of *sensor simulation* systems [35, 37, 77]. In such systems, a scene is visualized given the description of the objects, sensors, and light sources. These systems provide the framework for planning of sensor configurations.

The main characteristic of the prior work is that the objects or features to be viewed and the sources of occlusion are stationary. Many computer vision applications refer to non-stationary structures that can cause occlusions and loss of visibility. This requires statistical modeling of the structures and determining probabilistic answers for visibility as opposed to hard constraints imposed by prior algorithms. We will investigate such a scenario in this thesis. To our knowledge, such scenario has not be addressed in the literature.

1.2.2 Single Camera Algorithms

Many methods have been used for background modeling. Although most of these methods deal only with a fixed camera, they provide a good starting point for a moving camera scene. Simple methods include averaging the pixels at a particular location, taking the median of all the values at a location, and calculating spatially weighted values in order to reduce the effect of outliers. Such techniques, which do not explicitly model background versus foreground, are of limited value in practice. Koller et. al. [50], Ridder et. al. [80] and others employ a Kalman filter-based background model. Each pixel is modeled using a Kalman filter and is updated in each frame differently depending on whether it is hypothesized to be part of the background or not. This approach, however, is not well suited to a changing background or a multi-modal background. Moreover even when an observed pixel value is part of the foreground, it has an effect on the background model.

Friedman and Russell [24] and Stauffer et. al. [98] take approaches based on using mixture models to represent the background. Friedman and Russell try to classify the pixels into three distributions, corresponding to the road color, the shadows and the car colors. This makes their work somewhat restricted to such scenarios, although the method can probably be applied to other ones. Stauffer et. al. use a more general scheme, which is the basis of the method used in our work also.

There is a large literature on image mosaic techniques for constructing panoramic views ([101], [3], [41]). Most approaches to this problem assume that there is not significant motion parallax, that is, depth variations in the scene are not apparent from the motion of the camera. This can be guaranteed by rotating the camera about its optical center (an approach taken by commercial systems such as QuicktimeVR), and also holds true for most cameras once objects are a few tens of feet away. We follow this assumption of no motion parallax, solving for a planar projective transformation that registers one image with another. The most accurate techniques for constructing panoramic views solve for such a transformation between each image frame and the panoramic view that

has been constructed thus far. This helps avoid cascading errors that occur if each image frame is simply registered with the next one.

In general, image mosaic techniques assume a static scene. The presence of moving objects is problematic in two regards. First, and most important, such objects can throw off the image registration process because they provide incorrect information about how the images should be aligned. This results in a poor quality panoramic image. Registration errors can be addressed through the use of robust statistical techniques, and are also somewhat ameliorated by the use of pyramid-based registration methods ([3]). The second issue with moving objects is in the construction of the panoramic view. Simply taking the most recent pixel value, or the average pixel value, in constructing the panorama yields an overall image that contains bits and pieces of moving objects. The explicit background modeling of our approach addresses both of these problems.

1.2.3 Multi-Camera Algorithms

Haritaoglu et. al. [29] developed a single camera system which employs a combination of shape analysis and tracking to locate people and their parts (head, hands, feet, torso etc.) and tracks them using appearance models. In [30], they incorporate stereo information into their system. Kettnaker and Zabih [46, 45] developed a system for counting the number of people in a multi-camera environment where the cameras have a non-overlapping field of view. By combining visual appearance matching with mutual content constraints between cameras, their system tries to identify which observations from different cameras show the same person.

Darrell et. al. [12] developed a tracking algorithm which integrates stereo, color and face pattern detection. Dense stereo processing is used to isolate people from other objects and people in the background. Skin-hue classification is used to identify and classify body parts of a person and a face pattern detection algorithm is used to find the face of a person. Faces and bodies of people are then tracked.

All of these methods use a single viewpoint(using one or two cameras) for a particular part of the scene and would have problems in the case of objects occluded from that viewpoint.

The DARPA VSAM project at CMU developed a system [8] for video surveillance in an outdoor environment using multiple pan/tilt/zoom cameras. The system identifies blobs in the scene using motion detection and stores various information like blob size and color histogram for them. These blobs are then classified into different types of objects using neural networks.

Orwell et. al. [74] present a tracking algorithm to track multiple objects using multiple cameras using ”color” tracking. They model the connected blobs obtained from background subtraction using color histogram techniques and use them to match and track objects. In [73], Orwell et. al. present a multi-agent framework for determining whether different agents are assigned to the same object seen from different cameras.

Rosales and Sclaroff [83] use a single camera tracking system that unifies object

tracking, 3D trajectory estimation, and action recognition. An extended Kalman filter is used to compute trajectories, which are used to reason about occlusion.

These methods would have problems in the case of partial occlusions where a connected foreground region does not correspond to one object, but has parts from several of them.

Cai and Aggarwal [5] extend a single-camera tracking system by starting with tracking in a single camera view and switching to another camera when the system predicts that the current camera will no longer have a good view of the subject. The non-rigidity of the human body is handled by matching points of the middle line of the human image using a Bayesian classification scheme. Features like location, intensity and geometric features are used for tracking.

Intille et. al. [39, 38] present a system which is capable of tracking multiple non-rigid objects. The system uses a top-view camera to identify individual blobs and a "closed-world" assumption to adaptively select and weight image features used for matching these blobs. Putting a camera(s) on top is certainly a good idea since it reduces occlusion, but is not possible in many situations. Also, the advantage of a camera on top is reduced as we move away from the camera, which might require a large number of cameras. Such a camera system would also not be able to identify people or determine other important statistics (like height or color distribution) and hence may not be very useful for many applications. Therefore, we only consider cameras close to the height of the people.

Krumm et. al. [51] present an algorithm that has goals very similar to ours. They use stereo cameras and combine information from multiple stereo cameras (currently only 2) in 3D space. They perform background subtraction and then detect human-shaped blobs in 3D space. Color histograms are created for each person and are used for to identify and track people over time. The method of using short-baseline stereo matching to back-project into 3D space and integrating information from different stereo pairs has also been used by Darrell et. al. [13]. In contrast to [51] and [13], our approach utilizes the wide-baseline camera arrangement that has the following advantages:
(1) It provides more viewing angles with the same number of cameras so that occlusion can be handled better.
(2) It has higher accuracy in back-projection and lower sensitivity to calibration errors, and
(3) It provides many more camera pairs that can be integrated. Placing the cameras as far away from each other as possible and matching every one of them with every other using wide baseline stereo gives us C_2^n pairs. However, placing cameras in pairs of two for short baseline stereo yields only $n/2$ pairs for matching, using n cameras.

On the other hand, the short-baseline stereo pair camera arrangement used, e.g., in [13] has the advantages of
(1) more accurate correspondences due to small change in viewpoint, and
(2) better understood matching algorithms.

Our region matching algorithm can be considered to lie between wide-baseline stereo

algorithms, which try to match exact 3D points across the views, and volume intersection algorithms which find the 3D shape of an object by intersection in 3D space without regard to the intensity values observed (except for background subtraction). Wide-baseline stereo algorithms have the challenge of incorrect matches due to a substantial change in viewpoint, thus rendering traditional methods like correlation and sum of squared difference inappropriate. Although some work has been done to improve upon these methods([76, 26, 33]), they are still not very robust due to the fundamental difficulty of matching points seen from very different viewpoints.

On the other hand, volume intersection is very sensitive to background subtraction errors, so that errors in segmenting even one of the views can seriously degrade the recovered volume. Although there has been work recently ([95]) addressing some of these issues, occlusion is still a major problem. Back-projection in 3D space without regard to color also yields very poor results in cluttered scenes, where almost all of the camera view is occupied by the foreground. Some recent work [53, 21] has addressed some of these issues, but these methods fall in the category of full 3D surface reconstruction, which is very time-consuming and not possible for surveillance applications where computational time is a critical factor.

We do not match points exactly across views; neither do we perform volume intersection without regard to the objects seen. Rather, regions in different views are compared with each other and back-projection in 3D space is done in a manner that yields 3D points guaranteed to lie inside the objects. Clustering these points allows us to detect and track people.

1.2.4 Body Parts Identification Systems

Human Body pose estimation has received considerable interest in the past few years and several approaches have been tried for different applications.

There are many methods for incremental model-based body part tracking where a model of an articulated structure (person) is specified up front[14, 17, 4, 110, 16]. Delamarre and Faugeras [14] try to align the projection of an articulated structure with the silhouettes of a person obtained in multiple views by calculating forces that need to be applied to structure. Drummond and Cipolla [17] use Lie algebra to incrementally track articulated structures. Bregler and Malik [4] use twists and exponential maps to specify relationships between parts and to track an articulated structure incrementally. Sidenbladh[94] and Choo [7] use monte carlo particle filtering to incrementally update the posterior probabilities of pose parameters. These methods need to have both a 3D model of the human structure and a good initialization and have potential applications in motion-capture [16].

Another class of algorithms [40, 23, 96, 82] try to detect body parts in 2D using template matching and then try to find the best assembly using some criteria. Some other methods learn some models of human motion. These models can be based on optical flow [20], exemplars [69, 100], feature vectors [96], support vector machines

[82], or statistical mappings (SMA) [84]. These models can then be used to detect and estimate the pose of a human in an observed image.

Our work is most closely related to the work of Kakadiaris and Metaxas [43] who try to acquire 3D body part information from silhouettes extracted in orthogonal views. They employ a deformable human model so that any size of the human can be recognized. The distinguishing feature of our work is that it is able to work in a crowded scene so that in all of the views, the person might be fully or partially occluded. This is accomplished by explicitly modeling occlusion and developing prior models for person shapes from the scene. This helps us to decouple the problems of pose estimation for multiple people so that the degrees of freedom of the problem are decreased substantially.

Chapter 2

Sensor Planning by Stochastic Modeling of Visibility

2.1 Introduction

Many surveillance sites (for e.g. airports, subway stations and railway stations) are heavily crowded. Occlusion is a major factor to be considered in building surveillance systems for these sites. Typically, multiple sensors (i.e. cameras) are used to increase visibility and quite often the constraint that an object is observed by at least one sensor is to be fulfilled.

In manned systems where security personnel are looking at the video stream, visibility is the guiding factor for the selection of the sensor arrangement. In automated systems, where advanced algorithms are used to detect and track people, occlusion handling is again an important issue. In order to deal with this condition, such systems rely on the generation of motion models during visibility. Motion information is then used to "fill" in the trajectories during occlusion [58, 115]. Objects can then be found and correctly labeled when they become visible again. However, if objects are occluded for a significant amount of time, the motion models become unreliable and the tracking unstable. The use of appearance models can be considered to label these tracks so that common objects are detected. The accuracy of such a labeling depends to a great extent on the frequency and duration of the occlusion. Above a certain level of occlusion, the probability of recovering an inaccurate result is quite high. Therefore, it is important to limit the rate of occlusion by having a sufficient number of sensors and arranging them properly. Such arrangement has to guarantee that a minimum level of visibility is attained at all positions in the area under surveillance.

In this chapter, we present a novel theoretical framework to calculate the rate of visibility (or occlusion) in a surveillance area. The estimation process makes certain assumptions on the arrangement of the sensors and the number or density of people in the scene. This method can be used to find the optimal configuration that consists of a minimum number of sensors and their optimal placement w.r.t. a given visibility requirement.

The chapter is organized as follows. In section 2.2, we introduce the framework to calculate the visibility rate while in section 2.3, we present some deterministic results

for determining the minimum number and arrangement of sensors needed for full visibility regardless of the object configuration. Section 2.4 concludes the paper with some discussion on sensor placement.

2.2 Stochastic Reasoning

2.2.1 Visibility From At least One Sensor

Assume that we have a region $\mathcal{R}$ of area A observed by n sensors. The objective of our approach is to estimate the probability for an object O to be visible by at least one sensor. Such probability varies across space and can be recovered for a given object position.

Let $\mathcal{E}_i$ be the event that object O is visible from sensor i. The probability to be determined is the union $P(\cup_{i=1}^{n}\mathcal{E}_i)$ of these events, and it can be written using the inclusion-exclusion principle as:

$$\begin{aligned} P(\cup_i \mathcal{E}_i) = \sum_{\forall i} P(\mathcal{E}_i) - \sum_{i<j} P(\mathcal{E}_i \cap \mathcal{E}_j) + \\ \sum_{i<j<k} P(\mathcal{E}_i \cap \mathcal{E}_j \cap \mathcal{E}_k) ... + (-1)^{n+1} P(\cap_i \mathcal{E}_i) \end{aligned} \tag{2.1}$$

The main characteristic of this formula is that one can compute the terms on the RHS (right hand side) much easier than the one on the LHS.

In order to facilitate the introduction of the approach to be followed in computing $P(\cup_{i=1}^{n}\mathcal{E}_i)$, we consider the simple case of cylindrical objects of radius r moving on a ground plane. Furthermore, we assume that the sensors are placed at some known heights H_i from this plane (Figure 2.1). Also, we define visibility to mean that the center line of the object is visible for at least some length h. It can easily be shown that the distance d_i up to which another object can occlude an object is proportional to its distance D_i from sensor i (Figure 2.2). Mathematically,

$$d_i = (D_i - d_i)\mu_i = D_i \frac{\mu_i}{\mu_i + 1}, \tag{2.2}$$

where

$$\mu_i = \frac{h}{H_i}$$

An additional constraint can be derived from the maximum angle (α_{max})at which the observation of the object is meaningful. Such observation can be the basis for performing some other tasks, like object/action recognition, pose, gait analysis etc. This constraint translates into a constraint on the minimum distance from the sensor that an object must be in order to be detected. This constraint can easily be incorporated in to the visibility equations.

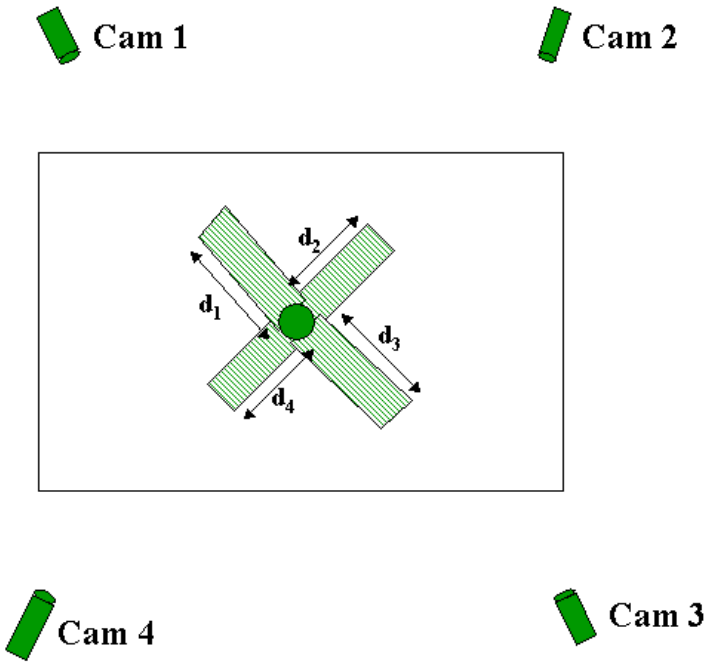

Figure 2.1: Scene Geometry used for stochastic reasoning.

Using a Fixed Number of Objects

Assume that there are a fixed number k of cylindrical objects of radius r in the scene. Assuming that objects are located randomly and uniformly in region $\mathcal{R}$, the first set of terms in Equation 2.1 can be written as:

$$P(\mathcal{E}_i) = (1 - \frac{d_i(2r)}{A})^k \tag{2.3}$$

In order to provide this formulation, we have neglected the fact that two discs (projections of a cylinder on the ground plane) cannot overlap each other. In order to incorporate this condition, we observe that the $(j+1)$th disc has a possible area of only $\mathcal{A} - j\pi(2r)^2$ available to it. The term $2r$ is used instead of r since an object "covers" an area of $\pi(2r)^2$. This is because another object cannot be placed anywhere within the circle of radius $2r$ centered at the first object without intersecting the object. Thus, we obtain:

$$P(\mathcal{E}_i) = \prod_{j=0}^{k-1} \left(1 - \frac{d_i(2r)}{A - j\pi(2r)^2}\right) \tag{2.4}$$

Note that, for simplicity, we have neglected higher order effects. It can be shown that j discs do not necessarily "cover" an area equal to $j\pi(2r)^2$. In order to illustrate this, we consider the case of two discs. In the absence of overlapping, the two discs "cover"

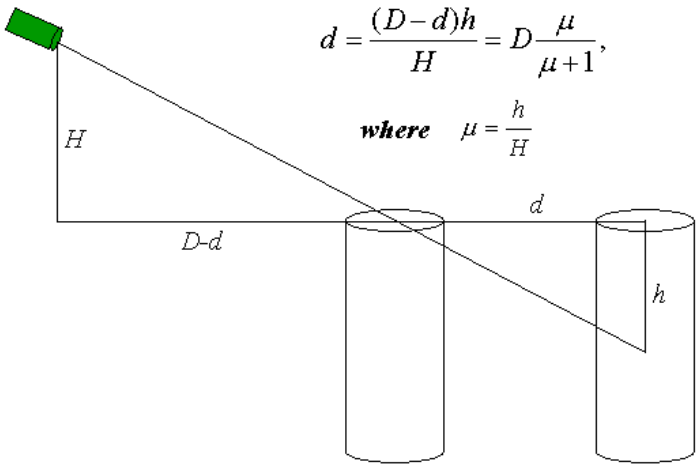

Figure 2.2: The distance up to which another object can occlude an object is proportional to its distance from the sensor

an area equal to $2\pi(2r)^2$. However, if the distance between their centers is between $2r$ and $3r$, then the total area "covered" by the two objects is less than that. Similar conditions exist for three discs when they are close to each other, though not for four or more discs because of geometric constraints. These effects are insignificant for most practical cases, but become considerable if the density of objects is high. Therefore, they must be taken into consideration for those cases.

Similar to the first set of terms, we can derive the equations for other terms in Equation 2.1:

$$P(\cap_i \mathcal{E}_i) = \prod_{j=0}^{k-1}(1 - \frac{(2r)\sum_i d_i}{A - j\pi(2r)^2}) \tag{2.5}$$

where we have neglected the possibility that occlusion regions for different sensors might overlap. Such overlap depends on the angular separation of the sensors and can be significant for when they are placed close to each other (for e.g. stereo sensors). However, we should still be able to find the occlusion area and obtain the corresponding formula for each set of sensors.

Stationary Object Density

Fixed assumptions on the number of objects in a region mean that the presence of an object at a particular location affects the object density everywhere in the area. A more realistic assumption is that the objects have a certain density of occupancy. Consequently, the presence of one object does not change the density of occupancy elsewhere. First, we consider the case of uniform density in the Region. This case can be treated as

a generalization of the finite object case introduced in the previous section. To this end, we increase the area A such that

$$k = \lambda A \tag{2.6}$$

where a constant object density λ is assumed. Equation 2.5 can then be written as

$$P(\cap_i \mathcal{E}_i) = \lim_{k\to\infty} \prod_{j=0}^{k-1} (1 - \frac{(2r)\sum_i d_i}{k/\lambda - j\pi(2r)^2}) \tag{2.7}$$

Defining

$$a = \frac{1}{\lambda(2r)(\sum_i d_i)}, b = \frac{\pi(2r)}{\sum_i d_i} \tag{2.8}$$

we get

$$P(\cap_i \mathcal{E}_i) = \lim_{k\to\infty} \prod_{j=0}^{k-1} (1 - \frac{1}{ka - jb}) \tag{2.9}$$

Combining terms for j and $k - j$, we get

$$\begin{aligned} &(1-\frac{1}{ka - jb})(1 - \frac{1}{ka - (k-j)b}) \\ &\quad = (\frac{ka - jb - 1}{ka - jb})(\frac{ka - (k-j)b - 1}{ka - (k-j)b}) \\ &\quad = \frac{k^2a^2 - k^2ab - 2ka + j(k-j)b^2 + bk + 1}{k^2a^2 - k^2ab + j(k-j)b^2} \end{aligned} \tag{2.10}$$

Assuming $a \gg b$, we can neglect terms involving b^2. Then, the above term can be written as

$$\approx (1 - \frac{1}{k}(\frac{2a - b}{a^2 - ab})) \tag{2.11}$$

There are $k/2$ such terms in Equation 2.7. Therefore,

$$P(\cap_i \mathcal{E}_i) \approx \lim_{k\to\infty} (1 - \frac{1}{k}(\frac{2a - b}{a^2 - ab}))^{k/2} = e^{-\frac{2a-b}{2a(a-b)}} \tag{2.12}$$

This problem has a similarity with the M/D/1/1 queueing model used in Queuing Theory [48, 49, 28] and similar results can be obtained using those methods. However, our problem is a special case of that problem and it is much easier to derive the results directly than by using those methods.

Non-stationary Object Density

In general, λ is a function of the location. For example, the object density near a door might be higher. Moreover, the presence of an object at a location influences the object density nearby since objects tend to appear in groups. We can incorporate both of these influences on the object density with the help of a density function $\lambda(\mathbf{X_c}, \mathbf{X_O})$. This function varies with both the current location ($\mathbf{X_c}$) at which we are calculating the density, and the object location $\mathbf{X_O}$ which is the location of the object for which we are calculating the visibility.

It can be shown that the non-stationarity of the density function affects the values a and b in Equation 2.12, such that:

$$a = \frac{1}{\int \lambda(\mathbf{X_c}, \mathbf{X_0})\, \mathrm{d}\mathbf{X_c}}, b = a\lambda_{avg}\pi(2r)^2 \tag{2.13}$$

where $\mathbf{X_c}$ is varied over the rectangular regions of width $2r$ and lengths d_i, and λ_{avg} is the average object density in the region.

2.2.2 Visibility from Multiple Sensors

In many applications, it is desirable to view an object from more than one sensor. Stereo reconstruction and depth recovery in an example where the requirement for visibility from at least two sensors is to be satisfied [22, 70, 60]. The probabilities for these requirements are also not too difficult to calculate. In order to evaluate the visibility from at least two sensors, we need to calculate the quantity:

$$P(\cup_{(i<j)}(\mathcal{E}_i \cap \mathcal{E}_j)) \tag{2.14}$$

This can be expanded exactly like Equation 2.1 treating each term $(\mathcal{E}_i \cap \mathcal{E}_j)$ as a single entity. All the terms on the RHS will then have only intersections in them which are easy to compute using the formulas developed in the previous sections (Eq. 2.5 or 2.12).

2.2.3 Simulations and Experiments

We have proposed a stochastic algorithm for recovering the optimal sensor configuration with respect to certain visibility requirements. In order to validate the proposed method, we consider various configurations and provide the visibility maps obtained for them. A preliminary C implementation is provided in the Appendix.

In Figure 2.3, the probability maps obtained for the case of one, two, three and four sensors respectively are shown. The sensors are mounted H = 10m above the ground in an area of size 50mX50m. We use Object density $\lambda = 1m^{-2}$, object height 150cm, object radius r=15cm, minimum visible height h=50cm and maximum visibility angle $\alpha_{max} = 30$. The average visibility probabilities obtained for the four cases were (a)

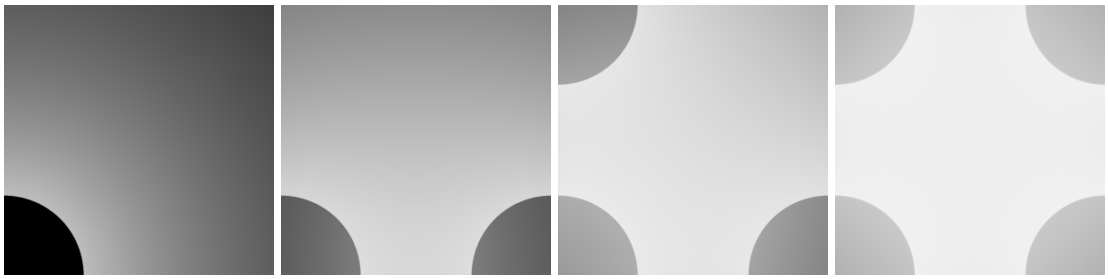

Figure 2.3: Visibility probability maps for 1,2,3 and 4 sensors in the scene. H=10m, R=50mX50m, $\lambda = 1m^{-2}$, , r=15cm, h=50cm, and α_{max} = 30 degrees. The average visibility probabilities were (a) 0.4296, (b) 0.672, (c) 0.8095, and (d) 0.888 respectively.

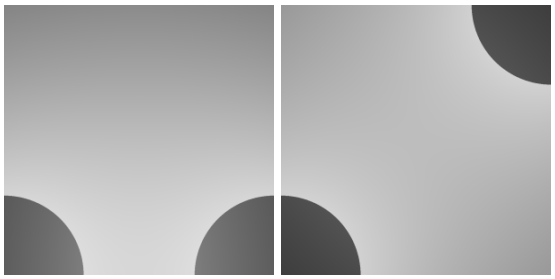

Figure 2.4: Visibility maps for two different configurations of two sensors in the scene. Again, H=10m, R=50mX50m, $\lambda = 1m^{-2}$, r=15cm, h=50cm, and α_{max} = 30 degrees. The average probabilities were (a) 0.672 and (b) 0.673

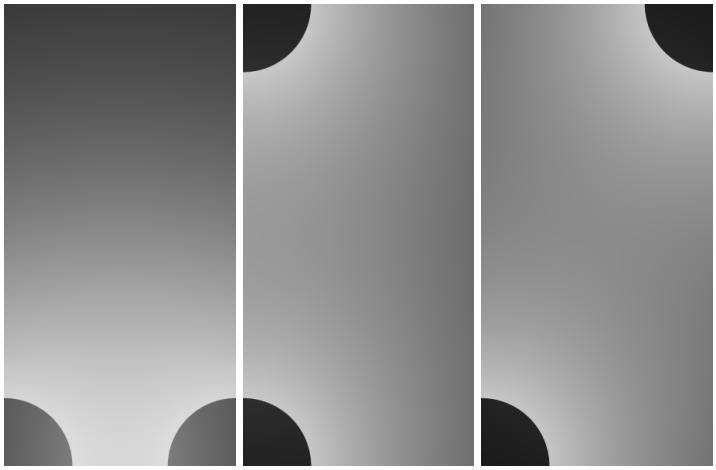

Figure 2.5: Visibility probability maps for two different configurations of two sensors in the scene. The average probabilities were (a) 0.5218, (b) 0.5499 and (c) 0.5514

0.4296, (b) 0.672, (c) 0.8095, and (d) 0.888 respectively. This information can be used to select the appropriate number of cameras based on the requirements.

Furthermore, for illustration purposes, we compare two configurations of two sensors in a square region. In the first case, the sensors were placed side by side and in the other, they were placed opposite each other. It turns out that the average visibility is more or less the same for the two arrangements with a slight improvement for the case with sensors placed at opposite corners. (Fig. 2.4).

Next, we compare the results when the region is rectangular of size 50mX100m. Keeping other parameters the same, we observe that there was a distinct improvement in the average visibility when the two sensors were placed along the long side of the rectangular region as compared to the short side. Placing them on opposite corners had only marginal effect (Fig. 2.5).

2.3 Deterministic Reasoning

Different techniques can be considered in order to improve the understanding of the problem. The formulas in the previous section can be shown to yield similar results, but the math is complicated and non-intuitive. Because of these limitations, we propose the following theorems:

2.3.1 Point Objects

For point objects, we can prove the following:

THEOREM 3.1

Part 1: For k point objects in the scene, k sensors are both necessary and sufficient to always "see" all the objects from at least one sensor.
Part 2: If we want at least m sensors to see an object always, then k+m-1 sensors are both necessary and sufficient.

PROOF:
Part 1
(a) *Necessary*
Suppose there are $n < k$ sensors and k objects in the scene. Then, consider the following configuration. Take one object, say O_1, and place $n - 1$ objects such that each of them is obstructing one of the sensors (Fig. 2.6 for the case of 6 sensors and 7 objects). In this situation, O_1 is not visible from any of the sensors.
(b) *Sufficient*
Consider any configuration of sensors where all of them are looking at a common area. Now, consider any one object, say O_1. This object has k line of sights to the sensors. We assume that the sensors are placed so that these line of sights are all distinct. However, there are only k-1 objects that can obstruct these lines of sight (the objects are assumed to be point objects, so they cannot obstruct two sensors simultaneously). Therefore, by simple application of the pigeon-hole principle, there must be at least one sensor viewing O_1. Since the choice of the object O_1 was arbitrary, this holds for all k objects in the scene.

Part 2
(a) *Necessary*
Similar to the reasoning of Part 1(a) above, suppose there are $n < k + m - 1$ sensors and k objects in the scene. Then, for an object O_1, we place $p = min(k - 1, n)$ objects such that each obstructs one sensor. The number of sensors having a clear view of the object are then equal to $n - p$ which is less than m (follows easily from the condition $n < k + m - 1$).

(b) *Sufficient*
Consider any object O_1. This object has k+m-1 lines of sights to the sensors, k-1 of which are possibly obstructed by other objects. Therefore, by the extended pigeon-hole principle, there must be at least $(k + m - 1) - (k - 1) = m$ sensors viewing O_1. Again, the arbitrary choice of O_1 means that this holds for all objects in the scene. □

The actual arrangement of sensors does not matter as long as no two sensors are along the same line of sight from any object. This is due to the peculiarity of considering only point objects.

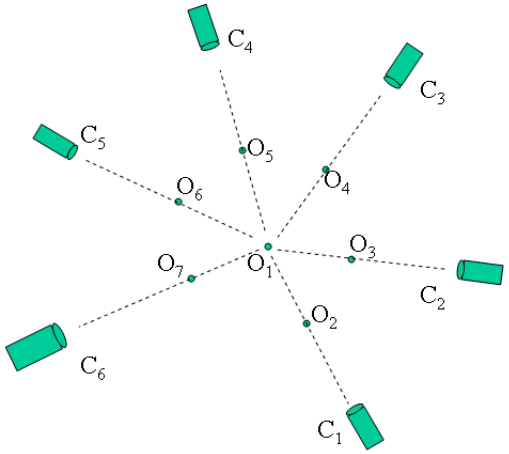

Figure 2.6: Six sensors are insufficient for always "seeing" seven point objects

2.3.2 2D Finite Objects

The above results hold for only point objects and no guarantees can be given for finite objects unless some structure is assumed for them. In this section, we present some results for finite objects assuming a flat world scenario where the objects and the sensors are in 2D.

THEOREM 3.2
Assume that there are k (possibly non-identical) objects in the scene such that the maximum angle that any object can subtend at the center of any other object is α. For example, for identical cylinders, $\alpha = 60$degrees and for identical square prisms, $\alpha = 90$degrees. The assumption that we make is that we want to see the center of the object. This center point can be arbitrarily defined. Then, the following results hold:
Part 1: For $k <= 360/\alpha$ objects in the scene, k sensors viewing the objects at at least a separation of α degrees between them (as seen from the objects) are both necessary and sufficient to always "see" all the centers of the objects from at least one sensor.
Part 2: If we want at least m sensors to always see an object, then $k+m-1 (k+m-1 <= 360/\alpha)$ sensors are both necessary and sufficient.

PROOF:
Part 1
The necessary condition follows directly from Theorem 3.1 Part1(a). For the sufficiency condition, consider an object, say O_1. The center of this object can be obstructed by another object for a maximum viewing angle of α (Figures 2.7 and 2.8). Therefore, if

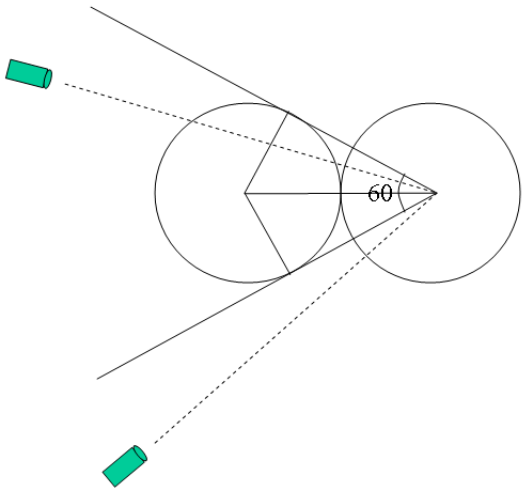

Figure 2.7: α = 60 degrees for identical cylinders. A 60 degree separation between sensors ensures that one cylindrical object can only obstruct one sensor.

the sensors are separated by an angle $> \alpha$, no single object can obstruct more than one sensor. Since there are only k-1 objects and k sensors, by simple application of the pigeon-hole principle, there must be at least one sensor viewing O_1. Since the choice of the object O_1 was arbitrary, this holds for all k objects in the scene.

An essential assumption here is that $k <= 360/\alpha$ since it is not possible to place $k > 360/\alpha$ sensors such that there is a separation of more than α degrees between them.

Part 2

The necessary condition follows trivially from Theorem 3.1 Part 2(a). For the sufficiency condition, consider an object O_1. If the angle of separation of the sensors is greater than α, then only one sensor can be obstructed by an object. There are k+m-1 lines of sights to the sensors, only k-1 of which are possibly obstructed by other objects. Therefore, by the extended pigeon-hole principle, there must be at least $(k + m - 1) - (k - 1) = m$ sensors viewing O_1. Again, the arbitrary choice of O_1 means that this holds for all objects in the scene.

Again, the condition $(k + m - 1 <= 360/\alpha)$ is essential so that it is possible to place the sensors with an angular separation of α.□

In order to track *people* from sensors viewing the scene from sides, one can assume that objects are cylindrical or have an elliptical cross-section. Therefore, α can be set to be 60 degrees or slightly more (for elliptical).

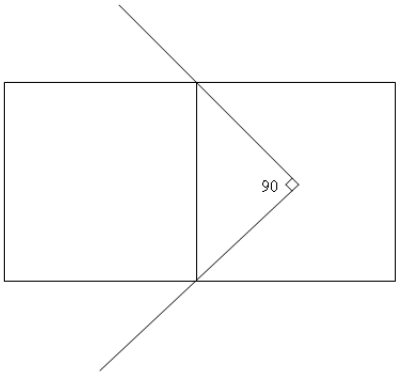

Figure 2.8: α = 90 degrees for identical square-prisms. A 90 degree separation between sensors ensures that one square-prism can only obstruct one sensor.

2.4 Sensor Configuration of Multi-Sensor Systems

2.4.1 Maximizing Visibility

Using both stochastic and deterministic reasonings, we have concluded that in order to achieve maximum visibility, the sensors must be placed as far from each other as possible. The "distance" measure that is relevant, is the angular separation between the sensors as seen from the object so that the view of one sensor does not affect the view of another sensor [1].

For a given minimum angular separation α, it can easily be shown that the region that two sensors cover is a circle passing through the centers of the two sensors such that the angle that the two centers subtend at the center of the circle is 2α (Figure 2.9). This result is derived from the condition that the angle subtended by a chord at any point on a circle is half the angle subtended by it at the center.

The generalization of this analysis to n sensors is straightforward. The area covered by n sensors is maximized by putting them on a circle with equal angular separation (Figure 2.10). In this circular region, from any vantage point, any two sensors will have an angular separation of at least α. The radius of the circular region is variable, but is constrained by the resolution of the sensors since objects at some distance from the sensors will not be clearly visible. The assumptions are that the possible area is

[1] Incidentally, this condition also gives a guarantee for a certain resolution in depth reconstruction from two stereo images.

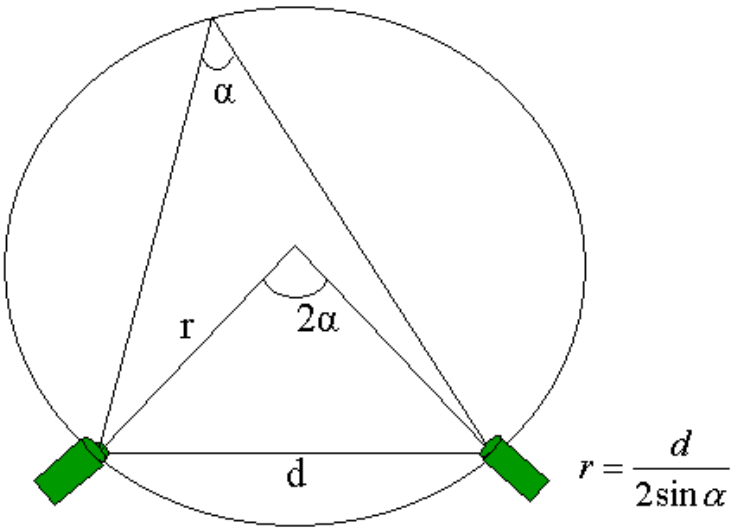

Figure 2.9: Every point within the circle has an angle $> \alpha$ to the sensors

unlimited, the sensors have unlimited field of view and that there is no constraint on the placement of the sensors.

When such constraints exist, such simple reasoning may be used as a rule of thumb. However, detailed analysis must be performed since there are exceptions to the rule. For example, the field of view of the sensors and the scene structure can greatly affect the optimum sensor placement.

2.4.2 Integrating Algorithmic Requirements in Automated Vision Systems: Discussion

The sensor arrangement providing the highest visibility is a necessity for manned systems. However, there are additional issues that need to be taken into consideration when the system is automatic and detection is done by a computer. The widely separated sensor arrangement is not conducive for point-matching (for e.g. stereo) algorithms that need a high similarity between the views. For these systems, there is a trade-off between visibility and matching accuracy.

The approach that most current systems [13, 12, 51] take is to sacrifice visibility for matching accuracy by using "stereo" pairs of sensors. Recovery of the scene structure is obtained by matching 2D information only within the sensor pair and then the 3D information is integrated across pairs. The sensor arrangement helps the algorithm detect people accurately *when* an object is visible, but visibility can be severely limited in

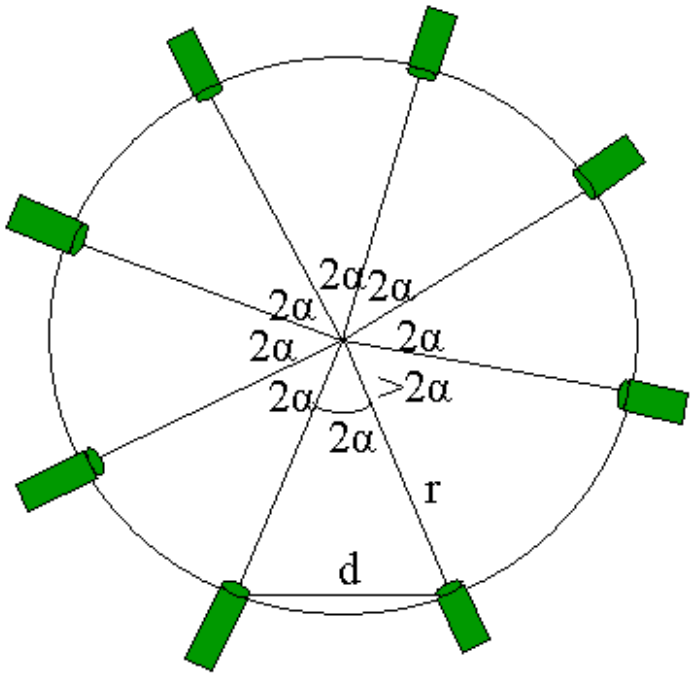

Figure 2.10: The best arrangement of sensors such that the common area where the angle to any two sensors is at least α is maximized

crowded conditions. Another approach [65] sacrifices matching accuracy for better visibility. This approach requires methods to deal with the problem of matching information across sensors placed far from each other ([76, 89, 107, 106]). A hybrid approach [8] has also been used where information from both close and widely separated sensors has been integrated.

An alternative [27, 44, 47] is to not match information directly across the views, but to merge the detections observed by multiple sensors in a consistent manner. Since 2D matching is not required, the sensors are not constrained to be close to each other and therefore, they can be placed for maximum visibility.

Some multi-sensor systems neglect occlusion and have sensors look at different parts of the scene, solving for the problem of matching detections observed at different times from different sensors as the object moves around the scene [46]. The sensor views can also have some overlap [5, 8] in order to simplify the object matching problem. Such a technique can only deal with minimal occlusion.

In conclusion, optimum sensor placement is a function not just of the visibility but also of the application and the vision algorithm used. If matching algorithms are used, visibility and matching accuracy have conflicting requirements that have to be met. For some other algorithms, this trade-off may not exist, but we may not be able to obtain as

good detection.

2.5 Conclusion

We have presented a stochastic framework for evaluation of the visibility probability given a certain configuration of sensors in a scene. Apart from yielding important performance characteristics of multi-sensor systems, it can be used to evaluate different sensor configurations, and to determine the one that should be used. Future work in this area might include extending the framework for dynamic environments, taking into account motion models of objects to better estimate the occlusion probability.

Chapter 3

Scene Analysis Using a Single Camera

In this chapter, we address the problem of monitoring activity over a wide area, using a single moving camera. There has been considerable recent work on constructing wide area or panoramic views from multiple images (e.g., [3], [41], [108] and [101]), however such work generally presumes that the scene is static (there are no moving objects in the field of view). There has also been recent work on activity monitoring and detection (e.g., [98], [50]), however such work generally assumes a non-moving camera. In contrast, we address the problem of activity monitoring and detection over a wide area, where a moving camera is used to "sweep" over the area of interest. Our approach is based on a combination of the image mosaic techniques that have been used for constructing panoramic views, and the mixture model techniques that have been used for activity monitoring. We create a model of the wide field of view that can be used to distinguish between the static (or stationary) *background* and the moving objects, or *foreground*.

Our method can be used for a number of applications, both in surveillance and monitoring and in image synthesis. For surveillance and monitoring, the method can be used to detect moving objects over a wide field of view area, to detect common patterns of activity over that field of view, and to detect activity that does not fit the common patterns. For image synthesis applications, our method can be used to create panoramic views that contain just the non-moving objects in a scene, to create synthetic panoramic views that place each moving object at just a single location, and to create synthetic videos that remove all or selected moving objects. In this paper we present examples illustrating these applications.

Background modeling in a wide field of view is not only useful for the above applications, it also enables improved accuracy in distinguishing between moving objects and the background. These improvements result from the fact that the wide field of view model has more information, and more stable information, than is present in individual image frames or adjacent frames. The same effect of increased accuracy is observed in standard image mosaic techniques for constructing panoramic views of static scenes. The most accurate image mosaic techniques are based on aligning each image frame with the overall mosaic rather than simply with the previous frame (or with a few

frames nearby in time). We illustrate these improvements in accuracy by contrasting results using our wide area background modeling method with previous techniques. We show both that mosaic quality is improved and that very small moving objects can be detected.

3.1 A General Overview of the Algorithm

Our method uses mixture models to form a model of the background. We represent each pixel location in the mosaic by a mixture of Gaussians. From these mixture models, we form an image representing the background by taking the mean of the highest weight Gaussian for each pixel. The highest weight Gaussian is the one that has the highest probability of occurrence. Without a lighting change, this will be the one that accounted for the largest number of observed pixel values, thus we assume that the background is observed more often than any foreground objects at each location. This background image can be used as a panoramic view of the "background scene", without any foreground objects.

For each new image obtained from the camera, we register it to this panoramic background image using any known good registration technique. The current image, having been registered with the background image, provides the input pixel values for updating the mixture models at each pixel where new data is observed. Once these mixture models have been updated, a new background image is computed from these updated mixture models. This process is repeated for each new frame obtained.

As noted in the previous section, the presence of moving objects can lead to an erroneous registration results. By aligning each image frame with the background image, our technique avoids this problem. As the background image does not contain the moving objects, there is generally no good match for the portions of the image corresponding to moving objects, and thus they have little effect on the solution for the best registration transformation.

For the problem of detecting moving objects, a background image can also be very useful. Most techniques for detecting moving objects operate by registering successive image frames, and then subtracting the registered frames to see where there are differences. This only detects regions where the images are different, which is not necessarily the entire moving object (e.g., when part of object overlaps in the two frames). In contrast, when registering an image frame to a panoramic background image, the entire moving object can be readily detected based on the difference.

3.2 Background Modeling

We model the scene using a mixture model for each of the pixel locations in the mosaic. The probability of a pixel belonging to a particular Gaussian is proportional to

the weight ascribed to that Gaussian. Within a particular model, the probability is distributed according to the Gaussian probability distribution scheme. More specifically, the probability of a pixel having an intensity value x, given that it belongs to a particular Gaussian j, is

$$Pr(X = x|J = j) = \frac{1}{\sigma_j\sqrt{2\pi}} e^{-\frac{1}{2}\frac{(x-\mu_j)^2}{\sigma_j^2}} \tag{3.1}$$

where the random variable J denotes the Gaussian that a pixel belongs to, and μ_j and σ_j are the mean and standard deviation of the jth Gaussian. This scheme can easily be extended to color images by using multi-dimensional Gaussians, but here, we will deal only with 1-d Gaussians and gray images rather than color ones.

The probability that a pixel has intensity value x is then simply

$$Pr(X = x) = \sum_{j=1}^{K} w_j Pr(X = x|J = j) \tag{3.2}$$

where $w_j = Pr(J = j)$ is the weight of the jth model.

This scheme is useful in modeling multi-modal backgrounds and scene changes, as well as modeling transient moving objects. A multi-modal distribution can result from the swaying of trees, blinking of lights, or any other periodic change in image intensity. In a slow scene change, such as occurs with most outdoor changes in illumination, the weights of the different models would gradually shift to the new background. Often, the background change is only transient and is due to some moving object. In such a case, the weights would gradually revert back after the moving object is removed.

One can determine the best mixture model given the previous n values using what is called the Expectation Maximization (EM) algorithm (see Dempster et. al. [15]). We could run this algorithm for each time frame by taking a window of the previous n frames, or calculating the result for all the pixels so far. However, this algorithm is quite time-consuming and is impractical to run for each frame. Neal and Hinton [71] provide a scheme for incrementally updating the solution using the new input values which has been used successfully by Friedman and Russell [24] to get good mixture models. The incremental scheme does not yield the best solution, but only guarantees to converge to a local minimum. However, it has the drawback of not being able to handle changing background scenes properly since it does not have any scheme to reduce the effect of previous input values. The results when the background has more than the given number of Gaussians are also not clear.

Instead of using the expectation maximization scheme, we use a conceptually simpler scheme which is computationally less expensive and we have found also yields better results for our problem. We determine the mixture models by incrementally updating the models using the intensity values of the pixels that are registered at a particular location. First of all, we determine which model (Gaussian) the observed pixel value belongs to. A pixel can belong to a particular model if its distance from the mean is within

a constant (a value between 2 and 3 is suitable) times the standard deviation. If a pixel belongs to more than one Gaussian, we take the one having the highest weight. This is done so that if we create more than one model corresponding to what is in actuality a single underlying distribution, we will eventually get only one Gaussian instead of two. The intensity value of the current pixel is then used to update the models. We can use a constant weight updating scheme and an exponential weighting scheme. However, since the different schemes are useful in different scenarios, we use both of them, switching between the two depending on some criteria which we will define shortly. Experimentally, this combined scheme works better than either of the schemes used alone. Below we discuss these two schemes and how to select between them.

The mean and variance of the matched distribution are updated as follows

$$\mu_{j,t} = (1-\rho)\mu_{j,t-1} + \rho x_t \tag{3.3}$$

$$\sigma^2_{j,t} = (1-\rho)\sigma^2_{j,t-1} + \rho(x_t - \mu_{j,t})^T(x_t - \mu_{j,t}) \tag{3.4}$$

where ρ is a constant which determines the rate of change of the mean and variance.

Note that the mean and variance are updated according to an exponential scheme where the recent pixels get exponentially higher weight. We can also have a scheme whereby all the pixels get equal weighting. However, the exponentially weighted scheme would be able to capture slowly changing scenes more easily and hence is used here.

If there is no model that the current pixel belongs to, a new Gaussian is added with a mean equal to that of the pixel value and a high variance. The weight is determined according to the model we are using. Due to a limited amount of memory space, we also require garbage collection when the number of models exceeds some maximum value. At that time, we simply remove the model with the least weight.

The weights of the distributions are also updated, using either a constant or exponential model, as we now describe.

3.2.1 The Constant Weight Updating Model

In this updating scheme, we update the weights of the models in a manner that awards equal weight to all the pixels registering to a particular location in the mosaic. To calculate the weights, we keep track of the number of pixels matching a particular mixture model and also the total number of pixels at that particular location. The weight for a given model is then simply the number of matching pixels divided by the total number of pixels. To update the weight of the model, we simply increment the number of pixels belonging to the matching model and the total number of pixels at a location. The mean and variance are updated using equations (3.3) and (3.4).

This scheme yields a result which works quite nicely in practice as long as the background does not change. The background image, corresponding to the means of the highest weight Gaussian models, would consist of those intensities that are visible for

the longest amount of time. Even if the view is obstructed for considerable time by a transient object such as a person standing for a long time, or a car stopping at an intersection, the background would still not change for a long time (presuming many observations of the background intensity beforehand). Thus, it would yield the correct result even in the case of slow moving objects on a road, or a road obstructed by transient objects for a long time.

This model would encounter problems, however, if the background changes abruptly due to a large obstructing object or lighting changes. A large obstructing object cannot be modeled properly by any scheme, but it can be detected by our algorithm easily by a large mean square error and the large number of unmatched points in the image. We handle the lighting changes by using the exponential model when there is a lighting change as opposed to a registration error.

3.2.2 The Exponential Weight Updating Model

In the exponential weighting scheme, we update the model parameters by awarding exponentially less weight to values that were observed earlier in time. The weights of the models are updated as follows. For each model j, the new weights are

$$w_{j,t} = (1 - \alpha)w_{j,t-1} + \alpha(M_{j,t}) \tag{3.5}$$

where α is a constant determining the rate of change of the models, and $M_{j,t}$ is 1 if the current pixel is matched to the model j, and is equal to 0 if it is not the matched distribution. As with the constant weight updating scheme, the mean and variance are updated using equations (3.3) and (3.4).

The exponential scheme has been previously used by Stauffer et. al. [98]. However, used alone, this creates a tradeoff as far as the rate of change of the model is concerned. A slow rate of change would be unable to model the lighting or background changes properly and fast enough. In a fixed camera scene, this could mean a wrong result for quite some time, while in a moving camera scene, it could even throw off the background registration. On the other hand, a fast rate of change would mean that a foreground object would start to appear as background in a very small amount of time. Also there are problems due to the exponential nature of the updating, which are especially visible when the number of pixels at a particular location is small or when there are slowly moving foreground objects.

3.2.3 Choosing the Weight Update Scheme

In our algorithm, we use the constant weight updating scheme in the normal mode. In the case that a lighting change is detected, the weights are updated using the exponential scheme where the number of pixels belonging to the models changes exponentially, i.e.

$$n_{j,t} = (1 - \alpha)n_{j,t-1} + \alpha(M_{j,t}) * (Totalnumberofpixels) \tag{3.6}$$

where the definitions are similar to equation (3.5). Note that this would require these variables to assume real values as opposed to integer values.

In order to distinguish between different scenarios, we calculate three quantities: (1) the normalized correlation between the background image and the current frame warped according to the calculated projective transformation values; (2) the mean square error between the background image and the warped current frame; and (3) the number of points in the current frame that do not match any of the Gaussian models with a large enough weight. These quantities are used to estimate whether there is a lighting change, registration error or other situation that warrants changing how the models are updated for the given frame.

A high normalized correlation and a high number of unmatched points is taken to indicate a lighting change, because the normalized correlation is not affected by a overall additive or multiplicative changes but most pixels will not find a good matching model in such a case. When this occurs, we use the exponential weight updating scheme for the models, so that the change in lighting is quickly adapted to by the background models.

A low normalized correlation, combined with a high number of unmatched points and a high mean square error is taken to indicate either a complete change of background or a registration error. In this case, the models are not updated; the frame is simply discarded. A low number of unmatched points and a high mean square error are taken to indicate a transient object occupying part of the image. The program should continue to run in the normal mode (constant weight updating).

3.3 Image Registration and Mosaicing

As we have discussed in the introduction, our algorithm provides a method for image registration and mosaicing that is robust with respect to moving objects. In contrast, most of the methods currently used for image registration and mosaicing can easily be thrown off by the presence of objects which have image motion that differs substantially from that of the background. This can introduce errors in the registration, and these errors can accumulate over time. In such cases the resulting mosaic will not be very good. In the results section below we illustrate this difference using an aerial video sequence.

Our method for image registration and mosaicing is based on registering the current frame with the background image that has been derived from the mixture models. This background image contains at each pixel, the mean of the highest weighted Gaussian, and is an approximation to the highest probability value at each pixel. The current frame can be aligned to this background image using any good registration technique. In our current implementation we use a hybrid registration method that first solves for an affine transformation based on feature correspondences, and then uses that transformation to initialize a direct method of solving for a projective transformation. The first step uses the KLT feature tracker (described in [92] and [57]) to find corresponding features in

the images and then solves for an affine transformation using a robust least squares fit. This affine transformation is used to initialize a direct method which yields a projective transformation between the images. The direct method operates in an iterative manner using the Levenberg-Marquardt method, a well known algorithm that is described in various papers (e.g., [3], [34]). It is also possible to use the Levenberg-Marquardt method directly, although use of the KLT tracker speeds up the registration.

Our algorithm provides a method for mosaicing that is robust to the presence of moving objects. First, as with several traditional mosaic techniques, we register each image to the entire mosaic rather than simply the previous frame. This limits cascading of registration errors. Second, when registering an image with the background mosaic, it is unlikely that there will be matches for moving objects. Thus the moving objects will not throw off the registration process. In general the background image does not contain any foreground objects (although when there are just a few images corresponding to a given pixel this may happen). Thus there is generally nothing in the background image that matches the foreground objects well, and the resulting registration is not affected by the moving objects.

This method also provides a good means of aligning successive frames, by constructing a background mosaic (although portions of the mosaic can be discarded over time if only successive frames are to be aligned).

3.4 Detecting Foreground Objects and Site Classification

To detect foreground (moving) objects in the current frame we first compute the registration of the frame to the panoramic background mosaic. Then we warp each pixel to the coordinate system of the mosaic, and search the nearby pixels for a matching Gaussian model. A small neighborhood is searched so as to allow for small alignment errors or small changes in background that may be caused by the swaying of trees etc. If the pixel does not match to any of the Gaussians in this neighborhood with a sufficiently large weight, we declare this point to be new, else it is part of the background. We finally run a connected components algorithm on this image to get rid of noise. Sufficiently large objects can then be taken as the foreground objects.

Note that slowly moving objects will not be detected as foreground objects since the weights of the Gaussians that such pixel values belong to will not be high enough to be counted as background. Thus slowly moving objects, which are often problematic in exponential forgetting algorithms, are not an issue here. The approach is able to quite accurately distinguish between moving objects and the background.

Based on the foreground objects that are detected in each frame, we can also perform site classification depicting the activity taking place at a particular location. This can be useful for surveillance applications of sites such as roads and parking lots, in which it is desirable to tell where in the image there is activity, or when that activity occurred. One

Figure 3.1: (a) Two views of a parking lot using a moving surveillance camera, (b) synthesized frames with foreground objects removed, and (c) the moving objects detected. Note that very small objects, such a person walking and a car entering behind trees were correctly detected.

such model can be obtained by simply keeping track of the amount of activity at each pixel in the panoramic mosaic of a scene. In the next section we show a gray image in which the individual pixels store the number of times that that a pixel of the mosaic registered a foreground object. Thus, areas in which more moving objects have been detected will show up as brighter, and areas in which no moving objects were detected will show up as black.

3.5 Results

In this section we present some of the results of our algorithm. The algorithm was found to work quite well over a range of scenarios including aerial video and ground-based surveillance of a region using a panning camera, for scenes containing multiple moving objects and a mix of roads, parking areas and walkways. Lighting changes were generally correctly detected and the algorithm was able to recover reasonably from these changes, yielding accurate models of the background within a few frames after the lighting change.

Figures 3.1 and 3.2 each show two frames from two video sequences (note these frames are not adjacent in time). The first row shows the original frames, the second row shows the synthesized frames in which all the foreground objects have been removed, and the third row shows the corresponding moving objects that were detected. The moving objects have clean boundaries and the entire motion area is detected, in contrast

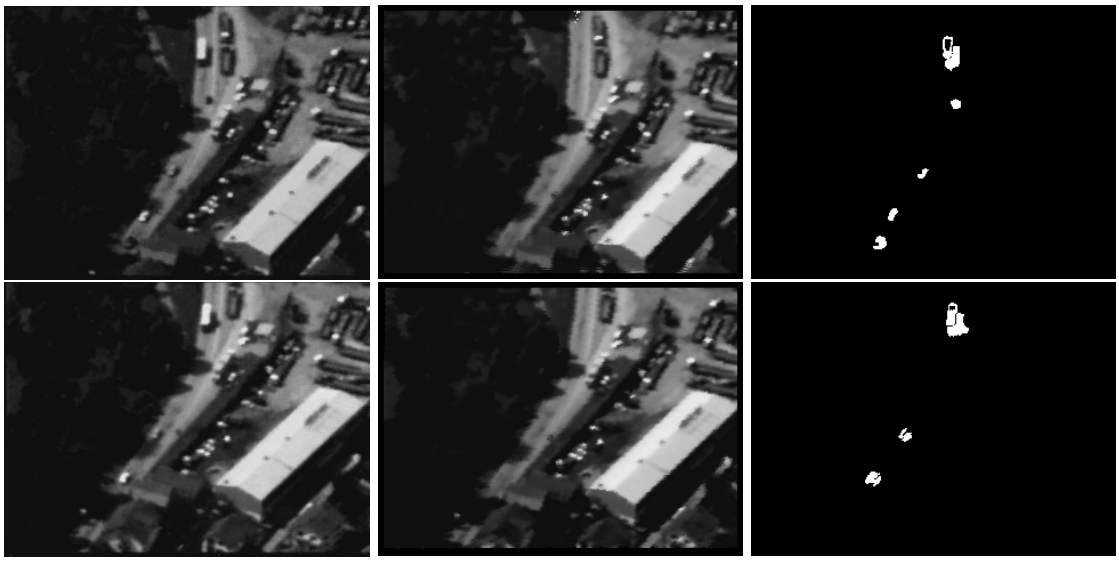

Figure 3.2: (a) Frame no. 42 and 57 from an aerial video, (b) synthesized frames with the moving objects removed, and (c) the corresponding moving objects detected.

with methods based on the difference between registered frames, which detect just the part of the image that is different between frames. Note that in Figure 3.1 two very small objects are successfully detected. In the first column, the small object to the left of the car corresponds to a person walking on the sidewalk. In the second column, the small object on the very right is a car entering behind the trees. The synthesized frames are also quite clean, not containing evidence of the moving objects or parts of them.

Figure 3.3 shows the background mosaic obtained for the 150 sequence from Figure 3.2. Note that at the upper left there is some evidence of the bus in this background mosaic, because that area was not observed for many frames without the bus present. Otherwise, however, there is no evidence of the moving objects in the mosaic. Figure 3.4 shows the corresponding "activity image" indicating the number of times that each pixel registered a moving object. The road is clearly visible in this image as a region of high activity (with the shadows cast by trees also visible by the fact that they break up a "stream" of activity).

In Figure 3.5 we present the background image obtained by registering the current frame with a more traditional image mosaic rather than the background image. This mosaic is formed by taking the intensity value of the last registered pixel at a given location, rather than using the mixture models to form a background image. This is a common technique, but as is visible from this mosaic from the first 75 frames of the sequence, the registration is not very robust. After 100 or so frames, the registration is totally off. This example illustrates the utility of the background image for accurate image registration in the presence of moving objects.

Finally, in Figure 3.6, we present a background mosaic of a region using 1000

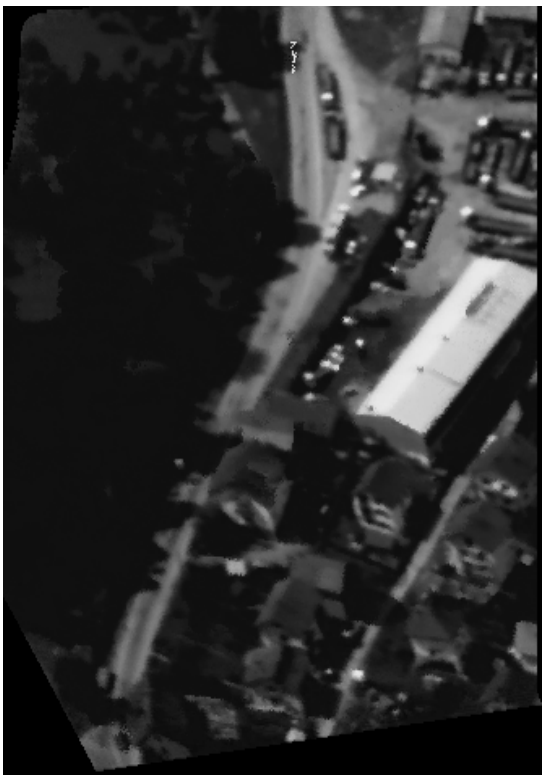

Figure 3.3: The background mosaic from an aerial video of 150 frames using our algorithm

frames. The mosaic is quite clear, indicating accurate and stable alignment over 1000 frames in the presence of many moving objects in the form of moving people on the pathways. There is no evidence of these moving objects in the mosaic, which illustrates the usefulness of the algorithm in constructing panoramic views that contain just the non-moving objects in the scene.

3.6 Summary and Conclusions

In this chapter, we have presented an algorithm for wide area surveillance and monitoring. The method uses a mixture of Gaussian model to represent pixels in the scene. From these mixture models, a model of the background is constructed using the mean of the highest weight Gaussian at each pixel. The background model provides a means of registering video frames that is robust in the presence of moving objects in the scene. The models can also be used to detect moving objects in a moving camera scene, and to create panoramic views and video sequences that do not contain any moving objects.

Figure 3.4: Figure where each individual pixel stores the number of times a foreground object was detected at that pixel. Note how the road is clearly distinguished from other areas due to objects moving on the road.

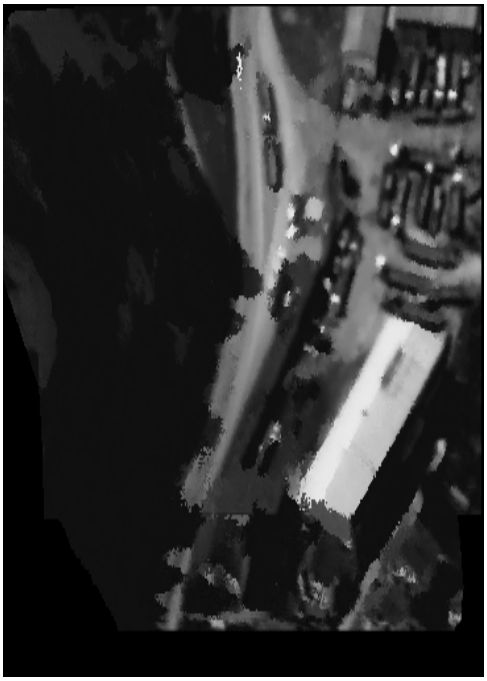

Figure 3.5: The background mosaic obtained when we register each frame with the mosaic where the mosaic is formed by taking the last pixel registered at that location.

Figure 3.6: The background mosaic from a sequence of 1000 frames captured from a surveillance camera moving in both vertical and horizontal directions

Chapter 4

A Multi-View Algorithm for Segmenting and Tracking People in a Crowded Scene

4.1 Introduction

When occlusion is minimal, a single camera is sufficient to reliably detect and track objects. However, when the density of objects is high, the resulting occlusion and lack of visibility necessitates the use of multiple cameras and collaboration between them so that an object is detected using information available from all the cameras in the scene.

In this chapter we describe a system for segmenting, detecting and tracking multiple people using a multi-perspective video approach. We address the situation that the scene being viewed is sufficiently "crowded" so that one cannot assume that any or all of the people in the scene would be visually isolated from any vantage point. Figure 4.1 shows images from a 6-perspective sequence that will be used to illustrate our algorithm. Notice that in all four images, no single person is viewed in isolation- i.e. neither occludes another person nor is occluded by another person. We assume that our cameras are calibrated, and that people are moving on a calibrated ground plane. We also assume that the cameras are frame synchronized.

Our algorithm takes a unified approach to the problem. We neither detect nor track objects from a single camera, or camera pair; rather evidence is gathered from multiple camera pairs and decisions of detection and tracking are taken at the end by combining the evidence in a robust manner taking occlusion into consideration. Also, we do not simply assume that a connected component of foreground pixels corresponds to a single object. Rather, we employ a segmentation algorithm to separate out regions belonging to different people. This helps us handle the case of partial occlusion and allows us to track people and objects in a cluttered scene where no single person is isolated in any view.

Good segmentation of people in a crowded scene is facilitated by models of the people being viewed. Unfortunately, the problem of detecting and finding the positions of the people requires accurate image segmentation in the face of occlusions. Therefore, we take a unified approach to the problem and solve both of them simultaneously. The

Figure 4.1: Images from a 6-perspective sequence at a particular time instant.

algorithm uses segmentation results to find people's ground plane positions and then uses the ground plane positions thus obtained to obtain segmentations; the process is iterated until the results are stable. This helps us obtain both good segmentations and ground plane position estimates simultaneously.

The chapter develops several novel ideas. The first and most important is the introduction of a region-based stereo algorithm that is capable of finding 3D points inside an object if we know the regions belonging to the object in two views. No exact point matching is required. This is especially useful in wide baseline camera systems where exact matching is very difficult due to self-occlusion and a substantial change in viewpoint. The second contribution is the development of a scheme for setting priors for use in segmentation of a view using Bayesian classification. The scheme, which assumes knowledge of approximate shape and location of objects, dynamically assigns

(1) Initialize using previous ground plane positions. In the beginning, assume there are no estimates available and all people are new.
(2) Segment images using person models and ground plane position estimates.
(3) Estimate ground plane positions using region matching.
(4) Repeat steps 2 and 3 till ground plane position estimates are stable.
(5) Update person models using information from images and the ground plane positions found.
(6) Repeat steps 1-5 for next time step.

Algorithm 1: Algorithm Structure

priors for different objects at each pixel so that occlusion information is encoded in the priors. These priors are used to obtain good segmentation even in the case of partial occlusions. The third contribution is a scheme for combining evidence gathered from different camera pairs using occlusion analysis so as to obtain a globally optimum detection and tracking of objects. Higher weight is given to those pairs which have a clear view of a location than those whose view is potentially obstructed by some objects. The weight is also determined dynamically and uses approximate shape features to give a probabilistic answer for the level of occlusion.

4.2 General Overview of the Algorithm

Our system models different characteristics of people by observing them over time. These models include color models at different heights of the person, "presence" probabilities along the horizontal direction at different heights, and the ground plane positions tracked using a Kalman filter. These models are used to segment images in each camera view. The regions thus formed are matched in pairs of camera views along epipolar lines. The matched segments are then used to yield 3D points potentially lying inside objects. These points are projected onto the ground plane and ground points are used to form an object location likelihood map using Gaussian kernels for a single image pair. The likelihood maps are combined using occlusion analysis so that pairs which have a clear view of a particular location are given higher weight than those whose views are obstructed by some other person. The algorithm is then iterated using these new ground plane positions and this process is repeated until the ground plane positions are stable. These iterations help to improve the quality and stability of the results. The final ground plane positions are then used to update the person models, and the whole process is repeated for the next time step.

4.3 Modeling People

We model the appearance and shape of the people in the scene. In order to simplify the problem, we assume that people are standing upright and that they do not squat or jump. These scenarios can also be included, but only at the cost of accuracy of the results since the models will have to be less discriminatory. These models are developed by observing people over time (method explained in section 11) and help us segment people in the camera views. These models are developed from the sequences automatically; no manual input is required.

4.3.1 *Color Models*

One of the attributes useful to model is the color distribution at different heights of the person. The distance from the ground plane to the top of the person is divided into regions of equal length and a color model is developed for each region. A single color model for the whole person would not be able to capture the vertical variation in color. On the other hand, modeling the horizontal distribution of color is very difficult without full 3D surface reconstruction, which would be too time-consuming and hence not too interesting for tracking and surveillance type of applications.

Pixels belonging to a particular person at a particular height are identified and the color model is developed using the well-known method of non-parametric kernel density estimation ([19]) which is well suited to our system. Let $c_1, c_2, ..., c_n$ be n observations determined to be belonging to the color model. The probability density function can be non-parametrically estimated [90] using the kernel estimator K as

$$Pr(c) = \frac{1}{n}\sum_{i=0}^{n} K(c - c_i) \tag{4.1}$$

If we choose our kernel estimator function, K, to be the Normal function, $N(0, \Sigma)$, where Σ represents the kernel function bandwidth, then the density can be written as

$$Pr(c) = \frac{1}{n}\sum_{i=0}^{n} \frac{1}{(2\pi)^{\frac{d}{2}} \mid \Sigma \mid^{\frac{1}{2}}} e^{-\frac{1}{2}(c-c_i)^T \Sigma^{-1}(c-c_i)} \tag{4.2}$$

If we assume independence between the different channels (dimensions of the observation vector) with different kernel bandwidths σ_j^2 for the jth channel, then

$$\Sigma = \begin{pmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ 0 & . & . & . \\ 0 & . & . & \sigma_d^2 \end{pmatrix} \tag{4.3}$$

Since the intensity levels change across cameras due to aperture effects, and due to shadow and orientation effects in the same view, we use normalized color (i.e. the ratios

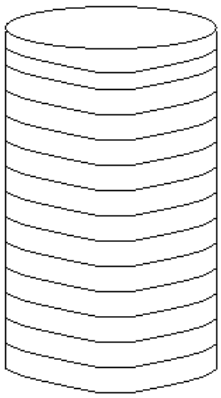

Figure 4.2: A color model is developed for each height slice of the person

$r/(r+g+b)$ and $g/(r+g+b)$). In order to have some differentiation between colors that are very different from each other in intensity but similar in chromaticity (for e.g. white and black), we add intensity as the third variable with high variance.

Note that having this color model does not restrict us to detecting people wearing uniform color clothes. Non-uniform colors can be handled as long as there are kernels belonging to all the colors present at a particular height. This is possible for us to do since we are viewing the people from all sides and are therefore able to build a color model that captures the color profile from all sides and hence is not viewpoint dependent.

4.3.2 *"Presence" Probabilities*

For our segmentation algorithm, we want to determine the probability that a particular person is "present" (i.e. occupies space) along a particular line of sight. Towards that end, we define "Presence" Probability (denoted by $L(h, w)$) as the probability that a person is present at height h and distance w from the vertical line passing through the person's center. This probability is a function of both the distance w and height h since, e.g., the width of a person near the head is less than the width near the center. This probability function also varies from person to person.

We estimate this probability by observation over time using the method described in section 10 (see Figure 4.3 for some samples). However, since this method is reliable only after a certain number of time steps, we employ a heuristic to model the "presence" probabilities in cases where we do not yet have sufficient data available. We model the width W of the person at a particular height h as a random variable with Gaussian distribution and assume that the person occupies all space from the center up to width W. Then, the probability that a person j is present at distance w and height h is given by

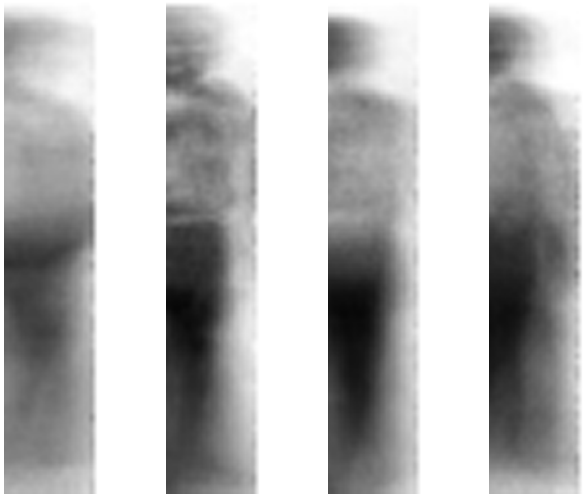

Figure 4.3: Sample Presence Probabilities of people observed over time. The x-axis is the width, y-axis is the height from the ground and the values shown are probabilities scaled by a factor of 256 for display purposes.

$$L_j(h, w) = P_{j,h}(w < W) = 1 - CDFG_W(w)$$

where $CDFG_W$ is the cumulative density function for the Gaussian function for W. The mean and standard deviation for W is varied to accommodate the fact that people have different distributions at different heights.

The model for presence probability used here is very effective in detecting the torso of the person, but is not very effective in detecting hands and legs far away from the center of the person. However, since our goal here is to find the location of the person, it is more important to find the torso than the hands and legs. If we want to detect hands and legs far from the person, one would have to use some other model for the shape prior, or would have to modify our model so that hands and legs are handled better.

4.4 Pixel Classification in a Single View

We use Bayesian Classification to classify each pixel as belonging to a particular person, or the background. The *a posteriori* probability that a pixel having observation $I(x)$ belongs to a person j (or the background) is

$$P_{posterior}(j/I(x)) \propto P^x_{prior}(j)P(I(x)/j)$$

The pixel is then classified as

$$\textit{Most likely class} = arg\max_j(P_{posterior}(j/I(x))$$

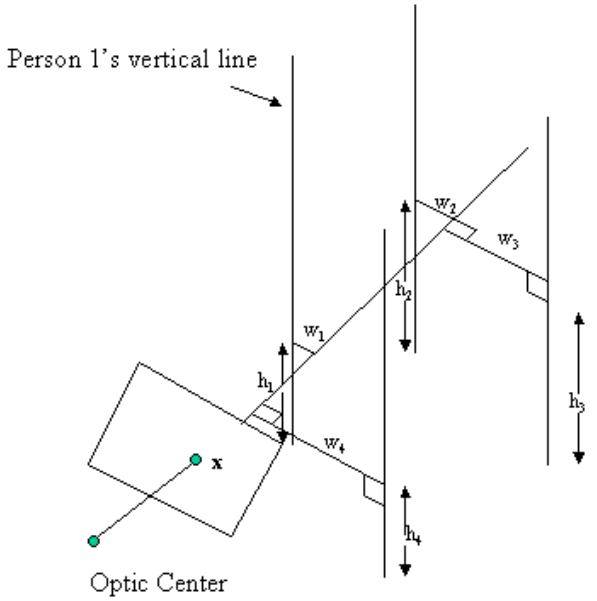

Figure 4.4: Measuring distances (and heights) from the line of sight

$P(I(x)/j)$ is given by the color model of the person at height h. For the background, we use a background model of the scene using the method described in [66].

We want the prior probabilities to include occlusion information so that the prior for a person in front is higher near his estimated position compared to the priors far away from him and compared to a person in rear. We employ the following methodology. For each pixel $\mathbf{x}$, we project a ray in space passing through the optical center of the camera (see Figure 4.4). We find the perpendicular distances w_j of this ray from the vertical lines passing through the currently estimated centers of the people. Also calculated are the heights h_j of these perpendicular lines(these line segments are horizontal). Then, the ***a priori*** probability that a pixel $\mathbf{x}$ is the image of person j is

$$\begin{aligned} P_{prior}(j) &= L_j(h_j, w_j) \prod_{k \text{ occludes } j} (1 - L_k(h_k, w_k)) \\ P_{prior}(bckgrnd) &= \prod_{all\ j} (1 - L_j(h_j, w_j)) \end{aligned} \tag{4.4}$$

where $L_j(h_j, w_j)$ is the "presence" probability described in section 4.2. A person k occludes" j if the distance of k to the optical center of the camera is less than the distance of j to the center.

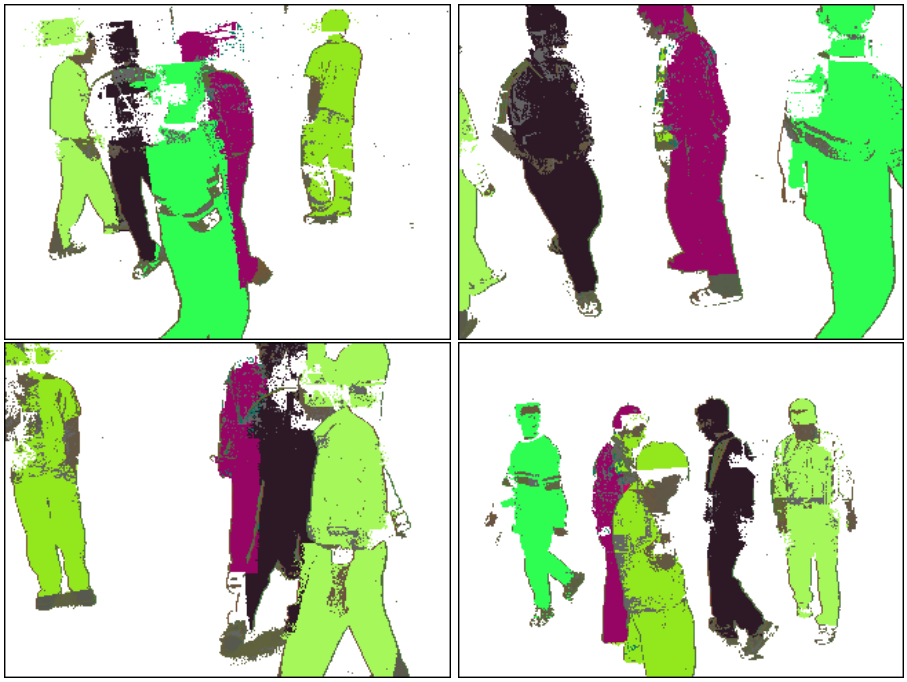

Figure 4.5: The result of segmenting four of the images shown in Figure 4.1

The motivation for the definition is that a particular pixel originates from a person if and only if (1) the person is present along that line of sight (Probability for this = L_j), *and* (2) no other person in front of her is present along that line of sight (Probability = 1 - L_k). If no person is present along a particular line of sight, we see the background. The classification procedure enables us to incorporate both the color profile of the people and the occlusion information available in a consistent and logical manner.

It is interesting to note that we expect to obtain better discrimination from the background using our algorithm than using traditional background subtraction methods because we take into account models of the foreground objects in the scene in addition to information about the background that is the only input for traditional background subtraction methods. Indeed, this is what we observe during experiments.

4.4.1 Detecting New People and Bootstrapping

The algorithm as described above assumes that we have information about the people visible in the scene and fails when there are people for whom we do not have any information or have inaccurate information. In order to detect "new" people in the scene and to correct inaccurate person models, we detect unclassified pixels as those for which $P_{prior} * P(c/j)$ is below a given threshold for all the person models and the background, i.e. none of the person models or the background can account for the pixel with a high enough probability. For these pixels, we use a simple color segmentation algorithm,

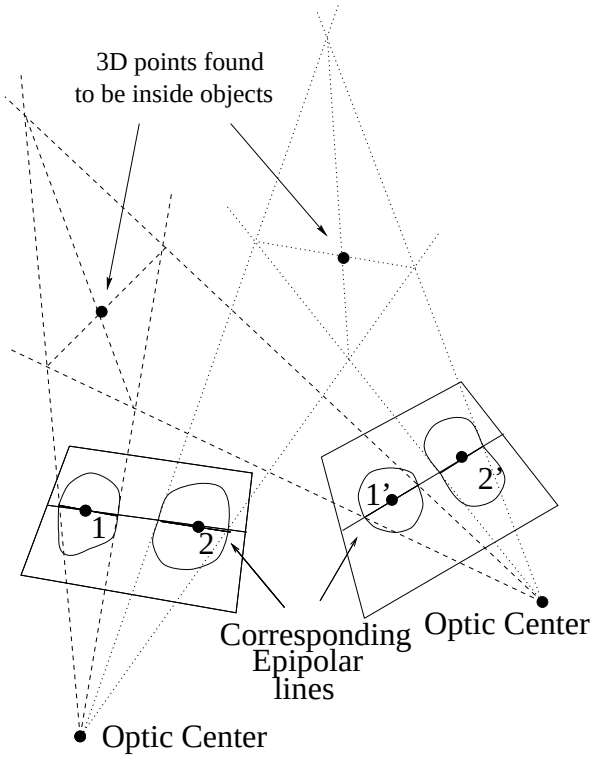

Figure 4.6: The point of intersection of the quadrilateral formed by back-projecting the endpoints of the matched segments yields a 3D point lying inside an object. The matching segments are 1 and 1', and 2 and 2' respectively.

which groups together pixels having similar color characteristics. This segmentation creates additional regions in the image and these regions are also matched across cameras as described in the next section. This method works not only for detecting "new" people entering the scene but also for bootstrapping the algorithm since in the beginning, all people are detected as "new".

4.5 Region-Based Stereo

After segmentation, we analyze epipolar lines across pairs of views. We first construct epipolar lines such that there are a fixed number of lines for two views. For instance, if the epipole lies inside a view, the lines are taken at an angular separation of $180/n$ where n is the number of lines desired. If the epipole lies outside the view, then the lines are constructed so that they have a fixed angular separation and cover the whole image. Along an epipolar line, we divide the line into "segments" belonging to different regions. If a line intersects a region twice, the two segments thus formed are merged to form one segment consisting of the two extreme end points. These segments from

one camera view are matched to the segments in the other view. Segments belonging to the same person in different views (as determined by the classification algorithm) are matched to each other. Regions corresponding to unclassified pixels are matched to each other based on color characteristics. Even if one segment matches to more than one segment in the other view, we do not select among these matches but consider all of the matched pairs as possible matches.

For each matched pair of segments, we project the end-points of the segments and form a quadrilateral in the plane of the corresponding epipolar lines. The point of intersection of the diagonals of this quadrilateral is taken to be belonging to the object (see Figure 4.6).

The motivation for taking the point of intersection of the diagonals of the quadrilateral is that, for a convex object, this is the only point that can be guaranteed to lie inside the object (see proof in Appendix). This is assuming that the complete object is visible and segmented completely as one region in each view. For any other 3D point in the plane of the epipolar lines, it is possible to construct a case in which this point will lie outside the object.

4.6 Producing Likelihood Estimates on the Ground Plane

Having obtained 3D points belonging to people, we want to detect and track people in a robust manner rejecting outliers. Assuming that people are standing upright or are otherwise extended primarily in the vertical direction, one natural way to do that would be to do the estimation on the ground plane after projecting the 3D points onto it. It is also possible to do clustering in 3D (as done by [51]) and this would be the method of choice for many applications. However, for our application, estimation on the ground plane is better since we are dealing with only walking people.

We define a "likelihood" measure which estimates whether a particular location on the ground plane is occupied by an object. Likelihood maps are developed for each camera pair and are then combined in a robust manner using the occlusion information available.

4.6.1 Likelihood from a Single Camera Pair

We develop likelihood maps on the ground plane using the ground plane points found using region matching. For each of the 2D ground points, we add a Gaussian kernel to the likelihood map. The weight and standard deviation of the kernel is based on the minimum width of the segments that matched to give rise to that point, and the camera instantaneous fields of view (IFOV). This helps us give higher weight to points originating from longer segments than from smaller ones. The likelihood from all these gaussian kernels is then integrated to obtain a likelihood map in the 2D space of the ground plane. This is done for each pair of cameras for which the segmentation and

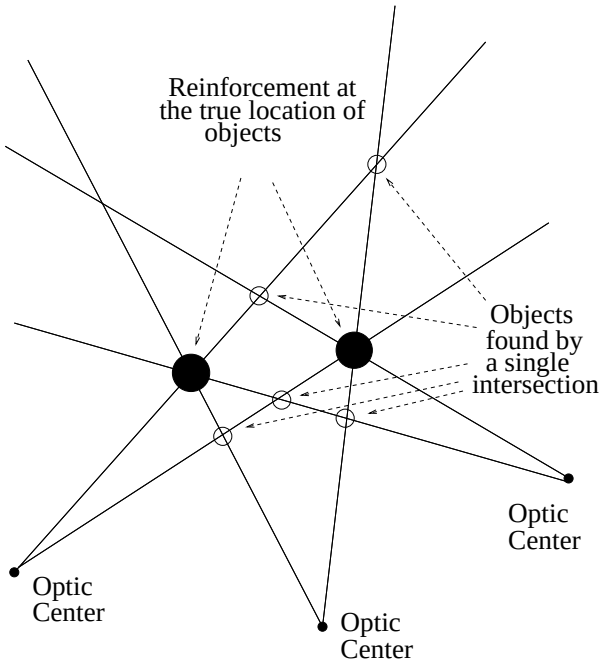

Figure 4.7: If segment matching fails to distinguish between two objects, the matches would reinforce each other only at the true location of the objects, and the false matches would get eliminated by the weighting scheme.

matching is performed. Thus, the likelihood associated with any point x on the 2D plane is given by

$$Lk(\mathbf{x}) = \Sigma_i \frac{w_i}{\sqrt{2\pi}\sigma_i} \exp\left(\frac{-(\mathbf{x}_i - \mathbf{x})^2}{2\sigma_i^2}\right)$$

where $\mathbf{x_i} = (x_i, y_i)$ and σ_i are the 2D position and standard deviation of the i-th Gaussian kernel and w_i is the weight assigned to this kernel.

4.6.2 Combining Results from Many Camera Pairs Using Occlusion Analysis

Given the likelihood maps from matching across pairs of cameras, we describe a method for combining likelihood maps that makes use of occlusion information available from the approximate position of the people. The simplest method would average the likelihood estimates obtained from different pairs. This method does yield good results because the likelihood values at the true locations of the objects reinforces each other, whereas the values at false locations are scattered about and occur at different 2D locations for different pairs of cameras. This can be seen in the example illustrated in Figure 4.7 where false matches are not reinforced but good ones are.

Although simple averaging of likelihood values yields good results, we can improve likelihood estimates significantly by determining (probabilistically) whether a particular location is visible or occluded from a camera and weighing the likelihood values accordingly.

For each of the cameras, we form a probability map that gives us the probability that a particular location **x** is visible from the camera. First of all, the camera center is projected onto the ground plane. Then, for each point **x** on the ground plane, we calculate the perpendicular distance w_j of each person j from the line joining the camera center and the point **x**. Then, defining "presence" probabilities $L_j()$ similar to section 4.2, but taking only the width as parameter (by averaging over the height parameter), we find the probability that the point **x** is visible from the camera c as

$$P_c(\mathbf{x}) = \prod_{j \textit{ occludes } \mathbf{x}} (1 - L_j(w_j)) \tag{4.5}$$

where j occludes **x** if its distance from the camera is less than **x**. Now, for a particular camera pair $(c1, c2)$, the weight for the ground point $\mathbf{x}$ is calculated as

$$w_{(c1,c2)}(\mathbf{x}) = P_{c1}(\mathbf{x}) P_{c2}(\mathbf{x}) \tag{4.6}$$

The weight is essentially the probability that **x** is visible from both the cameras. The weighted likelihood value is then calculated as

$$Lk(\mathbf{x}) = \frac{\sum_{(c1,c2)} w_{(c1,c2)}(\mathbf{x}) Lk_{(c1,c2)}(\mathbf{x})}{\sum_{(c1,c2)} w_{(c1,c2)}(\mathbf{x})} \tag{4.7}$$

This definition helps us to dynamically weigh the different likelihood values such that the values with the highest confidence level (least occlusion) are weighted the most. Note that the normalization constant is different for each ground plane point and changes over time.

4.7 Tracking on the Ground Plane

After obtaining the combined likelihood map, we identify objects by examining likelihood clusters and identifying regions where the sum of likelihoods exceeds a given threshold. The centroids of such likelihood "blobs" are obtained simply using

$$\mathbf{x}_{centroid} = \frac{\sum_{\mathbf{x}, \mathbf{x} \in region} \mathbf{x} * Lk(\mathbf{x})}{\sum_{\mathbf{x}} Lk(\mathbf{x})} \tag{4.8}$$

where $Lk(\mathbf{x})$ is the likelihood at point $\mathbf{x}$.

These centroids of the object blobs are tracked over time using a different Kalman filter for each of the blobs. The state vector of the filter consists of position and velocity of the object and prediction is made using the assumption of constant velocity during

Figure 4.8: The results of detection and tracking as seen in four of the images shown in Fig. 4.1.

the time step. The current observation of the object location is obtained by finding the centroid over a circular region around the current predicted object location.

Some simple heuristics are used to eliminate false detections, maintain robust person identities and to re-identify lost people who might be seen again. We keep track of the amount of time that a particular person has been tracked. If a person is not found near his predicted position by the ground plane position finding algorithm, then he is eliminated or retained depending on the amount of time he has been tracked continuously. Also, if two objects are too close to each other, then the one tracked for lesser duration is removed unless both of them have been tracked for some predefined duration. If a person that would normally have been removed is retained because of time considerations, then he is predicted to be at the position predicted by the Kalman filter for some time and this position and his models are used in the segmentation algorithm. However, after some time of non-detection, the person is removed from the list of tracked people and his model is stored in the "lost" people list. When a new person is detected, his model (mainly the color profile) is matched to the models for lost people and if it matches one of them, then he is identified as this "lost" person. We calculate the dissimilarity between two probability distribution functions P_1 and P_2 using the L_1 distance, i.e.

$$diss_{L_1}(P_1, P_2) = \int \mid P_1(\mathbf{x}) - P_2(\mathbf{x}) \mid d\mathbf{x} \tag{4.9}$$

The dissimilarity from the models at different heights is then simply summed up to get

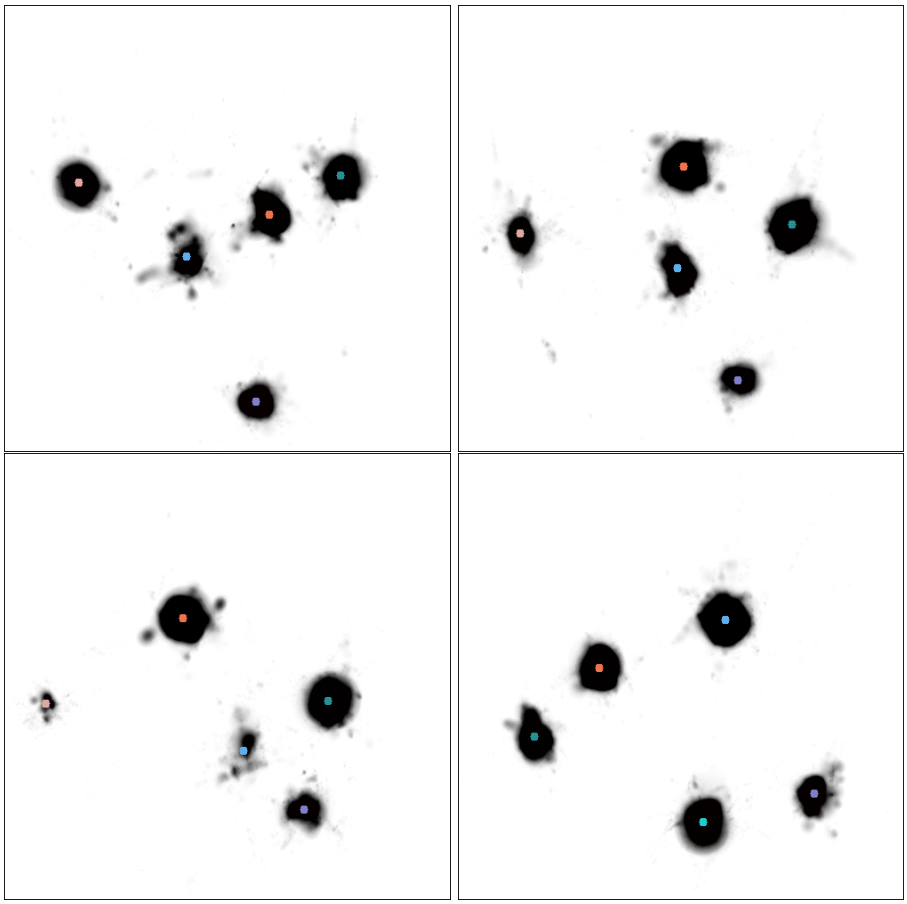

Figure 4.9: The first image is the likelihood map obtained for the image set shown in Figure 4.1 by applying the occlusion-analysis weighting scheme. The rest of the images are maps obtained for the sequence at other time steps. The dots show the position state variable of the Kalman filter tracking the person. Visit www.umiacs.umd.edu/~anurag for the full sequence and some additional results.

the dissimilarity measure between the color profiles of two people.

4.8 Updating Models of People

Observed images and information about the current position of the people are used to update models of people and create ones for the "new" people detected. In order to develop accurate models, we want to identify pixels for which we are very sure that they belong to a particular person. To do so, for each pixel, we calculate the "presence" probabilities L_j for each person as described earlier. We use the observed probability method only when the number of observed pixels for a particular width w is above a certain threshold. Then we determine if L_j is above a certain threshold for a particular person and below another (lower) threshold for all others. This helps us in ensuring that the pixel is viewing the particular person only and nothing else (except the background). In order to determine if the pixel belongs to the background or not, we use the background model to determine the probability that the pixel color originates from the background. If this probability is below a certain threshold, then we determine that the pixel belongs to the person; else it belongs to the background. If it belongs to the person, it is added as a kernel to the color model of the person at that height. We update the "presence" probability L_j for the person by incrementing the count for the total number of observations at height h and width w for the person and incrementing the count for positive matches only if this pixel is determined to belong to the person (according to the above mentioned method). The "presence" probability at that height and width is then simply the second count divided by the first.

For a new person, since we do not know the true height of the person, we develop models up to a maximum possible height. The true height of the person is estimated by observation over time and is used to delete the slices above that height.

4.9 Implementation and Experiments

Image sequences are captured using up to 16 color CCD cameras (Kodak ES-310C). These cameras, which are attached to "Acquisition" PCs via frame grabbers, are capable of being externally triggered for synchronization purposes. Cameras are located at positions surrounding the lab so that they see the objects from different viewpoints. Eight cameras are located in approximately a circle at a lower (in height) level and the other eight cameras are located directly on top of them at a higher level. Therefore, the angle between adjacent cameras at the same height is approximately 45 degrees. All of the cameras are calibrated using a global coordinate system and the ground plane is also determined. Frame synchronization across cameras is achieved using a TTL-level signal generated by a Data Translation DT340 card attached to a controller PC, and transmitted via coaxial cables to all the cameras. For video acquisition, the synchronization signal is

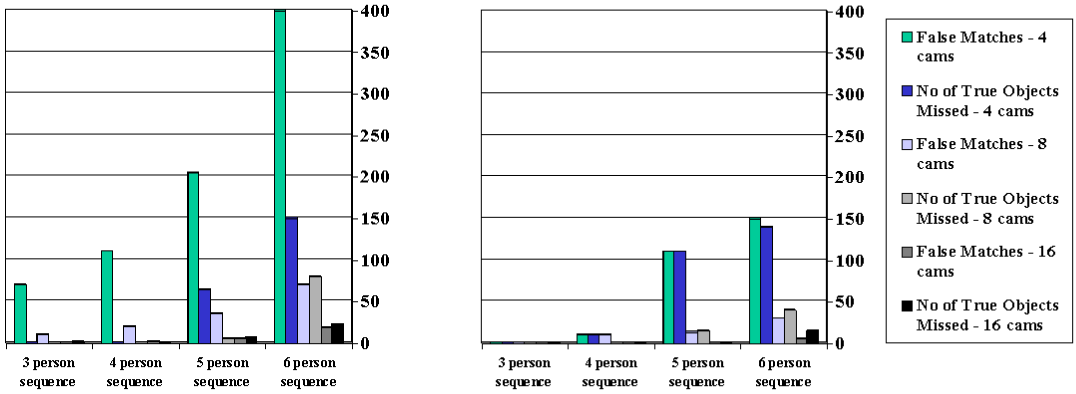

Figure 4.10: Cumulative errors for four sequences of 200 time steps each by averaging likelihoods and using (a) no occlusion analysis, and (b) using occlusion analysis.

used to simultaneously start all cameras. No timecode per frame is required. For details see [11].

In the distributed version of the algorithm where we use a Pentium II Xeon 450MHz PC for each of the cameras, the system currently takes about 2 seconds per iteration of the ground plane position finding loop. On the average, we need about 2 - 3 iterations per time step, so the running time of the algorithm is currently about 5 seconds per time step. We believe that by code optimizations and faster processors, we will be able to run the algorithm in real time.

In order to evaluate our algorithm, we describe experiments on four sequences containing 3, 4, 5 and 6 people respectively. Each sequence consisted of 200 frames taken at the rate of 10 frames/second and people were constrained to move in a region approximately 3.5mX3.5m in size. Matching was done for only adjacent pairs of cameras (n pairs)and not for all of the C_2^n pairs possible. This helps us control the time complexity of the algorithm, but reduces the quality of the results obtained.

For each of the sequences, we calculated the number of false objects found and the number of true objects missed by the algorithm. We calculated these metrics using 4, 8 and 16 cameras in order to study the effect of varying the number of cameras, thus enabling us to determine the minimum number of cameras required to properly identify and track a certain number of objects. For the experiment with four cameras, cameras were placed at an angular separation of 90 degrees, for the eight camera experiment, only the top row of cameras were used (so the angular separation between adjacent cameras is about 45 degrees) and for the sixteen camera experiment, all cameras were used (again the angular separation between adjacent cameras is 45 degrees). The angular separation for us is at least 45 and up to 90 degrees, which is much more than that used

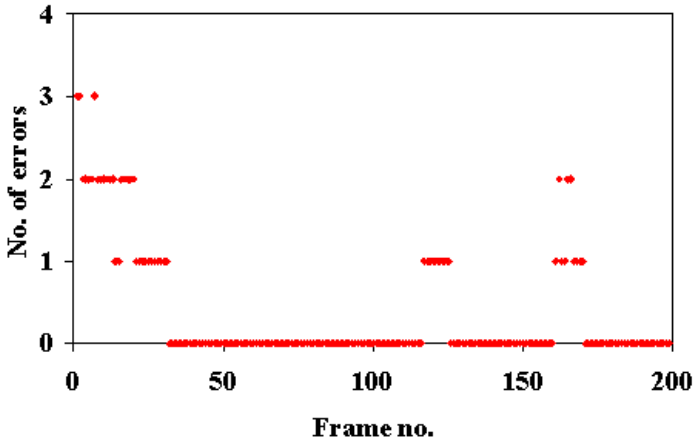

Figure 4.11: Total errors as a function of time for the sequence with 5 people using 8 cameras. Note how the errors decrease with time as the models become more robust. Errors after the initial period occur mainly because of people coming too close to each other.

by short baseline stereo cameras where the separation is of the order of 5 to 15 degrees. The cumulative errors over the 200 frames are shown in Figure 4.10(b). Also shown in Figure 4.10(a) are the error metrics obtained when the likelihood values obtained from different cameras are weighted equally and occlusion analysis is not used. This helps us observe the improvement obtained by using the occlusion analysis scheme.

4.9.1 Behavior during Initialization

In the beginning, we have no information as to where the people are, or what their models are, so the algorithm is trying to find people based solely on color matching across views. The results from an algorithm that does detection and tracking based solely on color matching can be found in our paper [64]. The results are decidedly of lower quality but, in a sense, are used to initialize the algorithm presented in this paper. Once an object has been detected, however, he can be tracked easily since we are able to build the models for him. Due to undetected people in the scene, the models developed for the detected people are sometimes not so accurate because pixels belonging to the undetected people are assigned to detected ones and are included even in their models. Apart from inaccurate detection and tracking results, this also results in an increase in the number of iterations required. This is a potential problem, especially when the number of people is very large and occlusions are very heavy. However, for cases of moderate occlusions, we have found that this stage is very short and the undetected people are also found after some time, allowing the algorithm to build more accurate models and

get very accurate results.

4.10 Summary and Conclusions

In this chapter, we have presented a method for detecting and tracking densely located multiple objects using multiple synchronized cameras located far away from each other. It is fully automatic and does not require any manual input or initializations. It is able to handle occlusions and partial occlusions caused by the dense location of these objects and hence can be useful in many practical surveillance applications.

Chapter 5

Human Body Pose Estimation Using Silhouette Shape Analysis

5.1 Introduction

Determining the pose of humans is an important problem in vision and has many applications. In this chapter, we target multi-camera surveillance applications where one wants to recognize the activities of people in a scene in the presence of occlusions and partial occlusions. One cannot assume that a person is visible in isolation or in full in either one or all of the views. Nor can one assume that we have a model of the person, or that the initial body pose is known. Such a system should also be reasonably fast. However, very accurate body pose values are typically not required, and an answer close to the actual body pose might be adequate. We describe an algorithm that can form the basis of such a surveillance system.

Our system estimates the 3D pose of a human body from multiple views. We make use of recent work in decomposition of a silhouette into 2D parts. These 2D part primitives are matched across views to build primitives in 3D which are then assembled to form a human figure. In order to search for the best assembly, we use a likelihood function that integrates information available from multiple views about body part locations. Greedy search strategies are employed so as to find the best assembly fast.

Our work is most closely related to the work of Kakadiaris and Metaxas [43] who try to acquire 3D body part information from silhouettes extracted in orthogonal views. They employ a deformable human model so that any size of the human can be recognized. The distinguishing feature of our work is that it is able to work in a crowded scene so that in all of the views, the person might be fully or partially occluded. This is accomplished by explicitly modeling occlusion and developing prior models for person shapes from the scene. This helps us to decouple the problems of pose estimation for multiple people so that the degrees of freedom of the problem are decreased substantially.

The chapter is organized as follows. Section 5.2 describes the method of extraction of multiple silhouettes in a crowded scene using the method described in the last chapter. Section 5.3 describes shape analysis of silhouettes and matching parts across views

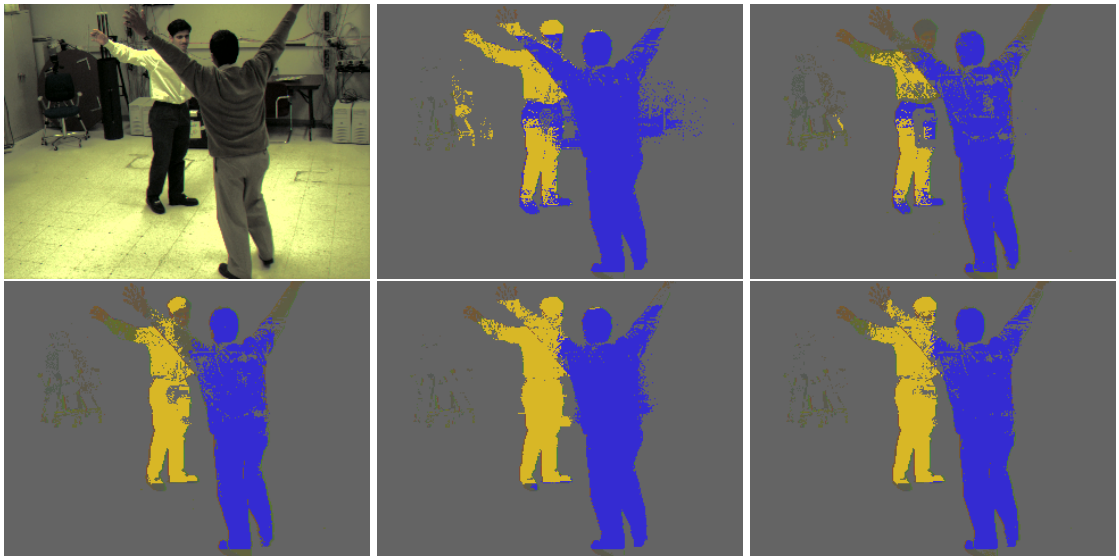

Figure 5.1: Multiple Segmentations Obtained for the image shown in the first image

to obtain 3D part primitives. Section 5.4 describes the likelihood function used for assembly evaluation. Section 5.5 describes the algorithm used to find the best assembly. We conclude with some preliminary results in section 5.6.

5.2 Obtaining Multiple Segmentations

We use the method described in the last chapter for obtaining segmentations of people in a crowded scene. There are several parameters in the segmentation algorithm. Accurate extraction of different parts of the person requires different parameters. Therefore, it is essential to vary the parameters so as to obtain multiple segmentations. The parameters that we vary are
(1) the relative weight given to the background model,
(2) the relative weight given to different foreground objects so that different objects are highlighted, and
(3) the threshold for determining whether a pixel is unclassified pixels.
The silhouettes thus obtained are segmented using the method described in the next section.

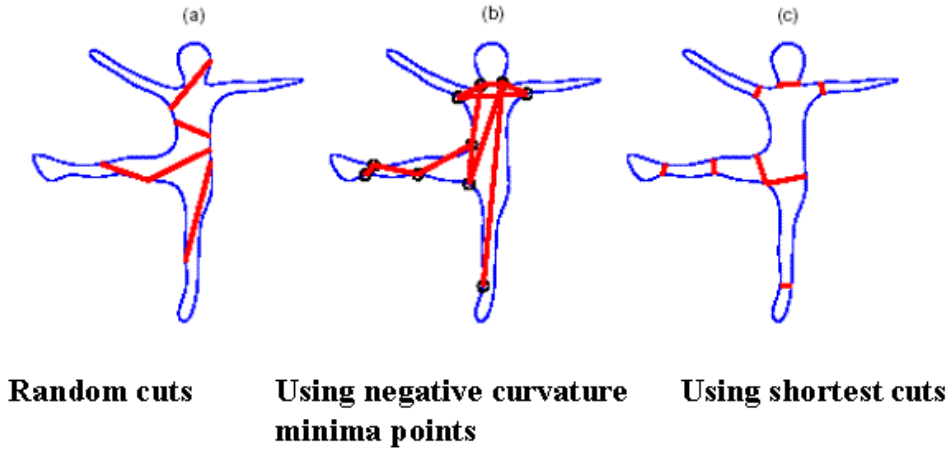

Figure 5.2: Silhouette Decomposition

5.3 Computing Body-part Primitives

5.3.1 2D Silhouette Shape Analysis

In order to recover the pose of a person, we break the silhouette of the person into parts. According to human intuition about parts, a segmentation into parts occurs at *negative minima of curvature* so that the decomposed parts are convex regions. Singh *et al.* noted that when boundary points can be joined in more than one way to decompose a silhouette, human vision prefers the partitioning scheme which uses the shortest cuts (A cut is the boundary between a part and the rest of the silhouette). They further restrict a cut to cross a symmetry axis in order to avoid short but undesirable cuts. However, most symmetry axes are very sensitive to noise and are expensive to compute. In contrast, we use the constraint on the salience of a part to avoid short but undesirable cuts. According to Hoffman and Singh's [32] study there are three factors that affect the salience of a part: the size of the part relative to the whole object, the degree to which the part protrudes, and the strength of its boundaries. Among these three factors, the computation of a parts protrusion (the ratio of the perimeter of the part (excluding the cut) to the length of the cut) is more efficient and robust to noise and partial occlusion of the object. Thus, we employ the protrusion of a part to evaluate its salience; the salience of a part increases as its protrusion increases.

In summary, we combine the short-cut rule and the salience requirement to constrain the other end of a cut. For example in Figure 5.3.1, let S be a silhouette, C be the boundary of S, P be a point on C with negative minima of curvature, and P_m be a point on C so that P and P_m divide the boundary C into two curves C_l, C_r of equal arc length.

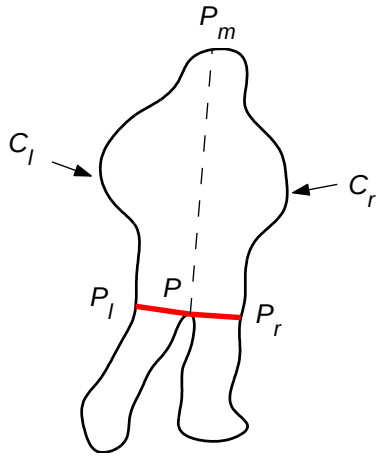

Figure 5.3: Computing the cuts passing through point P

Then two cuts are formed passing through point P: $\overline{PP_l}$, $\overline{PP_r}$ such that points P_l and P_r lies on C_l and C_r, respectively. The ends P_l and P_r of the two cuts are located as follows:

$$\begin{aligned} P_l &= arg\min_{P'} |\overline{PP'}| \\ \text{s.t.} \quad & \frac{|\overset{\frown}{PP'}|}{|\overline{PP'}|} > T_p, P' \in C_l, \overline{PP'} \in S \end{aligned} \tag{5.1}$$

$$\begin{aligned} P_r &= arg\min_{P'} |\overline{PP'}| \\ \text{s.t.} \quad & \frac{|\overset{\frown}{PP'}|}{|\overline{PP'}|} > T_p, P' \in C_r, \overline{PP'} \in S \end{aligned} \tag{5.2}$$

where $\overset{\frown}{PP'}$ is the smaller part of boundary C between P and P', $|\overset{\frown}{PP'}|$ is the arc length of $\overset{\frown}{PP'}$, and $\frac{|\overset{\frown}{PP'}|}{|\overline{PP'}|}$ is the salience of the part bounded by curve $\overset{\frown}{PP_l}$ and cut $\overline{PP_l}$.

Eq. (5.1) means that point P_l is located so that the cut $\overline{PP_l}$ is the shortest one among all cuts sharing the same end P, lying within the silhouette with the other end lying on contour C_l, and resulting in a significant part whose salience is above a threshold T_p. The other point P_r is located in the same way using Eq. (5.2).

Since negative minima of curvature are obtained by local computation, their computation is not robust in real digital images. We take several computationally efficient strategies to reduce the effects of noise. First, a B-spline approximation is used to moderately smooth the boundary of a silhouette, since B-spline representation is stable and

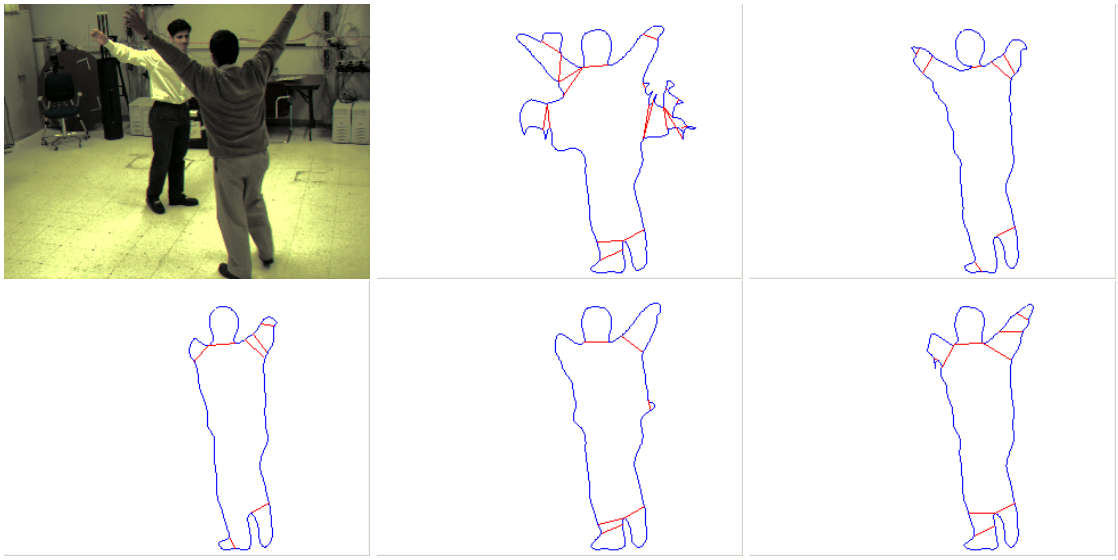

Figure 5.4: Multiple Body parts obtained using the segmentations shown in Fig. 5.1

easy to manipulate locally without affecting the rest part of the silhouette. Second, the negative minima of curvature with small magnitude of curvature are removed to avoid parts due to noise or small local deformations. However, curvature is not scale invariant (e.g. its value doubles if the silhouette shrinks by half). One way to transform curvature into a scale-invariant quantity is to first find the chord joining the two closest inflections which bound the point, then multiply the curvature at the point by the length of this chord. The resulting normalized curvature does not change with scale — if the silhouette shrinks to half size, the curvature doubles but the chord halves, so their product is constant.

This analysis yields 2D body parts for a person in a single view. The torso is not found directly by this method as the body part segmentations can only find protruded parts reliably. Since these protruded parts can overlap, there are a large number of torsos that can be formed from the remaining part of the silhouette. Therefore, we do not attempt to find the torsos directly and simply infer it from the other body parts.

Zhao [114] has used a similar method to develop a system for body part identification from a single view. However, body part identification from a single view is very difficult and labelings are often incorrect, especially in the case of partial-occlusions and self-occlusions where some body parts are not visible. The problem is also under-constrained since depth information is not available. Another difficulty is that their system requires extraction of good silhouettes which are not easy to obtain in a dense scene. As opposed to Zhao's work, we use multiple cameras and identify body pose in 3D using a global analysis.

5.3.2 Computing Body-part Primitives in 3D

2D part primitives obtained using Silhouette Analysis are used to obtain part primitives in 3D. First, parts that are relatively close to each other are combined with each other. Second, the decomposed parts are matched across views using epipolar geometry to yield 3D body parts. The two endpoints of a part in one view are matched to the corresponding endpoints in the other view. The matching is based on simply lying on the corresponding epipolar line. An additional constraint that can be used is the color profile of the body parts. The disadvantage is that if the viewpoints are substantially different, the color profiles can vary significantly. Also, the color profiles for different body parts can be very similar (for e.g. the two legs can have very similar color profiles.)

Once matching is done, a certain number of body parts are selected based on their matching score and their end points are projected in space to yield 3D body parts.

5.4 Assembly Evaluation using the Observation Likelihood

Labelings are assigned to these 3D parts by building an assembly that has the maximum likelihood according to an appropriate likelihood function. From the set of 3D body parts, we form sets of possible heads, hands and legs based on size constraints. Additional knowledge, if available, can be used. Such information might consist of the knowledge that the legs are close to the floor or that the person is standing (constraint on head and hand positions). Then, the problem reduces to finding a head, two hands (or a single or no hands, if not found) and two legs (or 0 or 1 legs), such that the assembly has the highest likelihood. The likelihood function we use is described in the next section.

5.4.1 Observation Likelihood

In order to evaluate a particular assembly $\mathcal{A}$, we determine the observation likelihood $Pr(I_1, I_2, ..., I_n/\mathcal{A})$, which is the likelihood of observing images $I_1, I_2, ..., I_n$ given the particular assembly $\mathcal{A}$. Assuming that assemblies have equal priors, the assembly having the highest likelihood is also the assembly with the highest posterior. Since we do not know the body pose of other people in the scene, the observation likelihood cannot be determined unless the problems of body pose determination of different people are coupled with one another. This leads to an exponential increase in the complexity of the algorithm.

We can decouple the problem, however, if we make some simplifying assumptions. Specifically, we can use the method developed in M_2Tracker[65] to determine priors using presence probabilities. Then, the general formula for the observation probability

at a particular pixel x can be written as:

$$p(I(x)) = \sum_j P_{prior}(j) Pr(I(x)/j) \tag{5.3}$$

where the summation is done over all persons j and the background, and $I(x)$ is the observation at pixel x. If the location of the assembly is given, the function $L_j(x)$ (Presence Probability defined in section 4.3.2) for the person m under consideration changes from a probabilistic to a fixed function so that:

$$L_m(x) = \begin{cases} 1 & \textit{if assembly } \mathcal{A} \textit{ projects to pixel x} \\ 0 & \textit{if } \mathcal{A} \textit{ does not project to pixel x} \end{cases} \tag{5.4}$$

Using this definition, one can re-determine the priors for all people using equation (4.4) and calculate the observation probability using equation (5.3). This would be the conditional probability $Pr(I(x)/\mathcal{A})$:

$$p(I(x)/\mathcal{A}) = \sum_j P_{prior}(j/\mathcal{A}) Pr(I(x)/j) \tag{5.5}$$

where

$$\begin{aligned} P_{prior}(j/\mathcal{A}) &= L_j(h_j, w_j) \prod_{\textit{k occludes j}} (1 - L_k(h_k, w_k)) \\ P_{prior}(bckgrnd/\mathcal{A}) &= \prod_{\textit{all j}} (1 - L_j(h_j, w_j)) \end{aligned} \tag{5.6}$$

such that L_m is given by Equation 5.4 and for all other persons, we use the presence probability function developed in the previous chapter.

Assuming that observations at different pixels are independent, the overall observation probability is then simply the product of the observation probabilities at each pixel in each view.

$$Pr(I_1, I_2, ..., I_n/\mathcal{A}) = \prod_{i=1}^{n} \prod_{\text{all pixels x}} Pr(I(x)/\mathcal{A}) \tag{5.7}$$

5.4.2 Projection of the Assembly

In order to determine the projection of the assembly on an image, we model the hands and legs as cylinders with approximate widths and the head as a sphere and determine their projections onto a view (Figure 5.5). The torso is built by filling in the polygon formed by taking the joint locations of the (five) parts as the vertices. More accurate projection can be formed by building a 3D structure based on the joint locations and finding its projection onto the views. That will, however, add to the running time of the algorithm.

Figure 5.5: Determining the Projection of an Assembly

5.4.3 Refining the Likelihood Function

M_2Tracker determines the probability $Pr(I(x)/j)$ used in equation (5.3) using color models at different height slices. This puts only occupancy constraints on the likelihood. However, apart from the hypothesis that the given assembly projects to a particular pixel, we also have information as to which part of the assembly projects to the pixel. Using this information, we can improve results by including in the likelihood function information available from the views about possible body part locations. For example, we might be able to find the head using a face detector. If we have a skin detector, we might want to exclude the torso from the set of body parts that can give rise to it. In the present work, we include an additional term in the likelihood $Pr(I(x)/j)$.

First, we determine the probability that a particular body part has a particular aspect ratio $Pr_{bp}(ar)$. This probability is modeled as a 1D Gaussian, its mean and standard deviation learnt using training data. Now, we consider body parts detected from the silhouettes extracted for the person and find their aspect ratios. Finding the value of the function $Pr_{bp}(ar)$, we assign this value to all pixels belonging to the part in the silhouette. Since the torso is not observed directly, we cannot determine this probability for pixels belonging to it and hence they are assigned a constant value. Since we have multiple silhouettes and hence multiple probability estimates for the aspect ratio at a given pixel, we average them to yield a single result. For pixels lying outside any silhouette, the probability is zero. This will yield the function $Pr(ar/bp)$ for each pixel x and each body part bp. During evaluation of an assembly, we can compute the value of this function since we know the projections of the body parts onto the image. This probability value can be multiplied with the color likelihood to yield the likelihood function $Pr(I(x)/j)$ used in equation (5.3).

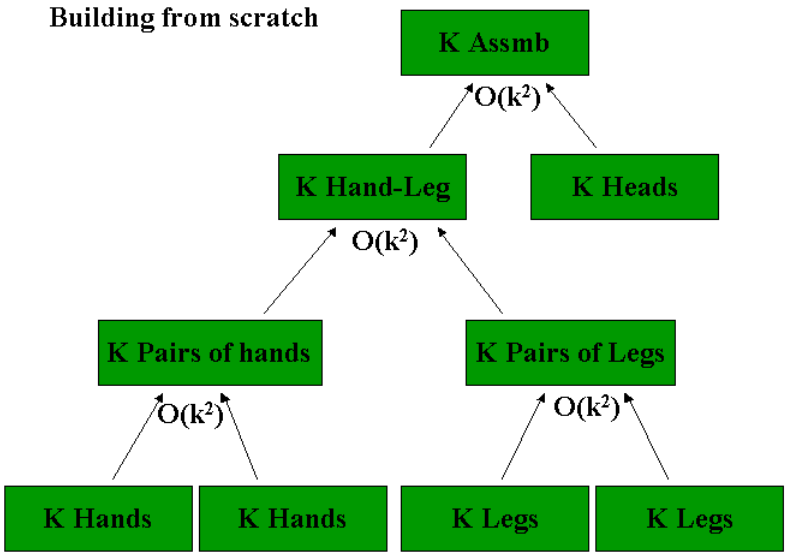

Figure 5.6: Schematic for the Initialization procedure

5.5 Searching for the Optimal Assembly

We believe that the best assembly can only be found by an exhaustive search in $O(n^5)$ time (where $n \sim O(10)$) is the number of possible primitives for each part). However, in practice, we have found that the same result can be obtained in $O(n)$ time if we have a good initial estimate of the body part positions , and in $O(n^2)$ time during the initialization phase. We first describe the incremental scheme.

5.5.1 *Incremental Algorithm*

If we have a sufficiently good estimate of the current body part locations, we use a greedy approach. The idea is to first try to replace each part with candidate parts. If the assembly with the original part has a higher likelihood than the ones with any of the new primitives, we keep the original one. This is repeated for different parts. We have found that, apart from being very fast, this method yields the best results (better than initialization method) since it is often the case that some body parts have no good candidates at a particular time step, in which case we can keep the old estimate.

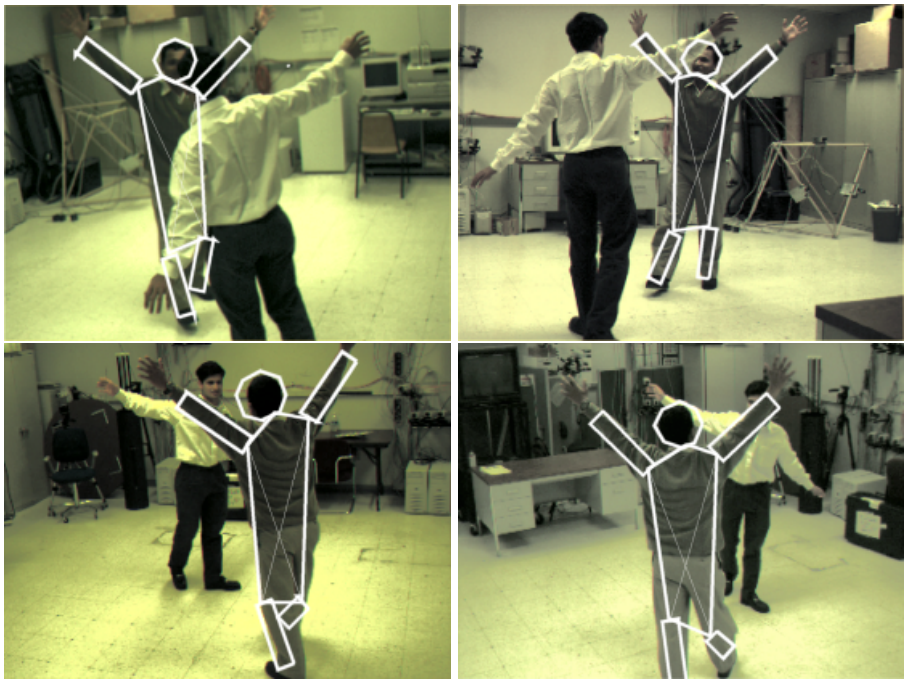

Figure 5.7: Results of the algorithm for a person at a particular time instant from multiple perspectives. Note how the person's body parts are correctly detected even though he is partially occluded from some views.

5.5.2 *Initialization*

In order to find an initial solution, or reinitialize the method if the incremental method fails, we use the following approach. First, we try to find good leg pairs. We find K best pairs (in $O(K^2)$ time) based on the likelihood function by building an assembly of just the two legs. Similarly, we find K best pairs of hands. Next, we find K best assemblies consisting of two hands and two legs using the hand and leg pairs found earlier (Figure 5.6). For this step, we construct the torso using the four joint locations. Finally the head is added and the best assembly is found. Although this method does not find the optimal assembly, we have found that it is extremely effective in practice and with the right choice of K, yields results very close to an exhaustive search.

If computational cost is available, we can find the result using both algorithms, taking the assembly with the higher likelihood as the answer.

5.6 Results

We have obtained promising results for the algorithm. We tested our algorithm on a 5-perspective sequence with two people partially occluding each other in several views. We were able to correctly identify the body parts of the people when they were extended

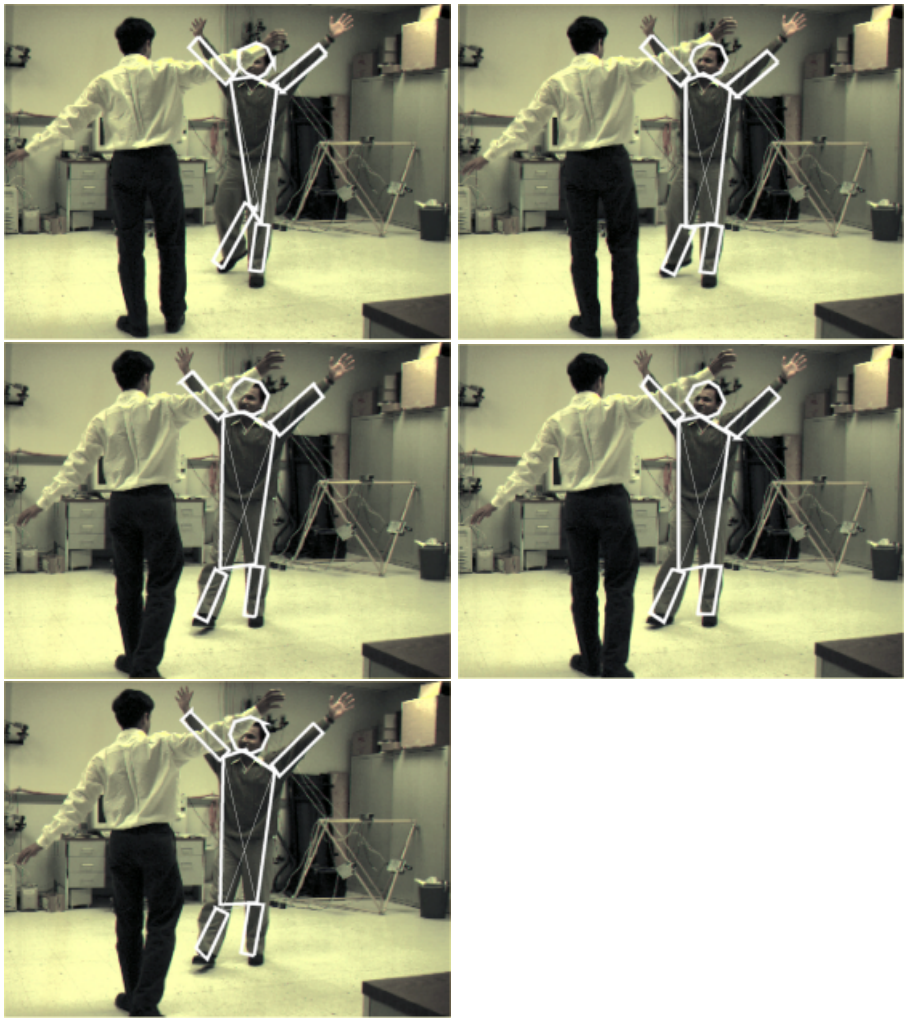

Figure 5.8: Results for five frames of the sequence. Note how the motion of the leg is correctly captured.

from the body. When the parts were close to the body, the algorithm labeled the part as missing and correctly identified the other parts. Figure 5.7 shows the result obtained at a particular time instant for the sequence. Figure 5.8 shows the results over time from a particular view. No initialization was done, nor any exact 3D model of the person specified. The algorithm took about 10s/frame on a Dual 933MHz Pentium III processor where most of the time was spent in evaluating different assemblies.

5.7 Summary and Conclusions

We have presented an algorithm for body pose estimation that does not require any initializations or models to be specified up front and is able to work in a crowded scene

so that occlusions - both full and partial - are present. These features make it especially useful for many surveillance applications. Future work in this area might involve investigation of others cues for body parts in an image (other than the silhouettes) like edge maps and texture regions, which might help us to reduce the number of cameras required to obtain a certain quality of results.

Chapter 6

Conclusions

In this thesis, we have addressed the problem of surveillance in situations where the density of objects is high.

Chapter 2 presented a stochastic framework for evaluation of the visibility probability given a certain configuration of sensors in a scene. Apart from yielding important performance characteristics of multi-camera systems, it can be used to evaluate different camera configurations, and to determine the one that is the best given the visibility requirements.

Chapter 3 described an algorithm for wide area surveillance and monitoring using a single camera. The method uses a mixture of Gaussian model to represent pixels in the scene. This background model provides a means of registering video frames that is robust in the presence of moving objects in the scene. The models can also be used to detect moving objects in a moving camera scene, and to create panoramic views and video sequences that do not contain any moving objects.

When multiple cameras are available, one can coordinate between them so that detection is done using all the cameras in the scene. Chapter 4 described a system for detecting and tracking densely located multiple objects from multiple synchronized cameras. Widely separated cameras are used as they provide the highest visibility.

Chapter 5 used the segmentation method of Chapter 4 to develop a system for body pose estimation. Techniques for silhouette decomposition are used to identify possible body parts in one view. These are then combined across views to build a 3D assembly. The assembly having the highest likelihood is selected.

The distinguishing feature of our methods is that they do not require any initializations or models to be specified up front and are able to work in a crowded scene so that occlusions - both full and partial - are present. Due to these features, they can form the basis for a fully automatic, real-time surveillance system for use in real-world scenarios that have a high density of objects.

Appendix A

Region-Based Stereo

In this section, we prove that, in the case of a convex object O, the point of intersection of the diagonals of the quadrilateral formed by backprojecting the end-points of corresponding segments of that convex object is guaranteed to lie inside the object; and that no other point can be guaranteed thus. (For a non-convex object, this point lies inside the convex hull of the object.)

We prove this with the help of an illustration showing the plane corresponding to the epipolar lines. (see Figure A.1). Let a and b be the rays back-projected from the left and right ends of the segment as seen from the first camera. Let c and d be the corresponding rays from the second camera. Now, let P_1, P_2, P_3 and P_4 be the points of intersection of a, b, c and d as shown in the diagram. Let P be the point of intersection of the diagonals of $P_1P_2P_3P_4$. Since camera 1 sees some point on line a that belongs to O, and O is guaranteed to lie between rays c and d, we can conclude that there exists a point on the line segment P_1P_2 that lies on the boundary of O. Let this point be called A. Similarly, we can conclude the existence of points from O on line segments P_2P_3, P_3P_4 and P_4P_1. Let these points be called B, C and D respectively. Since the object is convex, we can now conclude that all points lying inside the quadrilateral ABCD also lie within O.

Now, consider the line segment AB. Omitting details, we can easily prove that the point P lies on the same side of AB as the quadrilateral $ABCD$. Similarly, we can prove that P lies on the same side of lines BC, CD and DA as the quadrilateral $ABCD$. But this means that P lies inside $ABCD$, hence inside O.

For any point P' other than P, it is possible to place A, B, C and D such that the point P' lies outside the quadrilateral ABCD. For, it must lie on one side of at least one of the lines P_1P_3 and P_2P_4. If it lies on the side of P_1P_3 towards P_2, then we can place AB such that P' lies on the side of AB towards P_2, thus implying that it lies outside ABCD.

Therefore, the point P is the only point guaranteed to lie inside O.

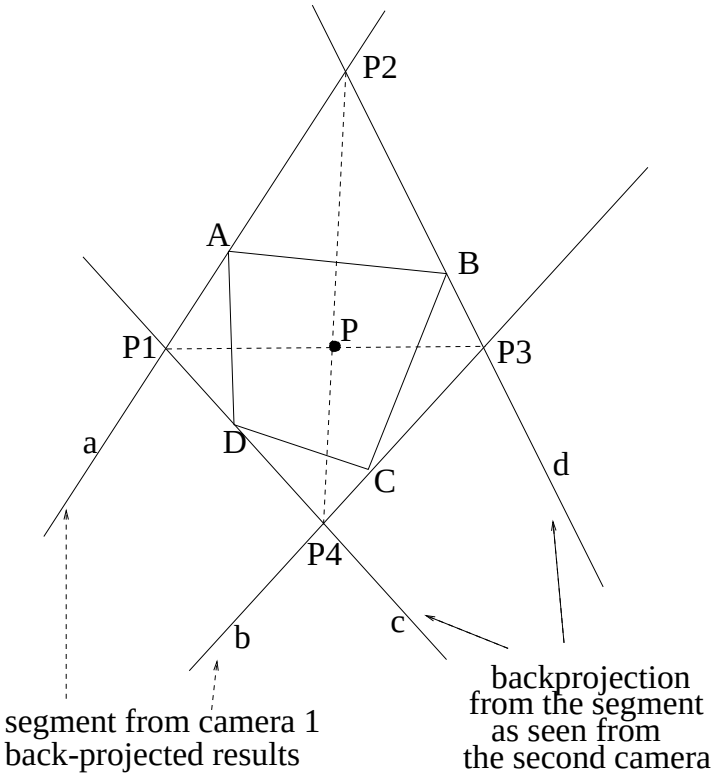

Figure A.1: Illustration for Appendix - shows that, for a convex object, the point of intersection of the diagonals of the quadrilateral formed by back-projecting the end-points of the matched segments is the only point guaranteed to lie inside the object

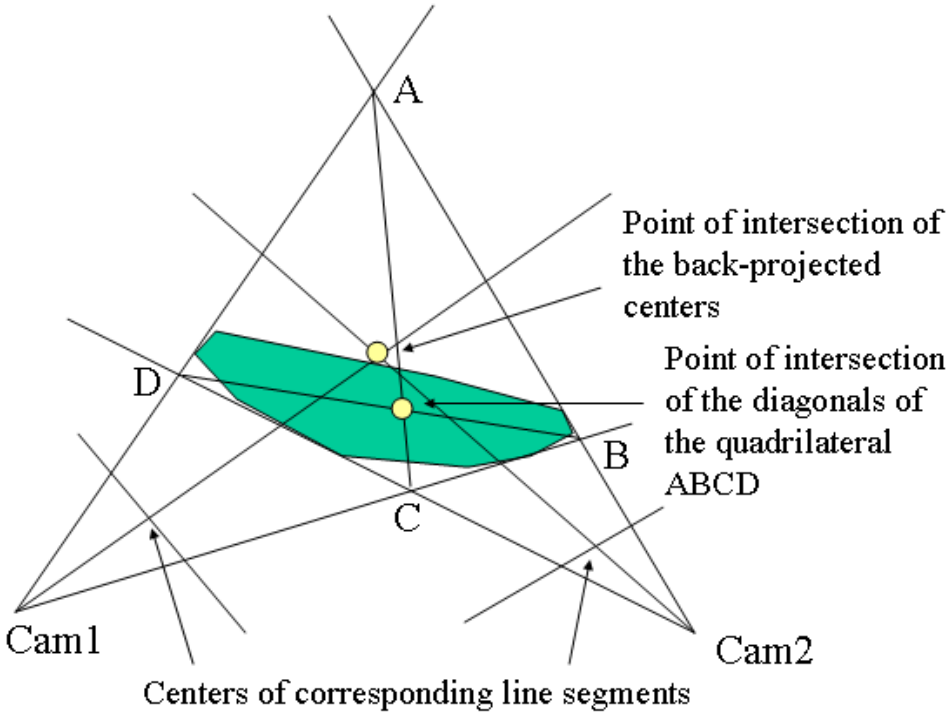

Figure A.2: Shows that the standard method of intersecting the projection of the centers of corresponding segments may result in a 3D point outside the object, but the point of intersection of the diagonals of quadrilateral ABCD is always inside the object if it is assumed to be convex.

Appendix B

Code for Visibility Analysis

```
#define PI 3.1416
#define SCALE 5

inline int sqr(int x){return(x*x);}

int updatevar(int var[], int i, int ncam){
   var[i-1] = var[i-1]+1;;
   if (var[i-1] >= ncam){
      if (i == 1) return 1;
      if (updatevar(var, i-1, ncam-1)) return 1;
      var[i-1] = var[i-2]+1;
   }
   return 0;
}

main(int argc, char *argv[]){
   int sizex = atoi(argv[1]);
   int sizey = atoi(argv[2]);
   int ncam = atoi(argv[3]);
   int pheight = atoi(argv[4]);
   int prad = atoi(argv[5]);
   int hvis = atoi(argv[6]);
   int maxangle = atoi(argv[7]);
   float pden = atof(argv[8]);

   int index = 9;
   int pcamx[100], pcamy[100], hcam[100];
   for (int camno = 0; camno < ncam; camno++){
      pcamx[camno] = atoi(argv[index++]);
      pcamy[camno] = atoi(argv[index++]);
```

```
   hcam[camno] = atoi(argv[index++]) - pheight;
}

float mu_rat[100];
float mincamdist[100];
for (camno = 0; camno < ncam; camno++){
  float mu = (float) hvis/(float) hcam[camno];
  mu_rat[camno] = mu/(mu+1.0);
  mincamdist[camno] = hcam[camno]/tan(maxangle*PI/180.0);
}

float probim[MAXY][MAXX];

float doccl[100];
int var[100];
float temp, temp2;

for (int yi = 0; yi <sizey/SCALE; yi++)
 for (int xi = 0; xi <sizex/SCALE; xi++){
  int x = SCALE*xi, y = SCALE*yi;

  for (int camno = 0; camno < ncam; camno++){
    float distcam =  sqrt(sqr(x - pcamx[camno])
       + sqr(y - pcamy[camno]));
    if (distcam > mincamdist[camno])
       doccl[camno] = distcam*mu_rat[camno];
    else doccl[camno] = -1.0;
  }

  prob[yi][xi] = 0.0;
  float sign = -1.0;

  for (int k = 1; k <= ncam; k++){
    sign = -sign;
    for (int i=0; i<k; i++)
      var[i] = i;

  int endcond = 0;
  while (!endcond){
    float sumdisoccl = 0.0;
    for (int i=0; (i<k) && (sumdisoccl > -0.5); i++){
      if (doccl[var[i]] >= 0.0)  sumdisoccl += doccl[var[i]];
```

```
            else sumdisoccl = -1.0;
        }

        float a = 1.0/(pden*2.0*prad*sumdisoccl);
        float b = PI*2.0*prad/sumdisoccl;
        if (sumdisoccl >= 0.0)
          probim[yi][xi] += sign*exp(-(2.0*a-b)/(2.0*a*(a-b)));

        endcond = updatevar(var, k, ncam);
    }}}}
```

BIBLIOGRAPHY

[1] D.P. Andersen. Efficient algorithms for automatic viewer orientation. *Computer and Graphics*, 9(4):407–413, 1985.

[2] P. Armstrong and J. Antonis. The construction of 3 dimensional models using an active vision system. In *European Conference on Computer Vision*, pages II: 182–196, Dublin, Ireland, May 2000.

[3] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *European Conference on Computer Vision*, May 1992.

[4] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 8–15, Santa Barbara, CA, June 1998.

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

[6] G.K.M. Cheung, T. Kanade, J.Y. Bouguet, and M. Holler. A real-time system for robust 3d voxel reconstruction of human motions. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 714–720, Hilton Head, SC, 2000.

[7] K. Choo and D. J. Fleet. People tracking using hybrid monte carlo filtering. In *IEEE International Conference on Computer Vision*, pages II: 321–328, Vancouver, Canada, July 2001.

[8] R.T. Collins, A.J. Lipton, H. Fujiyoshi, and T. Kanade. Algorithms for cooperative multi-sensor surveillance. *Proceedings of the IEEE*, 89(10):1456–1477, October 2001.

[9] D.J. Cook, P. Gmytrasiewicz, and L.B. Holder. Decision-theoretic cooperative sensor planning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):1013–1023, October 1996.

[10] C. K. Cowan and P.D. Kovesi. Automatic sensor placement from vision tast requirements. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3):407–416, May 1988.

[11] R.G. Cutler, R. Duraiswami, J.H. Qian, and L.S. Davis. Design and implementation of the university of maryland keck laboratory for the analysis of visual movement. Technical report, UMIACS, University of Maryland, 2000.

[12] T. Darrell, G. Gordon, M. Harville, and J. Woodfill. Integrated person tracking using stereo, color, and pattern detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 601–608, Santa Barbara, CA, June 1998.

[13] T.J. Darrell, D. Demirdjian, N. Checka, and P.F. Felzenszwalb. Plan-view trajectory estimation with dense stereo background models. In *IEEE International Conference on Computer Vision*, pages II: 628–635, Vancouver, Canada, July 2001.

[14] Q. Delamarre and O. D. Faugeras. 3d articulated models and multi-view tracking with silhouettes. In *IEEE International Conference on Computer Vision*, pages II: 716–721, Kerkyra, Greece, September 1999.

[15] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, 39 (Series B):1-38*, 1977.

[16] D. DiFranco, T. J.Cham, and J.M.Rehg. Reconstruction of 3-d figure motion from 2-d correspondences. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 307–314, Kauai, Hawaii, December 2001.

[17] T. Drummond and R. Cipolla. Real-time tracking of the multiple articulated structures in multiple views. In *European Conference on Computer Vision*, pages II: 20–36, Dublin, Ireland, May 2000.

[18] A. Elgammal, R. Duraiswami, and L.S. Davis. Efficient non-parametric adaptive color modeling using fast gauss transform. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages II:563–570, Kauai, Hawaii, December 2001.

[19] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. In *European Conference on Computer Vision*, pages II:751–767, Dublin, Ireland, May 2000.

[20] R. Fablet and M.J. Black. Automatic detection and tracking of human motion with a view-based representation. In *European Conference on Computer Vision*, page I: 476 ff., Copenhagen, Denmark, May 2002.

[21] O. Faugeras and R. Keriven. Complete dense stereovision using level set methods. In *European Conference on Computer Vision*, pages I: 379–393, Freiburg, Germany, June 1998.

[22] O.D. Faugeras. *Three-Dimensional Computer Vision, A Geometric Viewpoint*. MIT Press, 1993.

[23] P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 66–75, Hilton Head, SC, 2000.

[24] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. In *Proc. of the Thirteenth Conference on Uncertainty in Artificial Intelligence(UAI)*, August 1997.

[25] G. Gavrila and L. Davis. 3d model-based tracking of humans in action: a multi-view approach. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 73–80, San Francisco, CA, June 1996.

[26] N. Georgis, M. Petrou, and J.V. Kittler. Obtaining correspondences from 2d perspective views with wide angular separation of non-coplanar points. In *Proceedings of the European-Chinese Workshop on Computer Vision*, 1995.

[27] W.E.L. Grimson, C. Stauffer, R. Romano, and L. Lee. Using adaptive tracking to classify and monitor activities in a site. In *IEEE International Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, June 1998.

[28] D. Gross and C.M. Harris. *Fundamentals of Queueing theory: Second Edition*. Wiley Series in Probability and Mathematica Statistics, 1985.

[29] I. Haritaoglu, D. Harwood, and L. Davis. W_4: Who, When, Where, What: A Real-Time system for detecting and tracking people. In *IEEE Conference on Face and Gesture Recognition*, pages 222–227, 1998.

[30] I. Haritaoglu, D. Harwood, and L.S. Davis. W_4S: A Real-Time system for detecting and tracking people in 2 1/2-d. In *European Conference on Computer Vision*, pages I: 877–892, Freiburg, Germany, June 1998.

[31] I. Haritaoglu, D. Harwood, and L.S. Davis. Hydra: Multiple people detection and tracking using silhouettes. In *International Conference on Image Analysis and Processing*, 1999.

[32] D. D. Hoffman and M. Singh. Salience of visual parts. *Cognition*, 63:29–78, 1997.

[33] R. Horaud and T. Skordas. Stereo correspondence through feature grouping and maximal cliques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1168–1180, November 1989.

[34] B.K.P. Horn and E.J. Weldon-Jr. Direct methods for recovering motion. *International Journal of Computer Vision*, 2(1):51–76, June 1988.

[35] M. Huck, J. Raczkowsky, and K. Weller. Sensor simulations in robot applications. In *Advanced Robotics Program, Workshop on Manipulators, Sensors and Steps Towards Mobility*, pages 197–209, 1987.

[36] S.A. Hutchinson and A.C. Kak. Spar: A planner that satisfies operational and geometric goals in uncertain environments. *AIMag*, 11(1):31–61, 1990.

[37] K. Ikeuchi and J.C. Robert. Modeling sensor detectability with vantage geometric/sensor modeler. *IEEE Transactions on Robotics and Automation*, 7:771–784, 1991.

[38] S. Intille, J. Davis, and A. Bobick. Real-time closed-world tracking. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 697–703, Puerto Rico, June 1997.

[39] S.S. Intille and A.F. Bobick. Closed-world tracking. In *IEEE International Conference on Computer Vision*, pages 672–678, Boston, MA, 1995.

[40] S. Ioffe and D. Forsyth. Human tracking with mixtures of trees. In *IEEE International Conference on Computer Vision*, pages 690–695, Vancouver, Canada, July 2001.

[41] M. Irani and P. Anandan. Robust multi-sensor image alignment. In *IEEE International Conference on Computer Vision*, pages 959–966, Bombay, India, 1998.

[42] M. Irani and P. Anandan. A unified approach to moving object detection in 2d and 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(6):577–589, June 1998.

[43] I.A. Kakadiaris and D. Metaxas. Three-dimensional human body model acquisition from multiple views. *International Journal of Computer Vision*, 30(3):191–218, 1998.

[44] P. H. Kelly, A. Katkere, D.Y. Kuramura, S. Moezzi, S. Chatterjee, and R. Jain. An architecture for multiple perspective interactive video. In *Proceedings of the third ACM International Conference on Multimedia*, pages 201–212, 1995.

[45] V. Kettnaker and R. Zabih. Bayesian multi-camera surveillance. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages II:253–259, Ft. Collins, CO, June 1999.

[46] V. Kettnaker and R. Zabih. Counting people from multiple cameras. In *IEEE International Conference on Multimedia Computing and Systems*, pages II:253–259, 1999.

[47] S. Khan, O. Javed, Z. Rasheed, and M. Shah. Human tracking in multiple cameras. In *IEEE International Conference on Computer Vision*, pages I: 331–336, Vancouver, Canada, July 2001.

[48] L. Kleinrock. *Queueing Systems, Volumes I:Theory*. Wiley Interscience, New York, 1975.

[49] L. Kleinrock. *Queueing Systems, Volumes II:Computer Applications*. Wiley Interscience, New York, 1976.

[50] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russell. Towards robust automatic traffic scene analysis in real-time. In *International Conference on Pattern Recognition*, Israel, 1994.

[51] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S.A. Shafer. Multi-camera multi-person tracking for easyliving. In *Visual Surveillance*, 2000.

[52] K.N. Kutulakos and C.R. Dyer. Recovering shape by purposive viewpoint adjustment. *International Journal of Computer Vision*, 12(2-3):113–136, April 1994.

[53] K.N. Kutulakos and S.M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):199–218, 2000.

[54] S. Lee and H.S. Hahn. An optimal sensing strategy for recognition and localization of 3-d natural quadric objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1018–1037, October 1991.

[55] P. Lehel, E.E. Hemayed, and A.A. Farag. Sensor planning for a trinocular active vision system. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages II: 306–312, Ft. Collins, CO, June 1999.

[56] C.Y. Lin, S.W. Shih, Y.P. Hung, and G.Y. Tang. A new approach to automatic reconstruction of a 3-d world using active stereo vision. *Computer Vision and Image Understanding*, 85(2):117–143, February 2002.

[57] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *DARPA81*, pages 121–130, 1981.

[58] J.P. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. *International Journal of Computer Vision*, 39(1):57–71, August 2000.

[59] C.B. Madsen and H.I. Christensen. A viewpoint planning strategy for determining true angles on polyhedral objects by camera alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):158–163, February 1997.

[60] F. Martin and R. Horaud. Multiple camera tracking of rigid objects. *International Journal of Robotics Research*, 21(2):97–113, February 2002.

[61] S.O. Mason. Heuristic reasoning strategy for automated sensor placement. *PhEngRS*, 63(9):1093–1102, September 1997.

[62] S.O. Mason and A. Grun. Automatic sensor placement for accurate dimensional inspection. *Computer Vision and Image Understanding*, 61(3):454–467, May 1995.

[63] J. Maver and R.K. Bajcsy. Occlusions as a guide for planning the next view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(5):417–433, May 1993.

[64] A. Mittal and L.S. Davis. Unified multi-camera detection and tracking using region-matching. In *IEEE Workshop on Multi-Object Tracking*, pages 3–10, 2001.

[65] A. Mittal and L.S. Davis. M_2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene using region-based stereo. In *European Conference on Computer Vision*, page I : 18 ff., Copenhagen, Denmark, May 2002.

[66] A. Mittal and D.P. Huttenlocher. Scene modeling for wide area surveillance and image synthesis. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages II: 160–167, Hilton Head, SC, 2000.

[67] J. Miura and K. Ikeuchi. Task-oriented generation of visual sensing strategies. In *IEEE International Conference on Computer Vision*, pages 1106–1113, Boston, MA, 1995.

[68] H. Moon, R. Chellappa, and A. Rosenfeld. 3d object tracking using shape-encoded particle propagation. In *IEEE International Conference on Computer Vision*, pages II: 307–314, Vancouver, Canada, July 2001.

[69] G. Mori and J. Malik. Estimating human body configurations using shape context matching. In *European Conference on Computer Vision*, page III: 666 ff., Copenhagen, Denmark, May 2002.

[70] J. Mulligan, V. Isler, and K. Daniilidis. Trinocular stereo: A real-time algorithm and its evaluation. *International Journal of Computer Vision*, 47(1-3):51–61, April 2002.

[71] R. M. Neal and G. E. Hinton. A new view of the EM algorithm that justifies incremental and other variants. 1993.

[72] G. Olague and R. Mohr. Optimal camera placement to obtain accurate 3d point positions. In *International Conference on Pattern Recognition*, page AT11, 1998.

[73] J. Orwell, S. Massey, P. Remagnino, D. Greenhill, and G.A. Jones. A multi-agent framework for visual surveillance. In *International Conference on Image Analysis and Processing, Venice, Italy*, pages 1104–1107, 1999.

[74] J. Orwell, P. Remagnino, and G.A. Jones. Multi-camera colour tracking. In *Proceedings of the 2nd IEEE Workshop on Visual Surveillance, Fort Collins, Colorado*, 1999.

[75] R. Pito. A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1016–1030, October 1999.

[76] P. Pritchett and A. Zisserman. Wide baseline stereo matching. In *IEEE International Conference on Computer Vision*, pages 754–760, Bombay, India, 1998.

[77] J. Raczkowsky and K. H. Mittenbuehler. Simulation of cameras in robot applications. *Computer Graphics Applications*, pages 16–25, January 1989.

[78] M. K. Reed and P. K. Allen. Constraint-based sensor planning for scene modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1460–1467, December 2000.

[79] M.K. Reed, P.K. Allen, and I. Stamos. Automatic model acquisition from range images with view planning. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 72–77, Puerto Rico, June 1997.

[80] C. Ridder, O. Munkelt, and H. Kirchner. Adaptive background estimation and foreground detection using kalman filtering. In *Proc. International Conference on recent Advances in Mechatronics, 193–199*, 1995.

[81] D.R. Roberts and A.D. Marshall. Viewpoint selection for complete surface coverage of three dimensional objects. In *British Machine Vision Conference*, 1998.

[82] R. Ronfard, C. Schmid, and B. Triggs. Learning to parse pictures of people. In *European Conference on Computer Vision*, page IV: 700 ff., Copenhagen, Denmark, May 2002.

[83] R. Rosales and S. Sclaroff. 3d trajectory recovery for tracking multiple objects and trajectory-guided recognition of actions. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages II:117–123, Ft. Collins, CO, June 1999.

[84] R. Rosales, M. Siddiqui, J. Alon, and S. Sclaroff. Estimating 3d body pose using uncalibrated cameras. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 821–827, Kauai, Hawaii, 2001.

[85] S.D. Roy, S. Chaudhury, and S. Banerjee. Isolated 3-d object recognition through next view planning. *SMC-A*, 30(1):67–75, January 2000.

[86] S.D. Roy, S. Chaudhury, and S. Banerjee. Recognizing large 3-d objects through next view planning using an uncalibrated camera. In *IEEE International Conference on Computer Vision*, pages II: 276–281, Vancouver, Canada, July 2001.

[87] S. Sakane, M. Ishii, and M. Kakikura. Occlusion avoidance of visual sensors based on a hand eye action simulator system:heaven. *Advanced robotics*, 2(2):149–165, 1987.

[88] J.M. Sanchiz and R.B. Fisher. A next-best-view algorithm for 3d scene recovery with 5 degrees of freedom. In *British Machine Vision Conference*, pages 163–172, September 1999.

[89] F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo. In *IEEE International Conference on Computer Vision*, pages II: 636–643, Vancouver, Canada, July 2001.

[90] D.W. Scott. *Multivariate Density Estimation*. Wiley-Interscience, 1992.

[91] S.W. Sedas-Gersey. *Algorithms for Automatic Sensor Placement to Acquire Complete and Accurate Information*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 1993.

[92] J. Shi and C. Tomasi. Good features to track. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 593 – 600, 1994.

[93] A. Shmuel and M. Werman. 3d from an image sequence: Occlusions and perspective. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 712–713, 1991.

[94] H. Sidenbladh, M. J. Black, and D. J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *European Conference on Computer Vision*, pages II: 702–718, Dublin, Ireland, May 2000.

[95] D. Snow, P. Viola, and R. Zabih. Exact voxel occupancy with graph cuts. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages I:345–352, Hilton Head, SC, 2000.

[96] Y. Song, X. Feng, and P. Perona. Towards detection of human motion. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 810–817, Hilton Head, SC, 2000.

[97] I. Stamos and P. K. Allen. Interactive sensor planning. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 489–494, 1998.

[98] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *IEEE International Conference on Computer Vision and Pattern Recognition*, page II: 252, Ft. Collins, CO, June 1999.

[99] V.A. Sujan. Optimum camera placement by robot teams in unstructured field environments. In *International Conference on Image Processing*, pages III: 861–864, 2002.

[100] J. Sullivan and S. Carlsson. Recognizing and tracking human action. In *European Conference on Computer Vision*, page I: 629 ff., Copenhagen, Denmark, May 2002.

[101] R. Szeliski. Image mosaicing for tele-reality applications. In *WACV94*, pages 44–53, 1994.

[102] K. Tarabanis, R.Y. Tsai, and A. Kaul. Computing viewpoints that satisfy optical constraints. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 152–158, 1991.

[103] K. Tarabanis, R.Y. Tsai, and A. Kaul. Sensor planning in computer vision. *IEEE Transactions on Robotics and Automation*, 11(1):86–105, February 1995.

[104] K. Tarabanis, R.Y. Tsai, and A. Kaul. Computing occlusion-free viewpoints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(3):279–292, March 1996.

[105] K.A. Tarabanis, R.Y. Tsai, and P.K. Allen. The MVP sensor planning system for robotic vision tasks. *IEEE Transactions on Robotics and Automation*, 11(1):72–85, February 1995.

[106] D. Tell and S. Carlsson. Combining appearance and topology for wide baseline matching. In *European Conference on Computer Vision*, page I: 68 ff., Copenhagen, Denmark, May 2002.

[107] T. Tuytelaars and L.J. Van Gool. Wide baseline stereo matching based on local, affinely invariant regions. In *British Machine Vision Conference*, 2000.

[108] P. Viola and W. M. WellsIII. Alignment by maximization of mutual information,. In *IEEE International Conference on Computer Vision*, pages 16–23, Boston, MA, 1995.

[109] P. Whaite and F.P. Ferrie. Active exploration: Knowing when we're wrong. In *IEEE International Conference on Computer Vision*, pages 41–48, 1993.

[110] M. Yamamoto, A. Sato, S. Kawada, T. Kondo, and Y. Osaki. Incremental tracking of human actions from multiple views. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2–7, Santa Barbara, CA, June 1998.

[111] Y. Ye and J.K. Tsotsos. Sensor planning for 3d object search. *Computer Vision and Image Understanding*, 73(2):145–168, February 1999.

[112] S.K. Yi, R.M. Haralick, and L.G. Shapiro. Optimal sensor and light-source positioning for machine vision. *Computer Vision and Image Understanding*, 61(1):122–137, January 1995.

[113] H. Zhang. Two-dimensional optimal sensor placement. *SMC*, 25(5):781–792, May 1995.

[114] L. Zhao. *Dressed Human Modeling, Detection, and Parts Localization*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2001.

[115] T. Zhao, R. Nevatia, and F. Lv. Segmentation and tracking of multiple humans in complex situations. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 194–201, Kauai, Hawaii, 2001.

[116] S. Zhou and R. Chellappa. Probabilistic human recognition from video. In *European Conference on Computer Vision*, page III: 681 ff., Copenhagen, Denmark, May 2002.