Methods for the Analysis of Visual Motion

Thesis by

Xiaolin Feng

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2002

(Defended May 8, 2002)

To My Family

Acknowledgements

First and foremost, I would like to express my deepest thanks to my advisor, Professor Pietro Perona, whose keen insight has continuously provided me with invaluable advice for the work in this thesis. His great vision and wealth of knowledge in science and engineering have been a constant source of my inspiration. Undoubtedly, none of the results in this thesis would have been possible without his guidance. I also greatly appreciate his understanding and support when my personal life was at its down point. Without his encouragement, I would never be able to overcome those difficulties and finish this thesis. I cannot thank him enough for all he did for me.

I owe my special thanks to Professor John Doyle, who admitted me to Caltech Control and Dynamical System Department in 1995 and supported my study and research during my first two years. I thank Professor Stefano Soatto of UCLA, who introduced me to Caltech vision group and guided my first experimental project together with Pietro. He has continuously supported my research in computer vision by sharing his knowledge and ideas without any reservation in our discussions during the past six years. I would also like to thank Professors Joe Burdick, Christof Koch and Demetri Psaltis, who along with Pietro and Stefano, served on my thesis committee.

I am fortunate in spending these years in the Caltech vision group which is a fantastic place for doing research and enjoying life. I want to thank people in this group, Dr. Arrigo Benedetti, Dr. Enrico Di Bernardo, Dr. Jean-Yves Bouguet, Dr. Michael Burl, Dr. Luis Goncalves, Dr. Mario Munich, Dr. Marzia Polito, Dr. Markus Weber, Dr. Max Welling, Anelia Angelova, Claudio Fanti, Rob Fergus, Feifei Li, Pierre Moreels, Silvio Savarese, Yang Song, Catharine Stebbins, and Domitilla Del Vecchio. Special thanks go to Dr. Jean-Yves Bouguet and Yang Song for their respective collaboration in the work of 3-view geometry (Chapter 3) and human motion detection (Chapter 5). My friend Xinwei Yu provided great help in capturing the large image database for the recognition experiments (Chapter 6), which I appreciate very much.

Special thanks also go to our secretary Catherine Stebbins for all her precious help in making my study and research much more easier.

I am always grateful to many friends who have tremendously helped and supported me during my years at Caltech. I will never forget the continuous encouragement from my best friends Hongtao Sun, Li Zhang, Ning Sun, Yin He, Dr. Yin Su, Hailin Jin, Dr. Xiaoyun Zhu, Feifei Li, and Wei Yan. They are part of the treasure of my life. I also greatly appreciate the friendship and support from some colleagues, Dr. Jean-Yves Bouguet, Dr. Mario Munich and Dr. Luis Goncalves. Their advices and patience helped me so much in overcoming all the difficulties and coming to the finish of this thesis.

No words in the world can fully express my greatest appreciation to my parents, grandparents and sister for their lifelong support and unconditional love of me. I thank my brother-in-law for his sincere advices and care in these years. With my heart, I dedicate this thesis to my truly beloved family members.

Abstract

Vision is a primary sense that allows human beings to interact with their environment and motion is one of the most important cues that vision can explore and utilize.

In this thesis, we present computational approaches to the problems of inferring three-dimensional motion information and perceiving two-dimensional human motions from a sequence of images captured by a camera.

The three-dimensional structure of world can be represented by distinguishable features, such as points. Assume all the features move under the same rigid motion in space, this motion can be recovered from the projections of the features in three views by solving a set of trilinear constraints. The trilinear constraints have been considered only as algebraic equations so that their satisfactory performance in motion estimation is not easy to understand. This thesis solves this puzzle by discovering a geometrical interpretation of trilinear constraints. It is showed that those algebraic equations correspond to depth errors appropriately weighted by a function of the relative reliability of the corresponding measurements. When the assumption is relaxed to allowing features to move under different rigid motions, this thesis proposes a three-dimensional motion based expectation-maximization algorithm combined with the modified separation matrix scheme to cluster the features undergoing the same motion into a group and estimate the motion for every group at the same time.

The problem of detecting and recognizing human motions arises from many applications in computer vision. This thesis describes an algorithm to detect human body from their motion patterns in a pair of frames which is based on learning an approximate probabilistic model of the positions and velocities of body joints. It then presents a scheme to recognize human actions in a sequence of frames assuming the human body is detected. This scheme enables us to simultaneously recognize both the action and the body poses in the observed sequence.

All our theoretical work is supported by experimental results.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Motion Analysis in Computer Vision

Computer vision is the engineering to understand images by using computers. It has ties to many other disciplines, such as artificial intelligence, graphics, signal and image processing, robotics, and physiology. It has a multitude of applications: image analysis and processing, medical diagnostics, industrial quality control, security surveillance, robotics, and human-machine visual interactions.

Vision is the most complex and informative of the human senses, characterized by high resolution and efficient recognition. It is so advanced that it has been functioning so well without being noticed or credited sometimes. Take a moment to think about what our eyes can do for us. We recognize objects by looking at them, we estimate how far away an object is from us, we have an idea of the relative motion between us and an object or the environment. More fascinating is that we do not need to see things clearly to get what we want, sometimes part of the object can even be occluded. Occasionally our eyes can be confused by illusions, too. Nonetheless, the amazing capability of human vision sets a very high bar for computer vision.

For most of the so-called computer vision, the goal is to replicate the human visionary functions by a computer and camera system. Though we have probably taken them for granted, it has been an elusive problem in general for decades to make a computer see as we do. Computers have become masters in things that are so difficult for ordinary people, such as playing chess and deducing formulas, which is worth celebrating. Unfortunately computer vision is not a lucky one of them, all have been achieved in this field is not even close to what human vision can do. While enjoying what our eyes bring to us so easily everyday, we sometimes cannot deny the tiny bitter frustration of working in this field.

The conceptual approach taken in computer vision is first to extract information from images, then to recognize the subject behind the information, and finally use the realized information to achieve our goal. But in reality every step is much more complicated and most of the attention has been focused on the first two, in which the fundamental question is object and pattern recognition and scene reconstruction. Many cues have been worked on to explore this territory. Among them are motions, textures, shadings, contours, and colors, to name a few. While they accommodate many efforts and progress has been made in these directions, my research has been mainly focused on using the motion cues.

The problems of motion analysis in computer vision can be contemplated from different angles. But in general the questions can be put in three different categories:

What do we measure?

Due to the complex nature of objects, they usually cannot be considered in their full details because of various limitations such as our knowledge, methodology and computational efficiency. In the study of motion, objects are usually characterized by their representative features and these are the things we need to measure directly from images. Specifically we need to search for the special points, lines, and patches on objects which are least demanding but still carry the most relevant and revealing information. Due to the relative motion between camera and objects, it is also necessary to construct the correspondence between features in different image frames. Feature correspondence tells where a feature locates in the next image. There are two typical approaches to this problem: feature matching and feature tracking. The feature matching method selects features in two different images and matches them between the two. The feature tracking method selects features in one image and tracks them to the next one. It is more commonly used in recent years. Feature tracking has been extensively studied in the past decade [49, 88, 3, 10, 46, 43]. Most of the time a careful decision has to be made on what features we are going to use to model the object and study the underlying motion. Sometimes using simple point features can simplify the solution, while some other times we have to deal with more complicated patch features. Both will be demonstrated in my actual projects presented in this thesis, together with the consideration of the representative capabilities of the candidate features and the goals that need to be achieved by the experiments.

How do we separate multiple moving objects?

It is not often that the object we want to study is alone in the scene. Most likely there are more than one objects including the background structure and they may move differently in the scene. The question is how to separate these moving objects from each other. Again many cues can be employed, but motion is the primary one here. As the input to motion segmentation, most of the time distinguished features are selected with established correspondence to represent the objects in image frames, and occasionally all pixels in images are considered as features which are called "dense flow." Image features are clustered into groups according to their different motions, thus features from two different objects could end up with being in the same group as long as they take on the same motion relative to the camera. The grouping can follow through sequential image frames and judgment can be refined while more information is absorbed. On the other hand, the challenge could be pushed to the limit where only two consecutive image frames are available. Along the spatial dimension, motion segmentation can be considered in either two-dimensional (2D) image plane or three-dimensional (3D) space. 2D studies usually rely on parametric or nonparametric optical flow models for the projected motion in images, while 3D segmentation exploits the characters of the more complicated 3D motion (rotation and translation) itself. When the objects are separated according to their motions, the undertaken motions can be estimated and some structural information of the objects can be recovered as well.

What do we estimate?

First of all, we want to estimate the 3D motion itself. The absolute camera motion in space is called "ego motion" as opposed to the "object motion". However, the motion captured by a camera is a relative motion. No matter the actual motion is "ego" or "object" or a combination, all we can recover from an image sequence is the relative motion between the object and the camera. In addition, we can estimate 3D object structure, i.e., the positions in space of the points belonging to objects and background. The depth information is lost after the projection from 3D space to

2D image plane. A single image can only tell us from which directions image points are projected, but not how far away their true locations are along those directions. This is similar to the situation that we capture a snapshot of the world with only one eye in still. However using our two eyes, which locate at different positions, the depths of the objects can be recovered from the two different views and we are able to perceive their structures. It is certain that more views contain more information. This structure reconstruction from multiple views depends on the relative location of our eyes, or in the language of computer vision, the relative motion of camera between two or more views. Therefore, one classical problem in computer vision is how to recover the 3D motion and structure from a sequence of image frames.

If we are interested in a particular class of objects, possibly more interesting tasks can be carried out, for example, lip reading, gesture recognition, facial expression recognition, human motion detection, human action recognition, etc. Among them, human motion analysis has become an active research area in the last decade. From a sequence of image frames, the task for human motion detection is to answer if a subject is present in the scene by looking at the motions that might be taken by the subject, while for human action recognition the question is what action the subject is taking assuming the presence of the subject is already known. Solutions to both questions are knowledge based. To gain the knowledge of motions that it can recognize later, a computer system has to be trained first with some small but necessary amount of well-defined data related to the motions under concern, which is therefore referred to as "supervised learning". Afterwards, the perceived motions can be compared with the pre-acquired knowledge in database and decisions to the tasks can be made. Success in human motion detection and recognition will certainly discover many interesting and practical applications, but on the other hand, the answer to the challenge is complicated by the very nature of a human such as high degrees of freedom, rich textural characters and self-occlusion.

One common theme in the study of computer vision is the uncertainty of what is seen and measured. Distortion happens within the imaging apparatus. Measurement errors are generated in selecting and tracking image features. Large variances exist in the appearance and action performance of different individual subjects of the same kind. These uncertainties mandate that our approach be flavored by probabilistic analysis. The acceptance or rejection of an outcome is thus tied to a preset threshold or confidence level.

1.2 Outline of the Thesis

My research is on four different topics and has spanned the three general questions posed above. But they are all tied to each other, sharing the common thread of motion analysis by computer visual methods.

First, some basic introduction is given in Chapter 2 about the mathematics of projective geometry, camera calibration, and feature detection and tracking. The introduction facilitates the presentation of the following chapters.

Chapter 3 is about my work on a geometrical interpretation of trilinear constraints, which are commonly used in the study of motion and structure recovery. It is common knowledge in the computer vision community that these constraints are algebraic equations whose advantage is finding solutions in closed form. Surprisingly they are almost as accurate as implicit methods, such as the nonlinear reprojection error minimization method. An answer to this mystery is given in this chapter, accompanied by a related and robust scale propagation scheme and verification experiments.

Chapter 4 presents an approach to 3D motion segmentation. The task is challenged by the assumption that there are only two consecutive images available. A combination algorithm of expectation maximization and modified separation matrix is proposed. Some supporting experiment evidence is also presented.

The next two chapters describe my work on human motion analysis. The first is human motion detection in Chapter 5. The task is especially difficult because we want to detect a moving person from only two consecutive images. This is further complicated by the subject's self-occlusion and clutters around. Detection is achieved by setting a threshold to the summation of likelihoods from all possible labelings and localization of the human body is done by finding the labeling with the maximum likelihood. The two computations are performed efficiently by algorithms of dynamic programming or one similar. Satisfactory detection results are also shown by experiments.

In Chapter 6, an algorithm toward human action recognition is introduced. Both periodic and nonperiodic actions are studied. The special aspects of this work are patch features, the specially designed color-coded clothes for training, a special representation of human body poses – we call it "movelet", a simplified description of the huge movelet space by vector quantization, and a special utilization of HMMs as models for actions. Like the other three pieces of work, the experimental results are satisfactory. Influences of different descriptions of the movelet space and different lengths of image sequences to the performance of our recognition system are also studied.

Some final thoughts are given in Chapter 7 to conclude this thesis on visual methods for motion analysis. Now let us move on to see what motion can reveal to us.

Chapter 2 Background: Geometry and Features

This chapter introduces the background material to the thesis.

2.1 Projective Space and Euclidean Space

A point in n-dimensional projective space, $\mathcal{P}^n$, is represented by an $n+1$ vector of coordinates $\overline{x} = [x_1 \ \ x_2 \ \ \ldots \ \ x_{n+1}]^T$, where at least one of the x_i is nonzero. The numbers x_i are called homogeneous coordinates of the point. Two vectors $\overline{x} = [x_1 \ \ x_2 \ \ \ldots \ \ x_{n+1}]^T$ and $\overline{y} = [y_1 \ \ y_2 \ \ \ldots \ \ y_{n+1}]^T$ represent the same point if and only if there exists a nonzero scalar λ such that $x_i = \lambda y_i$ for $1 \leq i \leq n+1$. We denote this up to a nonzero scalar equality as $\overline{x} \simeq \overline{y}$.

Let E be the three-dimensional (3D) Euclidean space. For a given position of a camera in the space, we define the standard camera reference frame as $\mathcal{F} = (O_c, X_c, Y_c, Z_c)$, where O_c is the center of camera projection and the three axes (O_c, X_c), (O_c, Y_c) and (O_c, Z_c) are mutually orthogonal and right-handed. The axes (O_c, X_c) and (O_c, Y_c) are chosen parallel to the image plane and (O_c, Z_c) is along the camera optical axis.

The Euclidean space E can be viewed as embedded in the projective space $\mathcal{P}^3$. If we refer to a point in the reference frame $\mathcal{F}$ by its Euclidean coordinate vector $\overline{X} = [X \ \ Y \ \ Z]^T$, this point is alternatively represented by the homogeneous coordinate vector $\overline{\mathbf{X}} = [X \ \ Y \ \ Z \ \ 1]^T$ in the space $\mathcal{P}^3$.

In $\mathcal{P}^3$, a plane is represented by a homogeneous vector $\overline{\boldsymbol{\pi}} = [\pi_1 \ \ \pi_2 \ \ \pi_3 \ \ \pi_4]^T$. A point P lies on the plane if and only if its homogeneous coordinate vector $\overline{\mathbf{X}}$ satisfies $\overline{\boldsymbol{\pi}}^T \overline{\mathbf{X}} = 0$.

2.2 Perspective Projection and Image Plane

Consider the perspective projection of points in 3D space onto the 2D image plane. The center of the projection is denoted as O_c in the camera reference frame $\mathcal{F}$. The image plane, also called *focal plane*, is the plane at $Z = f$, where f is the focal length of the camera. The image reference frame is defined as (c, x_c, y_c), where c is the intersection point between optical axis and image plane and (c, x_c) and (c, y_c) are the two axes parallel to the axes (O_c, X_c) and (O_c, Y_c).

Under the perspective projection, a point P in space with coordinates $\overline{X} = [X \;\; Y \;\; Z]^T$ is mapped to the point $\overline{x} = [x \;\; y]^T$ on the image plane, where a line joining the point P to the projection center O_c meets the image plane (see Figure 2.1). By similar triangles, it can be shown easily that

$$\overline{x} = \begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} fX \\ fY \end{bmatrix} \tag{2.1}$$

This perspective projection model is also referred to as a " pinhole" camera model.

Similar to Euclidean space, the image plane can be viewed as a two-dimensional projective space $\mathcal{P}^2$. In this representation, a point p on the image plane has homogeneous coordinate vector $\overline{\mathbf{x}} = [x \;\; y \;\; 1]^T$. Using projective geometry, the projection operator defined in equation 2.1 becomes

$$Z\overline{\mathbf{x}} = K\,\mathbf{G}\overline{\mathbf{X}} \qquad \text{with} \qquad K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{G} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_{3\times1} \end{bmatrix} \tag{2.2}$$

where $\mathbf{I}_3$ is the 3×3 identity matrix and $\mathbf{0}_{3\times1}$ is the 3×1 zero vector. Since $Z\overline{\mathbf{x}}$ represents the same point as $\mathbf{x}$ in $\mathcal{P}^2$, the operator in equation 2.2 is a linear projection operator from $\mathcal{P}^3$ to $\mathcal{P}^2$.

A line on the image plane $\mathcal{P}^2$ is represented by a homogeneous vector $\overline{\mathbf{l}} = [l_1 \;\; l_2 \;\; l_3]^T$. An image point p lies on the line if and only if its homogeneous coordinate vector $\overline{\mathbf{x}}$ satisfies $\overline{\mathbf{l}}^T \overline{\mathbf{x}} = 0$.

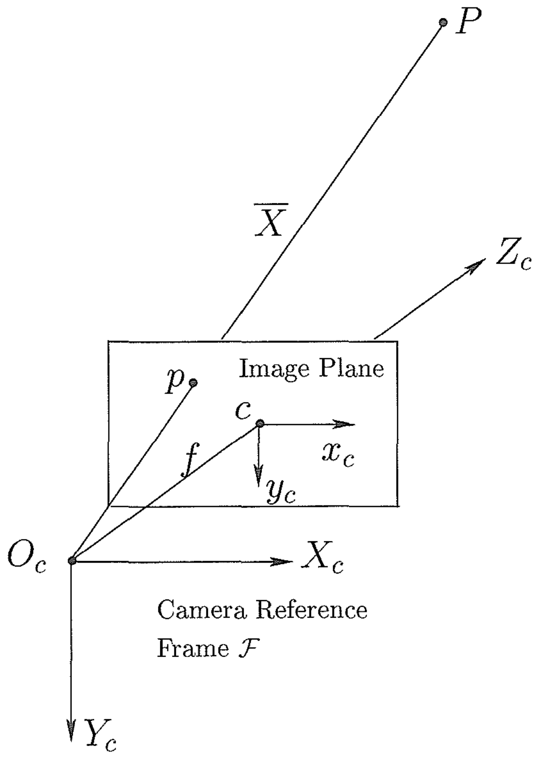


Figure 2.1: **Pinhole Camera Geometry.**

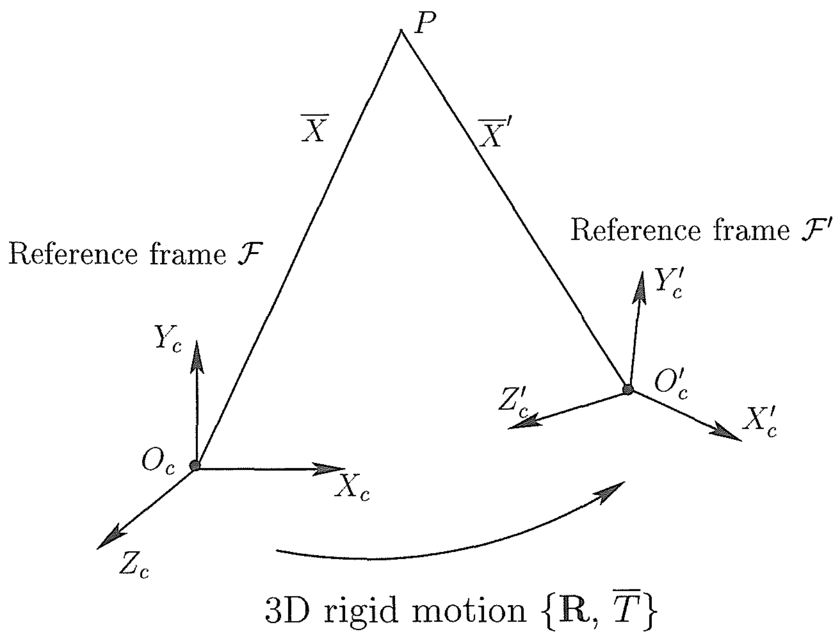

P
$\overline{X}$
$\overline{X}'$
Reference frame $\mathcal{F}$
Reference frame $\mathcal{F}'$
Y_c
O_c
X_c
Z_c
Y_c'
O_c'
Z_c'
X_c'
3D rigid motion $\{\mathbf{R}, \overline{T}\}$

Figure 2.2: **Rigid Body Motion Transformation** between camera frames $\mathcal{F} = (O_c, X_c, Y_c, Z_c)$ and $\mathcal{F}' = (O_c', X_c', Y_c', Z_c')$. The two coordinate vectors $\overline{X}$ and $\overline{X}'$ of point P in $\mathcal{F}$ and $\mathcal{F}'$ are related to each other through the rigid body motion transformation $\overline{X}' = \mathbf{R}\,\overline{X} + \overline{T}$.

2.3 Rigid Body Motion Transformation

Consider a point P in space, and let $\overline{X} = [X \ \ Y \ \ Z]^T$ be its coordinate vector in the camera reference frame $\mathcal{F}$. Suppose the camera moves to a new location in space, and let $\overline{X}' = [X' \ \ Y' \ \ Z']^T$ be the coordinate vector of the same point P in the new camera reference frame $\mathcal{F}'$ (see Figure 2.2). Then $\overline{X}$ and $\overline{X}'$ are related to each other through a rigid body motion transformation

$$\overline{X}' = \mathbf{R}\,\overline{X} + \overline{T} \tag{2.3}$$

where $\mathbf{R} \in SO(3)$[1] and $\overline{T}$ are respectively a 3×3 rotation matrix and a 3×1 vector that uniquely define the rigid motion between the two camera positions. The matrix $\mathbf{R}$ is defined by a rotation vector $\overline{\Omega} = [\Omega_x \ \ \Omega_y \ \ \Omega_z]^T$ such that

$$\mathbf{R} = e^{(\overline{\Omega}\wedge)} \tag{2.4}$$

where $(\overline{\Omega}\wedge)$ is the following skew-symmetric matrix

$$(\overline{\Omega}\wedge) = \begin{bmatrix} 0 & -\Omega_z & \Omega_y \\ \Omega_z & 0 & -\Omega_x \\ -\Omega_y & \Omega_x & 0 \end{bmatrix} \tag{2.5}$$

Equation 2.4 may also be written in a compact form using the Rodrigues' formula [60]

$$\mathbf{R} = \mathbf{I}_3 + \frac{\sin\theta}{\theta}\,(\overline{\Omega}\wedge) + \frac{1-\cos\theta}{\theta^2}\,(\overline{\Omega}\wedge)^2 \tag{2.6}$$

where $\theta = ||\overline{\Omega}||$.

The fundamental rigid body motion equation 2.3 may also be written in projective space $\mathcal{P}^3$. In $\mathcal{P}^3$, the point P has homogeneous coordinate vectors $\overline{\mathbf{X}} = [X \ \ Y \ \ Z \ \ 1]^T$ and $\overline{\mathbf{X}}' = [X' \ \ Y' \ \ Z' \ \ 1]^T$ in the first ($\mathcal{F}$) and second ($\mathcal{F}'$) reference frames,

[1]Special orthogonal 3×3 matrices.

respectively. Then, equation 2.3 may be rewritten as

$$\overline{\mathbf{X}}' = D\overline{\mathbf{X}} \qquad \text{with} \qquad D = \begin{bmatrix} \mathbf{R} & \overline{T} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{2.7}$$

where $\mathbf{0}_{1\times 3}$ is the 1×3 zero row vector. Observe that the inverse relation may also be written as follows

$$\overline{\mathbf{X}} = D^{-1}\overline{\mathbf{X}}' \qquad \text{with} \qquad D^{-1} = \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T\overline{T} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{2.8}$$

Let p' be the projection of point P on the image plane of the second camera, and $\overline{\mathbf{x}}' = [x' \;\; y' \;\; 1]^T$ be the homogeneous coordinate vector. Then, following equation 2.2, we have

$$Z'\overline{\mathbf{x}}' = K\,\mathbf{G}\,\overline{\mathbf{X}}' \tag{2.9}$$

By plugging equation 2.7 in, the above projection becomes

$$Z'\overline{\mathbf{x}}' = K\,\mathbf{G}\,D\overline{\mathbf{X}} = K\,\mathbf{G}'\,\overline{\mathbf{X}} \tag{2.10}$$

where

$$\mathbf{G}' = \mathbf{G}\,D = \begin{bmatrix} \mathbf{R} & \overline{T} \end{bmatrix} \tag{2.11}$$

The same model can also be applied to the situation where the camera is fixed and the 3D rigid scene is moving in the space. In this case, equation 2.3 represents how the coordinates of a point P in the 3D rigid scene change to those of the next position with respect to the same camera reference frame. Keeping the relative motion between the camera and the scene unchanged, this case can also be viewed equivalently as the camera moves and the scene is still.

2.4 Pixel Coordinates and Camera Matrices

2.4.1 Pixel Coordinates

The image reference frame defined in section 2.2 has the same metric as the Euclidean camera reference frame $\mathcal{F}$. Therefore, the point coordinates $[x \quad y]^T$ are metric coordinates (usually in meters). However, the position of a point p in a real image is originally expressed in pixel units. We can only observe that a point is located at the intersection of column u_x and row u_y on the given digitized image, which gives the pixel coordinates of the point as $[u_x \quad u_y]$. The process of establishing a correspondence between such pixel coordinates and metric ones is called *camera calibration.*

Since the origin of the image reference frame is at the principal point c, it is necessary to know the location of that point in the image: $\overline{c} = [c_x \quad c_y]^T$ (in pixels). Let d_x and d_y be the metric measurements of the x_c and y_c dimensions of one pixel in the imaging sensor, namely, the size of a pixel is d_x meter along x_c axis and d_y meter along y_c axis. Then the pixel coordinates $\overline{u} = [u_x \quad u_y]^T$ of a point on the image plane may be computed from its metric coordinates $\overline{x} = [x \quad y]^T$ through the following expression

$$\left\{ \begin{array}{rcl} u_x & = & x\,/\,d_x + c_x \\ u_y & = & y\,/\,d_y + c_y \end{array} \right. \tag{2.12}$$

This model assumes that the two axes of the imaging sensor are orthogonal. If both coordinates are represented by homogeneous vectors $\overline{\mathbf{x}} = [x \quad y \quad 1]^T$ and $\overline{\mathbf{u}} = [u_x \quad u_y \quad 1]^T$ in $\mathcal{P}^2$ respectively, the coordinate transformation equation 2.12 can also be written as

$$\overline{\mathbf{u}} = \begin{bmatrix} 1/d_x & 0 & c_x \\ 0 & 1/d_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \overline{\mathbf{x}} \tag{2.13}$$

2.4.2 Camera Matrices

Putting equation 2.13 together with equation 2.2 leads to

$$Z\,\overline{\mathbf{u}} = \mathbf{K}\,\mathbf{G}\,\overline{\mathbf{X}} \qquad \text{with} \qquad \mathbf{K} = \begin{bmatrix} 1/d_x & 0 & c_x \\ 0 & 1/d_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \; K = \begin{bmatrix} f/d_x & 0 & c_x \\ 0 & f/d_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.14}$$

which represents the global perspective projection from point P of metric coordinates in $\mathcal{P}^3$ to point p of pixel coordinates on the image plane. Let $f_x = f/d_x$ and $f_y = f/d_y$ which measure the focal length f in pixels. Notice that for most imaging sensors currently available, pixels may be assumed perfectly square, which implies that $d_x = d_y$ or $f_x = f_y$. But in the general, f_x and f_y can be different. With this denotation, the complete representation for the global perspective projection is

$$Z\,\overline{\mathbf{u}} = \mathbf{K}\,\mathbf{G}\,\overline{\mathbf{X}} = \mathbf{P}\,\overline{\mathbf{X}} \qquad \text{with} \qquad \mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{G} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_{3\times 1} \end{bmatrix} \tag{2.15}$$

The matrix $\mathbf{P} = \mathbf{K}\,\mathbf{G}$ is called *camera projection matrix* and the matrix $\mathbf{K}$ is called *camera calibration matrix.*

Similarly, if the camera undergoes a rigid motion transformation $\mathbf{R}$ and $\overline{T}$ in space, the projection associated with the camera's new location is obtained from equations 2.10, 2.11 and 2.13 as

$$Z'\,\overline{\mathbf{u}}' = \mathbf{K}\,\mathbf{G}'\,\overline{\mathbf{X}} = \mathbf{P}'\,\overline{\mathbf{X}} \qquad \text{with} \qquad \mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{G}' = \begin{bmatrix} \mathbf{R} & \overline{T} \end{bmatrix} \tag{2.16}$$

The matrix $\mathbf{P}' = \mathbf{K}\,\mathbf{G}'$ is the projection matrix associated with the second camera location.

2.4.3 Normalized Coordinates

Consider a camera projection matrix decomposed as $\mathbf{P} = \mathbf{K}\,\mathbf{G}$ and let $Z\,\overline{\mathbf{u}} = \mathbf{P}\,\overline{\mathbf{X}}$ be the camera projection of a point with coordinates $\overline{\mathbf{X}}$ in $\mathcal{P}^3$ to a point in the image with pixel coordinates $\overline{\mathbf{u}}$ in $\mathcal{P}^2$. If the calibration matrix $\mathbf{K}$ is known, its inverse can be applied to the point $\overline{\mathbf{u}}$ to obtain the point $\overline{\mathbf{x}} = \mathbf{K}^{-1}\overline{\mathbf{u}}$. Then $Z\,\overline{\mathbf{x}} = \mathbf{G}\,\overline{\mathbf{X}}$, where $\overline{\mathbf{x}}$ is the image point expressed in *normalized coordinates.* It may be thought of as the image of the point $\overline{\mathbf{X}}$ with respect to a camera transformation $\mathbf{G}$ having the identity matrix $\mathbf{I}_3$ as calibration matrix. The projection matrix $\mathbf{K}^{-1}\mathbf{P} = \mathbf{G}$ is called *normalized camera projection matrix* in which the effect of known calibration matrix is removed. If the world coordinate frame to represent $\overline{\mathbf{X}} = [X \quad Y \quad Z \quad 1]^T$ is the same as the camera reference frame, which implies $\mathbf{G} = [\mathbf{I}_3 \quad \mathbf{0}_{3\times1}]$, the normalized coordinates are simply $\overline{\mathbf{x}} = [X/Z \quad Y/Z \quad 1]^T$.

2.5 Camera Calibration

The assumption throughout this chapter has been that the camera is an ideal pinhole camera. Thus the projected point p in the image is on the line that joins point P in space and camera center O_c. However, real cameras do not have pinholes, but lens. This assumption will not hold for real cameras because lens will introduce certain amount of distortion in the image. The distortion makes the projected point to appear at a slightly different position on the image. The following expression is a simple first-order model that captures the distortion introduced by lens:

$$\begin{cases} \bar{a} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix} & \text{pinhole projection} \\ \bar{b} = \begin{bmatrix} b_x \\ b_y \end{bmatrix} = \bar{a}(1 + k_c\|\bar{a}\|^2) & \text{radial distortion} \\ \bar{u} = \begin{bmatrix} u_x \\ u_y \end{bmatrix} = \begin{bmatrix} f_x b_x + c_x \\ f_y b_y + c_y \end{bmatrix} & \text{pixel coordinates} \end{cases} \tag{2.17}$$

where k_c is called *radial distortion factor.* This model is also called first-order symmetric radial distortion model [11, 92, 91] ("symmetric" because the amount of distortion is directly related to the distance of the point to the principle point c). Observe that the systems (2.17) and (2.15) are equivalent when $k_c = 0$ (no distortion). The parameters c_x, c_y, f_x, f_y, k_c are called *camera intrinsic parameters.*

Therefore, if the position of the point P is known in camera reference frame, one can calculate its projection onto the image plane given the intrinsic camera parameters c_x, c_y, f_x, f_y and k_c. That is known as the *direct projection operation* and denoted by $\overline{u} = \Pi(\overline{X})$. However, most 3D vision applications require to solve the "inverse problem". That is to map pixel coordinates $\overline{u}$ to 3D world coordinates $[X \;\; Y \;\; Z]^T$. The only nontrivial aspect of this inverse map computation is in computing the vector $\overline{a}$ from $\overline{b}$. This step corrects the image measurements to those that would have been obtained under a perfect "pinhole" camera model and is named distortion compensation. For relatively small distortion, the distortion compensation can be very well approximated by the following equation

$$\overline{a} \approx \frac{\overline{b}}{1 + k_c \left\| \frac{\overline{b}}{1+k_c \|\overline{b}\|^2} \right\|^2} \tag{2.18}$$

Therefore, once the intrinsic camera parameters c_x, c_y, f_x, f_y,k_c are given, the projective ray joining camera center O_c and point P in space can be inversely computed from the pixel coordinates of projected point p on the image plane.

The procedure to recover the intrinsic camera parameters is called *camera calibration.* A standard method is to acquire an image of a known 3D object and search for the set of parameters that best matches the computed projection of the object with the observed projection on the image. The reference object is also called *calibration rig.* Since the camera parameters are inferred from image measurements, this approach is also called *visual calibration.* This technique was originally presented by Tsai [92, 91] and Brown [11]. An algorithm for estimation was proposed by Abdel-Aziz and Karara [1].

Figure 2.3 shows an example of calibration image when using a planar rig (checker

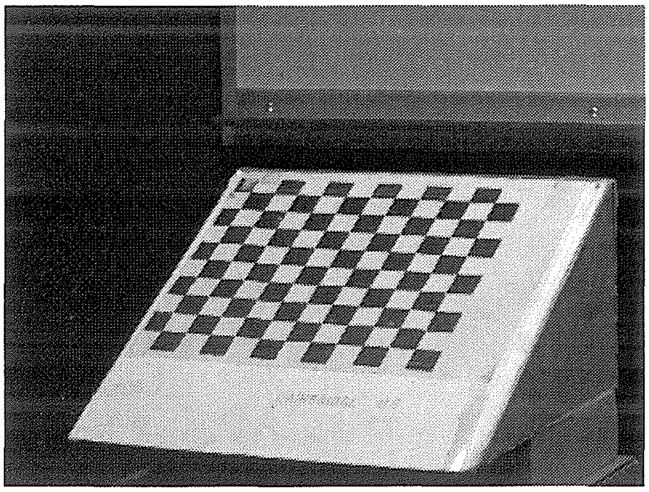


Figure 2.3: **Example of Camera Calibration Image** with a planar calibration rig (checker board pattern).

board pattern).

Notice that although the geometry of the calibration rig is known (i.e., the mutual positions of the grid corners in space), its absolute location with respect to the camera is unknown. Therefore, before applying the set of equations (2.17) to compute the image projection of every corner in the structure, it is necessary to find their 3D coordinates in the camera reference frame $\mathcal{F}$. For this purpose, a reference frame is first attached to the rig (called the object frame), in which all the corners P_i ($i = 1, \ldots, N$) are represented with the known coordinates $\overline{X}_o^i$. This set of vectors is known since the intrinsic rig structure is known. Then, the coordinate vector $\overline{X}_c^i$ of P_i in the camera frame $\mathcal{F}$ is related to $\overline{X}_o^i$ through a rigid motion transformation:

$$\forall i = 1, \ldots, N \qquad \overline{X}_c^i = \mathbf{R}_c \overline{X}_o^i + \overline{T}_c \tag{2.19}$$

where $\mathbf{R}_c$ and $\overline{T}_c$ define the pose of the calibration rig with respect to the camera. See Figure 2.4.

More unknown parameters $\mathbf{R}_c$ and $\overline{T}_c$ have been added to the calibration problem. Those parameters are called *extrinsic camera parameters* since they depend on the pose of the calibration pattern relative to the camera.

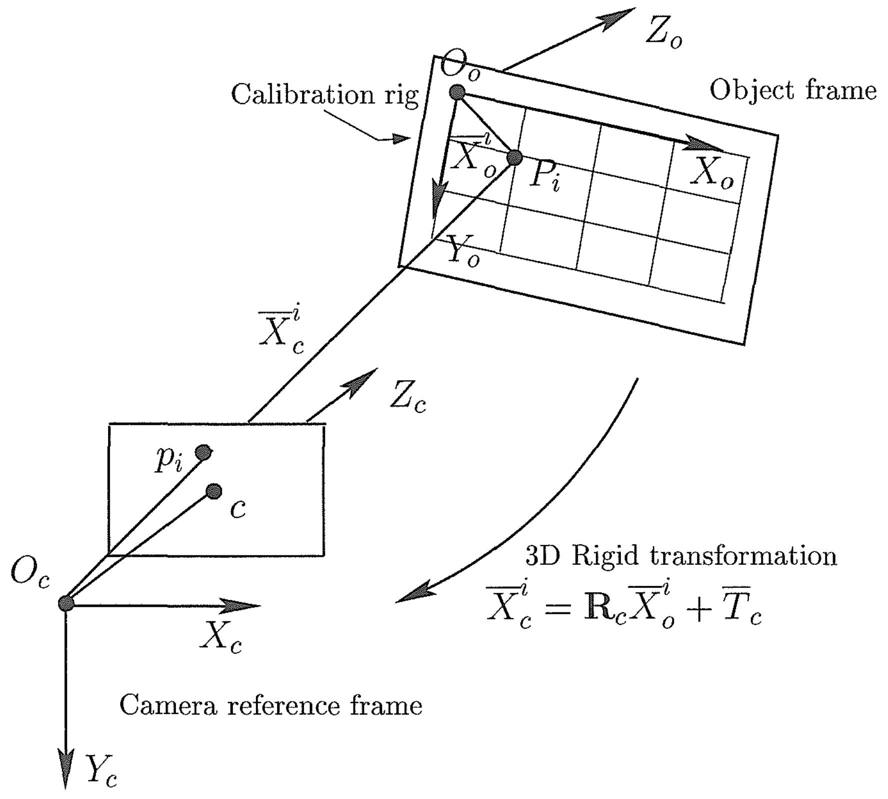


Figure 2.4: **Camera Calibration System.** This figure illustrates the case where a planar rig is used for calibration.

Let $\overline{\Omega}_c$ be the rotation vector associated to the rotation matrix $\mathbf{R}_c$ (see equation 2.4). Then, the complete set of unknowns to solve, or degree of freedom (DOF), for camera calibration is

- Focal length: f_x, f_y (2 DOF)
- Principal point coordinates: c_x, c_y (2 DOF)
- Radial distortion factor: k_c (1 DOF)
- Calibration rig pose: $\overline{\Omega}_c$, $\overline{T}_c$ (6 DOF)

Let p_i ($i = 1, \ldots, N$) be the observed image projections of the rig points P_i and $\overline{u}_i = [u_ix \quad u_iy]^T$ be their respective pixel coordinates. Experimentally, the points p_i are detected on the image using the standard Harris corner finders [29].

Then the estimation process consists of finding the set of calibration unknowns (extrinsic and intrinsic) that minimizes the reprojection error. Therefore, the solution to this problem may be written as

$$\{f_x, f_y, c_x, c_y, k_c, \overline{\Omega}_c, \overline{T}_c\} = \arg\min \sum_{i=1}^{N} \left\| \overline{u}_i - \Pi(\mathbf{R}_c \overline{X}_o^i + \overline{T}_c) \right\|^2 \tag{2.20}$$

where $\mathbf{R}_c = e^{(\overline{\Omega}_c \wedge)}$, $\Pi(\cdot)$ is the image projection operator defined in equation 2.17 (function of the intrinsic parameters f_x, f_y, c_x, c_y and k_c) and $\|\cdot\|$ is the standard distance norm in pixel unit. This nonlinear optimization problem may be solved using standard gradient descent technique.

2.6 Feature Detection and Tracking

Given two images captured by the camera at different locations, it is essential to find out the correspondent points p and p' projected from the same point P in space onto the two images, respectively. In the other words, it is necessary to recognize the point p (observed in the first image) on the second image as p', which is the so-called correspondence problem. However the correspondence problem is not always easy to solve for any kind of points. A point-feature is defined as a point that can be easily recognized from one frame to the other. In this section, we describe a method to automatically detect and track point-features in a sequence of images. The basic technique to solve the correspondence problem is *brightness constancy*, which assumes that the brightness surrounding a feature point keeps constant across the image frames in the sequence.

Many algorithms have been proposed to perform this task. We will concentrate on a well-known algorithm proposed by Lucas and Kanade and then refined by Tomasi and Kanade. It relies upon a differential technique, which is therefore effective only when the displacement across frames is small. When this assumption is not satisfied, however, it is possible to apply the same technique to the coarse-to-fine pyramid of images.

2.6.1 Feature Detection

First of all, it is not easy to identify single pixels, so a small neighborhood or a window $\mathcal{W}(x, y)$ is associated to each point with coordinate vector $[x \quad y]^T$ on an image. If a feature window has constant brightness, then it looks like a homogeneous patch and cannot be localized in a different image. If the window has a brightness gradient, then it is possible to localize its correspondence in a different image only in the direction of this gradient, since the image motion normal to the gradient does not modify the brightness pattern of the window patch. This is the well-known *aperture problem.* Therefore, in order to be able to solve the correspondence problem, a feature window should have significant gradients along two independent directions. Such a window is called *reliable window.* Let $I(x, y, t)$ be the brightness of the image at the point $[x \quad y]^T$ at time t, and $\nabla I(x, y, t) = [\frac{\partial I(x,y,t)}{\partial x} \quad \frac{\partial I(x,y,t)}{\partial y}]^T$ be the spatial gradient calculated at that point, then the feature window is reliable if and only if

$$\sigma_{\min}\left(\int_{\mathcal{W}(x,y)} \nabla I(x, y, t)\, \nabla I(x, y, t)^T \, dx\, dy\right) > \tau \qquad (2.21)$$

where $\sigma_{\min}$ denotes the smallest singular value of a matrix. The point p with coordinate vector $[x \quad y]^T$ is defined as a feature point if $\mathcal{W}(x, y)$ is a reliable window. With a preset threshold τ, the point features are those satisfying equation 2.21 and detected by searching through all the pixels in the image.

2.6.2 Feature Tracking

The feature points are tracked to the next frame under the brightness constancy constraint. Let $\overline{v} = [v_x \quad v_y]^T$ be the feature displacement from time t to $t + 1$. The brightness constancy is simply expressed as

$$I(x, y, t) = I(x + v_x, y + v_y, t + 1) \qquad (2.22)$$

Take the first-order Taylor expansion of the right-hand side and the above constraint becomes

$$\frac{\partial I(x,y,t)}{\partial x} v_x + \frac{\partial I(x,y,t)}{\partial y} v_y + \frac{\partial I(x,y,t)}{\partial t} = 0 \tag{2.23}$$

In matrix format, it is

$$\nabla I(x,y,t)^T \overline{v} = -\frac{\partial I(x,y,t)}{\partial t} \tag{2.24}$$

Since this is a scalar function of two unknown variables, it does not define a unique displacement. Right multiply the above equation by ∇I and integrate it over a small window $\mathcal{W}(x,y)$

$$\int\limits_{\mathcal{W}(x,y)} \nabla I(x,y,t)\, \nabla I(x,y,t)^T \, \overline{v} \, dx\, dy = - \int\limits_{\mathcal{W}(x,y)} \nabla I(x,y,t) \, \frac{\partial I(x,y,t)}{\partial t} \, dx\, dy \tag{2.25}$$

the constraint equation is augmented to a rank-two vector equation. Assume the displacement vector is constant at every point in the window $\mathcal{W}(x,y)$, the above integration can be simplified as

$$D\, \overline{v} = e \tag{2.26}$$

where

$$D = \int\limits_{\mathcal{W}(x,y)} \nabla I(x,y,t)\, \nabla I(x,y,t)^T \, dx\, dy \tag{2.27}$$

$$e = - \int\limits_{\mathcal{W}(x,y)} \nabla I(x,y,t) \, \frac{\partial I(x,y,t)}{\partial t} \, dx\, dy \tag{2.28}$$

The displacement can be estimated if D is invertible. This invertibility condition is exactly the same as that defined in equation 2.21, which all the detected feature points satisfy.

In order to further achieve sub-pixel accuracy, an iteration can be performed as

$$\overline{v}^{[k+1]} = \overline{v}^{[k]} + D^{-1}\, e^{[k+1]} \tag{2.29}$$

with

$$\overline{v}^{[0]} = D^{-1}\, e \tag{2.30}$$

$$e^{[k+1]} = \int_{\mathcal{W}(x,y)} \nabla I \; \big(I(x,y,t) - I(x+v_x^{[k]}, y+v_y^{[k]}, t+1)\big)\; dx\, dy \tag{2.31}$$

where $[k]$ means the k^{th} iteration step. In general, the brightness values should be interpolated to get the intensity at location $(x+v_x^{[k]}, y+v_y^{[k]})$ since it is not usually on the pixel grid.

2.6.3 Coarse-to-Fine Strategy for Tracking

All differential techniques fail when the displacement across frames is bigger than a few pixels. One possible solution to this problem is to apply coarse-to-fine strategy:

1. build a pyramid of images by smoothing and downsampling the original images
2. select features at the finest level of definition and propagate the selection to all the other ones
3. track the features at the coarser level to get the displacement
4. propagate that displacement to the finer level and use this new one as an initial step for the sub-pixel iteration
5. start from the coarsest level and repeat the last two steps till reaching the finest level

We have implemented the whole procedure from feature detection to tracking in real time on a Pentium PC.

Chapter 3 A New Geometrical Interpretation of Trilinear Constraints

3.1 Introduction

3.1.1 Motion and Structure Recovery

The recovery of three-dimensional motion and structure from a sequence of image frames is one of the classical problems in computer vision. The problem is generally posed as: given a camera undergoing unknown rigid motion and observing a scene represented by points and/or lines, how can one recover the 3D camera motion and the scene structure from the correspondent projected positions of points and/or lines in multiple images? These projected points and/or lines in images are called features. Section 2.6 described an algorithm to detect point features and find their correspondents between images by tracking. A method to detect line features can be found in [12] and their correspondence between images are usually established by matching [56].

The primary work towards 3D motion recovery can be traced back to Longuet-Higgins's paper [48] in 1981. The concept of *essential matrix* and *epipolar constraint* was first introduced in that paper and a quasi linear algorithm was proposed to recover 3D motion from correspondent point features between a pair of frames (2-views). It was followed by extensive research in the 1980's in motion and structure estimation from 2-views, and numerous new algorithms were developed [93, 105, 108, 99]. Sensitivity to noise was reported in [15, 93] and error analysis was studied in [101]. The properties of the essential matrix were mainly elucidated in the turning of 1980's to 1990's [42, 55, 40]. At this point, all the work is based on the assumption that the camera is calibrated, i.e., the intrinsic parameters of the camera are known and the features can be represented by their normalized homogeneous coordinates. The generalization to the uncalibrated situation was developed in the early 1990's independently by Faugeras et al. [16, 18] and Hartley et al. [30, 34], where the epipolar constraint is represented by the so-called *fundamental matrix.* Detailed analysis of linear and nonlinear techniques with respect to different object functions for estimation of the fundamental matrix can be found in [50]. All of these approaches can be categorized as *batch methods* for motion and structure estimation from 2-views in the sense that each pair of views is not related to the other pairs in the long image sequence. Therefore the estimation from any newly formed pairs of views can only be performed independently. In contrast to batch methods, recursive methods were proposed by Soatto et al. [77, 78] and McLauchlan et al. [58], where the motion parameters change temporally from frame to frame under a dynamical model.

Motion and structure estimation from multiple views (more than 2) was initially studied on line features in 1990 by Spetsakis and Aloimonos [84], who first introduced *trilinear constraints* and the concept of *trifocal tensor* arose from 3-view observations. Weng et al. [102] proposed some linear and nonlinear algorithms for the reconstruction from line features for calibrated cameras. It was later shown to be equally applicable to the uncalibrated case by Hartley [36]. Meanwhile Shashua in his independent work [70] introduced a set of trilinearity conditions concerning the image coordinates of correspondent point features in three views for uncalibrated cameras. At that time, the set of coefficients of trilinear constraints, which is later referred to as "trifocal tensor", was not considered as a tensor, but rather a set of three 3×3 matrices. The notation of tensor was first adopted by Vieville and Luong [95], while the name "trifocal tensor" was first introduced by Hartley [31, 32], who also showed in the same papers that the scene reconstructions from point features and from lines features arise from a common trilinearity framework and share the same trifocal tensor.

In subsequent work, properties of the tensor have been investigated. In particular, Faugeras et al. [19] studied the algebraic and geometric dependency between bilinear and trilinear constraints, Triggs [90] described the mixed covariant-contravariant behavior of the indices and Shashua et al. [72] described the geometry of the homographies encoded by the tensor. Further geometric properties of the tensor were given by Papadopoulo et al. in [62].

To compute the trifocal tensor and further estimate structure and motion, a linear method was first given by Hartley [32] where experimental results of estimation from both point and line features are reported. An iterative algebraic method based on the linear method for estimating the tensor was also proposed by Hartley [33]. Torr et al. [89] developed a nonlinear algorithm for computing the maximum likelihood estimate of the trifocal tensor which is parameterized using six point correspondences. Another parameterization of the tensor was proposed by Faugeras et al. [20] which also led to a nonlinear method for tensor estimation.

Following the study of 3-view geometry, quadrifocal tensor and qualinear constraints were discovered by Triggs [90] for the 4-view motion and structure estimation, and their properties were described in several papers [90, 19, 73, 38]. Then, it was shown by Faugeras and Mourrain [19] that, for a 3D point with its correspondent projections in an arbitrary number of images, there are only three types of algebraic relations between the coordinates of image points: bilinear, trilinear and quadrilinear. It was further proved that quadriliear constraints are redundant since they are algebraically dependent on the trilinear ones [19, 39], and trilinear constraints are algebraically dependent on bilinear ones when the centers of the cameras do not lie on a straight line [39, 53]. Much simpler proofs for these algebraic dependency were shown by Ma et al. [51], who also found the existence of nonlinear constraints for mixed points and lines in the same paper. Both these algebraic relations and the fact that trilinear constraints are necessary for the reconstruction from line features lead us to focus on the study of bilinear and trilinear constraints.

Although numerous linear and nonlinear methods have been proposed for reconstruction from 2-views or 3-views, in presence of image measurement noises of features, what is the optimum strategy? This problem was originally addressed by Weng et al. [102, 100], who proposed the maximum likelihood estimation of motion and structure from 2-views and compared linear algorithms against their proposal. The constraint generated by the maximum likelihood estimator is called *reprojection* *error*. Szeliski et al. [87] also addressed the optimal estimation issue by minimizing the reprojection error based on a linearized model. Kanatani [47] addressed the optimization problem in general, and gave some vision examples of 2-views as particular cases. However, the solutions provided by these algorithms depend on the initial conditions [100] and only guarantee convergence to local minima [100, 87]. Soatto et al. [76] presented a provably convergent algorithm which can lead to global solution under certain conditions without imposing restrictions on initial conditions. Ma et al. [52] then analyzed the sensitivity and robustness of different linear and nonlinear techniques and showed that the optimal estimation using reprojection error function can be practically approximated by two other commonly used methods: minimizing the statistically normalized epipolar constraints, and minimizing the geometrically normalized ones. The optimal strategy, i.e., minimizing the reprojection error, was also applied to the 3-view reconstruction [35].

3.1.2 Motivation to Our Work

Our attention focuses on developing a geometrical understanding of trilinear constraints. Trilinear constraints are linear with respect to both image coordinates and elements of the trifocal tensor. Therefore directly estimating the trifocal tensor from the trilinear constraints produces an efficient linear algorithm for reconstruction. However, as algebraic equations, the trilinear constraints do not have obvious geometrical interpretations. In a noiseless case, the algebraic equations are exactly satisfied by all the points. In the presence of noise, however, the residual errors generated by minimizing the constraint equations may vary dramatically from point to point. Thus, each point carries its own reliability that should be appropriately accounted for when building the cost function to minimize. The philosophy is similar to applying appropriate weights to different observation points. If the weights are not chosen properly, it is very likely that the best estimation will not be achieved. In that sense, one may think that pure algebraic constraints, taken in a least squares fashion, are not reliable to use from a statistical point of view. Other geometrically meaningful algorithms [20, 89, 35] were proposed, but they are more complicated with much higher computational cost, and usually still need the linear method as the first step to converge to the correct solution. Hartley then showed in [33] that directly minimizing the algebraic trilinear constraints without additional weighting coefficients does in practice give satisfying estimation results. To explain and understand this fact is the primary goal of this chapter by providing a geometrical interpretation to the trilinear constraints.

We revisit the trilinear constraints from the point of view of 3D structure depth and show that they are naturally weighted constraints enforcing depth equivalence of the point in space. A clear geometrical interpretation of the weight function is presented. We show theoretically, as well as experimentally, that the standard trilinear algebraic cost is almost equivalent to the optimally weighted depth matching cost. An extension to the uncalibrated case is also given. It provides a criterion to select a set of independent trilinear constraints. In addition, we propose a reliable scale propagation scheme that is insensitive to the measurement noise in the image point coordinates.

This chapter is organized as follows. Section 3.2 gives the background and brings up the main problem that this chapter tries to solve. The geometrical interpretation follows in the next section. In section 3.4 the scale propagation scheme is proposed. Experiments are reported in section 3.5. We briefly extend the work to the uncalibrated case in section 3.6 which is followed by the conclusion.

The majority work presented in this chapter was published in [22].

3.2 Background: Trilinear Constraints

In this chapter we adopt the standard tensorial notations for trilinearities. The coordinates of a point are specified with superscript and named contravariant vector, i.e., $\bar{\mathbf{x}} = [x^1 \ \ x^2 \ \ \ldots \ \ x^n]^T$. An element in its dual space is called a covariant vector and represented by subscripts, i.e., $\bar{\mathbf{l}} = [l_1 \ \ l_2 \ \ \ldots \ \ l_n]^T$. In a similar manner, the ij-th element of a matrix is denoted by a^i_j, where i and j represent the indices of its row and column respectively, and the triply indexed quantity $\mathcal{T}_i^{jk}$ is named trifocal tensor. We also use the usual covariant-contravariant summation convention in this chapter: any index repeated in covariant and contravariant forms implies a summation over the range of index values, i.e., $x^i l_i = x^1 l_1 + x^2 l_2 + \ldots + x^n l_n = \overline{\mathbf{x}}^T \overline{\mathbf{l}}$.

In this chapter we consider point features only. Suppose a point P with homogeneous coordinates $\overline{\mathbf{X}} = [X \;\; Y \;\; Z \;\; 1]^T$ in $\mathcal{P}^3$ is imaged in three views as point p, p' and p'', respectively (see Figure 3.1). The corresponding projected points $p \leftrightarrow p' \leftrightarrow p''$ are denoted by the homogeneous coordinate vectors as $\overline{\mathbf{x}} \leftrightarrow \overline{\mathbf{x}}' \leftrightarrow \overline{\mathbf{x}}''$, and the camera projections for the three views are described as

$$Z\overline{\mathbf{x}} = \mathbf{P}\overline{\mathbf{X}} \tag{3.1}$$

$$Z'\overline{\mathbf{x}}' = \mathbf{P}'\overline{\mathbf{X}} \tag{3.2}$$

$$Z''\overline{\mathbf{x}}'' = \mathbf{P}''\overline{\mathbf{X}} \tag{3.3}$$

It is well known that the camera projection matrices can be reconstructed from multiple views only up to a 3D projective transformation [32]. Therefore without loss of generality it can be assumed that $\mathbf{P} = [\mathbf{I}_3 \;\; \mathbf{0}_{3\times 1}]$. To save readers the trouble of counting primes, we denote the camera matrix $\mathbf{P}'$ by $\mathbf{P}' = [a^i_j] = [\mathbf{A} \;\; \overline{\mathbf{a}}_4]$, where a^i_j are the elements of $\mathbf{P}'$, $\mathbf{A}$ is a 3×3 matrix denoting the left three columns of $\mathbf{P}'$ and the vectors $\overline{\mathbf{a}}_j$ are the j-th columns of $\mathbf{P}'$. The ranges for the indices i and j are $i = 1, \ldots, 3$ and $j = 1, \ldots, 4$. Similarly the camera matrix $\mathbf{P}''$ is denoted by $\mathbf{P}'' = [b^i_j] = [\mathbf{B} \;\; \overline{\mathbf{b}}_4]$.

To recover the camera matrices, it is necessary to find out what constraints there are on the corresponding image points. The three back-projected rays from the camera optical centers to image points in each view must all meet in a single point in space, the 3D point P that projects to the corresponding points in the three images. Since in general three arbitrary lines in space do not intersect at one point, this incidence condition provides a genuine constraint on sets of corresponding points. However, in $\mathcal{P}^3$ lines do not have a convenient representation . Therefore, to explore the constraint, it is easier to represent the back-projected rays via the intersection of planes in $\mathcal{P}^3$.

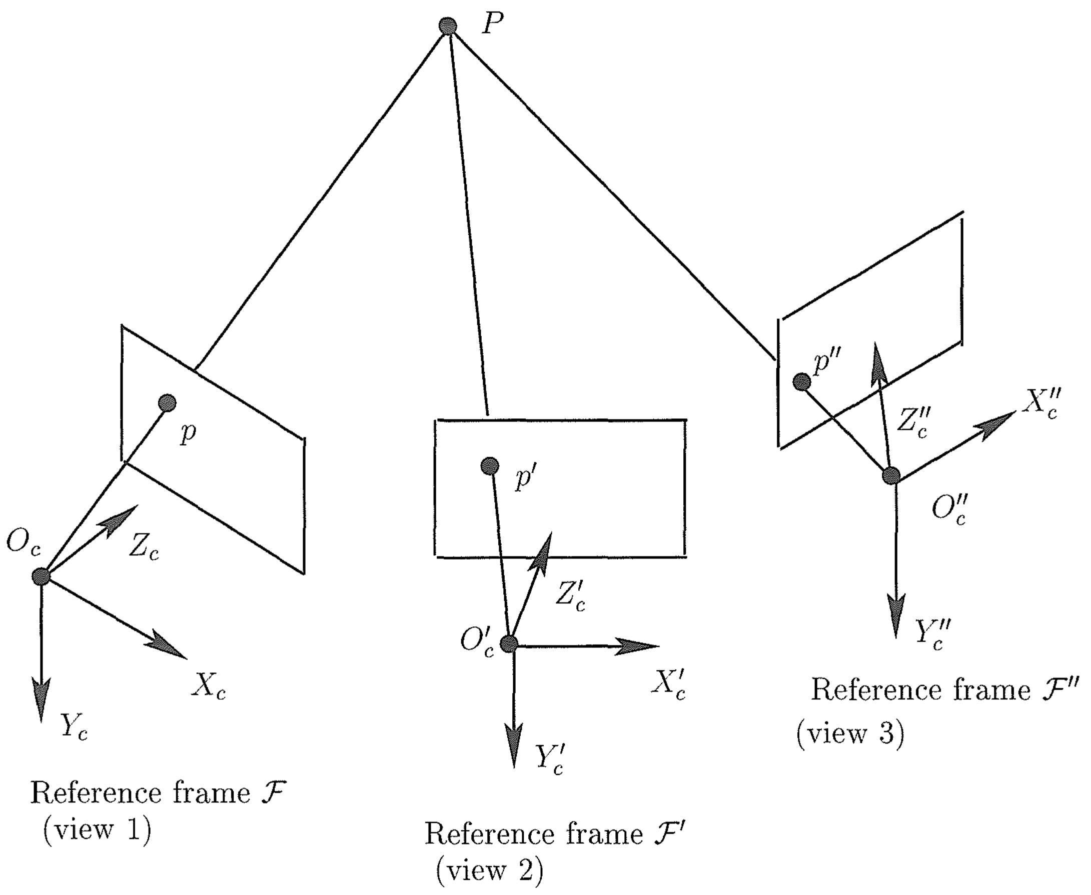


Figure 3.1: **Geometry of 3-View Projection.** A point P is projected in three views as point p, p' and p'', respectively.

Let us first consider one view. Given a point on the image plane $\mathcal{P}^2$ with homogeneous coordinates $\bar{\mathbf{x}} = [x^1 \ \ x^2 \ \ x^3]^T$, there are infinite lines passing through this point on the image plane. Among these lines, three canonical ones that have only one variable in each of their coordinates are

$$\bar{\mathbf{l}}_1 = \begin{bmatrix} x^3 \\ 0 \\ -x^1 \end{bmatrix} \qquad \bar{\mathbf{l}}_2 = \begin{bmatrix} 0 \\ x^3 \\ -x^2 \end{bmatrix} \qquad \bar{\mathbf{l}}_3 = \begin{bmatrix} x^2 \\ -x^1 \\ 0 \end{bmatrix} \tag{3.4}$$

All other lines passing through $\bar{\mathbf{x}}$ are linear combinations of any two lines from these three.

A line $\bar{\mathbf{l}}$ in $\mathcal{P}^2$ can be back-projected to $\mathcal{P}^3$ as a plane containing the camera center and the line. If the camera has projection matrix $\mathbf{P}$, the back-projected plane is $\overline{\boldsymbol{\pi}} = \mathbf{P}^T \bar{\mathbf{l}}$. Therefore, the planes from the three lines $\bar{\mathbf{l}}_1$, $\bar{\mathbf{l}}_2$ and $\bar{\mathbf{l}}_3$ are $\overline{\boldsymbol{\pi}}_1 = \mathbf{P}^T \bar{\mathbf{l}}_1$, $\overline{\boldsymbol{\pi}}_2 = \mathbf{P}^T \bar{\mathbf{l}}_2$ and $\overline{\boldsymbol{\pi}}_3 = \mathbf{P}^T \bar{\mathbf{l}}_3$, respectively. Figure 3.2 illustrates the geometrical relationship of these three lines and planes.

For the first view with the projection matrix $\mathbf{P} = [\mathbf{I}_3 \quad \mathbf{0}_{3\times 1}]$, the three planes constructed from image point $\bar{\mathbf{x}} = [x^1 \quad x^2 \quad x^3]^T$ are

$$\overline{\boldsymbol{\pi}}_1 = \begin{bmatrix} x^3 \\ 0 \\ -x^1 \\ 0 \end{bmatrix} \qquad \overline{\boldsymbol{\pi}}_2 = \begin{bmatrix} 0 \\ x^3 \\ -x^2 \\ 0 \end{bmatrix} \qquad \overline{\boldsymbol{\pi}}_3 = \begin{bmatrix} x^2 \\ -x^1 \\ 0 \\ 0 \end{bmatrix} \tag{3.5}$$

The planes constructed from the point $\bar{\mathbf{x}}' = [x'^1 \quad x'^2 \quad x'^3]^T$ on the second image plane with projection matrix $\mathbf{P}' = [a^i_j]$ are

$$\overline{\boldsymbol{\pi}}'_1 = \begin{bmatrix} x'^3 a^1_1 - x'^1 a^3_1 \\ x'^3 a^1_2 - x'^1 a^3_2 \\ x'^3 a^1_3 - x'^1 a^3_3 \\ x'^3 a^1_4 - x'^1 a^3_4 \end{bmatrix} \qquad \overline{\boldsymbol{\pi}}'_2 = \begin{bmatrix} x'^3 a^2_1 - x'^2 a^3_1 \\ x'^3 a^2_2 - x'^2 a^3_2 \\ x'^3 a^2_3 - x'^2 a^3_3 \\ x'^3 a^2_4 - x'^2 a^3_4 \end{bmatrix} \qquad \overline{\boldsymbol{\pi}}'_3 = \begin{bmatrix} x'^2 a^1_1 - x'^1 a^2_1 \\ x'^2 a^1_2 - x'^1 a^2_2 \\ x'^2 a^1_3 - x'^1 a^2_3 \\ x'^2 a^1_4 - x'^1 a^2_4 \end{bmatrix} \tag{3.6}$$

Similarly, the planes constructed from the point $\bar{\mathbf{x}}'' = [x''^1 \quad x''^2 \quad x''^3]^T$ in the third image with projection matrix $\mathbf{P}'' = [b^i_j]$ are

$$\overline{\boldsymbol{\pi}}''_1 = \begin{bmatrix} x''^3 b^1_1 - x''^1 b^3_1 \\ x''^3 b^1_2 - x''^1 b^3_2 \\ x''^3 b^1_3 - x''^1 b^3_3 \\ x''^3 b^1_4 - x''^1 b^3_4 \end{bmatrix} \qquad \overline{\boldsymbol{\pi}}''_2 = \begin{bmatrix} x''^3 b^2_1 - x''^2 b^3_1 \\ x''^3 b^2_2 - x''^2 b^3_2 \\ x''^3 b^2_3 - x''^2 b^3_3 \\ x''^3 b^2_4 - x''^2 b^3_4 \end{bmatrix} \qquad \overline{\boldsymbol{\pi}}''_3 = \begin{bmatrix} x''^2 b^1_1 - x''^1 b^2_1 \\ x''^2 b^1_2 - x''^1 b^2_2 \\ x''^2 b^1_3 - x''^1 b^2_3 \\ x''^2 b^1_4 - x''^1 b^2_4 \end{bmatrix} \tag{3.7}$$

All these nine planes should contain the 3D point P. In general, any three planes meet at a single point, but this point do not usually locate in another arbitrary plane. Therefore, the condition to let all nine planes meet at point P is equivalent to enforcing that any four planes of the nine intersect at this point. These four planes

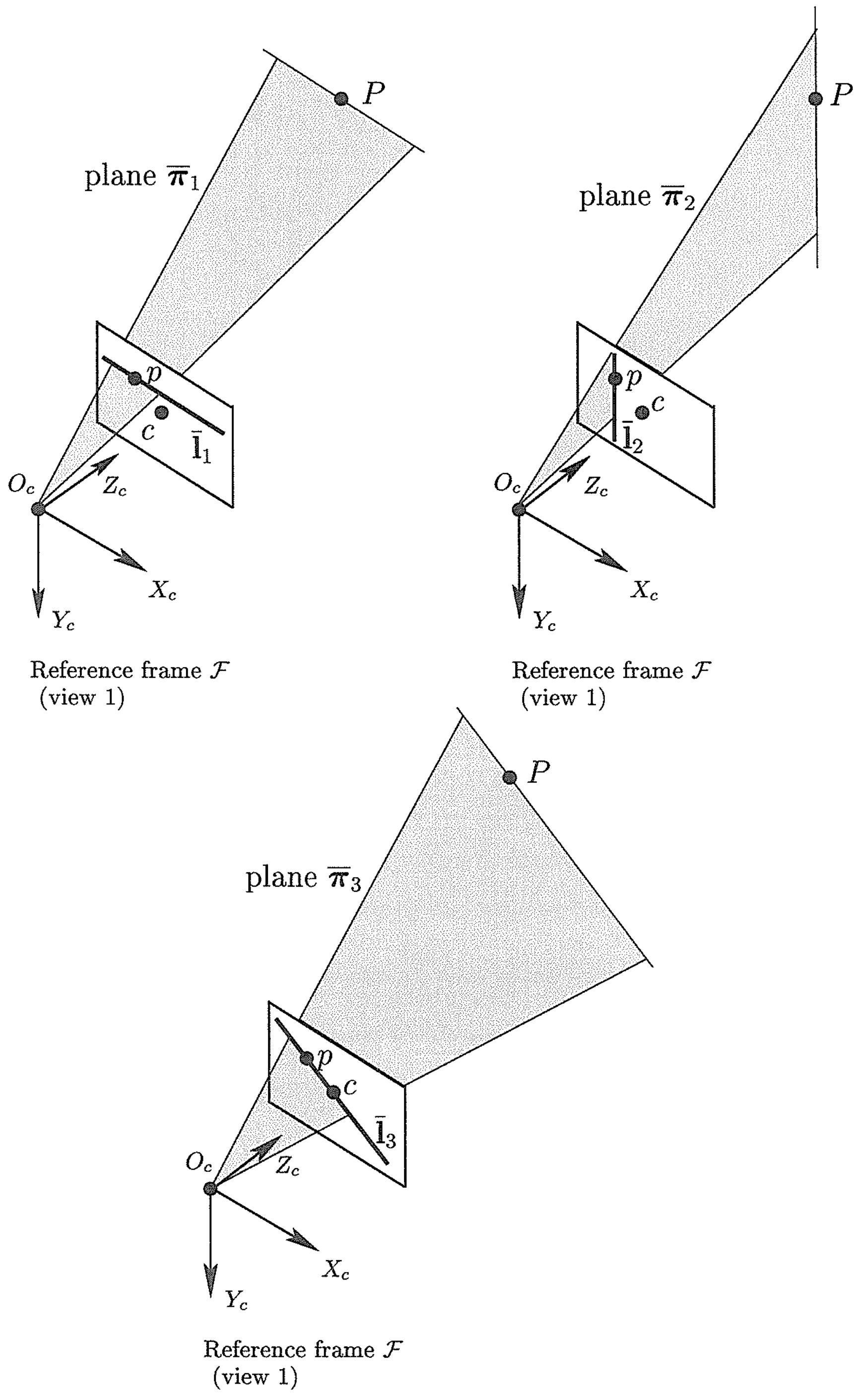


Figure 3.2: **Lines and Planes.** Three lines $\bar{l}_1$, $\bar{l}_2$, $\bar{l}_3$ which pass through point p in the first image and their corresponding planes $\overline{\pi}_1$, $\overline{\pi}_2$, $\overline{\pi}_3$ are shown in three plots, respectively.

should be chosen in such a way that two are generated from one same view and the other two from the rest two views, respectively. All other combinations are either trivial or not using three views. Without loss of generality, it can be assumed that two planes are from the first view. Any two planes from $\overline{\pi}_1$,$\overline{\pi}_2$ and $\overline{\pi}_3$ intersect at the same line, the projection ray joining camera center and image point $\overline{\mathbf{x}}$. Therefore, the set of four planes can be chosen as $\overline{\pi}_1, \overline{\pi}_2, \overline{\pi}'_r, \overline{\pi}''_s$, where $r = 1, 2$ or 3 and $s = 1, 2$ or 3. Geometrically, the plane $\overline{\pi}'_r$ meets the plane $\overline{\pi}''_s$ at a line $\overline{\pi}'_r \wedge \overline{\pi}''_s$ in $\mathcal{P}^3$ and $\overline{\pi}_1$ meets $\overline{\pi}_2$ at the projection ray of the first view as $\overline{\pi}_1 \wedge \overline{\pi}_2$. The constraint of four planes intersecting at one point is equivalent to that the projection ray $\overline{\pi}_1 \wedge \overline{\pi}_2$ meets the line $\overline{\pi}'_r \wedge \overline{\pi}''_s$ at one point (see Figure 3.3). This constraint can be expressed algebraically as

$$\det(\begin{bmatrix} \overline{\pi}_1 & \overline{\pi}_2 & \overline{\pi}'_r & \overline{\pi}''_s \end{bmatrix}) = 0 \tag{3.8}$$

where det() is the determinant of a square matrix.

Choosing different set of $\overline{\pi}'_r$ and $\overline{\pi}''_s$ ($r \in (1, 2, 3)$ and $s \in (1, 2, 3)$), and expanding the constraint 3.8 results in a set of nine algebraic equations, which are called *trilinear constraints* or *trilinearities* [32]

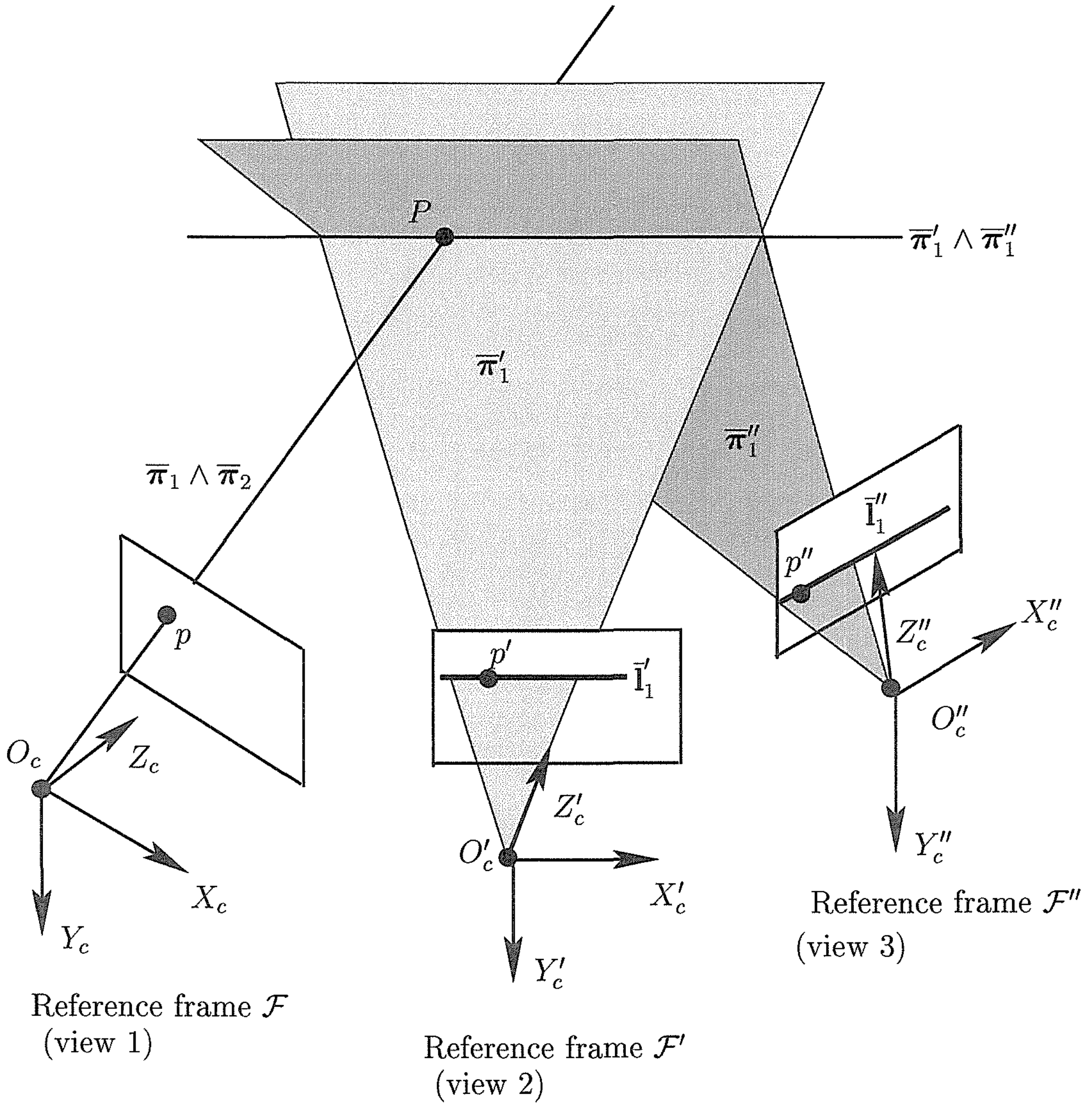
$$\mathcal{E}1^{ijlm} = x^k(x'^i x''^m \mathcal{T}_k^{jl} - x'^j x''^m \mathcal{T}_k^{il} - x'^i x''^l \mathcal{T}_k^{jm} + x'^j x''^l \mathcal{T}_k^{im}) = 0 \tag{3.9}$$

$$1 \leq i < j \leq 3 \qquad 1 \leq l < m \leq 3$$

where the trifocal tensor is defined as

$$\mathcal{T}_i^{jk} = a_i^j \, b_4^k \; - \; a_4^j \, b_i^k \tag{3.10}$$

The trilinear cost functions $\mathcal{E}1^{ijlm}$ are conveniently defined here for future reference. Different derivations of trilinear constraints can be found in [32, 72, 19]. Notice that these constraints are not independent. Although there are three back-projected planes $\overline{\pi}'_r$ for the second view, only two of them are independent, and the same is true for the planes $\overline{\pi}'_s$ in the third view. Hence, there are 4 linearly independent trilinearities in equation set 3.9.



P
$\overline{\pi}'_1 \wedge \overline{\pi}''_1$
$\overline{\pi}'_1$
$\overline{\pi}''_1$
$\overline{\pi}_1 \wedge \overline{\pi}_2$
$\bar{\mathbf{l}}''_1$
p''
X''_c
Z''_c
O''_c
Y''_c
p
p'
$\bar{\mathbf{l}}'_1$
O_c
Z_c
X_c
Y_c
Z'_c
O'_c
X'_c
Y'_c
Reference frame $\mathcal{F}''$ (view 3)
Reference frame $\mathcal{F}$ (view 1)
Reference frame $\mathcal{F}'$ (view 2)

Figure 3.3: **Trilinear Constraint**. The projection ray in the first view $\overline{\pi}_1 \wedge \overline{\pi}_2$ should meet the intersection line $\overline{\pi}'_1 \wedge \overline{\pi}''_1$ at point P in space.

Every term in the trilinear constraints involves a coordinate from each of the three image points. The camera projection matrices are embedded in the trifocal tensor, and the trilinear constraints are linear in each entry of the tensor. The state-of-the-art technique to recover the projection matrices is to estimate the trifocal tensor first [32] and then retrieve the projection matrices from the tensor. Details of retrieving the projection matrices are described in [35].

In real image sequences, due to measurement noises on the image coordinates, equations 3.9 will not be exactly satisfied by all the points. A linear technique is to minimize the residue errors of the constraints over the trifocal tensor $\mathcal{T}_i^{jk}$. The 27 entries of the trifocal tensor are not arbitrary but satisfy certain constraints. A trifocal tensor is said to be "consistent" or "satisfying all internal constraints" if there exist three camera matrices $\mathbf{P} = [\mathbf{I}_3 \;\; \mathbf{0}_{3\times1}]$, $\mathbf{P}'$ and $\mathbf{P}''$ such that $\mathcal{T}_i^{jk}$ corresponds to the three camera matrices according to equation 3.10. The internal constraints can be enforced by parameterizing the tensor with minimal 18 parameters [20, 62, 89] or solving a constrained linear algorithm [33]. The internal constraints and properties of the trifocal tensor are not the main concern of this chapter, for more details, please check the related references. Here we generally describe the internal constraints for $\mathcal{T}_i^{jk}$ as $S_{\mathcal{T}}$. The linear method to identify the unknowns [32, 33] is

$$\{\mathcal{T}_i^{jk}\}^* = \arg \min_{\mathcal{T}_i^{jk} \in S_{\mathcal{T}}} \sum_{P_i} \sum_{ijlm} (\mathcal{E}1^{ijlm})^2 \tag{3.11}$$

The term $\sum_{P_i}$ means summation over all the points P_i and the term $\sum_{ijlm}$ is the summation over 4 independent trilinearities with proper choices of i, j, l, m as in equation 3.9. The trilinear constraints are pure algebraic equations satisfied by the noiseless corresponding points in three views. No geometrical interpretation has been established for them in literature. In the presence of measurement noises, the residue errors generated by such algebraic constraints are not predictable and may vary dramatically from point to point.

The best estimate of trifocal tensor may be obtained from the maximum likelihood solution which minimizes *reprojection error*. Given a set of point correspondences

$\overline{\mathbf{x}}_i \leftrightarrow \overline{\mathbf{x}}'_i \leftrightarrow \overline{\mathbf{x}}''_i$ in three views, the reprojection error is defined as

$$\sum_{P_i} d(\overline{\mathbf{x}}_i, \hat{\overline{\mathbf{x}}}_i)^2 + d(\overline{\mathbf{x}}'_i, \hat{\overline{\mathbf{x}}}'_i)^2 + d(\overline{\mathbf{x}}''_i, \hat{\overline{\mathbf{x}}}''_i)^2 \tag{3.12}$$

where the points $\hat{\overline{\mathbf{x}}}_i \leftrightarrow \hat{\overline{\mathbf{x}}}'_i \leftrightarrow \hat{\overline{\mathbf{x}}}''_i$ exactly satisfy the trilinear constraints 3.9 for the estimated trifocal tensor and are the true projected positions of 3D point P_i in three views, $d(\overline{x}, \overline{y})$ is the Euclidean distance between two points $\overline{x}$ and $\overline{y}$ on the image plane. To minimize the reprojection error, one needs to introduce further variables $\overline{X}_i = [X_i \;\; Y_i \;\; Z_i]^T$ corresponding to each 3D point P_i. The minimization is performed over both the unknown structure and the unknown tensor:

$$\{\mathcal{T}_i^{jk}, \overline{X}_1, \ldots, \overline{X}_N\}^* = \arg \min_{\mathcal{T}_i^{jk} \in S_{\mathcal{T}}, \overline{X}_1, \ldots, \overline{X}_N} \sum_{P_i} d(\overline{\mathbf{x}}_i, \hat{\overline{\mathbf{x}}}_i)^2 + d(\overline{\mathbf{x}}'_i, \hat{\overline{\mathbf{x}}}'_i)^2 + d(\overline{\mathbf{x}}''_i, \hat{\overline{\mathbf{x}}}''_i)^2 \tag{3.13}$$

The reprojection error has clear statistical (maximum likelihood) and geometrical (error measured in image) interpretations. However, on the other hand, it is nonlinear and the minimization needs to be done iteratively in a much larger parameter space, which leads to much higher computational cost compared with the linear method from equation 3.11. In addition, the nonlinear minimization of reprojection error should be initialized with the estimate from linear method since otherwise it may converge to any one of many local minima. In [33], Hartley showed that surprisingly the quasi linear method actually gives results very close to those from the nonlinear method. The performance of the algebraic trilinear constraints is obviously beyond the scope that pure algebraic equations can reach. Here arises an interesting question: do the trilinear constraints have any geometrical meaning? Except for their algebra, no work along this line has been shown in any literature yet. This is what we are going to explore in this chapter.

In the following sections, we assume that the camera is calibrated. The camera matrices are normalized projection matrices as $\mathbf{P}' = [a_j^i] = [\mathbf{A} \;\; \overline{\mathbf{a}}_4] = \mathbf{G}'$ and $\mathbf{P}'' = [b_j^i] = [\mathbf{B} \;\; \overline{\mathbf{b}}_4] = \mathbf{G}''$ and $\overline{\mathbf{x}} \leftrightarrow \overline{\mathbf{x}}' \leftrightarrow \overline{\mathbf{x}}''$ are normalized image coordinates. Under this assumption, the matrix $\mathbf{A}$ and the vector $\overline{\mathbf{a}}_4$ represent the rotation and translation respectively from the first camera reference frame to the second one. Similarly, $\mathbf{B}$ and $\overline{\mathbf{b}}_4$ are the rotation and translation from the first camera to the third. The extension to the uncalibrated camera will be briefly discussed in section 3.6.

3.3 Geometrical Interpretation of Trilinear Constraints

We investigate the problem by rederiving the trilinear constraints from three-dimensional structure point of view.

Claim 1: The trilinear constraints in equations 3.9 are equivalent to depth matching constraints multiplied by geometrically meaningful weight.

Let us decompose the triple views into two pairs of views, (view 1 and 2) and (view 1 and 3). The depth Z of the 3D point P in the first camera reference frame can be estimated from both pairs of views. Denote the depth Z reconstructed from the first pair by Z_a, and from the second pair by Z_b. In the absence of measurement noises, both pairs of views should give same estimates of depth, $Z_a = Z_b$. To obtain Z_a and Z_b, consider the pair (view 1 and 2) first. The camera projection of the first view is $Z_a \overline{\mathbf{x}} = \mathbf{P} \overline{\mathbf{X}}$ with $\mathbf{P} = [\mathbf{I}_3 \quad \mathbf{0}_{3\times1}]$, from which we can write the coordinate vector of 3D point P as $\overline{\mathbf{X}} = [Z_a \overline{\mathbf{x}}^T \quad 1]^T$. Substituting this expression in the second camera projection equation gives

$$Z' \overline{\mathbf{x}}' = \mathbf{P}' \overline{\mathbf{X}} = \begin{bmatrix} \mathbf{A} & \overline{\mathbf{a}}_4 \end{bmatrix} \begin{bmatrix} Z_a \overline{\mathbf{x}} \\ 1 \end{bmatrix} \tag{3.14}$$

From this relation, the depth of the point $\mathbf{P}$ in these first two views, Z_a and Z', can be estimated by solving the following so-called triangulation equations

$$\begin{bmatrix} a^1_k x^k & -x'^1 \\ a^2_k x^k & -x'^2 \\ a^3_k x^k & -x'^3 \end{bmatrix} \begin{bmatrix} Z_a \\ Z' \end{bmatrix} = - \begin{bmatrix} a^1_4 \\ a^2_4 \\ a^3_4 \end{bmatrix} \tag{3.15}$$

From the other pair of views, (view 1 and 3), a similar triangulation equations can be written as

$$\begin{bmatrix} b^1_k x^k & -x''^1 \\ b^2_k x^k & -x''^2 \\ b^3_k x^k & -x''^3 \end{bmatrix} \begin{bmatrix} Z_b \\ Z'' \end{bmatrix} = - \begin{bmatrix} b^1_4 \\ b^2_4 \\ b^3_4 \end{bmatrix} \tag{3.16}$$

Considering all three views together, we may enforce the depth of the point from either pair of views to be identical:

$$\mathcal{E}2 = Z_a - Z_b = 0 \tag{3.17}$$

Equation 3.17 is called *depth matching constraint.* We are very aware that such depth matching constraint in 3D space is not statistically optimal for motion estimation compared with the reprojection error function 3.12, neither is the linear method shown in equation 3.15 and 3.16 for triangulation [37]. However, we can modify the constraint to a reasonable and reliable one by applying the probabilistically optimal weight as shown later in equation 3.25. The interpretation of the trilinear constraints will be based on that optimally weighted depth matching constraints.

Each triangulation set has 3 equations, one of them is redundant. Therefore, taking any two (i,j) from equations 3.15 and any two (l,m) from equations 3.16 to compute Z_a and Z_b, the depth matching constraint becomes

$$0 = \mathcal{E}2^{ijlm} = \frac{N_{Z_a}}{D_{Z_a}} - \frac{N_{Z_b}}{D_{Z_b}} \tag{3.18}$$

$$= \frac{\begin{bmatrix} x'^j & -x'^i \end{bmatrix} \begin{bmatrix} a_4^i \\ a_4^j \end{bmatrix}}{\begin{bmatrix} -x'^j & x'^i \end{bmatrix} \begin{bmatrix} a_k^i x^k \\ a_k^j x^k \end{bmatrix}} - \frac{\begin{bmatrix} x''^m & -x''^l \end{bmatrix} \begin{bmatrix} b_4^l \\ b_4^m \end{bmatrix}}{\begin{bmatrix} -x''^m & x''^l \end{bmatrix} \begin{bmatrix} b_k^l x^k \\ b_k^m x^k \end{bmatrix}}$$

For convenience N_{Z_a} and D_{Z_a} denote the numerator and denominator in the expression of Z_a and similarly for Z_b. For simplicity we do not specify superscript ij and lm for them. Different choices of (i, j) and (l, m) result in nine different matching constraints. By multiplying the product of the two denominators $D_{Z_a} D_{Z_b}$ on both sides of equation 3.18, we get again the same trilinear constraints shown in equation 3.9. That is

$$\mathcal{E}1^{ijlm} = \mathcal{E}2^{ijlm} D_{Z_a} D_{Z_b} = 0 \tag{3.19}$$

Therefore, the trilinear cost function $\mathcal{E}1^{ijlm}$ can be viewed as generated from the depth matching constraint $\mathcal{E}2^{ijlm}$ but weighted by the factor

$$\omega_{\text{tri}}{}^{ijlm} = D_{Z_a} D_{Z_b} \tag{3.20}$$

The importance of the weight ω_{tri} (subscript "tri" represents "trilinear") only depends on its absolute value. Therefore, we can always enforce the same sign on D_{Z_a} and D_{Z_b} so that ω_{tri} is positive. To understand the geometrical meaning of ω_{tri}, let us start from its factor D_{Z_a}. The normalized coordinate vector $\overline{\mathbf{x}} = [X/Z \;\; Y/Z \;\; 1]$ in $\mathcal{P}^2$ can be viewed as a vector in 3D Euclidean space from camera center to projected image point. The vector $\mathbf{A}\overline{\mathbf{x}}$ transforms $\overline{\mathbf{x}}$ from view 1 into view 2 so that it is represented in the same second camera reference frame as vector $\overline{\mathbf{x}}'$. Taking their (i, j) components is equivalent to projecting the vectors onto the $i - j$ plane (for example, $1 - 3$ plane is the $X - Z$ plane) in the second camera reference frame. Without considering the order of the three axes, we can describe these projected vectors in view 2 as

$$\overline{\mathbf{x}}_{\text{proj}} = \begin{bmatrix} a^i_k x^k \\ a^j_k x^k \\ 0 \end{bmatrix} \qquad \overline{\mathbf{x}}'_{\text{proj}} = \begin{bmatrix} x'^i \\ x'^j \\ 0 \end{bmatrix} \tag{3.21}$$

Consequently, taking the (i, j) equations from equations 3.15 for triangulation is to estimate the depth from a 2D triangle by projecting the 3D triangle onto this $i - j$ plane to get rid of one redundant equation.

The value of D_{Z_a} is exactly the norm of the cross product of these two projected vectors:

$$\begin{aligned} |D_{Z_a}| &= \left| \begin{bmatrix} -x'^j & x'^i \end{bmatrix} \begin{bmatrix} a^i_k x^k & a^j_k x^k \end{bmatrix}^T \right| \\ &= ||\overline{\mathbf{x}}'_{\text{proj}} \times \overline{\mathbf{x}}_{\text{proj}}|| \;=\; ||\overline{\mathbf{x}}'_{\text{proj}}||\, ||\overline{\mathbf{x}}_{\text{proj}}|| \sin\theta \end{aligned} \tag{3.22}$$

where θ is the angle between vectors $\overline{\mathbf{x}}'_{\text{proj}}$ and $\overline{\mathbf{x}}_{\text{proj}}$.

Since we project the triangle onto the $i - j$ plane, equation 3.22 contains both the information of 3D structured triangle and the goodness of the choice of $i - j$ as the projection plane. If the projection is ill conditioned, e.g., the $i - j$ plane is perpendicular to the 3D triangle plane, the 3D triangle will be projected only as a line and D_{Z_a} will be zero. If the projected triangle keeps the shape of the 3D triangle, which implies $||\overline{\mathbf{x}}'_{\text{proj}}||$ and $||\overline{\mathbf{x}}_{\text{proj}}||$ have relatively constant values, then the angle θ will represent the confidence or reliability of the triangulation. The term $\sin\theta$ changes from 0 to 1 while the directions of two vectors $\overline{\mathbf{x}}'_{\text{proj}}$ and $\overline{\mathbf{x}}_{\text{proj}}$ change from collinear to perpendicular. The more collinear the two vectors are, the more ill conditioned the depth triangulation is. This is because small noises on image point coordinates that change the orientations of the vectors may cause large errors on the depth estimates. Mathematically when the two vectors approach collinear, $\theta \to 0$, we have $D_{Z_a} \to 0$ from equation 3.22 which consequently leads to $\omega_{\text{tri}} \to 0$. Thus a very small weight ω_{tri} will be applied to this feature point. Same interpretation can be applied to the effect of D_{Z_b} to ω_{tri}. In the study of motion reconstruction from 2-views, Spetsakis and Aloimonos [85] had same intuitive observation that the algebraic epipolar constraint is equivalent to a weighted function of another geometrical constraint, but unfortunately there is no further study on the relationship and the weights.

Claim 2: The weight ω_{tri} is close to the statistically optimal weight for the depth matching constraint.

Let us model the feature localization noise on the image plane as independent identically distributed Gaussian $\mathcal{N}(0, \sigma_x^2)$. For each point, the variances of the triangulations from two pairs of views with projected planes $(i-j)$ and $(l-m)$ respectively can be derived and denoted as $\sigma_{Z_a}^2(ij)$ and $\sigma_{Z_b}^2(lm)$. According to equation 3.18, the depth matching cost function $\mathcal{E}2^{ijlm}$ has the variance[1]

$$\sigma_{\mathcal{E}2^{ijlm}}^2 = \sigma_{Z_a}^2(ij) + \sigma_{Z_b}^2(lm) \tag{3.23}$$

This variance encodes the reliability of both depth estimates through triangulation and depth matching. It can vary dramatically from point to point, which means the residue errors generated from their 3D depth matching constraint may be very different from point to point. Very small noises on the unreliable points may generate large residue errors, therefore, directly minimizing such residue errors over all points is not a feasible scheme for motion estimation. From statistic point of view, to recover the motion parameters, the constraint should be weighted by $1/\sigma_{\mathcal{E}2^{ijlm}}$ so that the weighted constraint implies the same variance for all points. The scheme to minimize this optimally weighted depth matching constraint may be written as

$$\{\mathbf{P}', \mathbf{P}''\}^* = arg \min_{\mathbf{P}', \mathbf{P}''} \sum_{P_i} \sum_{ijlm} \left(\frac{\mathcal{E}2^{ijlm}}{\sigma_{\mathcal{E}2^{ijlm}}} \right)^2 \tag{3.24}$$

$$\omega_{\text{opt}}{}^{ijlm} = \frac{1}{\sigma_{\mathcal{E}2^{ijlm}}} = \frac{1}{\sqrt{\sigma_{Z_a}^2(ij) + \sigma_{Z_b}^2(lm)}} \tag{3.25}$$

ω_{opt} is called optimal weight for depth matching.

To obtain the relationship between the trilinear weight ω_{tri} and the optimal weight

[1]The correlation between Z_a and Z_b is ignored

ω_{opt}, one needs to investigate first how D_Z reflects the term $\frac{1}{\sigma_Z}$ in one triangulation. Figure 3.4 illustrates a 2D projected triangulation viewed on its projection plane. Here $\overline{\mathbf{x}}'_{\text{proj}}$, $\overline{\mathbf{x}}_{\text{proj}}$ and $\overline{T}$ represent the projected vector of $\overline{\mathbf{x}}'$, $\mathbf{A}\overline{\mathbf{x}}$ and $\overline{\mathbf{a}}_4$ on the plane. The depth triangulation equation is $Z'\,\overline{\mathbf{x}}'_{\text{proj}} = Z\,\overline{\mathbf{x}}_{\text{proj}} + \overline{T}$. Without loss of generality, $\overline{\mathbf{x}}'_{\text{proj}}$ and $\overline{\mathbf{x}}_{\text{proj}}$ are assumed to have unit length for all features. If $(i = (1,2), j = 3)$ is chosen to be the projection plane for the triangle, this assumption introduces an approximation of at most 15% error for a field of view of 60^o.

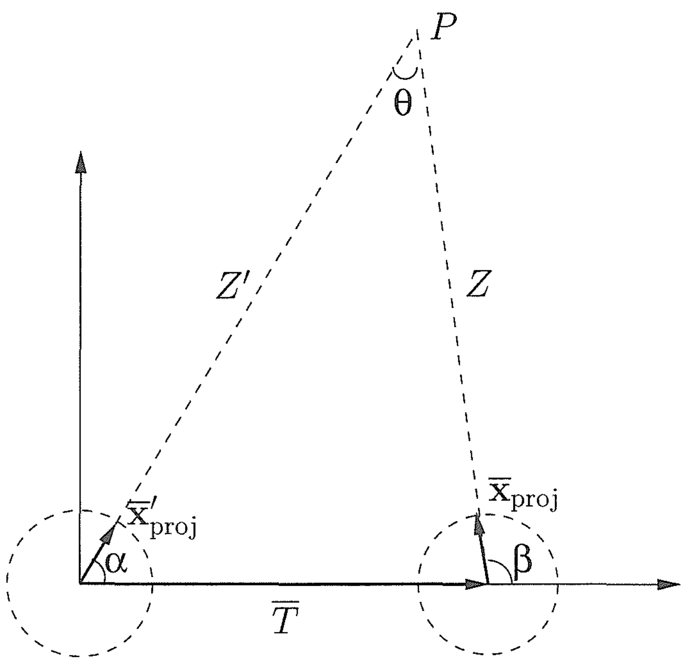

P
θ
Z'
Z
$\overline{\mathbf{x}}'_{\text{proj}}$
$\overline{\mathbf{x}}_{\text{proj}}$
α
β
$\overline{T}$

Figure 3.4: **2D Triangulation Illustration.**

Since the vectors are on the 2D projection plane, they can be denoted by two-dimensional vectors as

$$\overline{\mathbf{x}}'_{\text{proj}} = \begin{bmatrix} \cos\alpha \\ \sin\alpha \end{bmatrix} \qquad \overline{\mathbf{x}}_{\text{proj}} = \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix} \qquad \overline{T} = \begin{bmatrix} t \\ 0 \end{bmatrix} \tag{3.26}$$

where $(\alpha, \beta) \in (0, \pi)$ represent all possible camera orientations. The depth Z can be easily obtained from this parameterization as

$$Z = \frac{t \sin\alpha}{\sin(\beta - \alpha)} \tag{3.27}$$

and our goal is to check the relationship between its denominator

$$D_Z = \sin(\beta - \alpha) \tag{3.28}$$

and its reliability. Let us assume that Gaussian noise $\mathcal{N}(0, \sigma^2)$ is added to both α and β independently. This assumption transfers the noise model $\mathcal{N}(0, \sigma_x^2)$ of the measurements on the image plane to the rotation angle, which is reasonable since readers can verify, for the camera with 60^o view angle, σ varies only in the range $[0.75\sigma_x \ \ \sigma_x]$. From the first order Taylor expansion, the variance of depth Z is related to the variances of independent variables α and β by

$$\sigma_Z^2 = \left(\frac{\partial Z}{\partial \alpha}\right)^2 \sigma_\alpha^2 + \left(\frac{\partial Z}{\partial \beta}\right)^2 \sigma_\beta^2 \tag{3.29}$$

$$= \left(\left(\frac{\partial Z}{\partial \alpha}\right)^2 + \left(\frac{\partial Z}{\partial \beta}\right)^2\right) \sigma^2 \tag{3.30}$$

Equation 3.30 comes from our previous assumption that $\sigma_\alpha^2 = \sigma_\beta^2 = \sigma^2$. The two derivative terms can be computed from equation 3.27 as

$$\frac{\partial Z}{\partial \alpha} = \frac{t \sin\beta}{\sin^2(\beta - \alpha)} \tag{3.31}$$

$$\frac{\partial Z}{\partial \beta} = \frac{t \sin\alpha \cos(\beta - \alpha)}{\sin^2(\beta - \alpha)} \tag{3.32}$$

Substituting equations 3.31 and 3.32 in equation 3.30, one can obtain the reliability of Z as

$$\frac{1}{\sigma_Z} = \frac{\sin^2(\beta - \alpha)}{t\,\sigma \sqrt{\sin^2\alpha \cos^2(\beta - \alpha) + \sin^2\beta}} \tag{3.33}$$

Equation 3.33 intuitively hints that $1/\sigma_Z$ is strongly related to D_Z^2. The objective of this reliability weight is to "reject" unreliable points by assigning to them smaller

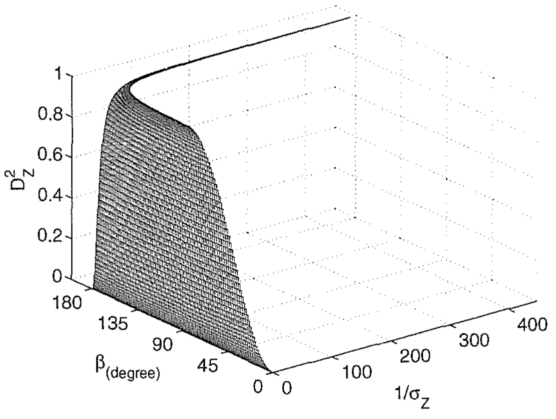


Figure 3.5: **Mesh Surface I.** General view of β and D_Z^2 vs. their corresponding reliability quantity $1/\sigma_Z$. More details are shown in Figures 3.6 and 3.7.

weight. Observe that the maximum of its denominator is only $\sqrt{2}t\sigma$. Therefore, for unreliable points, the cause of their reliability quantity $1/\sigma_Z \to 0$ must be $D_Z^2 \to 0$.

Since D_Z is a function of $\theta = \beta - \alpha$, we pick 2 independent angles β and θ to represent the triangle. Figure 3.5 shows in a form of 3D mesh plot the relation between $D_Z^2 = \sin^2\theta$, β and $1/\sigma_Z$ for all possible such triangles in 2D image plane. The values of the constants t and σ only effect the scale, but not the shape of the plot. Therefore, in this and the following mesh plots, they are arbitrarily set at $t = 10$ and $\sigma = 0.01$.

Observe that the majority of the mesh in Figure 3.5 is a planar structure, except for a small part of a long tail. This tail corresponds to the very unrealistic situation that the feature point is extremely close to the camera and has very small depth. Let us define *baseline* as the distance between projection centers of the two views, namely, t in our 2D triangulation model. Now if we add a very reasonable constraint: the depth must be larger than 10% of the baseline, and we draw the graph again, the relationship is shown in the left plot in Figure 3.6. Let us take a side view along the β axis: what we see from the right plot in Figure 3.6 is that for all possible values of angle β which corresponds to all possible qualified triangulations, there is an approximately linear relationship between D_Z^2 and $1/\sigma_Z$ except for the top part.

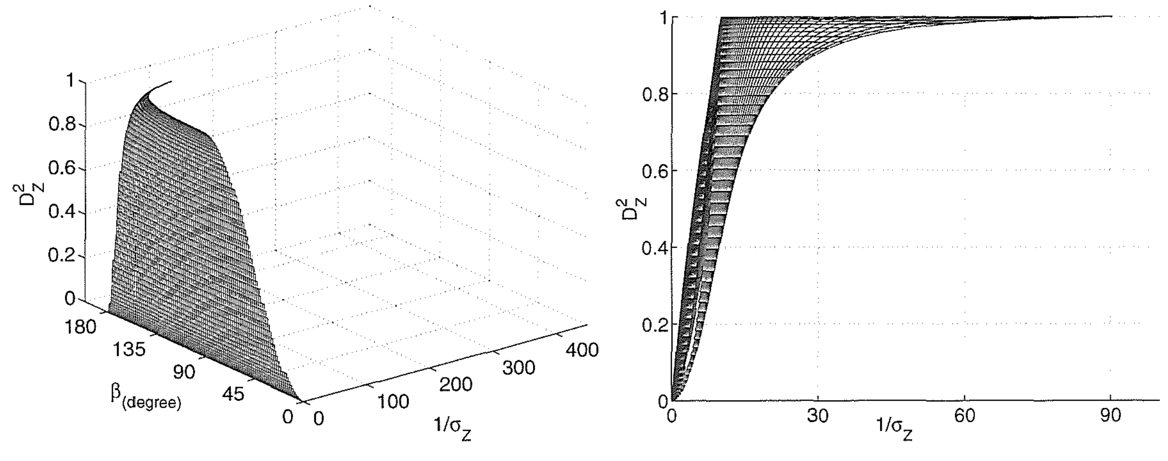


Figure 3.6: **Mesh Surfaces II**: under the constraint that depth is larger than 10% of the baseline. (Left): General view of β and D_Z^2 vs. their corresponding reliability quantity $1/\sigma_Z$. (Right): Side view along β axis. D_Z^2 vs. $1/\sigma_Z$ over all β. Approximate linearity is observed between D_Z^2 and $1/\sigma_Z$.

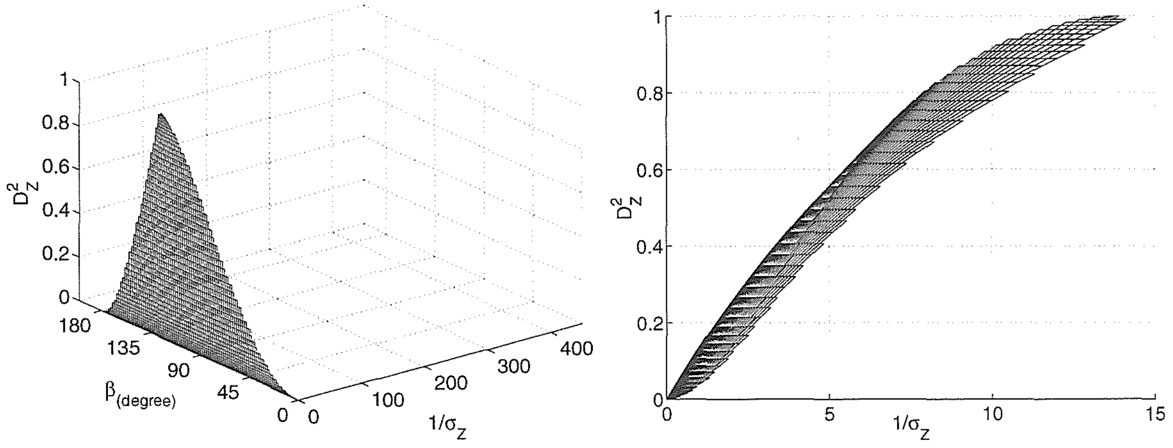


Figure 3.7: **Mesh Surface III**: under the constraint that depth is larger than 70% of the baseline. (Left): General view of β and D_Z^2 vs. their corresponding reliability quantity $1/\sigma_Z$. (Right): Side view along β axis. D_Z^2 vs. $1/\sigma_Z$ over all β. Strong linearity is observed between D_Z^2 and $1/\sigma_Z$.

Now we require further that the depth must be at least 70% of the baseline. The new relationship is shown in Figure 3.7. A linear relationship between D_Z^2 and $1/\sigma_Z$ is clearly observed from the right plot. The baseline in the motion estimation is actually very small, so this constraint on depth is practically trivial and satisfied by most realistic triangulation cases.

The narrower band and sharper end around 0 in the side view plot in Figure 3.7 indicate that the more unreliable is the triangulation, the better does D_Z^2 represent $1/\sigma_Z$ up to an overall scale factor. As a weight function, it satisfies the criteria to be able to "reject" correctly the unreliable points. Therefore, D_Z^2 is a good approximation of the reliability quantity $1/\sigma_Z$ up to an overall scale for the triangulation. By computing the first-order derivative of $1/\sigma_Z$ as a function of D_Z^2 at $D_Z^2 = 0$, we derive a closed form expression for the proportionality factor between the two quantities: $D_Z^2 \approx \sqrt{2}t\sigma|\sin\beta|\frac{1}{\sigma_Z}$. In the case $t = 10, \sigma = 0.01$, the maximum slope shown in Figure 3.7(right) is about 0.14.

In general, we can assume that the motions are smooth so that $\sigma_{Z_a} \approx \sigma_{Z_b}, D_{Z_a} \approx D_{Z_b}$. Then we have $\omega_{\text{opt}} \approx 1/(\sqrt{2}\sigma_{Z_a})$ and $\omega_{\text{tri}} \approx D_{Z_a}^2$. Therefore, ω_{tri} is approximately linear to ω_{opt}. Consequently, ω_{tri} is a good substitute to the optimal weight for depth matching constraint and trilinear constraints are quasi equivalent to the optimal weighted depth matching constraints. The clear geometrical meaning of trilinear constraints enables us to understand their excellent performance in 3-view motion estimation.

3.4 Scale Propagation Scheme

In order to explore the performance of the depth matching constraint under different weighting conditions in the following experiment section, we only leave the relative scale $s = \frac{||\bar{\mathbf{a}}_4||}{||\bar{\mathbf{b}}_4||}$ as the unknown motion parameter which makes it easier for comparison and also demonstrates the problem well. The other motion parameters, i.e., motions between each pair of two views with unit length translation are given by the standard 2-view motion estimation scheme. Denote the point depth Z by Z_{au} and Z_{bu} which are triangulated respectively from the first pair (view 1 and 2) and the second pair (view 1 and 3) whose motions are recovered independently assuming unit length translation. The variances of Z_{au} and Z_{bu} are denoted by $\sigma^2_{Z_{au}}$ and $\sigma^2_{Z_{bu}}$. Consequently the depth matching constraint 3.17 can be modified as

$$Z_{au} - s\, Z_{bu} = 0 \tag{3.34}$$

As the only unknown parameter, the relative scale s is invariant to all the points P_i while the depth Z_{au}, Z_{bu} are functions of points P_i. The least squares solution for s which minimizes $\sum_{P_i}(Z_{au} - sZ_{bu})^2$ is

$$s^* = \frac{\sum_{P_i} Z_{bu}\, Z_{au}}{\sum_{P_i} Z^2_{bu}} \tag{3.35}$$

However, this is an asymmetric solution because we can also write equation 3.34 as a function of $1/s$

$$\frac{1}{s} Z_{au} - Z_{bu} = 0 \tag{3.36}$$

Then we obtain the least squares solution for $1/s$ which minimizes $\sum_{P_i}(Z_{au}/s - Z_{bu})^2$ as

$$\frac{1}{s^*} = \frac{\sum_{P_i} Z_{au}\, Z_{bu}}{\sum_{P_i} Z^2_{au}} \tag{3.37}$$

Both equations 3.35 and 3.37 give reasonable estimates for s^* though they minimize different cost functions. A practical solution is obtained by taking the square root of the product of these two estimates as

$$s^* = \sqrt{\frac{\sum_{P_i} Z^2_{au}}{\sum_{P_i} Z^2_{bu}}} \tag{3.38}$$

Trilinear constraints are also linear functions of the scale s

$$\mathcal{E}1^{ijlm} = N_{Z_{au}} D_{Z_{bu}} - s N_{Z_{bu}} D_{Z_{au}} = 0 \tag{3.39}$$

The symmetric solution for s using trilinear constraints is

$$s^*_{\text{tri}} = \sqrt{\frac{\sum_{P_i} \sum_{ijlm} (N_{Z_{au}} \, D_{Z_{bu}})^2}{\sum_{P_i} \sum_{ijlm} (N_{Z_{bu}} \, D_{Z_{au}})^2}} \tag{3.40}$$

The solution for s using the optimally weighted cost function according to equations 3.24 and 3.25 is

$$\begin{aligned} s^* &= arg\min_s \sum_{P_i} \sum_{ijlm} \left(\frac{Z_{au} - s\, Z_{bu}}{\sqrt{\sigma^2_{Z_{au}} + s^2 \, \sigma^2_{Z_{bu}}}} \right)^2 \\ &\approx arg\min_s \sum_{P_i} \sum_{ijlm} \left(\frac{Z_{au} - s\, Z_{bu}}{\sqrt{\sigma^2_{Z_{au}} + \frac{Z^2_{au}}{Z^2_{bu}} \, \sigma^2_{Z_{bu}}}} \right)^2 \\ &= \frac{\sum_{P_i} \sum_{ijlm} \frac{Z^2_{bu}}{Z^2_{bu}\, \sigma^2_{Z_{au}} + Z^2_{au}\, \sigma^2_{Z_{bu}}} \, Z_{bu} \, Z_{au}}{\sum_{P_i} \sum_{ijlm} \frac{Z^2_{bu}}{Z^2_{bu}\, \sigma^2_{Z_{au}} + Z^2_{au}\, \sigma^2_{Z_{bu}}} \, Z_{bu} \, Z_{bu}} \end{aligned} \tag{3.41}$$

During the derivation, the scale s appearing in the weight function is approximated by the individual scale Z_{au}/Z_{bu}, which gives the exact individual weight function and generates a linear estimator. The same solution can also be reached by another method: weight the individual scale by the inverse of its variance, i.e.,

$$\frac{\sum_{P_i} \sum_{ijlm} \left(s(P_i, ijlm) \, / \, \sigma^2_{s(P_i, ijlm)} \right)}{\sum_{P_i} \sum_{ijlm} \left(1 \, / \, \sigma^2_{s(P_i, ijlm)} \right)} \tag{3.42}$$

This can be verified easily.

The solution for $1/s$ using the optimally weighted constraint can be similarly derived. Unlike equation 3.35 and 3.37, to find the symmetric solution for s^*_{opt}, there are no common terms between s^* and $1/s^*$ that can cancel each other out, because two weights $\omega_{\text{opt},s}, \omega_{\text{opt},1/s}$ are different:

$$\omega_{\text{opt},s} = \sqrt{\frac{Z_{bu}^2}{Z_{bu}^2\,\sigma_{Z_{au}}^2 + Z_{au}^2\,\sigma_{Z_{bu}}^2}} \tag{3.43}$$

$$\omega_{\text{opt},1/s} = \sqrt{\frac{Z_{au}^2}{Z_{bu}^2\,\sigma_{Z_{au}}^2 + Z_{au}^2\,\sigma_{Z_{bu}}^2}} \tag{3.44}$$

The optimal solution s_{opt}^* is much more complicated than s^* in equation 3.38 and s_{tri}^* in equation 3.40.

$$s_{\text{opt}}^* = \sqrt{\frac{(\sum_{P_i}\sum_{ijlm}\omega_{\text{opt},s}^2\,Z_{bu}\,Z_{au})(\sum_{P_i}\sum_{ijlm}\omega_{\text{opt},1/s}^2\,Z_{au}\,Z_{au})}{(\sum_{P_i}\sum_{ijlm}\omega_{\text{opt},s}^2\,Z_{bu}\,Z_{bu})(\sum_{P_i}\sum_{ijlm}\omega_{\text{opt},1/s}^2\,Z_{bu}\,Z_{au})}} \tag{3.45}$$

3.5 Experiments

3.5.1 Simulated Translation

This experiment is designed to demonstrate the basic results in section 3.3. A total of 500 features are uniformly distributed inside a $60 \times 60 \times 60\ cm^3$ cube, which is put in front of the first view 70 cm away from the camera center. Therefore, the 3D point coordinates in the camera reference frame of view 1 are in the range of $X \in [-30, 30]$, $Y \in [-30, 30]$, $Z \in [40, 100]$ which corresponds to 73.7^o view angle. The normalized homogeneous coordinate vector for point features on the image plane is $\bar{\mathbf{x}} = [X/Z \quad Y/Z \quad 1]^T$. The camera translates in the $X - Z$ plane to its second (view 2) and third (view 3) positions respectively with $\overline{T}' = [1.5 \quad 0 \quad 1.5]^T\ cm$ and $\overline{T}'' = [-3 \quad 0 \quad -3]^T\ cm$. The projection matrices are $\mathbf{P} = [\mathbf{I}_3 \quad \mathbf{0}_{3\times1}]$, $\mathbf{P}' = [\mathbf{I}_3 \quad \overline{T}']$ and $\mathbf{P}'' = [\mathbf{I}_3 \quad \overline{T}'']$. We make $\|\overline{T}''\|$ twice larger than $\|\overline{T}'\|$ here so that the triangulation from (view 1 and 3) is more reliable than that from (view 1 and 2). Points are projected onto the three image planes and Gaussian noise $\mathcal{N}(0, 0.002)$ is added to the normalized feature coordinates independently.

Figure 3.8(a) and (b) show the relationship D_Z^2 vs. $1/\sigma_Z$ of all the points P_i when the $X - Z$ planes ($i = l = 1, j = m = 3$) are chosen as the projection planes for

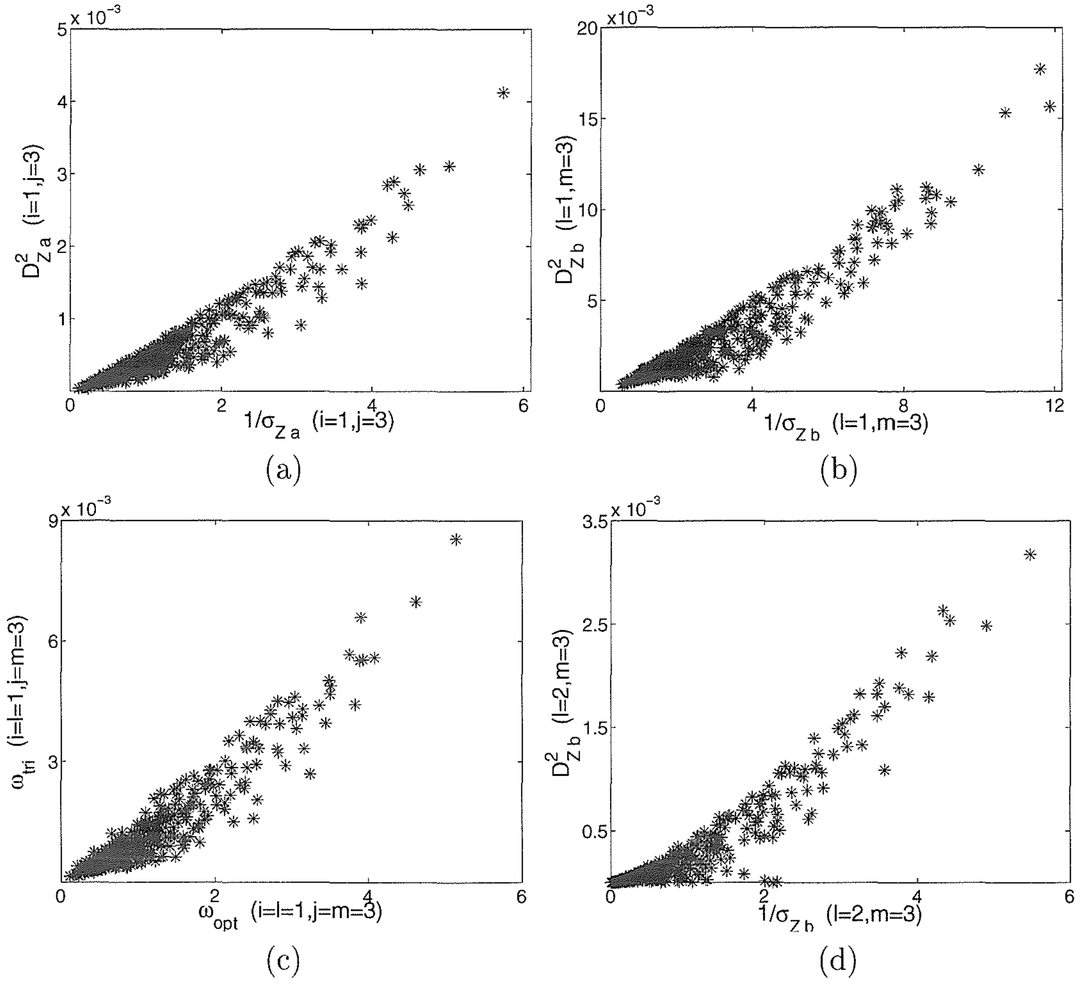


Figure 3.8: **Characteristic of weight functions in simulated translation experiment.**

both triangulations. Each plot is approximately linear which confirms that D_Z^2 does represent well the triangulation reliability. Notice from the axes of the plots, the two triangulations have different levels of reliability and also different approximated slopes due to their very different baselines. This case is more complicated than the smooth motion which we assumed in the last part of section 3.3 to infer the relationship of ω_{tri} and ω_{opt}. However, Figure 3.8(c) shows that ω_{tri} still varies linearly as a function of the optimal weight ω_{opt}. In this experiment, the $X-Z$ plane is the best for projection (Figure 3.8(a)(b)). If we project the 3D triangles of points onto the $Y-Z$ plane, the triangles will be deformed, especially for the points whose Y components are close to zero. Therefore the reliability of the depth estimation from such triangles will decrease dramatically. Figure 3.8(d) illustrates this effect, where the triangles in the pair (view 1 and 3) are projected onto the $Y-Z$ plane ($l=2, m=3$). Reliability $1/\sigma_{\lambda_b}$ shown in Figure 3.8(d) is four times smaller than that in Figure 3.8(b) where the same 3D triangles are projected on the $X-Z$ plane. The dominant plane varies from case to case between the $X-Z$ and $Y-Z$ planes, but in general will not be the $X-Y$ plane.

3.5.2 Corridor Navigation

When a camera is moving forward in an environment, the so-called *epipoles*, the point of intersection of the baseline between two camera positions with the image planes, of any two frames are close to the image centers. Therefore, the triangulation is ill conditioned for the feature points that appear close to the image center. In this case, it is crucial to appropriately reject those points by assigning to them a scalar weight close to zero. We test our strategy on a real image sequence taken with a calibrated CCD video camera mounted on a cart moving forward along a building corridor. The data are captured by Bouguet [8]. Features are detected and tracked using the algorithm described in section 2.6. A sample image with detected features is shown in Figure 3.9. The sequence contains 400 frames. For this experiment, we choose every fourth frame which generates a sequence of 100 frames with baseline four times longer

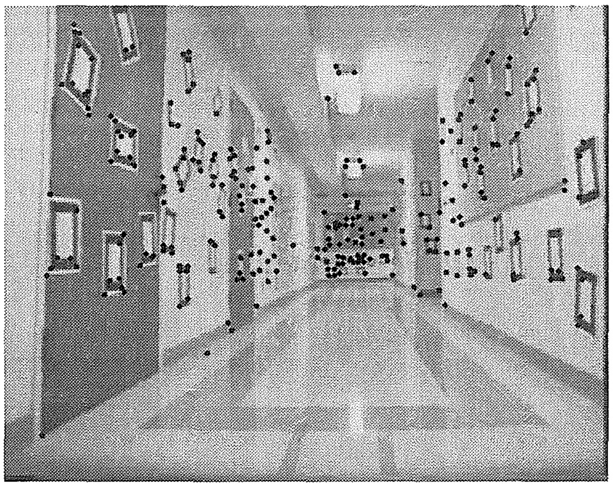


Figure 3.9: **Sample Image in the Real Corridor Sequence**. Black dots indicate the features detected on the image.

than that of the original sequence. The sequence is difficult for motion estimation because the features in the far end of the corridor appear almost around the image center and their depth triangulations are not reliable at all.

The motion parameters with unit length of translation between every two consecutive frames are estimated using the recursive Newton-Raphson method. Given those parameters, the relative scale between camera translations from time $t-1$ to time t ($\overline{T}(t-1,t)$) and from time t to time $t+1$ ($\overline{T}(t,t+1)$) for all $t \in (1, 100)$ is recovered by three different schemes proposed in section 3.4: direct depth matching (equation 3.38), optimally weighted depth matching (equation 3.45), and trilinear constraints (equation 3.40). Here are some implementation details: the depth Z and its variance σ_Z^2 appearing in equation 3.38 and equation 3.45 are calculated by least square method using all 3 triangulation equations. Therefore summation $\sum_{ijlm}$ is not necessary in equation 3.45. For the trilinear constraints, only 4 independent equations with combinations of $i = (1,2), l = (1,2), j = m = 3$ are used. Figure 3.10 shows the recovered scale s over time. Although we do not have ground truth for s, it should be around 1 since the cart was moving at approximately constant velocity. The first method (Figure 3.10(left)) does not give acceptable result because the estimated s oscillates dramatically between 0.5 and 3. However the scale estimation becomes much more stable in its final phase, the reason is that the camera is approaching to the end of the hallway and the features located there are now projected far away from the image center. A big improvement is achieved by the second method (Figure 3.10(middle)) which uses ω_{opt} as the weight of the depth matching. The result shown in Figure 3.10(right) demonstrates that the trilinear scheme is comparable to the optimal solution.

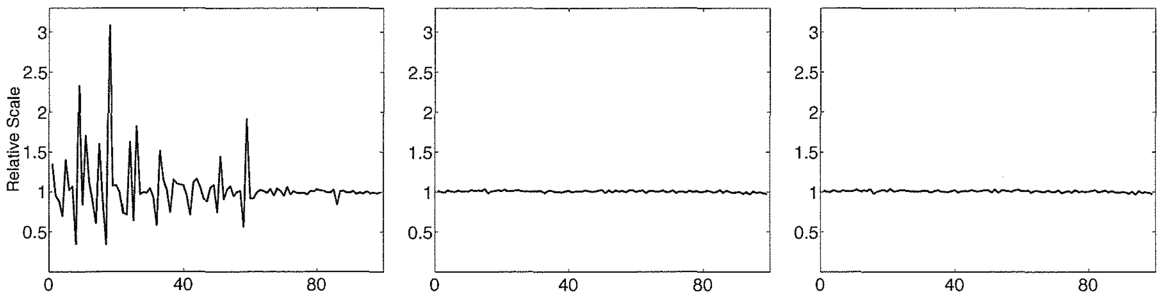


Figure 3.10: **Relative Scale s over Frames for Navigation Sequence**. (left): s recovered by direct depth matching. (middle): s recovered by depth matching with optimal weight. (right): s recovered from trilinear constraints.

Let us pick three frames within the whole sequence (frames 16, 17 and 18 in this example) and draw some relationship in detail. With $X - Z$ as projection plane for both triangulations ($i = l = 1, j = m = 3$), Figure 3.11(a) shows that D_Z^2 is almost proportional to the optimal reliability measurement $1/\sigma_Z$. Figure 3.11(b) shows that the trilinear weight ω_{tri} is also approximately linear to the optimal weight $\omega_{\text{opt},s}$. The property that both of them assign very small weights to the unreliable triangulations is a crucial factor that makes the trilinearly and optimally weighted scale recovery schemes successful. The other two projection planes are less informative since they are less important and less reliable than the $X - Z$ projection plane in this example. Figure 3.11(c) and (d) show how the weight relates to the depth and individual scale. The depth Z_a, Z_b here are calculated using all 3 triangulation equations, therefore summation over $(ijlm)$ is not necessary. According to Figure 3.11(c), as Z_a increases, $D_{Z_a}^2$ decreases. This provides a supporting argument to our analysis since a larger Z_a means that the point is closer to the image center, and therefore unreliable for

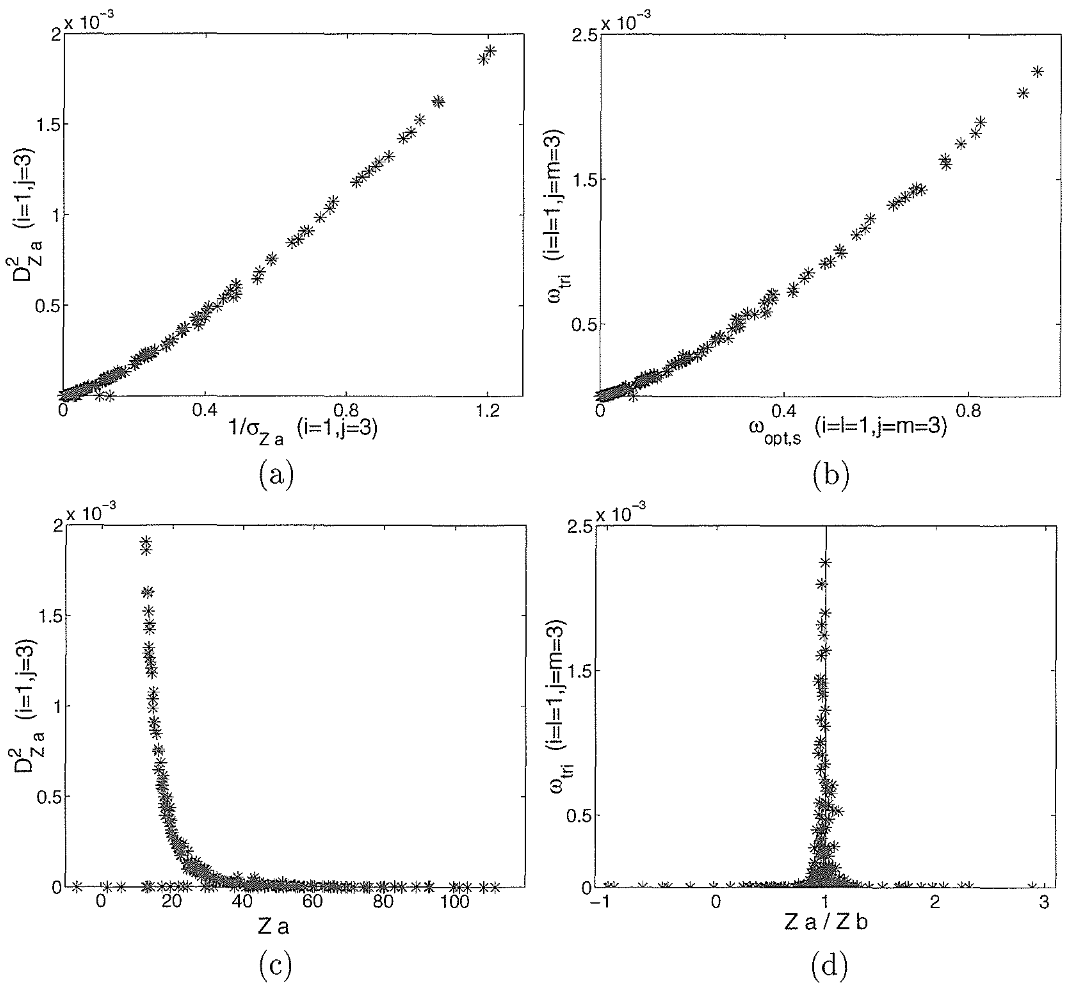


Figure 3.11: **Characteristic of the trilinear weight functions of 3 views in navigation sequence.**

triangulation. It justifies the small value for $D^2_{Z_a}$. Figure 3.11(d) indicates that larger weights are assigned to features with individual scales Z_a/Z_b close to the believed true scale 1, which makes the estimation of scale s using weighted schemes reliable.

3.5.3 Rotating Rock Experiment

Another experiment is done on a rotating rock sequence acquired by Bouguet [7]. A textured rock is placed on a turn table. Between two consecutive images, the rotating stage is turned by 2 degrees. A total of 225 frames are acquired. A sample image with detected features is shown in Figure 3.12.

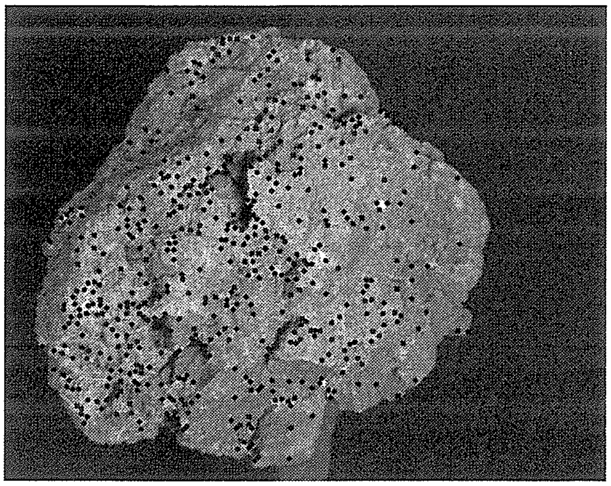


Figure 3.12: **Sample Image in the Rotating Rock Sequence.** Black dots indicate the features detected on the image.

For this sequence, the relative scale s is recovered reliably with or without weighting factors. The three plots in Figure 3.13 are almost identical. The scale s varies in the range 1 ± 0.04, which is quite acceptable.

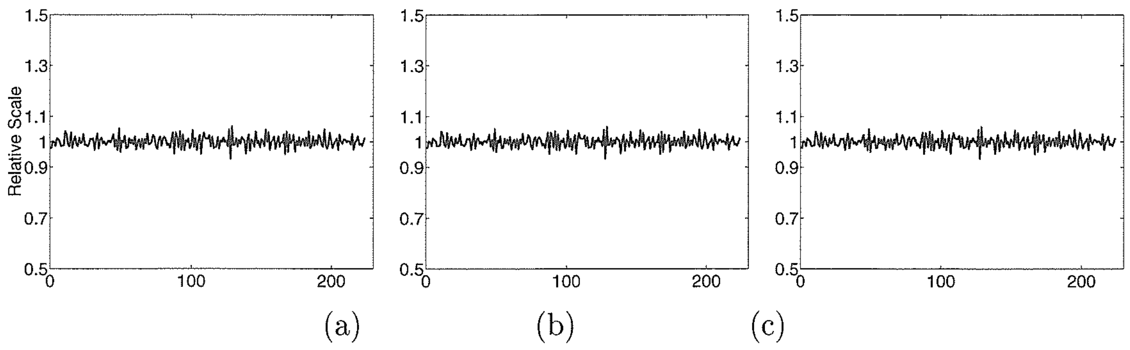


Figure 3.13: **Relative Scale s over Frames for the Rotating Rock Sequence.** (left): s recovered by direct depth matching. (middle): s recovered by depth matching with optimal weight. (right): s recovered from trilinear constraints. Three plots are similar because all points are weighted almost equally.

The reason for the similar performance of scale recovery using either direct or weighted methods can be explained by Figure 3.14(a) and (b). As we expected, linear relationship is observed between $D^2_{Z_a}$ and $1/\sigma_{Z_a}$ for single 2D triangulation, and between ω_{tri} and $\omega_{\text{opt},s}$ for depth matching. Different from the previous navigation

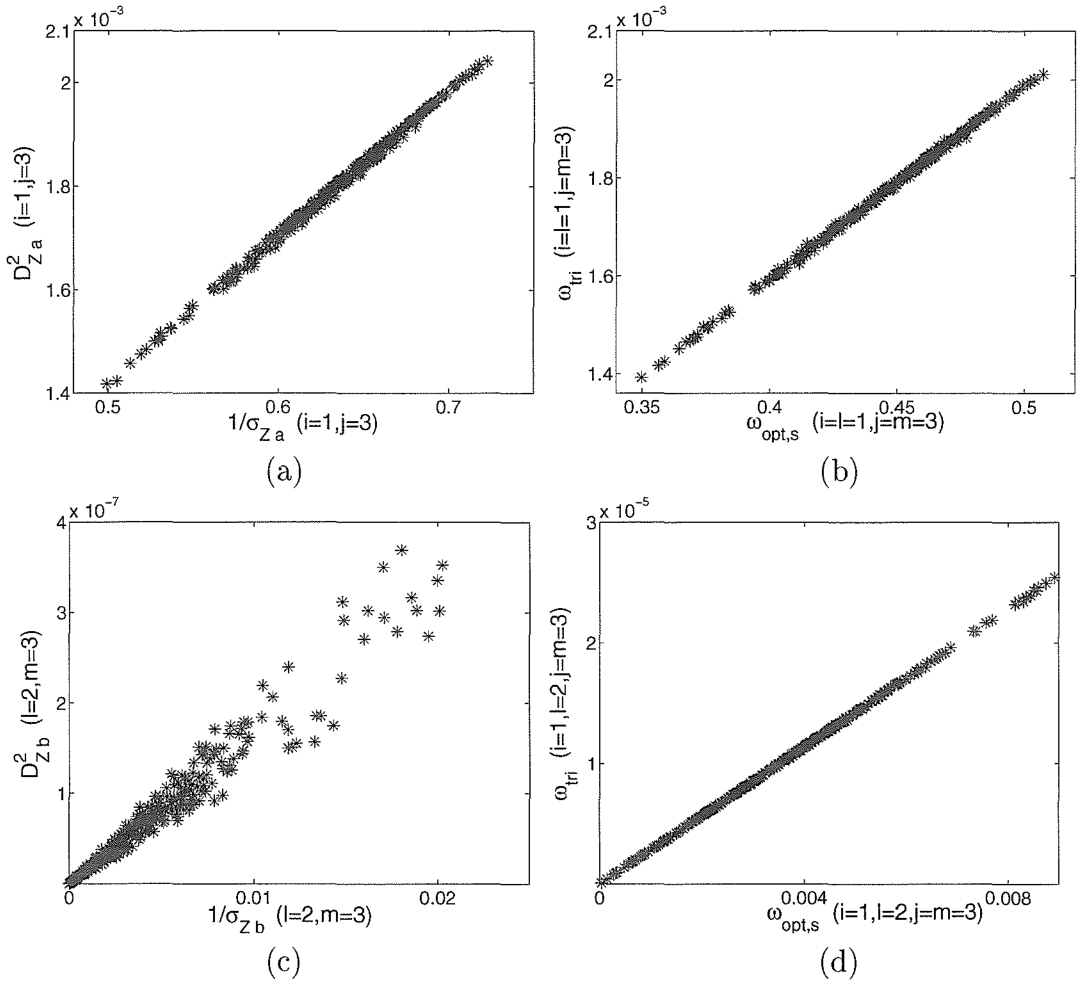


Figure 3.14: **Characteristic of the weight functions of 3 sample views in rotating rock sequence.**

experiment that has many features weighted close to 0, the ranges of the weight functions in this experiment shown in Figure 3.14(b) are very small, $\omega_{\text{tri}} \in [1.4, 2.1] \times 10^{-3}$ and $\omega_{\text{opt},s} \in [0.35, 0.5]$. This indicates that all the features are almost equally reliable. Therefore, there is not much difference among the three different scale recovery schemes and they generate almost identical results. Figure 3.14(c) shows the reliability of the triangulation if the $Y - Z$ plane is chosen as the projection plane. The order of D^2 goes down to 10^{-7}, which is trivial compared with that in Figure 3.14(a) (order of 10^{-3}). If we pick one triangulation in the $X - Z$ plane and another one in the $Y - Z$ (i=1, l=2, j=m=3) to form the depth matching (which is one of the 9 trilinear constraints), the corresponding weight functions $\omega_{\text{opt},s}$ and ω_{tri} shown in Figure 3.14(d) are 100 times smaller than that in Figure 3.14(b). Therefore, among the 9 trilinear constraints, only a few dominate the estimation.

3.6 Extension to Uncalibrated Case

For an uncalibrated sequence, let us simply assume $f_x = f_y = f$ and the unknown calibration matrices be

$$\mathbf{K}' = \mathbf{K}'' = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.46}$$

The homogeneous pixel coordinate vector for point in the second image is denoted by $\overline{\mathbf{u}}'$, which relates to the normalized homogeneous coordinates as $\overline{\mathbf{u}}' = \mathbf{K}'\overline{\mathbf{x}}'$. Unlike the case of a calibrated camera, in the uncalibrated case the normalized vector $\overline{\mathbf{x}}'$ can no longer be computed from pixel coordinate vector $\overline{\mathbf{u}}'$ since the camera matrix $\mathbf{K}'$ is unknown. The same is true for other views. Denote the new projection matrix for view 2 by $\mathbf{P}'_{\text{new}} = \mathbf{K}'[\mathbf{A} \;\; \overline{\mathbf{a}}_4]$. The camera projection for view 1 is kept the same as $Z\overline{\mathbf{u}} = [\mathbf{I}_3 \;\; \mathbf{0}_{3\times1}]\overline{\mathbf{X}}$ and the camera projection for view 2 is $Z'\overline{\mathbf{u}}' = \mathbf{P}'_{\text{new}}\overline{\mathbf{X}}$. The same triangulation equation as equation 3.15 can be obtained except that $\overline{\mathbf{x}}, \overline{\mathbf{x}}', \mathbf{A}$ and $\overline{\mathbf{a}}_4$ are substituted by $\overline{\mathbf{u}}, \overline{\mathbf{u}}', \mathbf{K}'\mathbf{A}$ and $\mathbf{K}'\overline{\mathbf{a}}_4$, respectively. By choosing different (i, j) combination of 2 equations to estimate the depth, the denominator associated to the uncalibrated case $D_{Z_a,\text{new}}$ is related to that of the calibrated case D_{Z_a} as

$$D_{Z_a,\text{new}} = f D_{Z_a} \qquad i = (1,2), j = 3 \tag{3.47}$$

$$D_{Z_a,\text{new}} = f^2 D_{Z_a} + f(c_y u'^1 a_k^3 u^k + c_x a_k^2 u^k - c_y a_k^1 u^k - c_x u'^2 a_k^3 u^k) \qquad (i = 1, j = 2) \tag{3.48}$$

The derivations are straightforward and omitted here. Equations 3.47 and 3.48 illustrate how to choose the 4 independent trilinear constraints among the nine for the uncalibrated camera. If we assume $c_x = c_y = 0$, the principal point is assured to be at the center of the image, then $D_{Z_a,\text{new}} = f^2 D_{Z_a}$ for $(i = 1, j = 2)$, which is weighted f times larger than the other two triangulation combinations $(i = 1, j = 3)$ and $(i = 2, j = 3)$. The focal length f is usually around 500-2000 pixels. When we form the trilinear constraint, the depth matching weight is $\omega_{\text{tri}} = D_{Z_a} D_{Z_b}$. The weight ω_{tri} of the constraint with $(i = l = 1, j = m = 2)$ will be f^4 times larger than that of the calibrated case, while the weight of the constraint with $(i = l = 1, j = m = 3)$ will only be f^2 times larger than that of its corresponding calibrated case. Such unbalanced weight may affect the whole estimation using trilinear constraints. The influence of c_x, c_y is not as significant as f, however, the closer they are to zero, the better the results will be. Therefore, the proper choice for the set of independent trilinear constraints should be the 4 trilinearities with $i = (1,2), l = (1,2), j = m = 3$ out of the total 9 constraints for 3 views motion estimation with uncalibrated camera [71].

3.7 Conclusion

In this chapter we find a new geometrical interpretation of the algebraic trilinear constraints. It is shown that minimizing the trilinear algebraic equations in a least square sense is a good approximation of minimizing a geometric error in Euclidean space with appropriate weight applied on every point. This fundamental result provides an explanation as to why, in practice, there is no need of adding extra weighting coefficient to each trilinear constraint in order to achieve satisfactory motion estimates. In other words, the sum of the squares of every algebraic equation is a valid cost function to minimize for estimating motion parameters. All the theoretical statements are further supported by experimental results. One additional contribution in this chapter is a robust scheme to propagate scale information which is significantly insensitive to noise in the measurement data. Although most of the derivations are done assuming a calibrated camera, natural extensions of the results to the uncalibrated case are also provided. It is shown theoretically how to choose 4 independent trilinear constraints for reconstruction of motion and structure.

Chapter 4 Scene Segmentation from 3D Motion

4.1 Motivation

An intrinsic assumption for the problem of three-dimensional motion and structure recovery discussed in the previous chapter is that the relative motion between a camera and its observed three-dimensional scene is rigid. However, in practice this assumption is easy to be violated and more often the camera observes multiple moving objects in the scene. In this situation, the reconstruction has to be decomposed into two stages: segment each object from the cluster, then recover the motion and structure for each separate object which undergoes a rigid motion. Scene segmentation remains a difficult topic in computer vision in all these years, but it is very important in studying the problem of structure from motion assuming multiple moving objects observed. Despite that there may exist many cues to be exploited for segmentation, such as texture, color, structure models, etc., motion is the most common factor contained in this kind of sequences and the cue we are going to explore in this chapter.

Motion based segmentation can be classified into 2 major categories: 2D optical flow segmentation [44, 96, 97] and 3D motion based segmentation [106, 75, 79, 54]. Recent development in 2D segmentation applies the expectation-maximization (EM) algorithm [14] to segment and estimate the region-based optical flow. Either parametric [44, 96, 98] or non-parametric [97] 2D flow model is assumed. For the research of 3D structure and motion recovery, as shown in previous chapter, two and three views are the common framework to explore. However, it is too hard to consider the difficult segmentation task together with the complicated 3-view analysis at the same time. The common approach is to perform motion segmentation based on two views, which is also the framework that we take in this chapter. Therefore the input to our motion segmentation system is the correspondent positions of point features on two image frames.

Given the detected features and their correspondents in the next frame, we observe that in most cases the objects or their feature flow are separated very well from each other in space. For these cases, we propose a modified separation matrix method [79] by applying the normalized cuts [74] on the separation matrix to obtain global grouping indicator, and expected segmentation results are achieved. However, when the objects are overlapped spatially but undergo independent motions, such as Ullman's co-axial transparent cylinder demonstration, there will be no proper affinity to perform segmentation by that way. Pure underlying 3D motion becomes the only cue to segment the scene. We explore the EM algorithm to deal with such difficult cases and achieve excellent results. However, EM deals with highly nonlinear expression in 3D motion parameters, which only leads to convergence after many iterations. In addition, convergence to the global minimum is usually not guaranteed. To efficiently perform segmentation, it is reasonable to propose a combination method: the modified separation matrix scheme is first applied, then for the cases in which this approach fails to segment the scene correctly, the EM algorithm is applied in addition. The scheme is tested on vast number of synthetic image sequences. Results with real image sequence are also given in this chapter. The most closely related work for 3D EM based motion segmentation in literature is the paper by MacLean et al. [54], who use EM algorithm to cluster image patches based on their subspace motion constraints.

A calibrated camera is assumed in our study. The chapter is organized as follows. In section 4.2 we describe the first step of our approach: separation matrix and normalized cuts grouping scheme. Section 4.3 presents the EM based 3D motion segmentation. The experiments are reported in section 4.4 and followed by the conclusion.

The majority work presented in this chapter was published in [23].

4.2 Separation Matrix and Normalized Cuts

We first consider the case that rigid objects move independently in the image sequence and their spatial positions and motions are quite different. An example of the feature locations and velocities is shown in Figure 4.1(a). To properly and quickly segment this kind of scene, we combine the separation matrix scheme [79] with the normalized cuts [74].

4.2.1 Essential Matrix for Two View Motion Estimation

Denote the rigid relative motion between a camera and an object by the rotation about the axis $\overline{\Omega}$ and translation $\overline{T}$. Let $(\overline{T}\wedge)$ and $(\overline{\Omega}\wedge)$ be the skew-symmetric matrices of $\overline{T}$ and $\overline{\Omega}$, respectively (see equation 2.5), and $\mathbf{R} = e^{(\overline{\Omega}\wedge)}$ be the rotation matrix of $\overline{\Omega}$. Let the Euclidean coordinate vector of a point P_i in the two camera reference frames be $\overline{X}_i$ and $\overline{X}'_i$, respectively. The motion transformation between the two is then expressed as $\overline{X}'_i = \mathbf{R}\overline{X}_i + \overline{T}$. This point is imaged in 2-views with normalized homogeneous coordinate vectors as $\overline{\mathbf{x}}_i$ and $\overline{\mathbf{x}}'_i$, respectively. Since $\overline{X}_i = Z\,\overline{\mathbf{x}}_i$ and $\overline{X}'_i = Z'\,\overline{\mathbf{x}}'_i$, the two image coordinate vectors relate to each other as $Z'\,\overline{\mathbf{x}}'_i = Z\,\mathbf{R}\,\overline{\mathbf{x}}_i + \overline{T}$. The three vectors $\mathbf{R}\,\overline{\mathbf{x}}_i$, $\overline{\mathbf{x}}'_i$ and $\overline{T}$ are all represented in the second camera reference frame, and lie in the same plane. Algebraically, this coplanarity constraint of these three vectors can be expressed as

$$\overline{\mathbf{x}}'^T_i \mathbf{Q} \overline{\mathbf{x}}_i = 0 \tag{4.1}$$

where $\mathbf{Q} = (\overline{T}\wedge)\mathbf{R}$ is called *essential matrix* and equation 4.1 is called *epipolar constraint*. Epiploar constraint was first introduced by Longuet-Higgins [48] in 1981. Both $\overline{T}$ and $\overline{\Omega}$ can be recovered from the essential matrix $\mathbf{Q}$. Notice that equation 4.1 is linear with respect to the components of $\mathbf{Q}$. If we interpret $\mathbf{Q}$ as a 9×1 vector $\overline{\mathbf{q}}$ obtained by stacking the 3 columns of the matrix $\mathbf{Q}$ on top of each other, we can rewrite the epipolar constraint as

$$\chi_i^T\,\overline{\mathbf{q}} = 0 \tag{4.2}$$

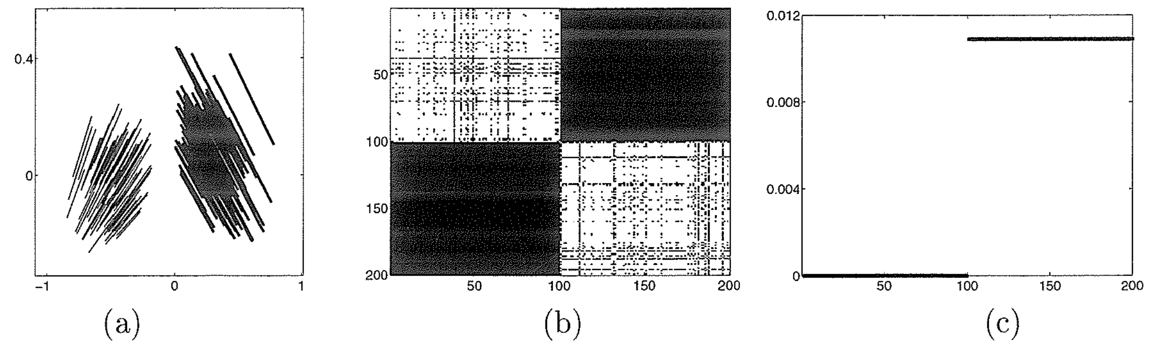


Figure 4.1: **Separation Matrix Scheme**. (a) Feature positions and velocities. Different line widths of the flow represent different motions. (b) Separation matrix. The values of $\mathbf{W}$, which are in the region (0,1), are represented uniformly by the 256 gray values between black and white. (c) Indicating vector $\overline{\mathbf{y}}$ of normalized cuts.

where $\overline{\chi}_i$, a 9×1 vector, is a function of the components of $\overline{\mathbf{x}}_i$ and $\overline{\mathbf{x}}'_i$.

4.2.2 Separation Matrix

Soatto [79] proposed a scheme to capture the grouping information in the so-called residual space. Due to the measurement noises of feature positions in an image and the error of motion estimation, epipolar constraint 4.2 is not exactly satisfied but some residual error $\epsilon_i = \overline{\chi}_i^T \overline{\mathbf{q}}$ exists for each point P_i. For objects undergoing different rigid motions, the intuition in [79] is, if we pick up an arbitrary motion $\overline{\mathbf{q}}$, the residuals ϵ_i corresponding to points which belong to the same group tend to cluster, especially when $\overline{\mathbf{q}}$ is close to the true motion. Therefore, picking a motion family $< \overline{\mathbf{q}}_k >_{k=\{1:K\}}$ which produces a residual family $\varepsilon_i = [\epsilon_{i1} \ \epsilon_{i2} \ \ldots \ \epsilon_{iK}]$ for each feature point P_i, one can construct a separation matrix $\mathbf{W}$ with the elements

$$W_{ij} = e^{-\frac{\|\varepsilon_i - \varepsilon_j\| \,/\, K}{\sigma^2}} \tag{4.3}$$

The matrix $\mathbf{W}$ can also be viewed as affinity matrix between feature points. Its element W_{ij} close to 1 implies features P_i and P_j have high probability to belong to same group and W_{ij} close to 0 means not. To choose the motion family $< \overline{\mathbf{q}}_k >_{k=\{1:K\}}$, 10 random motions in the motion space $\mathbb{R}^5$ are picked. Notice that here we drop one dimension, the length of $\overline{T}$, from the original $\mathbb{R}^6$ motion space $(\overline{\Omega}, \overline{T})$. This corresponds to the well-known scale ambiguity which says that objects with the same shape, which are twice as big, twice as far and translate twice as fast, look identical. Therefore, the structure and motion can only be recovered up to a scale factor of the translation velocity and the depth of points. Usually we set $||\overline{T}|| = 1$ and model the motion parameters in $\mathbb{R}^5$. The separation matrix generated from data in Figure 4.1(a) is shown in Figure 4.1(b), where the value of W_{ij} from 0 to 1 is represented by the gray scale from black to white in the graph. The vertical axis is the feature index i of W_{ij} and the horizontal axis is the feature index j. Each row represents how a feature point P_i relates to the others. To allow one visually see this relationship from the separation matrix, the features are reordered in such a way that the first 100 points belong to one group, and the rest 100 features belong to another group. After the reordering, what we observe from Figure 4.1(b) is two white blocks along the diagonal and two black blocks along the cross-diagonal of the separation matrix. Such matrix clearly shows the structure of groups.

In [79], the proposed separation matrix is only used to initialize some filters for motion estimation and segmentation in the following way. Assume there are N feature points in the scene. For each point P_i at the initial time point $t = 0$, a cluster is generated by finding all points P_j whose association with point P_i, W_{ij}, is higher than a preset threshold. Given these N clusters, N essential filters are then initialized accordingly from them to estimate motion in a long sequence while outliers are consistently discarded and filters with similar motions are dynamically merged. For details, please refer to [79]. Unlike that work which performs the segmentation using a long sequence of image frames, we are interested in the more challenging question: how well can the segmentation be done with only two frames? In [79], only local grouping information at each point is extracted from the separation matrix. For global clustering based on this separation matrix $\mathbf{W}$, we propose a modified method by applying normalized cuts method [74] to complete the separation matrix approach for segmentation.

4.2.3 Normalized Cuts

Shi and Malik [74] proposed normalized cuts approach as an optimal graphical grouping algorithm. Normalized cuts treats $\mathbf{W}$ as an affinity matrix of a graph V which has two disjoint clusters A and B. The solution to the defined optimal cut criterion is obtained by solving for the second smallest eigenvector $\bar{\mathbf{y}}$ of the generalized eigenvalue system: $(\mathbf{D} - \mathbf{W})\,\bar{\mathbf{y}} = \lambda\,\mathbf{D}\,\bar{\mathbf{y}}$, where $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{ii} = \sum_j \mathbf{W}_{ij}$. The eigenvector $\bar{\mathbf{y}}$ can then be the indicator of the partition and ideally will take two distinct values. One can continue with the same strategy to partition in each subgroup until $\bar{\mathbf{y}}$ fails to have distinct values. Applying the normalized cuts to the separation matrix shown in Figure 4.1(b), we obtain the indicator vector $\bar{\mathbf{y}}$ as shown in Figure 4.1(c). The horizontal axis is the index of feature points, the vertical axis is the value of $\bar{\mathbf{y}}$. The two distinct values of $\bar{\mathbf{y}}$ allows one to tell apart two groups of features right away. It should be pointed out that the ordering of feature points is only for the convenience of showing the separation matrix and results from normalized cuts. In practice, two distinct values of $\bar{\mathbf{y}}$ are obtained for the separation matrix generated from the randomly ordered feature points.

4.3 Mixture Models for 3D Motion

One of the problems of the modified separation matrix method proposed in the previous section is that it is based on some hypothetic spurious motions. We do not use the true motions that can be possibly estimated to segment the scene. The neat separation matrix we obtained so far is mainly because of the spatially well-separated feature positions and velocities in the 2D image plane. When the 3D motions in the scene are such that no partition information can be obtained from 2D feature motions, we have to investigate the case from pure 3D motion classification.

Figure 4.2 is a demonstration of the 3D motion segmentation problem proposed in [94]. Two transparent cylinders are rotating about the same axis, but with opposite directions. Assume that $\frac{N}{2}$ trackable features are placed uniformly on each cylinder.

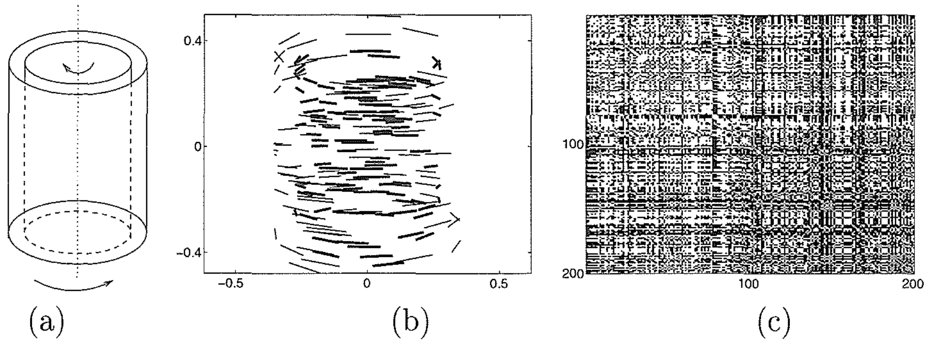


Figure 4.2: **Cylinder Demonstration.** (a) is the structure of two cylinders in space. (b) shows their feature flows in the image sequence when they rotate about the same axis with opposite direction. Different line widths of the flows represent different motions. (c) Separation matrix.

All of them are perspectively projected and can be viewed in the image because of the transparency. The positions and velocities of these N features are shown in Figure 4.2(b). It is very hard to tell the groups from these flows because they are totally mixed up, and that even for one cylinder, the 2D flows of features close to each other in the image may move along opposite directions. The straight separation matrix (see Figure 4.2(c)) fails to provide global grouping information in this demonstration. However, we propose an expectation-maximization algorithm based on 3D motions to solve this kind of difficult segmentation problems.

4.3.1 Motion Segmentation Using Expectation Maximization

We model the segmentation and estimation of multiple motion processes as a maximum likelihood estimation problem. Assume there are G independent motions in the scene. Each motion $g \in (1, \ldots, G)$ is modeled by its motion parameters $\theta_g = (\overline{T}_g, \overline{\Omega}_g)$. The feature points can be viewed as arising from a mixture of these G motion models in some proportions π_g such that $\sum_{g=1}^{G} \pi_g = 1$. Let z_{gi}, a binary variable, indicate the membership of the point P_i belonging to motion model g. If P_i is moving under the motion model g, then $z_{gi} = 1$, otherwise, $z_{gi} = 0$. Denote all the indicator variables for a point P_i as a vector $\overline{Z}_i = [z_{1i} \ z_{2i} \ \ldots \ z_{Gi}]^T$. Let us adopt a few notations for simplicity: the set of motion parameters for mixture models is $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_G)$, the set of model proportions is $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_G)$, and finally the homogeneous coordinate vectors for the imaged positions of point P_i on 2-views, $\overline{\mathbf{x}}_i$ and $\overline{\mathbf{x}}'_i$, are denoted by $\mathbf{x}_i = (\overline{\mathbf{x}}_i, \overline{\mathbf{x}}'_i)$.

The standard maximum likelihood estimation for the motion model parameters is to maximize the log-likelihood of the observed image data $\mathbf{x}_i$ $(i = 1, \ldots, N)$, which is given by

$$\begin{aligned} L(\boldsymbol{\theta}, \boldsymbol{\pi}) &= \log P(\mathbf{x}_1, \ldots, \mathbf{x}_N \,|\, \boldsymbol{\theta}, \boldsymbol{\pi}) && (4.4)\\ &= \sum_{i=1}^{N} \log P(\mathbf{x}_i \,|\, \boldsymbol{\theta}, \boldsymbol{\pi}) && (4.5)\\ &= \sum_{i=1}^{N} \log \sum_{g=1}^{G} P(z_{gi} = 1)\, P(\mathbf{x}_i \,|\, z_{gi} = 1, \theta_g, \pi_g) && (4.6)\\ &= \sum_{i=1}^{N} \log \sum_{g=1}^{G} \pi_g\, P(\mathbf{x}_i \,|\, z_{gi} = 1, \theta_g) && (4.7) \end{aligned}$$

It is difficult to tackle this optimization problem directly since it involves logarithm of sum. The expectation-maximization (EM) algorithm proposed in [14] provides an indirect solution. Let us treat the unknown membership $\overline{Z}_i$ as hidden data which together with observed data $\mathbf{x}_i$ consists of the complete data. Thus the log-likelihood of the complete data is [57]

$$\begin{aligned} L_c(\boldsymbol{\theta}, \boldsymbol{\pi}) &= \log P(\mathbf{x}_1, \ldots, \mathbf{x}_N, \overline{Z}_1, \ldots, \overline{Z}_N \,|\, \boldsymbol{\theta}, \boldsymbol{\pi}) && (4.8)\\ &= \sum_{i=1}^{N} \log P(\mathbf{x}_i, z_{1i}, \ldots z_{Gi} \,|\, \boldsymbol{\theta}, \boldsymbol{\pi}) && (4.9)\\ &= \sum_{i=1}^{N} \log \prod_{g=1}^{G} (P(\mathbf{x}_i, z_{gi} = 1 \,|\, \boldsymbol{\theta}, \boldsymbol{\pi}))^{z_{gi}} && (4.10)\\ &= \sum_{i=1}^{N} \log \prod_{g=1}^{G} (\pi_g\, P(\mathbf{x}_i \,|\, z_{gi} = 1, \theta_g))^{z_{gi}} && (4.11)\\ &= \sum_{i=1}^{N} \sum_{g=1}^{G} z_{gi}(\log \pi_g + \log P(\mathbf{x}_i \,|\, z_{gi} = 1, \theta_g)) && (4.12) \end{aligned}$$

The derivation from equation 4.9 to equation 4.10 comes from the fact that there is one and only one element z_{gi} in the vector $\overline{Z}_i = [z_{1i} \;\; z_{2i} \;\; \ldots \;\; z_{Gi}]^T$ such that $z_{gi} = 1$, and all other elements in this vector are equal to zero.

The EM algorithm proceeds iteratively in two steps, E-step (for expectation) and M-step (for maximization), to maximize a sequence of following functions

$$Q(\boldsymbol{\theta},\, \boldsymbol{\pi},\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]}) = E[L_c(\boldsymbol{\theta},\, \boldsymbol{\pi}) \,|\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]}, \mathrm{x}_1,\, \ldots,\, \mathrm{x}_N] \tag{4.13}$$

$$= \sum_{i=1}^{N} \sum_{g=1}^{G} E[z_{gi} \,|\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]},\, \mathrm{x}_i]\,(\log \pi_g + \log P(\mathrm{x}_i \,|\, z_{gi} = 1,\, \theta_g)) \tag{4.14}$$

where $E[.]$ refers to the expectation value. The parameters $\boldsymbol{\theta}, \boldsymbol{\pi}$ in Q are the ones we are optimizing for, and $\boldsymbol{\theta}^{[k]}, \boldsymbol{\pi}^{[k]}$ are the optimized values from the previous k^{th} iteration. In the E-step, the function $Q(\boldsymbol{\theta},\, \boldsymbol{\pi},\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]})$ is computed, which indeed is to compute the expectation value $E[z_{gi} \,|\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]},\, \mathrm{x}_i]$. In the M-step, the parameters $\boldsymbol{\theta}, \boldsymbol{\pi}$ are updated by maximizing Q. In detail,

E-Step:

$$l_{gi}^{[k]} = E[z_{gi} \,|\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]},\, \mathrm{x}_i] \tag{4.15}$$

$$= P(z_{gi} = 1 \,|\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]},\, \mathrm{x}_i) \tag{4.16}$$

$$= \frac{P(\mathrm{x}_i,\, z_{gi} = 1 \,|\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]})}{\sum_{j=1}^{G} P(\mathrm{x}_i,\, z_{ji} = 1 \,|\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]})} \tag{4.17}$$

$$= \frac{\pi_g^{[k]}\, P(\mathrm{x}_i \,|\, z_{gi} = 1,\, \theta_g^{[k]})}{\sum_{j=1}^{G} \pi_j^{[k]}\, P(\mathrm{x}_i \,|\, z_{ji} = 1,\, \theta_j^{[k]})} \tag{4.18}$$

M-Step:

$$\pi_g^{[k+1]} = \sum_{i=1}^{N} l_{gi}^{[k]} \,/\, N \tag{4.19}$$

$$\theta_g^{[k+1]} = \arg\max_{\theta_g} \sum_{i=1}^{N} l_{gi}^{[k]} \log P(\mathrm{x}_i \,|\, z_{gi} = 1,\, \theta_g) \tag{4.20}$$

EM algorithm starts from random initialization and iteratively maximizes the function Q over the model parameters. It has been proven [14] that

$$\text{if} \qquad Q(\boldsymbol{\theta}^{[k+1]},\, \boldsymbol{\pi}^{[k+1]},\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]}) \geq Q(\boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]},\, \boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]}) \tag{4.21}$$

$$\text{then} \qquad L(\boldsymbol{\theta}^{[k+1]},\, \boldsymbol{\pi}^{[k+1]}) \geq L(\boldsymbol{\theta}^{[k]},\, \boldsymbol{\pi}^{[k]}) \tag{4.22}$$

Therefore EM algorithm guarantees that the log-likelihood of observed data can never decrease after an EM iteration step. This means that EM theory guarantees the series of model parameters $\boldsymbol{\theta}$, $\boldsymbol{\pi}$ obtained by maximizing the function Q converges to those corresponding to a local maximum of $L(\boldsymbol{\theta},\, \boldsymbol{\pi})$. Therefore, this EM approach is a maximum likelihood estimation. The computed l_{gi}, which is the conditional expectation of z_{gi} from equation 4.15, takes on continuous values between 0 and 1 and can be viewed as a soft membership assignment. It tells at each iteration step how likely a point P_i belongs to a motion process g. The segmentation result is finally given after convergence by setting $z_{g^*i} = 1$ and $z_{ji} = 0$, $\forall j \neq g^*$, where $g^* = \arg\max_g l_{gi}$.

4.3.2 3D Motion Constraint

The tricky part left is to build the posterior probabilistic model, $P(\mathbf{x}_i \,|\, z_{gi} = 1,\, \theta_g)$, to judge how well each motion model fits each feature point. To our knowledge, Gaussian distribution is most preferable for EM algorithm with simple solution. If we define a constraint $C_{gi}(\theta_g, \mathbf{x}_i) = 0$ which is satisfied by the true motion and noiseless feature measurements, then we would further assume that the error space of this constraint caused by non-accurate motion estimation or image measurement noises is a Gaussian distribution

$$P(\mathbf{x}_i \,|\, z_{gi} = 1,\, \theta_g) = \frac{1}{\sqrt{2\pi}\,\sigma_g}\, e^{-\frac{C_{gi}^2}{2\sigma_g^2}} \tag{4.23}$$

Equation 4.2 as a constraint is widely used in estimating the rigid motion between two views. One approach is to find the least square estimate of $\overline{\mathbf{q}}$ in the linear space of its elements, and then project it back to the motion space. Alternatively, one can also directly estimate motion parameters instead of $\overline{\mathbf{q}}$ which is a nonlinear minimization

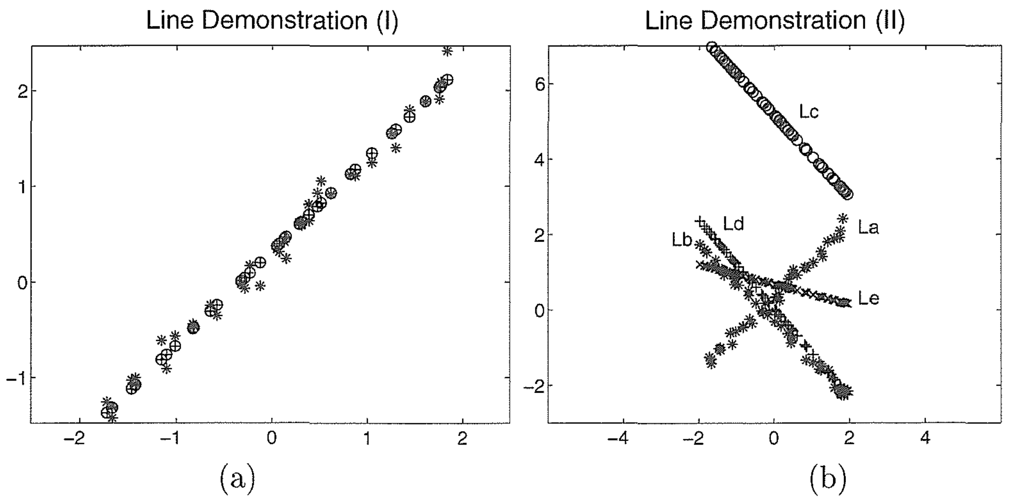


Figure 4.3: **Line Fitting.**

problem: $(\overline{T}^*, \overline{\Omega}^*) = \arg\min_{\overline{T},\overline{\Omega}} \sum_i (\overline{\chi}_i^T \overline{\mathbf{q}}(\overline{T}, \overline{\Omega}))^2$. Intuitively one may want to use this constraint as C_{gi} directly in the segmentation problem since it is a criterion to judge if an estimated motion fits observed data. However, things are not the same in multiple motions as in a single motion case.

Let us demonstrate the difference of fittings between single and multiple groups in a lower dimensional case – line fitting. In Figure 4.3(a), $'*'$s are noisy data generated from a planar line function. To fit a line to these data, we have two schemes. A line can be described as $Ax + By + C = 0$. Similar to the motion estimation from equation 4.2, one method is to find nonzero coefficients A, B, C which minimize the residual of the line function constraint, i.e.,

$$(A^*, B^*, C^*) = \arg \min_{A,B,C:\, A^2+B^2+C^2=1} \sum_i (A\, x_i + B\, y_i + C)^2 \tag{4.24}$$

The minimization is performed under the scale constraint $A^2 + B^2 + C^2 = 1$ because the line function is unique only up to a scale. Construct a matrix E by row-wise stacking the vector $[x_i \;\; y_i \;\; 1]$ for all points. It is easy to prove [17] that the solution for 4.24 is the unit norm eigenvector corresponding to the smallest eigenvalue of $E^T E$. The fitted line is shown in Figure 4.3(a) by marker $'\circ'$ which corresponds to points (x_i, y_i^*), where y_i^* are computed values from the estimated line function.

Another approach is to find the line which minimizes the sum of squares of the distances between points and the line

$$(A^*, B^*, C^*) = \arg \min_{A,B,C:\, A^2+B^2+C^2=1} \sum_i \left(\frac{A\,x_i + B\,y_i + C}{\sqrt{A^2+B^2}}\right)^2 \qquad (4.25)$$

This is a nonlinear minimization problem and the fitted line is represented by marker $'+'$ in Figure 4.3(a). It is clear from this figure that the two methods generate two almost identical fitted lines and both are reasonable solutions to fitting this one line data.

Unfortunately this does not always happen in multi-line fittings. Figure 4.3(b) shows such an example. Another noisy line data L_b is added to the original one L_a, but we try to fit both of them together by only one line. What we expect is the fitting line is either at the middle of two line data L_a and L_b or close to one of them. The results from the two different methods 4.24 and 4.25 are shown as L_c and L_d, respectively, in Figure 4.3(b). One can see that L_c is quite far away from the original data set and L_d is very close to one of the two lines. Line L_c is then not acceptable in this case because it is too far from both lines. Also shown in Figure 4.3(b) is a line L_e which minimizes the error along the y axis and has a little larger error on the point-line distance than L_d. To treat x and y equally, L_d is obviously the best solution. This demonstration is equivalent to the initial step of EM algorithm when all points are assigned with almost the same weight l_{gi} for the motion model g. As we all know, initialization is crucial in the convergence to the true motion.

Motion segmentation can be thought as an extension of the line demonstration from $\mathbb{R}^2$ to $\mathbb{R}^5$ in motion space or $\mathbb{R}^8$ in $\overline{\mathbf{q}}$ space with the scale dimension dropped. In the M-step of EM, the only modification needed to be done is to estimate motion parameters for each model by putting different weights on different features. Therefore, in multiple motion fittings we have to avoid the unnormalized approach which only minimizes the residuals of the linear function. Instead we need to use some normalized criterion similar to the one which minimizes the point-line distance. Therefore, the distance measured on the image plane from an image point to the epipolar line

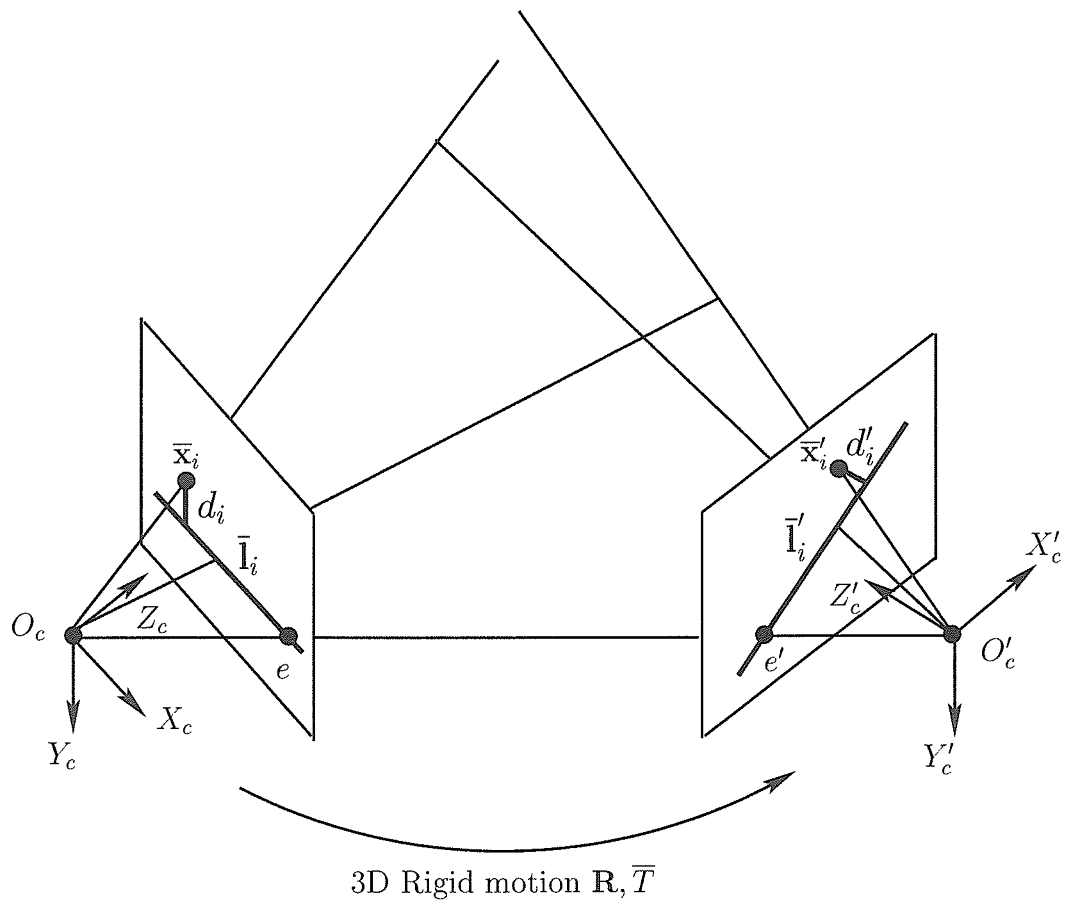

$\overline{\mathbf{x}}_i$
d_i
$\overline{\mathbf{l}}_i$
O_c
Z_c
e
X_c
Y_c
$\overline{\mathbf{x}}'_i$
d'_i
$\overline{\mathbf{l}}'_i$
Z'_c
e'
X'_c
O'_c
Y'_c
3D Rigid motion $\mathbf{R}, \overline{T}$

Figure 4.4: **Epipolar Geometry**.

becomes our choice.

Figure 4.4 illustrates the epipolar geometry. Given the measurements $\overline{\mathbf{x}}_i$ and $\overline{\mathbf{x}}'_i$ of feature positions in two views and the estimated 3D motion, we construct an epipolar line $\overline{\mathbf{l}}'_i$ which is the projection of the line joining O_c and $\overline{\mathbf{x}}_i$ onto the second image plane. The distance between the corresponding feature point measurement $\overline{\mathbf{x}}'_i$ and this epipolar line $\overline{\mathbf{l}}'_i$ in the second image is

$$d'_i = f \frac{|\, \overline{\mathbf{x}}'^T_i \, \mathbf{Q} \, \overline{\mathbf{x}}_i \,|}{\sqrt{(\mathbf{Q}\,\overline{\mathbf{x}}_i)_1^2 + (\mathbf{Q}\,\overline{\mathbf{x}}_i)_2^2}} \tag{4.26}$$

where the notation $(\mathbf{Q}\,\overline{\mathbf{x}}_i)_k$ with subscript k means taking the kth component of that vector. This distance is called *epipolar distance.* If there are no measurement noises on the feature positions and $\mathbf{Q} = (\overline{T}\wedge)\mathbf{R}$ is the true motion, the epipolar distance will be zero and image point $\overline{\mathbf{x}}_i'$ will be on the line $\overline{\mathbf{l}}_i'$. Otherwise, the epipolar distance tells how far the measured point $\overline{\mathbf{x}}_i'$ is from its most likely true position with the current motion estimate if we assume the measurements $\overline{\mathbf{x}}_i$ in the first image is accurate. This nonzero distance d_i' is caused by error in either motion or image measurements, and is measured directly on the image plane. Therefore it is a normalized criterion to be minimized. However, notice that one can also construct another epipolar line $\overline{\mathbf{l}}_i$ by projecting the ray joining the second camera center O_c' and image point $\overline{\mathbf{x}}_i'$ back onto the first image plane. Another distance d_i is generated between image point $\overline{\mathbf{x}}_i$ and this epipolar line $\overline{\mathbf{l}}_i$ in the first image, where the measurements $\overline{\mathbf{x}}_i'$ in the second image is assumed to be accurate. Ideally $\overline{\mathbf{x}}_i$ will be on the epipolar line $\overline{\mathbf{l}}_i$, and d_i is equal to zero if there is no any measurement noise and motion estimate is accurate. Though both are normalized criteria, neither of the two epipolar distances is good enough by itself since when one is minimized the other is usually not minimized at the same time. Therefore, $d_i^2 + d_i'^2$ is used as the constraint to be minimized for motion estimation. This will be our criterion to judge how well the current motion parameters fit the feature measurements $\overline{\mathbf{x}}_i$ and $\overline{\mathbf{x}}_i'$. By dropping the constant focal length, the constraint C_{gi} can be written as

$$C_{gi}^2 = \left(\frac{\overline{\mathbf{x}}_i'^T\,\mathbf{Q}\,\overline{\mathbf{x}}_i}{\sqrt{(\overline{\mathbf{x}}_i'^T\,\mathbf{Q})_1^2 + (\overline{\mathbf{x}}_i'^T\,\mathbf{Q})_2^2}} \right)^2 + \left(\frac{\overline{\mathbf{x}}_i'^T\,\mathbf{Q}\,\overline{\mathbf{x}}_i}{\sqrt{(\mathbf{Q}\,\overline{\mathbf{x}}_i)_1^2 + (\mathbf{Q}\,\overline{\mathbf{x}}_i)_2^2}} \right)^2 \qquad (4.27)$$

Gradient decent method is applied to minimize this nonlinear constraint over its motion parameters in $\mathbb{R}^5$.

In summary, the 3D motion segmentation scheme we propose is as follows:

1. Construct the separation matrix and use normalized cuts to group the features.

2. Check the correctness of the grouping by estimating the motion of each group and examine the generated residues. If all segmented features do belong to this group, the residues should be very small.

3. Otherwise, apply 3D motion based EM algorithm with random initialization to segment the scene. With the proposed probabilistic model for the constraint, the EM algorithm becomes

   E-Step:

$$l_{gi}^{[k]} = \frac{\pi_g^{[k]}\, e^{-C_{gi}^2 / 2\sigma_g^2} \,/\, \sigma_g}{\sum_{j=1}^{G} \pi_j^{[k]}\, e^{-C_{ji}^2 / 2\sigma_j^2} \,/\, \sigma_j} \tag{4.28}$$

   M-Step:

$$\pi_g^{[k+1]} = \sum_{i=1}^{N} l_{gi}^{[k]} \,/\, N \tag{4.29}$$

$$(\overline{\Omega}_g^{[k+1]}, \overline{T}_g^{[k+1]}) = \arg \min_{\overline{\Omega}_g, \overline{T}_g} \sum_{i=1}^{N} l_{gi}^{[k]}\, C_{gi}^2(\overline{\Omega}_g, \overline{T}_g) \tag{4.30}$$

   where C_{gi} is given by equation 4.27.

4.4 Experiments

In this section we demonstrate some experiments that we have done to validate our method. Sections 4.4.1 and 4.4.3 present only the experiments for which the 3^{rd} step of our method (EM based motion segmentation) is needed to perform the segmentation. The other cases are trivial, whenever the separation matrix can clearly separate the motions, the normalized cuts will give the correct solution.

4.4.1 Cylinder Demonstration

The transparent cylinder demonstration mentioned in section 4.3 is a difficult problem in segmentation since there are no 2D cues that can be retrieved. However, it turns out that our 3D motion based EM algorithm works very well on this problem. To make it more challenging, the two cylinders are set to have the same size. This is difficult even for human observers to distinguish the different motions. These two cylinders are generated in 3D space with the same dimensions as 0.5 m radius, 1 m height and 1.5 m far away from the camera, but are rotated to opposite directions about their common axis by 10 degrees between two consecutive frames. Each cylinder is represented by 100 features randomly generated by simulation. The two projected images are generated before and after the rotation assuming the focal length is 1.

Figure 4.5(a) shows the feature positions and velocities. The corresponding separation matrix shown in Figure 4.5(b) does not provide any useful information for segmentation. Figure 4.5(c) shows the grouping result by our EM based motion segmentation assuming no measurement noises to feature positions. The two parts of Figure 4.5(c) indicate two motion models $g = (1, 2)$ for the two opposite rotations and each vertical line represents the value l_{gi} of each feature point at convergence. Similar to Figure 4.2, to visually show the results, the features are reordered such that the first 100 belong to one cylinder and the rest to the other one. Notice that $l_{1i} = 1$ when $i \leq 100$ and $l_{2i} = 1$ when $i > 100$. Therefore, the EM algorithm gives perfect segmentation results for this noiseless example. If independent Gaussian noises with 0.002 standard deviation are added to the image measurements of all features according to the feature tracking scheme [3], the resulted l_{gi} shown in Figure 4.5(e) is not as perfect as that in Figure 4.5(c) due to the noises but the group segmentation is still quite clear.

Another interesting experiment [94] is to abruptly rotate the common axis by an angle θ ($0° \leq \theta \leq 90°$) in $X - Y$ plane while the two cylinders are still oppositely rotated about the axis by 10 degrees between two frames as before. Figure 4.5(d) shows the flow when $\theta = 90°$. It looks messy, but in fact there are only two different 3D motions which are possible to be separated in the motion space. Our 3D EM segmentation scheme also works fairly well in this case. The results are similar to those shown in Figure 4.5(c) and (e) without or with measurement noises. As discussed in [98], the number of models can be estimated at the same time while grouping is done. Figure 4.5(c) and (e) are actually obtained from initialization with a higher number ($g \geq 3$) of models which finally merged into the correct number of groups. This experiment has been repeated several times with different random features and noises, occasionally some spurious models may remain without being merged into the correct groups. However in these cases, there are very few feature points assigned

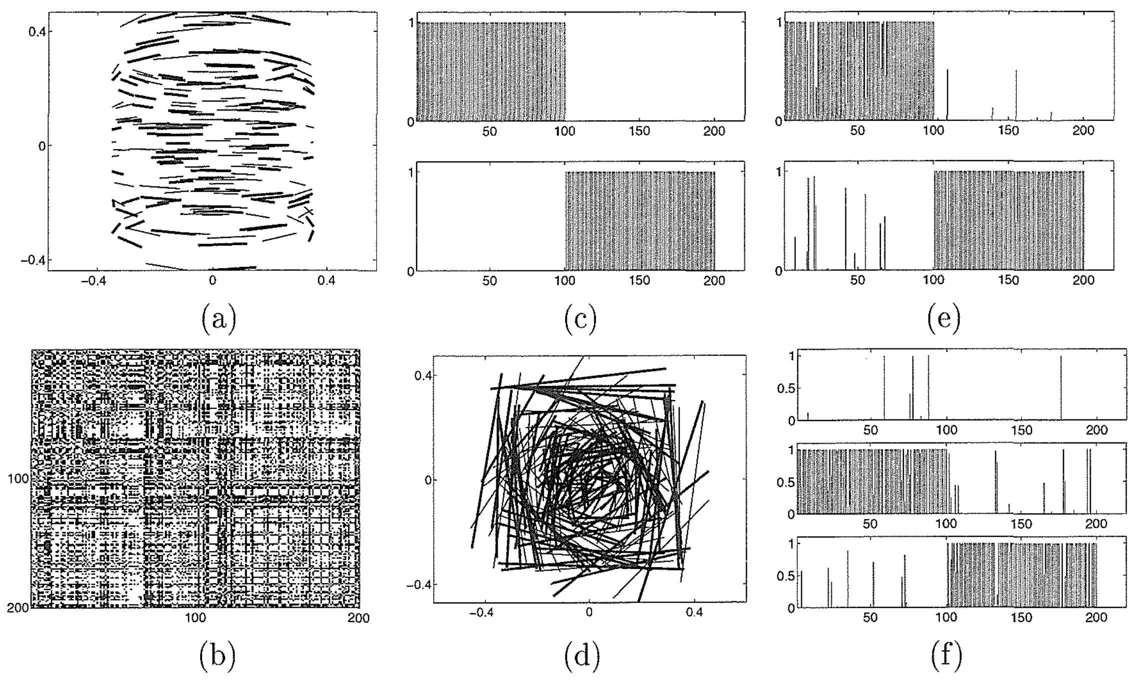


Figure 4.5: **EM Based Motion Segmentation of Two Co-axial Cylinders Rotating to Opposite Directions**. (a) Feature flows for two same-sized cylinders rotating about the same axis to opposite directions. (b) Separation matrix does not collect grouping information. (c)(e) Membership results l_{gi} of the 3D motion segmentation using EM algorithm without or with Gaussian noises (0.002 standard deviation) added to the feature position measurements. (d) Feature flows for abruptly rotating the common axis 90° in X-Y plane between two frames while also rotating cylinders as before. (f) Starting from higher number of models, usually the models will converge to correct number of models as shown in (c) and (e), but occasionally we may get this result too.

to the spurious groups. Figure 4.5(f) shows an example with one spurious group remained. The vast majority of the features are segmented properly.

4.4.2 Rotation, Translation and Full Transformation

The purpose of designing and conducting this experiment is to show how the two different segmentation schemes perform under different situations of data distribution and 3D motions. The schemes are tested on a vast number of synthetic image sequences.

According to the motion transformation equation $\overline{X}_i' = \mathbf{R}\overline{X}_i + \overline{T}$ in 3D space, two series of simulated experiments are generated for two point clusters undergoing different rigid motions. Each experiment is repeated 200 times. In the first series

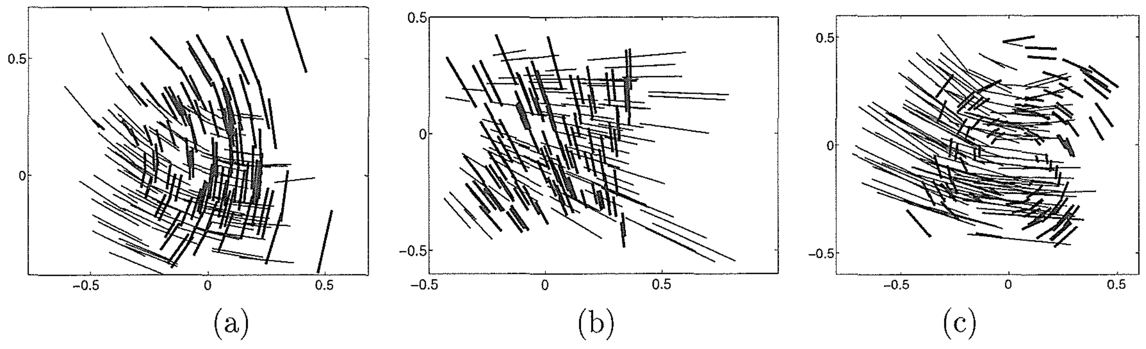


Figure 4.6: **Typical Feature Flow from Two Point Clusters.** (a) with pure rotation. (b) with pure translation. (c) with full transformation.

called 'S1', the structures of both clusters are random points in a $1 \times 1 \times 1m^3$ box whose center is 1.5 m far away from the optical center. The focal length is assumed to be 1 m. The separation matrix approach and EM based motion segmentation scheme are tested under the following different motion conditions:

Pure Rotation. Two clusters of points undergo pure rotations about different axes passing through the optical center, $\mathbf{R}_1 \neq \mathbf{R}_2 \neq \mathbf{I}_3$ and $\overline{T}_1 = \overline{T}_2 = \mathbf{0}_{3\times1}$. In practice, since this is a singular condition with $\overline{\mathbf{q}} = \mathbf{0}_{9\times1}$, we add very small translation to $\overline{T}_1$ and $\overline{T}_2$ to make sure the motion can be estimated once grouping is done. See Figure 4.6(a).

Pure Translation. Two clusters of features undergo the pure translations without any rotation, $\mathbf{R}_1 = \mathbf{R}_2 = \mathbf{I}_3$ and $\overline{T}_1 \neq \overline{T}_2 \neq \mathbf{0}_{3\times1}$. See Figure 4.6(b).

Full Transformation. The general case, $\mathbf{R}_1 \neq \mathbf{R}_2 \neq \mathbf{I}_3$ and $\overline{T}_1 \neq \overline{T}_2 \neq \mathbf{0}_{3\times1}$. See Figure 4.6(c).

All the motion parameters in each scenario are chosen as uniformly distributed random variables in the range $(-0.25\quad 0.25)$ which can generate motions both small and large.

There are two methods to determine the variances $\sigma_g, g \in (1, G)$ in the constraint probabilistic model. One is to estimate each σ_g at every iteration in order to keep it continuously updated. Another way is to fix it according to some prior knowledge. Each experiment set is run twice, with the variances updated or fixed at every step.

The second series 'S2' is the same as the first one except that the original two point clusters are already well separated from each other in space. One is shifted to be centered at $[0.5 \quad 0.5 \quad 1.5]$, and the other one at $[-0.8 \quad 0.3 \quad 1.5]$.

A statistical summary of all the experiments is shown in table 4.1.

	Pure $\mathbf{R}$	Pure $\mathbf{T}$	Full $\mathbf{R}+\mathbf{T}$
S1, SM	67%	60%	41%
S1, EM, Upσ	95%	5%	77%
S1, EM, Fxσ	97%	5%	88%
S2, SM,	90%	99%	52%
S2, EM, Upσ	42%	38%	45%
S2, EM, Fxσ	50%	51%	66%

Table 4.1: **Percentage of Successful Segmentation** under different conditions. 'EM': EM based scheme. 'SM': Separation matrix scheme. 'Upσ': Update variance. 'Fxσ': Fix variance.

The table shows that the 3D motion based EM segmentation scheme works better for 'S1' series than 'S2' under the conditions of pure rotation and full transformation while the modified separation matrix method shows opposite results. Under pure translation, the modified separation matrix outperforms the EM algorithm. The EM algorithm works better for pure rotation than translation. To some extent, the two methods are compensative to each other. Therefore, their combination, as outlined in section 4.3, will enhance the overall segmentation performance under most situations. Table 4.1 also illustrates that the EM approach with properly fixed variance has higher success rate than that with updated variance.

4.4.3 Experiments with a Real Image Sequence

A real image sequence with two moving objects is acquired and used for this experiment. In order to obtain enough good features, objects are chosen with checkerboard patterns. The object in the background moves both forward and backward. The foreground object is a box which is hung by a string and can rotate about the vertical axis. The sequence is shown to several people to test their ability in segmentation. It turns out that people have no trouble to separate the two moving objects. The key cues they can rely on are 3D motion, texture, shape, etc.

Figures 4.7(a) and (b) are two image frames extracted from the sequence. They are 3 frames apart so that there is large enough motion in between. Assume the feature positions and their correspondents are known (here for convenience, the data are obtained manually), the feature flows are observed as shown in Figure 4.7(c). The background flow is divergent. The magnitudes of the flows of both the background and foreground objects are not uniform and spreading in a large range. Therefore, it is very difficult to perform the segmentation by 2D flow. So the separation matrix scheme, as the first step in our algorithm, does not give well-segmented groups. To illustrate the result, we reorder the features of the two groups without mixing them. The background features are in the first group. In Figure 4.7(d) the separation matrix seems to show obvious lines between the four blocks, but inside each of at least three, it is not even close to uniformly dark or white as the expected. Consequently, the normalized cuts vector (Figure 4.7(e)) does not separate the two groups as two distinct values.

Fig 4.7(f) shows the membership results of the two groups using the 3D motion based EM segmentation scheme. The two clusters corresponding to the feature positions in Figure 4.7(b), marked by $'o'$ and $'*'$, respectively, are shown in Figure 4.7(g). Though a few features are classified incorrectly, they do not have a membership value as high as the others. Collecting the most likely members for each group and estimating the undergoing motion, the depth of each feature can also be recovered. Since the relative scale factor between the two objects is known as ground truth, the structures of two objects with correct relative position can be reconstructed. The top view of the reconstructed structures is shown in Figure 4.7(h). The background plane is apparently not parallel to the X-Y plane, so its feature points have variant depths. There are a few features on the background plane whose depth reconstructions are somehow

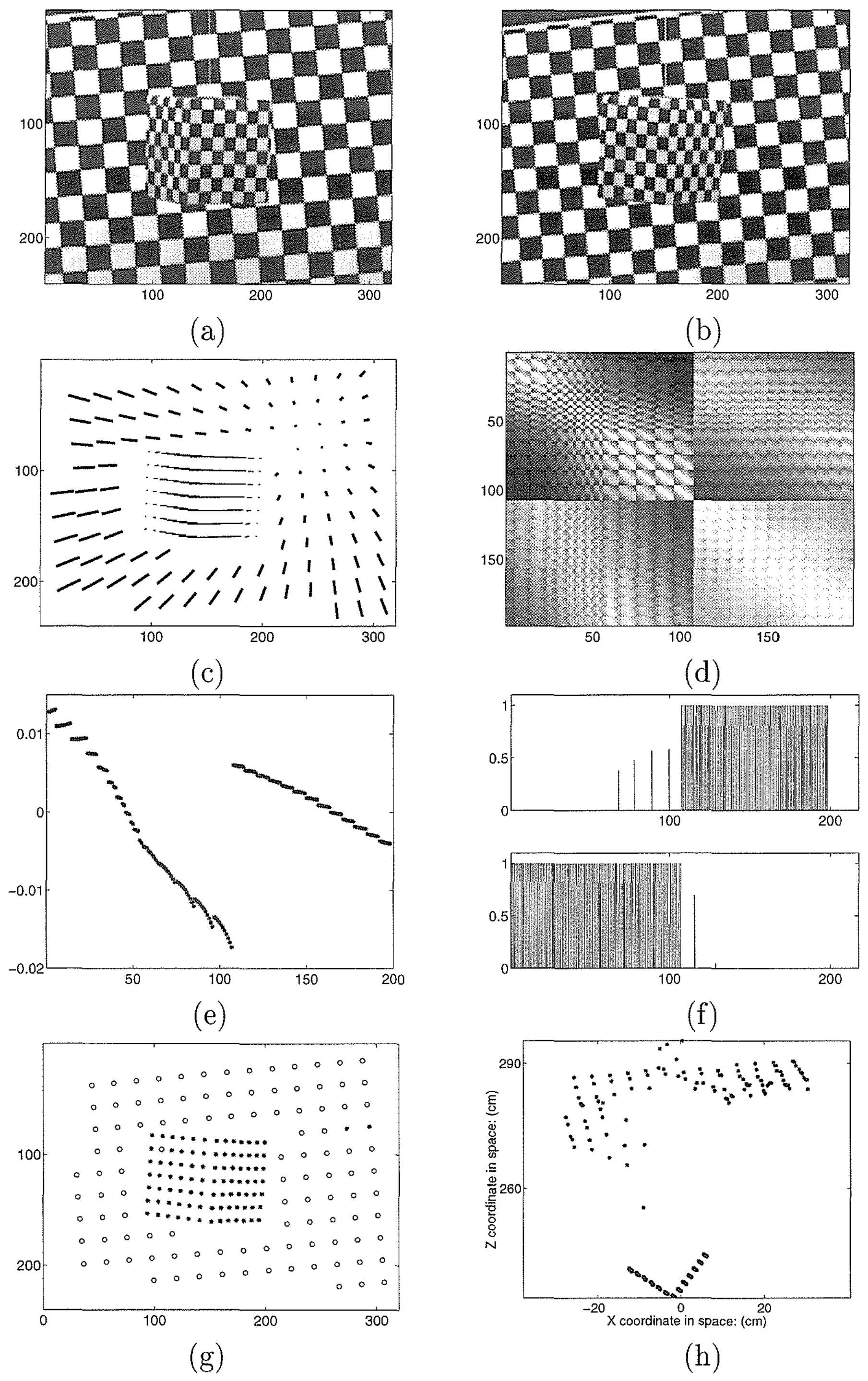


Figure 4.7: **Plane-box Motion Segmentation.** (a)-(b) Two frames in the image sequence. (c) Feature flows between frame (a) and (b). (d) Separation Matrix. (e) The indicating vector of normalized cuts cannot be separated as two distinct values. (f) Membership result of the motion segmentation scheme based on EM. (g) Same grouping result is shown for the features in frame (b). (h) Reconstructed scene structure.

ill conditioned in this case. The estimated structure for the foreground object looks excellent as two perpendicular planes.

In this experiment, using matlab on Ultra 2/200 sun station, the EM approach takes 33 seconds to converge after 31 iterations, while the separation matrix method only takes about 3 seconds but failed in yielding any good result.

4.5 Conclusion

Motion segmentation is an essential step in the success of motion analysis. In this chapter, we described a combination method of improved separation matrix scheme and 3D motion based EM algorithm for scene segmentation concerning multiple moving objects. It is showed that the improved separation matrix by applying normalized cuts afterwards is an efficient approach in dealing with some motion segmentation problems. However, there are still lots of other very difficult cases where separation matrix scheme cannot provide correct grouping information any more. For these cases, we have developed a 3D motion based EM algorithm for scene segmentation. Experiments illustrate that to some extent these two methods are compensative to each other. Therefore, the combination approach we proposed herein is a more powerful tool for scene segmentation.

Chapter 5 Human Motion Detection

In Chapters 3 and 4, we discussed the general 3D motion segmentation and estimation problem. It is said to be "general" because there is no more assumption about the objects other than that each cluster of their representative feature points undergoes rigid motion. Here an individual object is not of the concern, but each individual 3D motion process. As long as objects undergo the same rigid motion, for example, several different objects consisting of a still environment through which a camera is navigating, they are clustered as one single group and the unique undergone 3D motion is recovered. However, other factors of objects are not considered, such as object type, object shape, mutual relationship between feature points, etc.

More often in practice we may be interested in a particular type of objects and carry on more interesting tasks on their motion analysis such as detection, tracking and recognition, etc. Human motion is one of the most common things we observe everyday and therefore attracts our attention to study. We are going to address the problem of human motion detection in this chapter, and devote the next one to human action recognition.

The detection and recognition of human motion throughout these two chapters will only be based on 2D image sequences and the behind 3D model is not going be considered.

5.1 Introduction and Motivation

Perceiving the motion of a human body is not an easy job. First of all, the human body is richly articulated – even a simple stick model describing the poses of arms, legs, torso and head requires more than 20 degrees of freedom. The body moves in 3D and this makes the estimation of these degrees of freedom a challenge in a monocular setting [26, 41]. Image processing is also a challenge. People wear clothes which may be loose and textured, and part of the body is usually self-occluded. All these make it difficult to identify limb boundaries, and even more so to segment the major parts of the body. However, it is surprising how the human visual system has evolved to be so good at perceiving Johansson's stimuli [45, 61] – a fairly poor information where each joint of the body is shown as a moving dot.

Human motion perception may be divided into two phases: first, detection and, possibly, segmentation; and then tracking. Of the two, tracking has recently been the subject of much attention, and considerable progress has been made on it [67, 66, 26, 28, 10]. On the contrary, detection (given two frames: is there a human body, and where?) remains an open problem. Current trackers have to be initialized either by hand, or by ad hoc heuristics. Song et al. [81] have focused on detection in the context of Johannson stimuli. A method was proposed based on probabilistic modeling of human motion and on modeling the dependency of the motion of body parts with a triangulated representation, which makes it possible to solve the combinatorial problem of labeling body parts in polynomial time. Excellent and efficient performance of the method has been demonstrated on a number of motion sequences.

However, the application of that work is limited to Johansson stimuli where all moving dots belong to the human body and there is no clutter and very limited occlusion. In a realistic situation there is no guarantee that the body joints constitute good features that can be tracked easily by the early vision front-end. Moreover, significant occlusion and possibly large amount of moving clutters may be present. To solve these complicated problems, here we propose a scheme that extends Song's work to real images. The task we carry on is to detect human bodies by their motion patterns in monocular real image sequences in which extraneous motions and occlusions may be present. Our method does not rely on segmentation, or grouping, and it is assumed that the vision front-end is limited to observing the motion of key feature points between pairs of frames. Features are not assumed to be tracked for more than two frames. Our algorithm is based on learning an approximate joint probabilistic model of the positions and velocities of different body features. Detection is performed by comparing the sum of observation likelihoods from all possible labelings with a preset threshold. The localization results from the algorithm may be used to compute 3D pose as in [26, 41]. The experiments on a dozen of walking sequences indicate that the algorithm is accurate and efficient.

The majority work presented in this chapter was published in [80] and I primarily worked in the experimental design and execution for this project.

5.2 System Overview

Given two consecutive image frames, the goal is to detect whether a moving human body is present. A human body is represented by its joints. As shown in Figure 5.1, our system requires a training phase. For this we first manually construct a training set containing positions and velocities of labeled joints on the human body in a number of motion sequences. A probabilistic model of human motion is then learned from the training set. This model contains the probability density function of positions and velocities of joint triplets.

The system has a feature-tracking front-end to measure at run time the positions and velocities of all the observable features between two frames. From these features, we first detect if there is a person in the scene by summing over the observation probabilities from all possible labelings. Localization is further done by finding the labeling of all features which maximizes the likelihood of the probabilistic model.

5.3 Approach

The set of feature points and associated velocities can be obtained by applying the feature detection and tracking scheme (see section 2.6) to the entire image. An example is shown in Figure 5.2. In the following, we will address two problems: *detection* – if there is a person in the scene; and *localization* – finding the most human-like configuration, i.e, the best labeling, given a set of features.

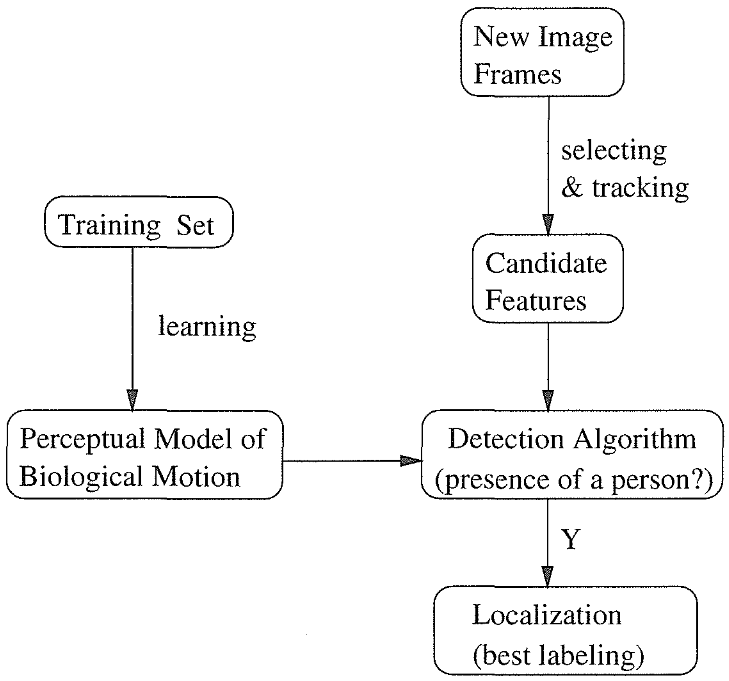

New Image Frames
selecting & tracking
Candidate Features
Training Set
learning
Perceptual Model of Biological Motion
Detection Algorithm (presence of a person?)
Y
Localization (best labeling)

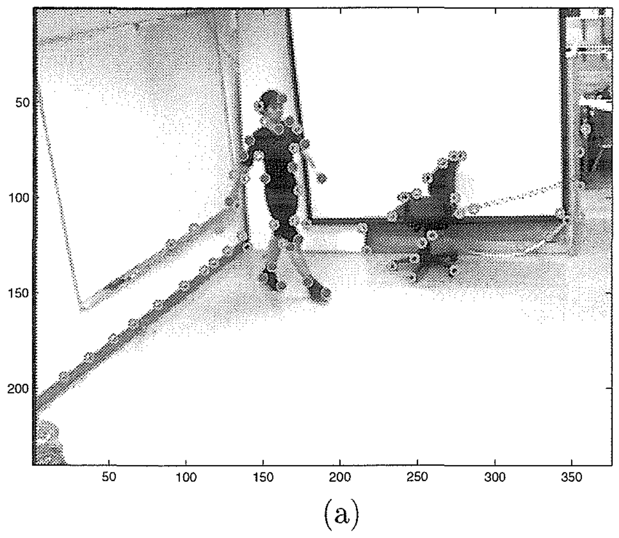
Figure 5.1: **System Overview.**

5.3.1 Notation

Suppose that we observe N points (as in Figure 5.2), and $\overline{X} = [X_1 \ \ X_2 \ \ \ldots \ \ X_N]$ is the vector of measurements. Each feature measurements X_i $(1 \leq i \leq N)$ include the two-dimensional positions and two-dimensional velocities of that feature on the image plane. Let O_1 denote that a person is present in the image, and O_0 denote the opposite case. The detection task is to determine whether the ratio

$$\frac{P(O_1 \,|\, \overline{X})}{P(O_0 \,|\, \overline{X})} = \frac{P(\overline{X} \,|\, O_1)\, P(O_1) \,/\, P(\overline{X})}{P(\overline{X} \,|\, O_0)\, P(O_0) \,/\, P(\overline{X})} = \frac{P(\overline{X} \,|\, O_1)}{P(\overline{X} \,|\, O_0)} \cdot \frac{P(O_1)}{P(O_0)} \tag{5.1}$$

is greater than 1 or not. If we assume the priors are equal, the second term of the above equation is 1. Let $\mathcal{S}_{\text{body}} = \{LW, LE, LS, H, \ldots RT\}$ be the set of M body parts, for example, LW is the left wrist, RT is the right toe, etc., and BG be the background label. In addition, let $\overline{L} = [L_1 \ \ L_2 \ \ \ldots \ \ L_N]$ denote a possible labeling, where $L_i \in \mathcal{S}_{\text{body}} \cup \{BG\}$ is the label of X_i $(1 \leq i \leq N)$. Assume $\mathcal{L}$ is all the possible



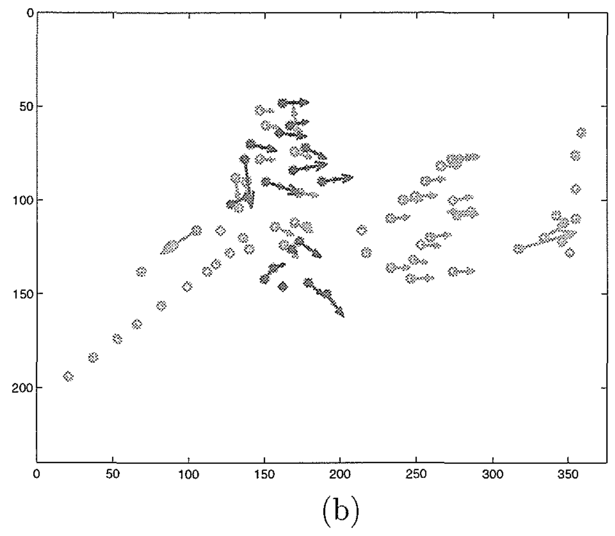


Figure 5.2: **Illustration of the Approach.** For a given image (a), features are first selected and then tracked to the next frame. Dots in (a) are the features, and (b) shows the features with velocities. From all the candidate feature points (with positions and velocities), we first want to detect if there is a person in the scene and then find the best labeling – the most human-like configuration (dark dots in (a) and (b)) according to a learned probabilistic model.

labelings when a person is present (O_1), then

$$\begin{aligned} P(\overline{X} \mid O_1) &= \sum_{\overline{L} \in \mathcal{L}} P(\overline{X}, \overline{L} \mid O_1) \\ &= \sum_{\overline{L} \in \mathcal{L}} P(\overline{X} \mid \overline{L}, O_1)\, P(\overline{L} \mid O_1) \end{aligned} \tag{5.2}$$

When there is no person in the scene, $\overline{L}_0 = [BG \quad BG \quad \ldots \quad BG]$ is the only possible labeling. Then,

$$\begin{aligned} P(\overline{X} \mid O_0) &= P(\overline{X}, \overline{L}_0 \mid O_0) \\ &= P(\overline{X} \mid \overline{L}_0, O_0)\, P(\overline{L}_0 \mid O_0) \\ &= P(\overline{X} \mid \overline{L}_0, O_0) \end{aligned} \tag{5.3}$$

If there is not any prior information about the labeling, then it can be assumed that $P(\overline{L} \mid O_1) = 1 / |\mathcal{L}|$ for any labeling $\overline{L}$, where $|\mathcal{L}|$ is the number of possible labelings. To compute equation 5.1, the rest is to estimate $\sum_{\overline{L} \in \mathcal{L}} P(\overline{X} \mid \overline{L}, O_1)$ and $P(\overline{X} \mid \overline{L}_0, O_0)$.

Given a labeling $\overline{L}$, each point feature i has a corresponding label L_i. Therefore given this label, each feature measurement X_i can also be written as X_{L_i}, which is the measurement corresponding to a specific body part with label L_i. For example if $L_i = LW$, i.e., the i^{th} label is associated with the left wrist, then $X_i = X_{LW}$ is the position and velocity of the left wrist.

Let us define

$$\begin{aligned} \overline{\mathcal{L}}_{\text{body}} &= \{L_i;\, i \in 1, \ldots, N\} \cap \mathcal{S}_{\text{body}} \\ & \qquad \text{set of body parts appearing in } \overline{L} \\ \overline{X}_{\text{body}} &= [X_{i_1} \;\; X_{i_2} \;\; \ldots \;\; X_{i_K}] \\ & \qquad \text{such that } \{L_{i_1}, L_{i_2}, \ldots, L_{i_K}\} = \overline{\mathcal{L}}_{\text{body}} \\ \overline{X}_{\text{bg}} &= [X_{j_1} \;\; X_{j_2} \;\; \ldots \;\; X_{j_{N-K}}] \\ & \qquad \text{such that } L_{j_1} = L_{j_2} = \ldots = L_{j_{N-K}} = BG \end{aligned}$$

where K is the number of body parts appearing in $\overline{L}$. If we assume that the positions and velocities of the visible body parts are independent of those of clutter points, then

$$\begin{aligned} P(\overline{X} \,|\, \overline{L},\, O_1) &= P_{\overline{\mathcal{L}}_{\text{body}}}(\overline{X}_{\text{body}})\, P_{\text{bg}}(\overline{X}_{\text{bg}}) && (5.4) \\ P(\overline{X} \,|\, \overline{L}_0,\, O_0) &= P_{\text{bg}}(\overline{X}) && (5.5) \end{aligned}$$

where $P_{\overline{\mathcal{L}}_{\text{body}}}(\overline{X}_{\text{body}})$ is the marginal probability density of the whole body according to $\overline{\mathcal{L}}_{\text{body}}$. If independent uniform background noise is assumed, $P_{\text{bg}}(\overline{X}_{\text{bg}}) = (1/S)^{N-K}$, where $N-K$ is the number of background points, and S is the volume of the possible domain of X_i. We will use this assumption about background features throughout this chapter. Under this assumption, part of the background terms in $P(\overline{X} \,|\, \overline{L},\, O_1)$ and $P(\overline{X} \,|\, \overline{L}_0,\, O_0)$ can be cancelled out so that detection can be performed by setting a threshold to the summation of the "modified" foreground likelihoods without accurately estimating the background probabilities. More details of the procedure are explained below.

5.3.2 Summation of Likelihoods

We first consider the problem that there are no missing body parts. Under this situation, if a person is present, then all the body parts can be seen, i.e., $\overline{\mathcal{L}}_{\text{body}} = \mathcal{S}_{\text{body}}$. From the above discussion, it follows that if the background (clutter) features are assumed to be independent and uniform, then the detection without occlusion is reduced to calculating

$$\frac{1}{|\mathcal{L}|} \sum_{\overline{L} \in \mathcal{L}} P_{\mathcal{S}_{\text{body}}}(\overline{X}_{\text{body}}) \tag{5.6}$$

If the summation is done simply as all the terms appear, the computational cost is exponential with regard to the number of body parts (M). Thus the brute-force summation is computationally prohibitive. The method proposed in [81] provides a way to approximate the foreground probability density $P_{\mathcal{S}_{\text{body}}}(\overline{X}_{\text{body}})$ so that the summation can be calculated efficiently. By using the kinematic chain structure of a human body, the whole body can be decomposed into parts as shown in Figure 5.3. If the appropriate conditional independence (Markov property) is valid, then

$$\begin{aligned} P_{\mathcal{S}_{\text{body}}}(\overline{X}_{\text{body}}) &= P_{LW,LE,LS}(X_{LW} \mid X_{LE}, X_{LS})\, P_{LE,LS,LH}(X_{LE} \mid X_{LS}, X_{LH}) \\ &\quad \cdots P_{RA,RHE,RT}(X_{RA}, X_{RHE}, X_{RT}) \\ &= \left(\prod_{t=1}^{T-1} P_t(X_{A_t} \mid X_{B_t}, X_{C_t})\right) P_T(X_{A_T}, X_{B_T}, X_{C_T}) \end{aligned} \quad (5.7)$$

where T is the number of triangles in the decomposition shown in Figure 5.3, t is the triangle index, and A_t is the first label associated to triangle t. The structure of the decomposable graph ([2, 81]) allows the summation to be rewritten as follows:

$$\begin{aligned} \sum_{\overline{L}\in\mathcal{L}} P_{\mathcal{S}_{\text{body}}}(\overline{X}_{\text{body}}) &= \sum_{\overline{L}\in\mathcal{L}} \left(\prod_{t=1}^{T-1} P_t(X_{A_t} \mid X_{B_t}, X_{C_t})\right) P_T(X_{A_T}, X_{B_T}, X_{C_T}) \\ &= \sum_{X_{A_T}, X_{B_T}, X_{C_T}} P_T(X_{A_T}, X_{B_T}, X_{C_T}) \sum_{X_{A_{T-1}}} \cdots \\ &\quad \sum_{X_{A_2}} P_2(X_{A_2} \mid X_{B_2}, X_{C_2}) \sum_{X_{A_1}} P_1(X_{A_1} \mid X_{B_1}, X_{C_1}) \end{aligned} \quad (5.8)$$

Finally the summation in equation 5.8 can be done by an algorithm similar to dynamic programming [81, 2]. Let

$$\begin{aligned} \Psi_t(X_{A_t}, X_{B_t}, X_{C_t}) &= P_{A_t \mid B_t C_t}(X_{A_t} \mid X_{B_t}, X_{C_t}) \\ &\quad \text{for} \quad 1 \le t \le T-1 \end{aligned} \quad (5.9)$$

$$\begin{aligned} \Psi_t(X_{A_t}, X_{B_t}, X_{C_t}) &= P_{A_T B_T C_T}(X_{A_T}, X_{B_T}, X_{C_T}), \\ &\quad \text{for} \quad t = T \end{aligned} \quad (5.10)$$

be the cost function associated with each triangle, then the summation algorithm can be described as follows:

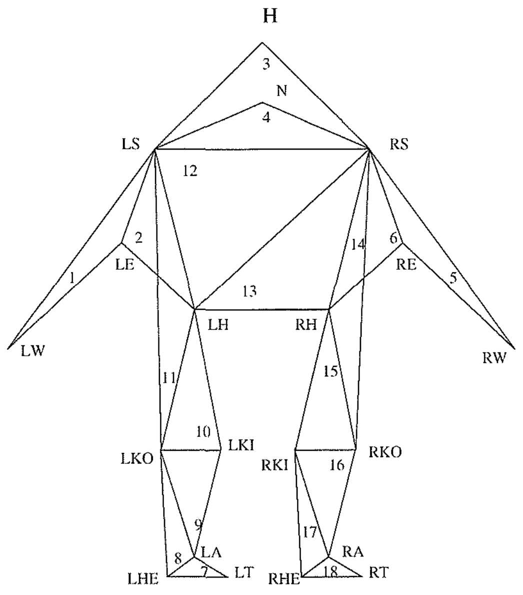


Figure 5.3: **Decomposition of the Human Body into Triangles.** 'L' and 'R' in label names indicate left and right. H:head, N:neck, S:shoulder, E:elbow, W:wrist, H:hip, KI:inside knee, KO:outside knee, A:ankle, HE:heel and T:toe. The numbers inside triangles show the order in which the algorithm proceeds.

Stage 1: for every pair (X_{B_1}, X_{C_1}),

Compute $\Psi_1(X_{A_1}, X_{B_1}, X_{C_1})$ for all possible X_{A_1}

Define $T_1(X_{A_1}, X_{B_1}, X_{C_1})$ the total value so far

Let $T_1(X_{A_1}, X_{B_1}, X_{C_1}) = \Psi_1(X_{A_1}, X_{B_1}, X_{C_1})$

Store $\Gamma_1(X_{B_1}, X_{C_1}) = \sum_{X_{A_1}} T_1(X_{A_1}, X_{B_1}, X_{C_1})$

```
**Stage t**, $2 \le t \le T$: for every pair $(X_{B_t}, X_{C_t})$

        Compute $\Psi_t(X_{A_t}, X_{B_t}, X_{C_t})$ for all possible $X_{A_t}$
        Compute the total value so far (till stage t):
          — Define $T_t(X_{A_t}, X_{B_t}, X_{C_t})$ the total value so far.
          — Initialize $T_t(X_{A_t}, X_{B_t}, X_{C_t}) = \Psi_t(X_{A_t}, X_{B_t}, X_{C_t})$
          — If edge $(A_t, B_t)$ is contained in previous stages and
            $\tau$ is the last one of such stages, multiply $T_t(X_{A_t}, X_{B_t}, X_{C_t})$ by
            $\Gamma_\tau(X_{A_t}, X_{B_t})$ or $\Gamma_\tau(X_{B_t}, X_{A_t})$ if the edge is reversed
          — Likewise, multiply $T_t(X_{A_t}, X_{B_t}, X_{C_t})$ by the values of
            the last stage of previous ones containing
            edge $(A_t, C_t)$ and edge $(B_t, C_t)$ respectively
        Store $\Gamma_t(X_{B_t}, X_{C_t}) = \sum_{X_{A_t}} T_t(X_{A_t}, X_{B_t}, X_{C_t})$
```

When stage T calculation is complete, the overall sum can be obtained by

$$\sum_{\overline{L}\in\mathcal{L}} P_{S_{\text{body}}}(\overline{X}_{\text{body}}) = \sum_{X_{B_T}, X_{C_T}} \Gamma_T(X_{B_T}, X_{C_T}) \tag{5.11}$$

The computational complexity of the above procedure is of the order of $M * N^3$.

5.3.3 Detection and Localization with Occlusion

In the case of no occlusion, detection can be done easily by setting a threshold to $(1/|\mathcal{L}|) \cdot \sum_{\overline{L}\in\mathcal{L}} P_{S_{\text{body}}}(\overline{X}_{\text{body}})$ obtained from the previous section. Assuming equal priors and independent and uniform background features, localization and labeling can be obtained by finding the labeling $\overline{L}^*$

$$\begin{aligned}
\overline{L}^* &= \arg\max_{\overline{L}\in\mathcal{L}} P(\overline{L}\,|\,\overline{X},\,O_1) \\
&= \arg\max_{\overline{L}\in\mathcal{L}} P(\overline{X}\,|\,\overline{L},\,O_1)\,P(\overline{L}\,|\,O_1)\,/\,P(\overline{X}) \\
&= \arg\max_{\overline{L}\in\mathcal{L}} P(\overline{X}\,|\,\overline{L},\,O_1) \\
&= \arg\max_{\overline{L}\in\mathcal{L}} P_{\mathcal{S}_{\text{body}}}(\overline{X}_{\text{body}}) \qquad (5.12)
\end{aligned}$$

This optimization can be done by dynamic programming as in [81, 82].

When some body parts are occluded, the foreground probability $P_{\overline{\mathcal{L}}_{\text{body}}}(\overline{X}_{\text{body}})$ is the marginalized version of $P_{\mathcal{S}_{\text{body}}}(\overline{X}_{\text{body}})$ – marginalization over the missing body parts. If we assume that the background features are independent and uniformly distributed, detection can be done by thresholding

$$\frac{1}{|\mathcal{L}|}\sum_{\overline{L}\in\mathcal{L}} P_{\overline{\mathcal{L}}_{\text{body}}}(\overline{X}_{\text{body}})\,(1/S)^{M-K_{\overline{L}}} \qquad (5.13)$$

where M is the total number of body parts, $K_{\overline{L}}$ is the number of body parts present in labeling $\overline{L}$, and $1/S$ is the volume of the space X_i lies in. If the local cost function $\Psi_t(X_{A_t},\,X_{B_t},\,X_{C_t})$ associated with triangle t, $(1 \le t \le T-1)$, is defined as

- $P_{A_t\,|\,B_t,C_t}(X_{A_t}\,|\,X_{B_t},X_{C_t})$ if all the three body parts are observed
- $1/S$ if A_t is missing or two or three of A_t, B_t, C_t are missing
- $P_{A_t\,|\,C_t}(X_{A_t}\,|\,X_{C_t})$ or $P_{A_t\,|\,B_t}(X_{A_t}\,|\,X_{B_t})$ if B_t or C_t is missing and the other two body parts are observed

With such definition similarly applied to the last triangle T, the summation algorithm described in section 5.3.2 can also be used to calculate expression 5.13.

Similar to equation 5.12, the localization and labeling can be found by

$$\begin{aligned}
\overline{L}^* &= \arg\max_{\overline{L}\in\mathcal{L}} P(\overline{L}\,|\,\overline{X},\,O_1) \\
&= \arg\max_{\overline{L}\in\mathcal{L}} P_{\overline{\mathcal{L}}_{\text{body}}}(\overline{X}_{\text{body}})(1/S)^{M-K_{\overline{L}}} \qquad (5.14)
\end{aligned}$$

Given the above described local cost function, dynamic programming can be used to get the optimal labeling.

The detailed analysis of and explanation to equation 5.13 and 5.14 can be found in [82]. One intuitive explanation is that for each triangle, the dimensions of the local cost functions are the same for different numbers of missing body parts, which makes it reasonable to sum (or get the maximum of) them locally. Also, the dimension of the domain of $P_{\overline{\mathcal{L}}_{\text{body}}}(\overline{X}_{\text{body}})\,(1/S)^{M-K_{\overline{L}}}$ is fixed regardless of the number of candidate features and the number of missing body parts in the labeling $\overline{L}$, so that we can directly compare the likelihood of different hypotheses, even these from different images.

Another way to perform detection [82] is to first get the most likely labeling (the labeling with highest $P_{\overline{\mathcal{L}}_{\text{body}}}(\overline{X}_{\text{body}})\,(1/S)^{M-K_{\overline{L}}}$), then compare the likelihood of such labeling to a threshold. If the likelihood is higher than the threshold, then a person is believed to be present. Both methods are tested. Their performances are compared and presented in the experiment section.

5.3.4 Using Information from Multiple Frames

So far, we have assumed that we can only use information from two consecutive frames to obtain positions and velocities of a number of features. In this section we would like to extend our previous results to the case where multiple frames are available. However, in order to maintain generality, we assume that it is impossible to track features across more than 2 frames. This is a simplified model for the situation where, due to extreme body motion and/or loose and textured clothing, tracking is extremely unreliable and each individual feature's lifetime is very limited. Neri et al. [61] use a similar assumption when conducting their psychophysical investigation in biological motion perception in the human visual system.

Let $P(O\,|\,\overline{X})$ denote the probability when a person is present given $\overline{X}$ observed. As before, we use the approximation that $P(O\,|\,\overline{X})$ is proportional to $\Phi(\overline{X})$ which is defined as $\Phi(\overline{X}) \overset{\text{def}}{=} (1/|\mathcal{L}|)\sum_{\overline{L}\in\mathcal{L}} P_{\overline{\mathcal{L}}_{\text{body}}}(\overline{X}_{\text{body}})\,(1/S)^{M-K_{\overline{L}}}$. Now if we have n

observations $\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_n$, then the decision depends on

$$\begin{aligned} P(O \,|\, \overline{X}_1, \overline{X}_2, \ldots, \overline{X}_n) &= \frac{P(\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_n \,|\, O)\, P(O)}{P(\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_n)} \\ &= \frac{P(\overline{X}_1 \,|\, O)\, P(\overline{X}_2 \,|\, O) \ldots P(\overline{X}_n \,|\, O)\, P(O)}{P(\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_n)} \end{aligned} \quad (5.15)$$

The last line of equation 5.15 holds if observations $\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_n$ are assumed to be independent. Assume the priors are equal, $P(O \,|\, \overline{X}_1, \overline{X}_2, \ldots, \overline{X}_n)$ can then be represented by $P(\overline{X}_1 \,|\, O)\, P(\overline{X}_2 \,|\, O) \ldots P(\overline{X}_n \,|\, O)$, which is proportional to $\prod_{i=1}^{n} \Phi(\overline{X}_i)$. If a threshold is set to $\prod_{i=1}^{n} \Phi(\overline{X}_i)$, then detection can be performed given multiple observations $\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_n$. With more information to exploit, it is expected that better detection will be resulted.

5.4 Experiments

The image sequences used in the experiments are captured by a CCD camera at 30 Hz. There are three different types of motions: (1) A subject walks from the rear left corner to the front right corner, facing about 60 degrees away from the front view (middle column of Figure 5.4). For this motion, we have about 1000 frames (8 sequences, each has around 120 frames) as the training set, and another 1500 frames (12 sequences) as testing set. (2) A chair moves from left to right, about 1000 frames (8 sequences) (right column of Figure 5.4). (3) While a subject walks as in the motion type (1), a chair also moves as a background moving object (left column of Figure 5.4). 2000 frames (16 sequences) have been collected. The goal is to detect if there is a person walking in the scene and further localize and label the person if there is one.

5.4.1 Training of Probabilistic Models

We chose 20 features to represent the human body configuration. Most of these features are close to the major body joints. The dark dots in Figure 5.2 show 17 of

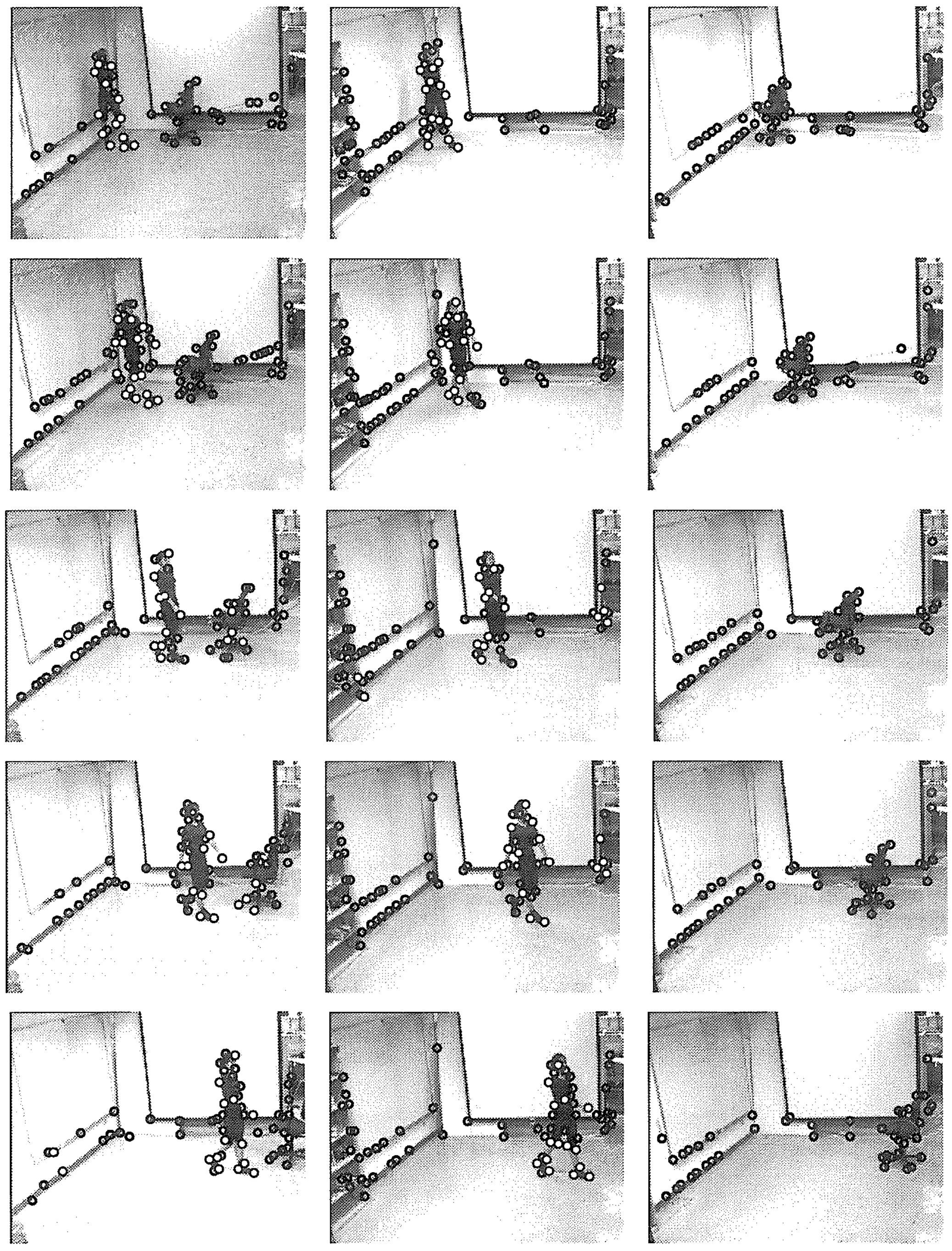


Figure 5.4: **Sample Frames**. (left column): with both walking person and moving chair. (middle column): with walking person only. (right column): with moving chair only. The black and white dots are the features selected by Tomasi-Kanade feature detection algorithm, while the white ones are the most human-like configuration found by our algorithm.

them (being correctly labeled in that frame), the other 3 are missing: two at the left knee and one at the right heel.

On the 8 training sequences with about 1000 frames in total, we manually construct the ground truth of feature positions and velocities in the following way: on the first frame of each sequence, we manually select the positions of all the visible body model features. Then the features are tracked automatically to the next frame using the Lucas-Tomasi-Kanade tracking algorithm (section 2.6) and their velocities between the two frames are computed. At each frame after tracking, we monitor the results and discard the features which have obvious tracking errors. The correct positions of these mistracked features and those newly appeared after being occluded are located manually, so that we have again the positions of all the features appearing in this frame and they can be tracked to the next frame and their velocities can be calculated. The features are also labeled manually at the same time. Occlusion is common in our training set: each feature is present in approximately 85% of frames (see Figure 5.5 (a)).

The training is done by estimating the joint (or conditional) probabilistic density functions (pdf) for all the triplets as described in section 5.3. As in [81, 82], all the pdfs are assumed to be Gaussian, and the distribution parameters are estimated from the training set.

5.4.2 Testing Set

For the testing sequences, the system automatically selects features for each frame, and tracks them to the next one. The feature selection and tracking algorithm is the standard Tomasi-Kanade version. We do not track features over more than 2 frames, but select all the features again on the next frame after tracking. Thus, there is no feature correspondence between sequential frames and each frame has a unique set of features. This is arguably the most difficult situation under which labeling and detection have to be performed, as mentioned in section 5.3.4. The dots in Figures 5.2 and 5.4 are the features generated from the automatic selection and tracking.

5.4.3 Test of Probabilistic Model

To validate the triangulated probabilistic model (Figure 5.3), we first do experiments on the manually tracked features (with ground truth, as in section 5.4.1). We have a total of 8 such sequences (with 120 frames each). To test a sequence, frames from all the other seven sequences are used as the training set. A label error happens when a body part appears but is labeled as either a different part, or background. The error also happens when a body part is missing but its label is assigned to another point. Figure 5.5(a) shows the statistics of the number of body parts present in all the sequences used in this experiment. Since the points are tracked manually, most of the un-occluded body parts are detected correctly. Figure 5.5(b) shows the correct labeling rate as a function of the number of body parts present, with the overall (considering all the frames) correct labeling rate being 85.89%. If the average number of features detected is N, ($N \approx 17$ in this experiment), the probability of a body part being assigned with a correct candidate feature by random selection is only $1/(N+1)$ (with one more background point). The correct labeling rate from our probabilistic model is much higher than that random selection probability. From Figure 5.5(b), it can be seen that the correct labeling rate goes up as the number of detected body parts increases. This is consistent with the fact that with more body parts present, the probability decomposition as shown in equation 5.7 is a more accurate approximation.

5.4.4 Detection

For a given image pair, the detection task is to determine whether or not there is a moving person in the scene. We perform detection experiments using three different types of sequences: one with a walking person only (middle column in Figure 5.4), one with both the walking person and a moving chair (left column in Figure 5.4), and the last one with the moving chair only (right column in Figure 5.4). Figure 5.4 shows sample images from the three types of sequences. The black and white dots are features detected by the Tomasi-Kanade tracker. There are a total of 1500 frames

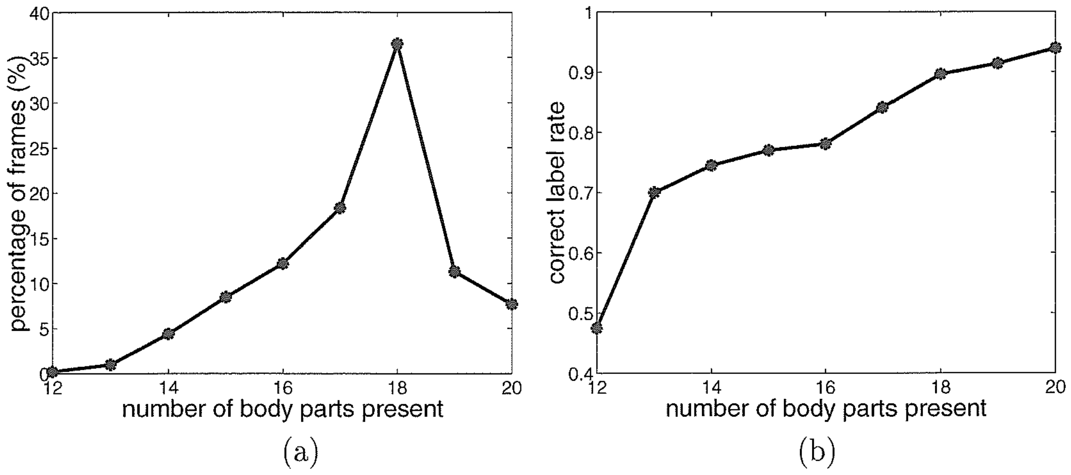


Figure 5.5: **Validation of Probabilistic Model.** (a) percentage of frames corresponding to the number of body parts present in the manually constructed (as in section 5.4.1) data set; (b) correct labeling rate vs. the number of body parts present. The chance level of a body part being assigned a correct candidate feature is around 0.06. The correct rates here are much higher than that.

and with an average of 64 feature points detected per frame in the sequences with only the person walking, 2000 frames in total and on average 58 points selected per frame in the sequences of both the person and the chair moving, and 1000 frames and average 46 feature points selected per frame in the sequences of only the chair moving.

Figure 5.6 shows two curves of receiver operating characteristics (ROC) constructed from the summation of likelihoods expressed in equation 5.13. The solid curve is the ROC when the sequences with both the person and chair and the ones with the chair only are combined together to compute the false alarm and detection rates. With $P_{detect} = 1 - P_{false-alarm}$, the detection rate is 87.54%. The dashed curve is the ROC when the results of the sequences with the person only and the ones with the chair only are combined. In this case the detection rate as $P_{detect} = 1 - P_{false-alarm}$ is 89.61%. The two curves are very similar, showing that adding a distracter (moving chair) to the scene has little negative impact to the performance of the person detector. In fact, the small difference between the two ROCs is more likely attributable to some different experiment settings that are not completely under control: the backgrounds are slightly different in the sequences, especially there is a bookshelf with

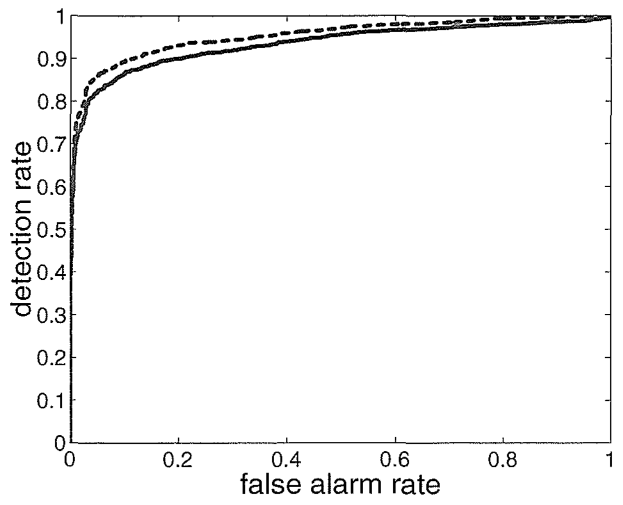
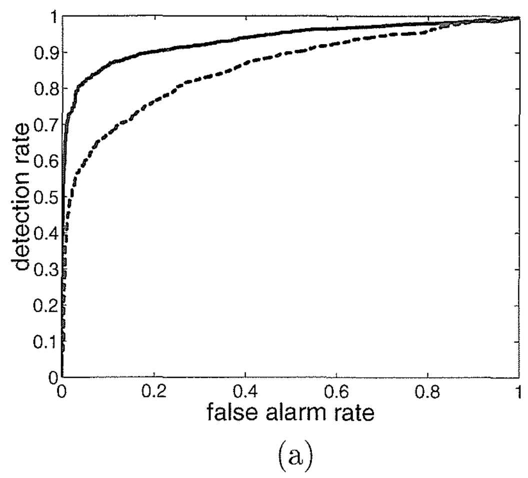


Figure 5.6: **ROC Curves** of the detection experiment in section 5.4.4. Solid line: images with body and chair vs. images with chair only, the detection rate is 87.54% when $P_{detect} = 1 - P_{false-alarm}$; Dashed line: images with body only vs. images with chair only, the detection rate is 89.61% when $P_{detect} = 1 - P_{false-alarm}$.

some features selected in the sequence with the moving person only, and the motions and paths of both the person and the chair are not identical.

The localization results are shown in Figure 5.4 as well. For each image, the white dots correspond to the best labeling $\overline{L}^*$ as in equation 5.14. For most frames, the person is localized correctly. Notice that the white dots consisting of the best configuration can be erroneously far away from each other in an image. For example, for the frame in the middle of the left column in Figure 5.4, though most white dots are correctly on the body, two are on the wall, and four more on the chair. A detailed study shows that the algorithm takes the two white dots on the wall as 'left elbow and left wrist', and the four on the chair as 'left outside knee, left ankle, left toe and left heel.' The reason for this is that for a triangulated body decomposition such as the one we use as shown in Figure 5.3, one invisible body part may cause trouble to its connected ones. For this example, 'left shoulder and left hip' are missing, then both 'left elbow and left wrist' and 'left outside knee, left ankle, left toe and left heel' are disconnected with other body parts. Therefore, the optimal labeling is composed of several independent components, possibly far away from each other. It is clear



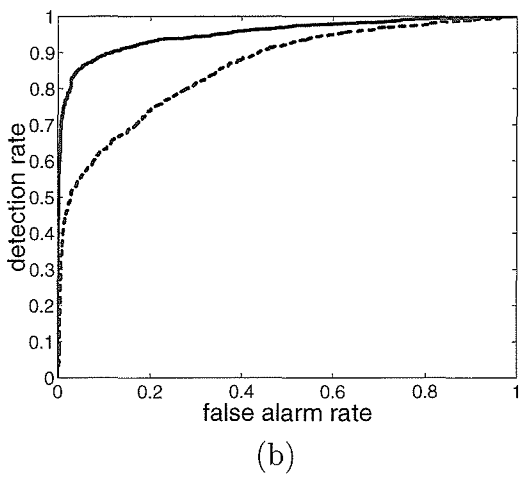


Figure 5.7: **ROC Curves.** (a) Results of images with body and chair vs. images with chair only. (b) Results of images with body only vs. images with chair only. Solid line: using method in section 5.3.3; dashed line: using method in [82].

that in this case the conditional independency required by equation 5.7 is not a good approximation any more.

Experiments are also conducted to compare the performance of thresholding the summation of likelihood of all possible labelings (as in section 5.3.3) and thresholding the likelihood of the most human like configuration (as in [82]). The solid curves in Figure 5.7 show the results by using the method described in section 5.3.3, and the dashed lines are from the method proposed in [82]. Figure 5.7 (a) and (b) are respectively results from images with body and chair vs. images with chair only and from images with body only vs. images with chair only. Both plots show that our method described in section 5.3.3 works better than the one in comparison.

5.4.5 Using Information from Multiple Frames

In this experiment, we test how the detection rate improves by integrating more information over time, using the approach described in section 5.3.4. The sequences with body and chair and the ones with chair only are studied. Figure 5.8(a) shows ROC curves of using 1 to 4 pairs of frames, respectively. Figure 5.8(b) plots the detection rates (with $P_{detect} = 1 - P_{false-alarm}$) vs. the number of frame pairs integrated. With more frames used, the detection rate gets higher. The detection rate is more than 98% when more than 7 frames (around 200 ms) are used.

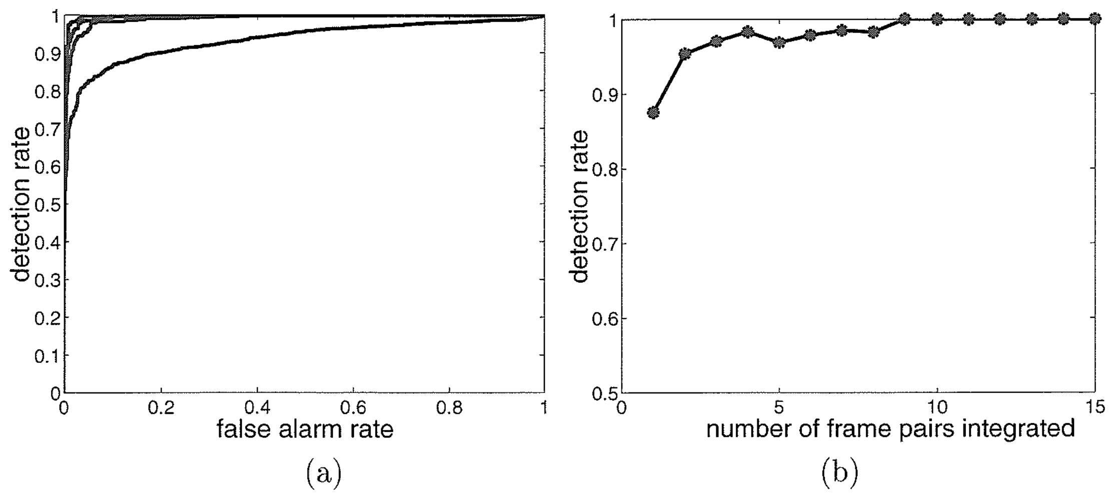


Figure 5.8: **Results from Integrating Multiple Frames.** (a) Four curves are ROCs of integrating 1 to 4 frame pairs, respectively. The more frames integrated, the better the ROC curve is. (b) Detection rate (when $P_{detect} = 1 - P_{false-alarm}$) vs. number of frame pairs used.

5.4.6 Experiments on Different Subjects

In previous experiments, the sequences for training and testing are from the same subject. In this section we test the performance on another subject, who also walks with a chair moving in the scene. Four sequences are used with about 120 frames in each. Figure 5.9(a) shows the comparison result. The solid line is the ROC curve from detecting the new subject, with 75.19% detection rate (when $P_{detect} = 1-P_{false-alarm}$), and the dashed line is that from the same subject as in training. Figure 5.9(b) shows the detection rates (with $P_{detect} = 1 - P_{false-alarm}$) vs. the number of frame pairs integrated for the new subject. The detection performance improves with more frames integrated: it is almost certain when more than 10 pairs of frames used.

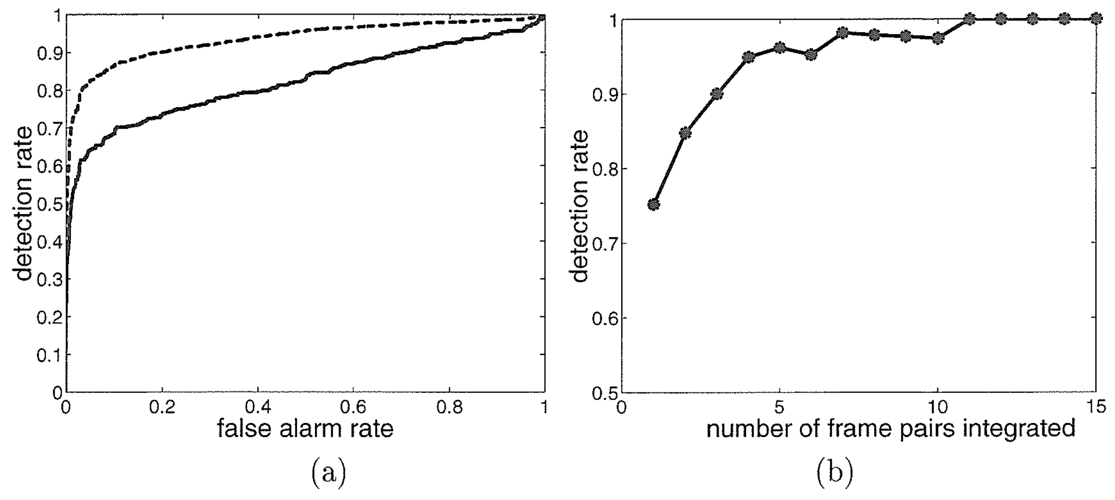


Figure 5.9: **Results from Detecting a Different Subject.** (a) ROC curves. Solid line: another subject (different from training set); Dashed line: Subject of the training set (the same as the solid line of Figure 5.6). (b) Detection rate (when $P_{detect} = 1 - P_{false-alarm}$) vs. number of frame pairs used to detect the new subject.

5.5 Conclusion

In this chapter we have presented a method to detect and label human motions in monocular image sequences. The method takes as its input the positions and velocities of the most salient features in the image, as generated from the Lucas-Kanade feature tracker. No prior image segmentation is required. The method models human motion with an approximated joint probability density of the positions and velocities of features associated with a human body. Given a (possibly cluttered) motion sequence, detection is performed by setting a threshold to the summation of the likelihoods of all possible labelings. When a person is detected, localization is done by finding the subset of detected features that is most likely to be associated with a human body. The model is trained with a manually labeled training set.

We have tested our method on a number of image sequences containing either a walking person, or some nonhuman motion, or both. The results are encouraging. The detection rate is around 90% from 2 frames, or 60 ms, and in excess of 98% from 7 frames, or 200 ms. It also shows that the method can be fairly generalized to detect a person different from the one used from training. Neither labeling nor detection take more than 1 second per frame by an implementation in Matlab running on a 450MHz Pentium PC. It is promising to have a real-time implementation in C.

Our ideas may be extended and improved along a few directions. For instance, human motion is currently modeled by Gaussian distribution. This choice is arbitrary and needs to be re-examined in the light of our training data. We do not experiment with different structures for the triangulated model, either. Many other reasonable choices may exist and can be tested. Furthermore, some form of hierarchical modeling will be needed to account for long-range dependency of body parts. This is critical in the case of occlusion as discussed in the experimental section. Song et al. [81] have demonstrated that their system can be generalized well to changes in viewpoint and to different types of motions when using un-occluded Johansson stimuli. This gives good reason for us to believe that our system could be equally robust. However, systematic testing needs to be done on a variety of body motions and under a number of different viewing conditions in order to assess the full capacity of our system.

Chapter 6 Human Action Recognition

6.1 Introduction

The system discussed previously focuses on human motion detection in the shortest period possible – only a pair of frames. The multiple frame integration is a factorial of detections from frame pairs which are assumed to be independent from each other. However, frame pairs are not independent: human motion is a complex pattern which develops over time. When human motion is detected and observed long enough in an image sequence, it is possible to tell what kind of action the subject is performing, and this is called *human action recognition*.

The concept of *movemes* is introduced in [25] and defined as a set of elementary human motions which would roughly correspond to the "elementary units of motion" used by the brain. A typical moveme lasts on the order of 1 second. Some examples of movemes are: taking a step forward, stepping over an object, reaching an object, and turning head to look in a new direction, etc. A human action is a simple concatenation of movemes and can be regarded as a temporal repeat of a moveme. Its time scale can be from the order of 1 second (the same as moveme) to the order of 10 minute. For instance, walking is an action since it is a repeat of the stepping forward moveme, reaching an object can be viewed as both a moveme and an action. The complicate serial or parallel combination of different actions creates human activities. For examples, an activity to pick an object far away is consisted of two actions in serial: walking to the location and reaching the object, an activity to go to an assigned room is decomposed as walking forward and turning your head to look for the room number at the same time. On the time scale, a human activity typically lasts a few minutes to hours. The recognition of human action is what we will study in this chapter.

There are many challenges to the problem of human action recognition.

- Actions develop over longer time periods, at least 0.5 to 2 seconds, and therefore one needs to integrate information over at least 15 to 60 image frames (sequences are usually captured at 30 frames per second).

- Different actions may be associated to similar patterns, therefore, recognition has to be based on the discrimination between actions.

- Different individuals perform the same actions in slightly different ways. Repeated performances of the same action by the same person may vary. Recognition should be invariant with respect to these factors.

- The onset and ending of the actions are not predictable and can be very different from sequence to sequence.

- Recognition should be invariant to the viewpoints.

- Self-occlusions of body parts occur frequently during actions.

The projections of body part motions onto image planes depend on the observation viewpoint. Therefore, to make the learning-based recognition from 2D images invariant to the viewpoints, actions should be learned from different view angles.

As we switch our attention to the problem of human action recognition, we change the features that we use to model a human body from points to patches. The candidate point features are usually not difficult to select from images, and excellent detection can be achieved based on the triangulated model of human body joints (Chapter 5). However, there are some disadvantages in using them in recognition. The human body is composed mainly of body parts, which cannot be represented fully by the point features corresponding to body joints. Information could be lost from the original body image by this relatively simple joint representation. On the other hand, recognition is more subtle than detection because it is aimed to telling apart similar motions. Consequently patch features that contain richer information are more desirable than joint features. In addition, due to frequent self-occlusion of the human body during motion and the fact that people often wear nontextured clothes, the joint features can barely be observed in many cases. Such instability in feature observation tends to fail the body joint model in correctly detecting and recognizing a person from images. To avoid the problem, the subject can be purposely dressed as in the experiment section in Chapter 5 where clothes are altered on purpose to make sure that the neck, wrists, knees as well as ankles can be detected easily as point features, or these joints can be selected manually as the input to the recognition system as done in [5]. Obviously these assumptions are too restrictive to be natural. Due to the undesirable requirements by the model of joint features, we explore the territory of action recognition with patch features, instead. The generalization of detection based on joint features to more natural scenes is explored further by Song in [83].

Typical approaches to human action recognition include top-down [27, 13] and bottom-up methods [104, 9]. The top-down methods have action models which can predict human body poses and velocities in the images. The original test images are processed. Then the action is recognized as the one whose predictions best match the processed images. The bottom-up methods start from the original images and extract the low level image features, such as points and specified patches. The low level features are then grouped according to some constraints. The human action and body poses are recognized as the one that best represents the human among all possible grouping results. It is well known that the advantage of bottom-up methods is speed and its disadvantage is poor signal-to-noise ratio. On the contrary, top-down approaches have far better detection and recognition properties but at the cost of both algorithmic and computational complexity. In the case of human body pose recognition, typical features for bottom-up methods are boundaries and "patches," which are regions roughly corresponding to body parts. Sometimes the key assumption of these methods is that the color and/or texture of each body part is known in advance [21, 43]. We do not want to make any assumption on the clothing style (or absence thereof) and/or the presence of good boundaries between limbs and torso. Therefore a good segmentation to obtain low-level patch features becomes extremely difficult with the current technology and we choose to take the top-down approach.

We briefly describe here the main characteristics of our approach. Each topic is discussed in detail in the following sections. Our basic assumption, similar to those in [13, 27], is that a sequence of regions of interest (ROI) containing a foreground moving object is obtained by background subtraction. The space of poses is then discretized into a number of *codewords* which are fitted to the ROI for recognition. In order to further improve the signal-to-noise ratio, we consider poses in space-time which are called *movelets*. A movelet is a collection of the shape, motion and occlusion of image patches corresponding to the main parts of the body. We model each action as a stochastic concatenation of an arbitrary number of movelet codewords, each one of which may generate images (also stochastically). Hidden Markov models are used to represent actions and the images they generate. The Viterbi algorithm is used to find the most likely sequence of codewords and action that correspond to a given observed sequence. The movelet codewords are learned in advance from training examples and are shared amongst all actions (similarly to an alphabet of letters which are shared by all words).

Results are presented later on the model training, in addition to the recognition of 3 periodic actions imaged from multiple viewpoints and 8 nonperiodic actions. We explore experimentally what is the necessary number of movelet codewords to represent the movelet space. Our experiments also show how the performance of action recognition varies as a function of the number of frames that are used.

6.1.1 Relation to Previous Work

There has been a lot of work in the past several years in the literature about the problem of human action recognition. They can be classified by several different criteria, such as the type of model used (e.g., stick figure-based, volumetric and statistical), dimensionality of representation (2D vs. 3D), sensor multiplicity (monocular vs stereo), sensor mobility (stationary or moving), etc. Our approach to human action recognition is based on 2D appearance from monocular image sequences. In particular, it is most closely related to two categories of recognition: 1) recognizing body poses by fitting pose models to images, which is sometimes referred to as "labeling problem"; 2) recognizing actions as sequences of poses.

As mentioned earlier, a human body pose can be represented by either a combination of body parts with certain mutual relationship constraints (bottom-up methods) or a whole pose with identified body parts (top-down methods). Bottom-up methods for pose recognition [80, 21, 43] generate correct pose representations (or labels of parts) from the detected candidates of low level features. Joint feature is used in [80] (or Chapter 5) and patch features are explored in [21, 43] along this line. The low level patch features are actually very special. It is assumed that the human body is in nude [43] or the patch texture is known in advance [21]. Due to the lack of robust segmentation techniques to obtain general patch candidates for body parts, it is difficult to apply the bottom-up methods proposed in [21, 43] to general situations. Top-down methods for pose recognition and parts identification match the learned whole pose models with the extracted body silhouettes [27, 107] or the extracted special features [69]. In this chapter, we reconstruct body poses by fitting the models (movelet codewords) to the extracted silhouettes. Our method is therefore top-down. Unlike [27, 107], our primary goal is to recognize actions, hence we do not intend to label body parts on images after pose being identified.

To recognize a human action, one general bottom-up approach is to bypass the step of pose recognition and describe the action as a sequence of ROI represented by low level features [63, 104, 13]. Polana and Nelson referred this method as "how to get your man without finding his body parts" [64]. Their work uses a periodicity measure to detect activities and a low level feature vector based on the motion information in a periodic cycle to recognize the action. A similar low level feature vector based on ROI appearance is used for action recognition in [104], but the periodicity requirement is relaxed and hidden Markov model (HMM) is applied. HMM is widely used in modeling temporal related actions, such as gesture recognition [86, 59]. Action recognition based on the cumulative binary motion image (MEI) and motion history image (MHI) is described in [13] as a top-down approach. This method compresses all information in the image sequence into MEI and MHI so that recognition becomes a simple task.

The same technique is also applied in [68]. However, we believe such simplification does not catch the true mechanism of how human vision perceives actions.

Some other approaches to action recognition require that human poses are given in advance by manual initialization. In [103], the body parts are initially segmented by hand and then tracked through the sequence. Recognition is carried out by comparing the trajectories of body parts of the observed action with those of the training action "examplars " via their principal components. This approach requires temporal alignment of actions. A dynamical model of geometric feature trajectories is taken into account in [5], where the features (body joints) are labeled manually in each image frame. A drawback of this dynamical approach is that the problem of self-occlusion is not addressed.

A systematic framework has been proposed to segment, track and classify complex human dynamics based on probabilistic analysis in [9]. It involves the propagation from low level areas of coherent motions, through mid level simple dynamical movements, to high level HMM represented actions. The framework is excellently constructed in theory. Nevertheless, the segmentation of motion patches is such a difficult task that in practice the system can only be applied to a single body part, one lower leg in this paper.

Our approach to human action recognition in this chapter is also based on HMM. However, unlike typical HMM approaches, we predefine the hidden states of HMM as the learned pose models which are called *movelet codewords.* A binomial distribution is applied to model the probability that an observed configuration is generated by a codeword and an action is modeled by HMM as stochastic concatenation of codewords. By modeling the problem in such a framework, the final estimation of maximum likelihood by HMM can generate the recognition results of both human action and body poses at the same time. In addition, in our experiments we test the influence of the number of codewords and sequential frames on the performance of our algorithm. To the best of our knowledge, these issues have not yet been addressed in literature.

The majority work presented in this chapter was published in [24].

6.2 Movelet and Codeword: Modeling Human Body

Human body is an articulated structure with many degrees of freedom. Although complicated, the relative motion between body parts is constrained by the kinematic structure and motor control plans in the brain to perform a coordinated action. An action is a sequence of human body configurations. The set of all possible configurations for human actions is very large and nonlinear. Our approach here is to discretize this set into a small collection of codewords, and then represent actions as concatenation of these codewords.

If we do not look at details, a human body configuration can be characterized by the shapes, mutual positions and motions of 10 major parts: the head, torso, 4 parts of upper limbs and 4 lower ones. We name such configuration representation *movelet*. The shape of each corresponding body part on the image plane is modeled by a rectangle S_j with 5 degree of freedom (DOF): 2 for the center position, 2 for the long and short axes, and 1 for the orientation of long axis. Assume this shape transforms to S_j' (also 5 DOF) in the next frame. Since the change of the part appearance can be caused by 3D motion of body part, variation of loose clothes, as well as partial occlusion, etc., this shape alteration is no longer a 3DOF rigid motion on the image plane, but a nonrigid motion in its full parametric space as $S_j' - S_j$. Therefore, a human body movelet may be represented as $\mathbf{M} = (S, S') = (\bigcup_{j=1}^{10} S_j, \bigcup_{j=1}^{10} S_j')$, which is mathematically denoted as a 100×1 vector. The advantage of this parameterization over the direct representation of the shape and motion is that its parameters are in the same metric space.

Consider a set of human actions we are interested in. Each action may be represented as a sequence of movelets and different actions may contain similar movelets. We collect the set of the movelets of all actions that have been observed over a learning time period, and cluster them into N codewords using an unsupervised vector quantization method. For this purpose, the K-means algorithm [4] is used in our approach. Let $\mathbf{M}_m$ $(m = 1, \ldots, M)$ represent the movelets, and z_{im}, a binary variable, be the membership of movelet $\mathbf{M}_m$ being clustered to the codeword $\mathbf{C}_i$ $(i = 1, \ldots, N)$.

The K-means algorithm for the codewords learning consists of iterations between the following two steps till convergence (k is the iteration step):

Step 1:

$$\begin{aligned} z_{im}^{[k]} &= 1 \quad if \quad \forall j \neq i \quad \|\mathbf{M}_m - \mathbf{C}_i^{[k]}\|^2 < \|\mathbf{M}_m - \mathbf{C}_j^{[k]}\|^2 \\ &= 0 \quad otherwise \end{aligned} \tag{6.1}$$

Step 2:

$$\mathbf{C}_i^{[k+1]} = \sum_m z_{im}^{[k]} \mathbf{M}_m \Big/ \sum_m z_{im}^{[k]} \qquad i = 1, \ldots, N \tag{6.2}$$

The vector quantization procedure coarsely divides the whole movelet space into N regions represented by codewords.

There are two practical issues that merit further discussion.

Origin Selection: A movelet is defined relative to the origin which is the center of the rectangle representing the head. The positions of all the other parts are relative to this origin. The DOF of a movelet is therefore reduced by 2. Because of this special treatment, the center position of the head, $(0, 0)$, has to be removed from the mathematical representation of all movelets to perform the vector quantization.

Self-Occlusion: We observe in our experiments that over half of all movelets have some of the body parts occluded. Therefore the dimensions of those movelets are reduced to lower DOF. We divide the movelets into subgroups according to their occlusion patterns. The learning scheme should be modified accordingly:

1. For each movelet, identify the parts present in both S and S'. Mark the other parts as occluded.

2. Classify each movelet into one of the 2^{10} (10 is the number of parts we model the body) possible occlusion patterns.

3. Select the occlusion patterns with significant number n_o of associated movelets. Discard all the other patterns. Let $r_o = \frac{n_o}{\sum_o n_o}$ be the relative frequency of

occlusion pattern o.

4. Distribute the total number N of codewords to these selected occlusion patterns proportionally according to their membership ratio r_o. Let N_o be the number of codewords the occlusion pattern o is assigned to, we have $N_o \approx r_o N$ and $\sum_o N_o = N$.

5. For each occlusion pattern o, apply the vector quantization algorithm to learn N_o codewords from all the movelets classified to this occlusion pattern. These movelets have same DOF and the occluded parts are removed from their representation.

6. The union of the learned codewords for all the selected occlusion patterns is the final set of codewords.

6.3 Configuration Recognition

To recognize an observed action, we assume that the image of a moving body can be segmented automatically as the foreground in the image sequence, which sometimes can be done by applying background subtraction. Our approach will proceed in two steps: recognize each single configuration in the action first, and then recognize the action as a concatenation of configurations, which are represented by the codewords. In this section, we try to solve the first problem, the second will be addressed in the following section.

For each pair of frames, an observed configuration is defined as the foreground pixels $\mathbf{X} = \{X, X'\}$, where X and X' are the 2D positions of the foreground pixels in the first and second image frame, respectively. Although we model the human body by its parts in the previous section, due to the lack of techniques to reliably segment human body parts directly from the image, we consider the observed configuration of foreground pixels in their entirety. For our recognition purpose, a top-down approach is applied to fit the codewords as whole poses to the observations. Examples are shown in figure 6.2 where we only plot the fittings in the first frame. Given a codeword with shape S and transformed shape S' with fixed origin location, we want to calculate the likelihood of observing X and X'. The issue of how to search for this origin location will be addressed in the experiment section 6.5. We assume, given the shape S, a priori probability to detect a pixel X_i in the image as a foreground(fg) or background (bg) point is a binomial distribution conditioned on whether the pixel is in the shape S or not. Table 6.1 shows this probability distribution.

	Foreground pixel $X_i \in \text{fg}$	Background pixel $X_i \in \text{bg}$
Pixel is inside the shape: $X_i \in S$	α	$1 - \alpha$
Pixel is outside the shape: $X_i \notin S$	β	$1 - \beta$

Table 6.1: **Binomial Distribution** for $P(X_i|S)$

where $\alpha \gg \beta$. An illustration of how the probability $P(X_i\,|S)$ changes with the pixel position is shown in Figure 6.1.

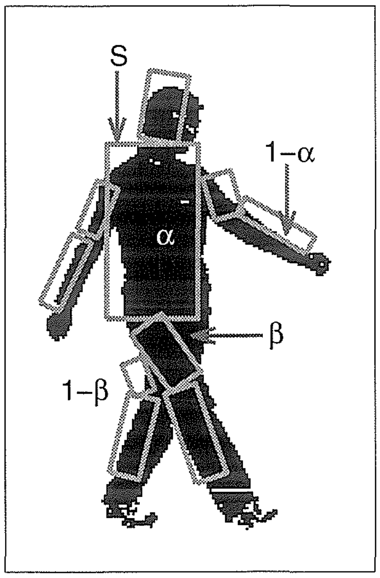


Figure 6.1: **Likelihood $P(X_i\,|S)$ as a Function of Location X_i.** The union of all gray rectangles is the shape S of a codeword. This figure corresponds to the distribution in Table 6.1. For the illustration purpose, here the codeword is not placed at its best location for fitting.

Assume the independence of pixels, the likelihood of observing X as a collection

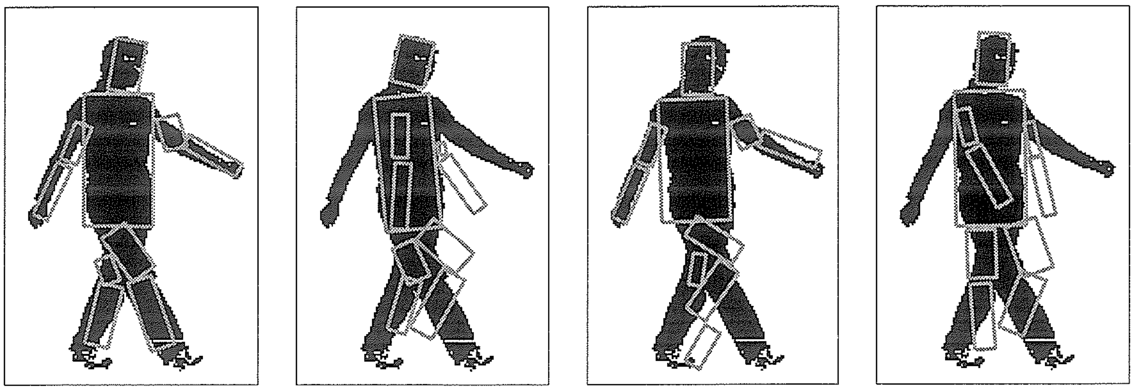


Figure 6.2: **Examples of fitting codewords to an observed configuration**. Set $\alpha = 0.9$ and $\beta = 0.1$, the log-likelihood $\log P(\mathbf{X}|\mathbf{C}_i)$ for fitting these four different sample codewords from left to right are: -2.66×10^4, -3.38×10^4, -3.14×10^4 and -3.33×10^4, respectively. The codeword shown in the most left plot is the best among these four to represent this observed configuration.

of all foreground pixels given the codeword shape S is

$$P(X|S) = \prod_{X_i \in image} P(X_i|S) \tag{6.3}$$

A codeword $\mathbf{C}$ includes both the human body shape S and its transformed shape S' in the next frame. When the location of the initial shape is fixed, the transformed shape is also known in the next frame since both of them are relative to the same origin. The likelihood of observing the configuration $\mathbf{X}$ given a codeword $\mathbf{C} = (S, S')$ can then be computed as

$$P(\mathbf{X} \mid \mathbf{C}) = P(X,\, X' \mid S,\, S') = P(X \mid S)\, P(X' \mid S') \tag{6.4}$$

The likelihood $P(X'|S')$ is estimated in the same way as in equation 6.3.

Usually the estimation of the observation probability $P(\mathbf{X}|\mathbf{C})$ is too small to be calculated directly, therefore only its log-likelihood $\log P(\mathbf{X}|\mathbf{C})$ is computed, instead. The log-likelihood that the observed configuration $\mathbf{X}$ is generated by every codeword $\mathbf{C}_i$ in the codeword database, namely $\log P(\mathbf{X}|\mathbf{C}_i)$ $(i = 1, \ldots, N)$, is computed and preserved for action recognition addressed in the next section. A few examples of fitting codewords to an observed configuration as well as their computed log-likelihoods are shown in Figure 6.2.

The best codeword representing the observation $\mathbf{X}$ is simply obtained from the codeword database as the one with the maximum likelihood:

$$\mathbf{C}^* = \arg\max_{\mathbf{C}_i} P(\mathbf{X} \mid \mathbf{C}_i) \tag{6.5}$$

Setting a threshold to the resulted maximum observation likelihood $P(\mathbf{X}|\mathbf{C}^*)$ enables us to determine if a single foreground observation $\mathbf{X}$ from a pair of frames is caused by human action.

6.4 HMM for Action Recognition

Assume the observation contains a sequence of the foreground configurations as $(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_T)$, where $\mathbf{X}_t$ $(t = 1, \ldots, T)$ are defined as same as $\mathbf{X}$ in section 6.3. To recognize the observed sequence as one of the learned actions, a slightly modified hidden Markov model (HMM) is applied. HMM is a stochastic state transition model [65] which can deal with time-sequential data. In essence, it models the system space by a set of discrete hidden states, which are connected to the observations by the probabilistic function. The sequential observation is viewed as an output generated from a state sequence with certain probability.

For the problem of human action recognition, we predefine the states of HMM as the trained codewords. The probability of an observed configuration to be generated from a particular state is obtained as $P(\mathbf{X} \mid \mathbf{C}_i)$ in section 6.3. This probability is not obtained through the learning scheme of usual HMM because here we know exactly by construction what the states are.

Assume there are K human actions to recognize: $\mathbf{H}_k$ $(k = 1, \ldots, K)$. The actions share the same set of states (codewords), but are distinguished by the different transition matrices of their HMMs. Different actions take different spatial and temporal paths which are modeled by transition matrices in the state space. In detail, a transition matrix $A = \{A_{ij}\}$ of a HMM represents the probability of the transition from one state at time t to another state at time $t+1$, namely, its element $A_{ij} = P(q_{t+1} = \mathbf{C}_j \,|\, q_t = \mathbf{C}_i)$, where q_t denotes the actual state at time t. In our case, each movelet $\mathbf{M}_m$ from training actions has been assigned to one codeword $\mathbf{C}_i$ and the membership of $\mathbf{M}_m$ belonging to $\mathbf{C}_i$ is equal to 1, $z_{im} = 1$ (see section 6.2). Therefore, for each training action k, a corresponding codeword sequence is generated from its original movelet sequence. Its HMM state (codeword) transition matrix A^k can be learned easily from this codeword sequence. The ij element of A^k is computed by counting the number of transitions from state $\mathbf{C}_i$ to state $\mathbf{C}_j$ and normalizing it by the total number of transitions from state $\mathbf{C}_i$ in the codeword sequence of action $\mathbf{H}_k$.

Notice that the sequence of the best representative codewords, which are obtained from equation 6.5 for each individual configuration in the observed sequence, may not form a valid action. To recognize which action the observed sequence $(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_T)$ represents and what its best representative codeword sequence is, the following HMM solutions are applied. First, for each action hypothesis $\mathbf{H}_k$ $(k = 1, \ldots, K)$, find the single best representative sequence of states (codewords) that maximizes the log-likelihood of the state sequence as

$$\begin{aligned} & q_{k1}^*,\, q_{k2}^*,\, \ldots,\, q_{kT}^* \\ & = \arg \max_{q_{k1}, q_{k2}, \ldots, q_{kT}} \log P(q_{k1},\, q_{k2},\, \ldots,\, q_{kT} \,|\, \mathbf{X}_1,\, \mathbf{X}_2,\, \ldots,\, \mathbf{X}_T,\, A^k) \end{aligned} \tag{6.6}$$

where $q_{kt} \in (\mathbf{C}_1,\, \mathbf{C}_2,\, \ldots,\, \mathbf{C}_N)$.

By Bayesian rule, the maximum log-likelihood estimation in equation 6.6 is equivalent to the following maximization problem:

$$\begin{aligned} & q_{k1}^*,\, q_{k2}^*,\, \ldots,\, q_{kT}^* \\ & = \arg \max_{q_{k1}, q_{k2}, \ldots, q_{kT}} \log P(q_{k1},\, q_{k2},\, \ldots,\, q_{kT},\, \mathbf{X}_1,\, \mathbf{X}_2,\, \ldots,\, \mathbf{X}_T \,|\, A^k) \end{aligned} \tag{6.7}$$

Denote the elements of transition matrix A^k by $\{A_{ij}^k\}$. The optimization of equation 6.7 is performed by applying the following Viterbi algorithm:

1. Initialization

$$\begin{aligned}\delta_1(i) &= \log P(\mathbf{X}_1 \,|\, \mathbf{C}_i) \qquad 1 \leq i \leq N \\ \phi_1(i) &= 0\end{aligned}$$

2. Recursion

$$\begin{aligned}\text{for } 2 &\leq t \leq T \qquad 1 \leq j \leq N \\ \delta_t(j) &= \max_{1\leq i\leq N} \left(\delta_{t-1}(i) + \log A^k_{ij}\right) + \log P(\mathbf{X}_t \,|\, \mathbf{C}_j) \\ \phi_t(j) &= \arg\max_{1\leq i\leq N} \left(\delta_{t-1}(i) + \log A^k_{ij}\right)\end{aligned}$$

3. Termination at $t = T$

$$\begin{aligned}V^*_k &= \max_{1\leq i\leq N} \delta_T(i) \\ q^*_{kT} &= \arg\max_{1\leq i\leq N} \delta_T(i)\end{aligned} \tag{6.8}$$

4. State sequence backtracking

$$\begin{aligned}\text{for } t &= T-1, T-2, \ldots, 1 \\ q^*_{kt} &= \phi_{t+1}(q^*_{k(t+1)})\end{aligned}$$

Then the human action $\mathbf{H}_{k^*}$ that best represents the observed sequence is determined by

$$\begin{aligned}k^* &= \arg\max_k \log P(q^*_{k1}, q^*_{k2}, \ldots, q^*_{kT}, \mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_T \,|\, A^k) \\ &= \arg\max_k V^*_k\end{aligned} \tag{6.9}$$

where V^*_k is computed in Viterbi algorithm as shown in equation 6.8.

Consequently the sequence $q^*_{k^*1}, q^*_{k^*2}, \ldots, q^*_{k^*T}$ is the recognized most likely codeword sequence to represent the observed sequence $(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_T)$.

Theoretically the recognized action $\mathbf{H}_{k^*}$ should be the one which maximizes the probability $P(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_T | A^k)$. However, the classical forward algorithm [65] to estimate this likelihood accurately is prohibited because it involves the summation of probabilities $P(\mathbf{X}_t|\mathbf{C}_i)$ which is too small to be computed directly and thus can only be represented by $\log P(\mathbf{X}_t|\mathbf{C}_i)$. Therefore we maximize the likelihood of the observation with the best state sequence as in equation 6.9, which is proposed in the score evaluation section in [65].

6.5 Experiments

6.5.1 Experimental Design for Codewords Training

The objective of codewords training is to obtain the set of codewords from all the movelets of training actions. A human body movelet is represented by the union of body parts and their motions. Therefore, we need to identify each body part in the images of training actions. For this purpose, we made special clothes which have distinguishable colors for each body part. So it is made sure that any two neighboring parts have different colors and any two parts with the same color are located far away from each other. For this reason, every body part can be segmented by its special color and different location from the others.

To estimate the rectangular shape of a body part from its segmented pixels, we exploit some useful properties of a rectangle here. If we take all the pixels $\overline{X}_i = [x_i \quad y_i]^T$ $(i = 1, \ldots, N)$ inside a rectangle, the first and second moment of this rectangle can be estimated as

$$\overline{\mu} = \sum_i \overline{X}_i \,/\, N \tag{6.10}$$

$$\Sigma = \sum_i (\overline{X}_i - \overline{\mu})(\overline{X}_i - \overline{\mu})^T \,/\, (N-1) \tag{6.11}$$

Let the parameters to define this rectangle be: c_x, c_y, a, b, θ which are the center, half width, half height, orientation, respectively. The following relationships between

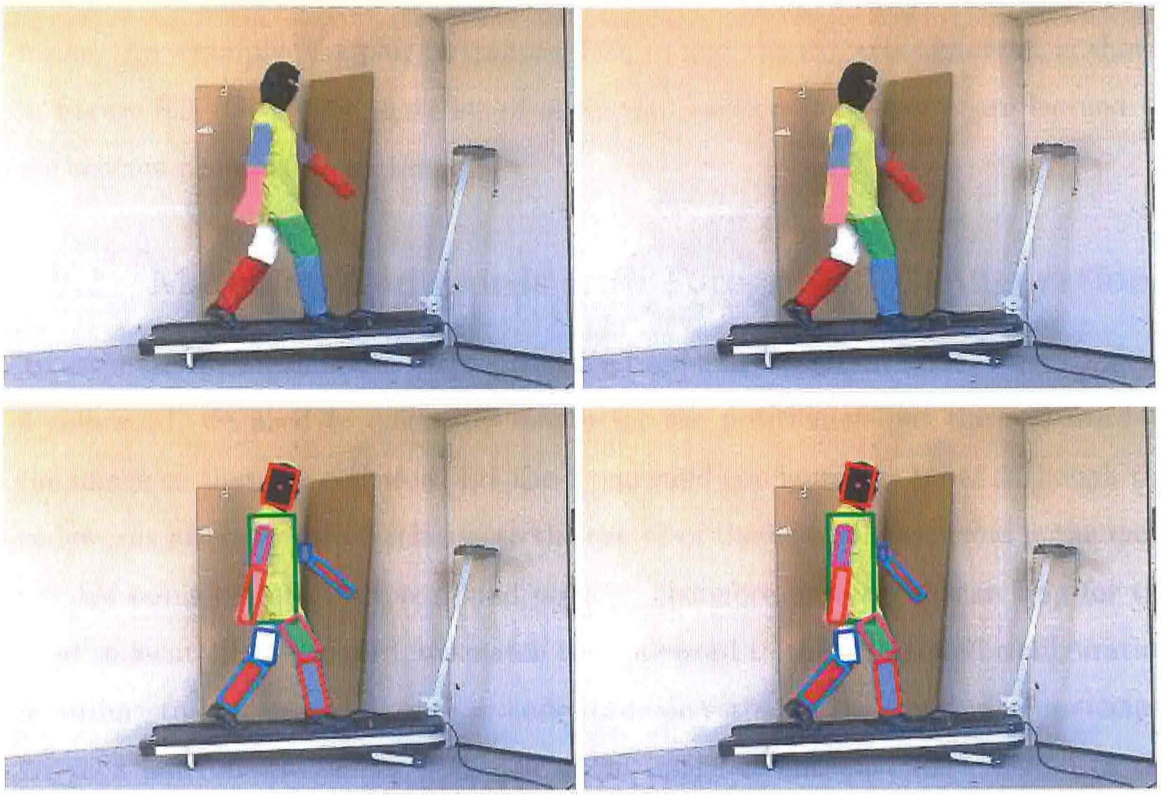


Figure 6.3: **Examples of Training Images and Movelet.** Top: two consecutive images in the training sequence. Bottom: estimated rectangular shapes for both frames to form a movelet.

the rectangle shape parameters and its first two moments can be proved:

$$\begin{bmatrix} c_x \\ c_y \end{bmatrix} = \overline{\mu} \qquad \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \frac{a^2}{3} & 0 \\ 0 & \frac{b^2}{3} \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}^T = \Sigma \tag{6.12}$$

The center $[c_x \quad c_y]^T$ is equal to the first moment $\overline{\mu}$ and the other parameters a, b, θ can be obtained from the singular value decomposition (SVD) of the second moment Σ. Therefore, while the specially designed color clothes for the training actions provide us an easy way to segment the associated pixels of each body part from the rest, the parameters c_x, c_y, a, b, θ of this rectangular shape can be estimated directly by equation 6.12 from the pixels resulted from segmentation. A movelet is formed by stacking together the rectangular shapes of all body parts visible in both of the consecutive frames. For a movelet, the shapes of all rectangles in both frames are relative to the same origin which is the center of the head in the first frame. An example of a pair of training frames and the extracted movelet is shown in Figure 6.3. Given the movelets of all training actions, codewords are learned by the scheme proposed in section 6.2.

6.5.2 Matching Codewords with Foreground Configurations

To estimate the likelihood $P(\mathbf{X} \mid \mathbf{C})$ that a foreground configuration is observed given a codeword, we need to efficiently search for the position to put the codeword in the image so that the codeword fits the foreground configuration best. Although the codewords are represented relative to the center of the head, the centroid is the most reliable point to detect a foreground object. Therefore, instead of searching for the head to locate the codeword, we match the codeword to the foreground configuration according to its centroid. For each codeword, we estimate the centroid of its shape. Given a pair of test frames with 320×240 image resolution , the centroid of the foreground pixels in the first frame is computed. We then lock in the square of 49×49 pixels centered at this point and search this region for the location of codeword shape centroid where we have the best fit. In Figure 6.4, the left plot shows an example of the computed foreground centroid as well as the search region. The right one shows how the log-likelihood $\log P(\mathbf{X} \mid \mathbf{C}^*)$ varies with respect to the assumed location of the codeword centroid in this search region with $\mathbf{C}^*$ chosen to be the best fitting codeword in this case. For every codeword, we search in this region for the location where the log-likelihood of the observed configuration in the first frame given this codeword shape is maximized. The transformed shape of the codeword is correspondingly placed on the second frame. After locating the codeword, the log-likelihood of $P(\mathbf{X} \mid \mathbf{C})$ can be determined by equations 6.3 and 6.4. The parameters of the binomial distribution are chosen as $\alpha = 0.9$ and $\beta = 0.1$ arbitrarily.

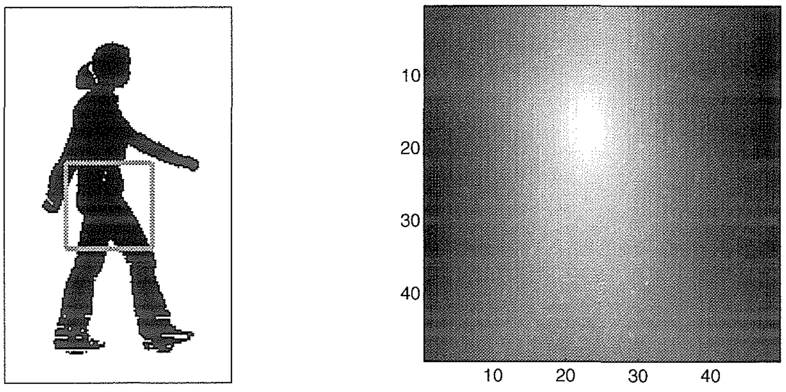


Figure 6.4: **Log-likelihood Varies With Centroid Position.** (left): the 49×49 search region for the codeword centroid. (right): Log-likelihood $\log P(\mathbf{X}|\mathbf{C})$ varies with respect to the assumed centroid position in the search region. It is gray-scaled. Brighter indicates higher log-likelihood.

In our experiment we actually search for the location of a codeword centroid on every third pixel in the search region, which ends up with 16×16 possible position to search. It takes about 0.012 seconds to search for one codeword with a Pentium 750 MHz machine. If there are 200 codewords in the database, the total time required to recognize a single configuration is therefore about 2.4 seconds.

6.5.3 Recognition of Periodic Actions

In this section we illustrate the recognition experiment of a set of 15 view-based periodic actions: 3 different gaits (stepping, walking and running) captured from 5 different view angles ($0°, 45°, 90°, 135°, 180°$ between the camera and subject orientations). To make the action and view angle controllable, a treadmill is used on which all these periodic actions are performed. The image sequences used for training and testing in our experiments are captured by a digital camcorder at 30 frames/sec. Figure 6.5 shows one key frame for each action.

Training Actions

The training actions are performed by only 1 subject wearing the specially designed color-coded clothes. For each training action, 1800 frames are collected and processed by the codewords training algorithm. Several frames for some training ac-

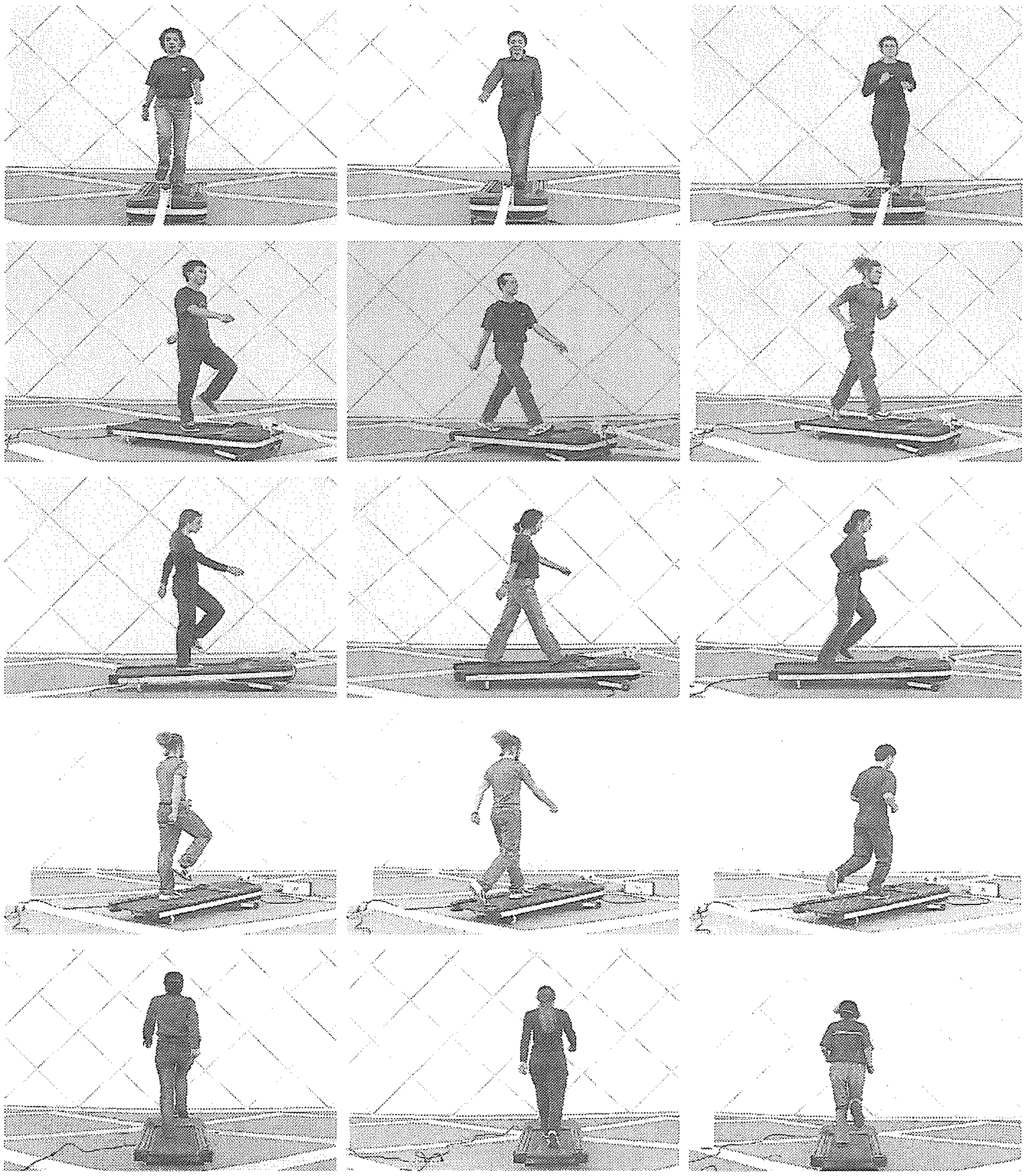


Figure 6.5: **Key Frames of Periodic Actions**. (left column): stepping. (middle column): walking. (right column): running. Each column contains 5 images captured respectively at 0°, 45°, 90° ,135°, 180° angles of view

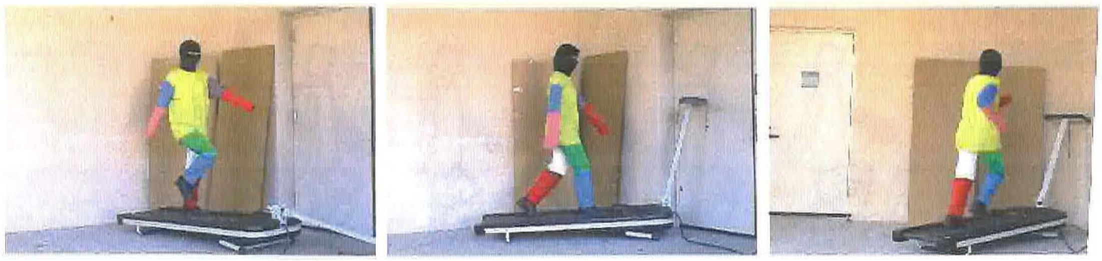


Figure 6.6: **Key Frames of 3 Periodic Training Actions.** (left): 45° stepping. (middle): 90° walking. (right): 135° running.

tions are shown in Figure 6.6. Movelets are obtained from each pair of frames using the method described in section 6.5.1.

From this set of movelets, we first train $N = 450$ codewords. This number is chosen because on average an action cycle is about 30-40 frames long and we have 15 different periodic actions. The recognition performance with different number N of trained codewords is reported at the end of this section. As described in section 6.2, the movelets are grouped according to their occlusion patterns. There are 16 common occlusion patterns in this training set. A proportional number of codewords are trained from the group of movelets associated with each occlusion pattern.

Given the trained codewords and their associated movelets, we learn the transition matrix of HMM for each action as discussed in section 6.4. The transition matrix represents the possible spatial and temporal paths of codewords that an action can pass through. In Figure 6.7, we show a few examples of codewords that are passed by actions of stepping, walking and running all recorded at 90° angle of view.

Recognition of the Actions

6 subjects participate in the test experiments. A sequence of 1050 frames (35 seconds video) is captured for each of the 15 actions performed by every subject. Foreground data are obtained by background subtraction. We answer three questions by this experiment: 1) how well can the actions be recognized? 2) how many frames are needed to recognize the action with certain accuracy? 3) how does the number of trained codewords influence the recognition performance?

Human vision does not need to observe the whole sequence (1050 frames) to

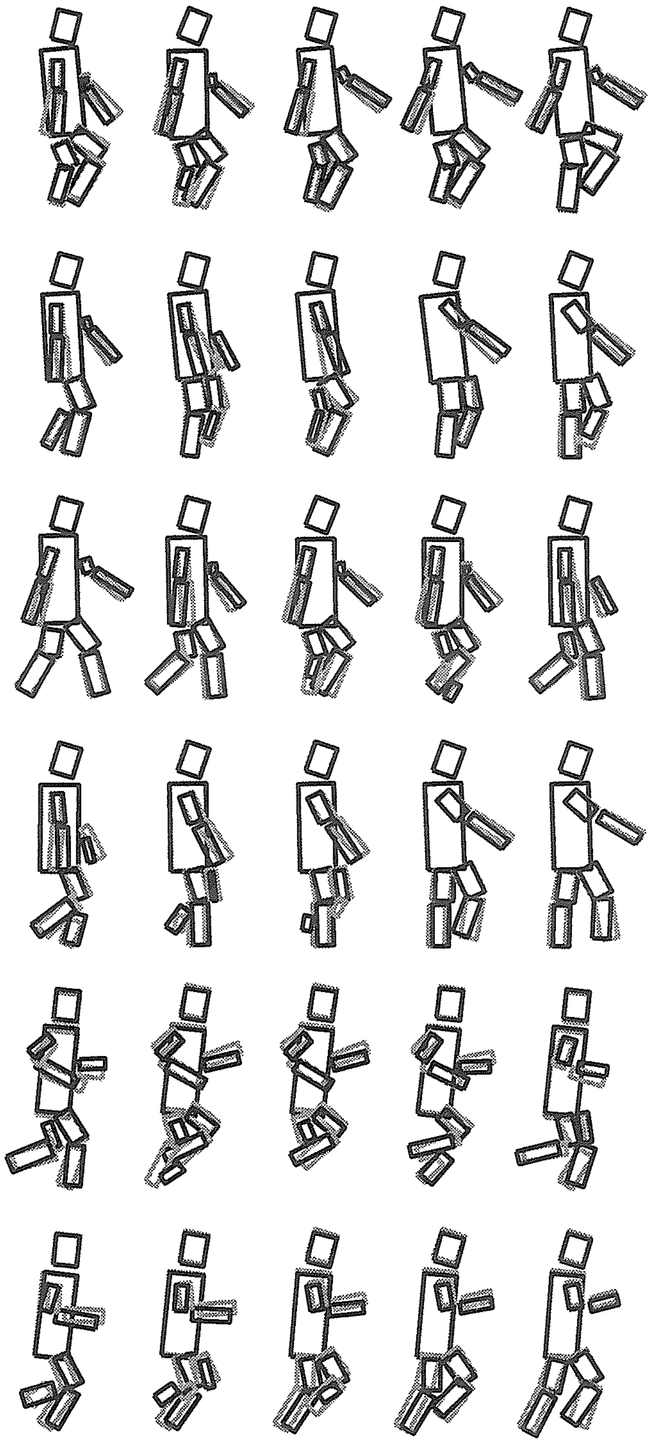


Figure 6.7: **Samples of Trained Codewords** (transformed shape S' is in gray, superimposed by the shape S in black). These are samples of codewords that actions 90° stepping (row 1 and row 2), 90° walking (row 3 and row 4), and 90° running (row 5 and row 6) can pass, obtained from their HMM transition matrices.

recognize the action. To test how many frames are necessary for a good recognition, each sequence is split into smaller segments of T frames long. The tail frames are ignored. Therefore, for a segment length T, there are $\lfloor 1050/T \rfloor \times 15 \times 6$ segments in total generated from splitting up the long sequences and these smaller segments are used for testing (15 is the number of actions and 6 that of subjects). We classify each segment into a learned action. Table 6.2 shows the confusion matrix for $T = 20$. The row and column labels are the action names. Each number in the matrix is the percentage of an action with the column label being recognized as that with the row label. Thus all the numbers in a column sum up to 100%.

Act	0°S	180°S	0°W	180°W	0°R	180°R	45°S	90°S	135°S	45°W	90°W	135°W	45°R	90°R	135°R
0°S	25%														
180°S	34%	40%	4%	36%											
0°W	10%	14%	52%	24%											
180°W		10%		9%											
0°R	31%	28%	36%	22%	80%	50%									
180°R		8%	8%	9%	20%	50%									
45°S							95%		70%						
90°S								100%							
135°S							5%		30%						
45°W										65%	3%	29%			
90°W											93%				
135°W										35%	4%	71%			
45°R													83%		71%
90°R														86%	
135°R													17%	14%	29%

Table 6.2: **Confusion Matrix for Recognition of Periodic Action** ($T = 20$)**.** In the labels, 'S', 'W' and 'R' are the abbreviations of stepping, walking and running actions, respectively.

We make the following three observations from Table 6.2.

First, the gaits from front (0°) and back (180°) views are difficult to recognize. This is because the silhouettes of different gaits from these two views are not very distinguishable. Therefore, we will not discuss them any more in this chapter.

Secondly, consider the other three views 45°, 90°, and 135°, the gaits of the action is easier to recognize than the directions. The three lower 3×3 blocks along the diagonal line in the table for three different gaits indicate that there is no problem in telling if the action is stepping, walking, or running.

Thirdly, we claim that an action viewed from the 45° view angle is basically not separable from that from the 135° view angle. This observation might be surprising. The direction of an action captured at either 45° or 135° view angle can be easily recognized from the original sequence by human vision because there is more information to be used, such as face, hair, texture, etc. However, if all these clues are removed by background subtraction and only the shape and motion are remained as the black-white foreground image sequence, human vision is very confused to tell these two directions either. Figures 6.8, 6.9, 6.10, and 6.11 show some samples of the original images, the extracted foreground data and the best fitting codewords from HMM of 4 test actions (45° walking, 135° walking, 90° running, and 90° stepping, respectively). Compare Figures 6.8 and 6.9: although the two actions, 45° walking and 135° walking, are performed by two different subjects (one female and the other male), their individual foreground data are so similar to each other (the middle columns in two figures) that they are represented by the same codewords (the right columns in two figures). Therefore 45° and 135° walking can be regarded as one action, so are 45° and 135° stepping, as well as 45° and 135° running.

Now let us discard the actions captured from front and back views, and focus on those taken at 45°, 90°, and 135° view angles. Once again we train $N = 270$ codewords from the 9 training actions of these three views and test the recognition algorithm with various segment length T. The recognition rate is the percentage of an action in all the containing segments being recognized correctly. Figure 6.12 demonstrates how the recognition rate varies when the segment length T goes from 2 to 40 for six actions (here we regard the 45° and 135° actions as one already). For visual clarity, we show the relationship in two plots, each for 3 actions. From the two plots, it is shown that better performance is gained with longer segment length T. The action can be reliably recognized after about 20 frames, which is 0.66 seconds of action. The chance level for recognizing the actions at 90° view angles shown in the left plot of Figure 6.12 is 1/9, and that for recognizing the actions in the right plot is 2/9 since the actions of the same gait captured at 45° and 135° view angles are regarded as one action. Our recognition rates are much higher than these chance levels.

What is the appropriate number of codewords we should train to represent the whole space of movelets? This question can be answered by experiments. We train different number $N = (5, 9, 18, 27, 36, 45, 90, 135, 180, 225, 270)$ of codewords from the training set of movelets for actions taken at $45°$, $90°$, and $135°$ view angles, and repeat the recognition experiments with each group of trained codewords. Figure 6.13 shows how the recognition rate at segment length $T = 20$ varies with respect to different number of trained codewords. For all the actions, above 80% recognition rates are achieved with $N \geq 135$.

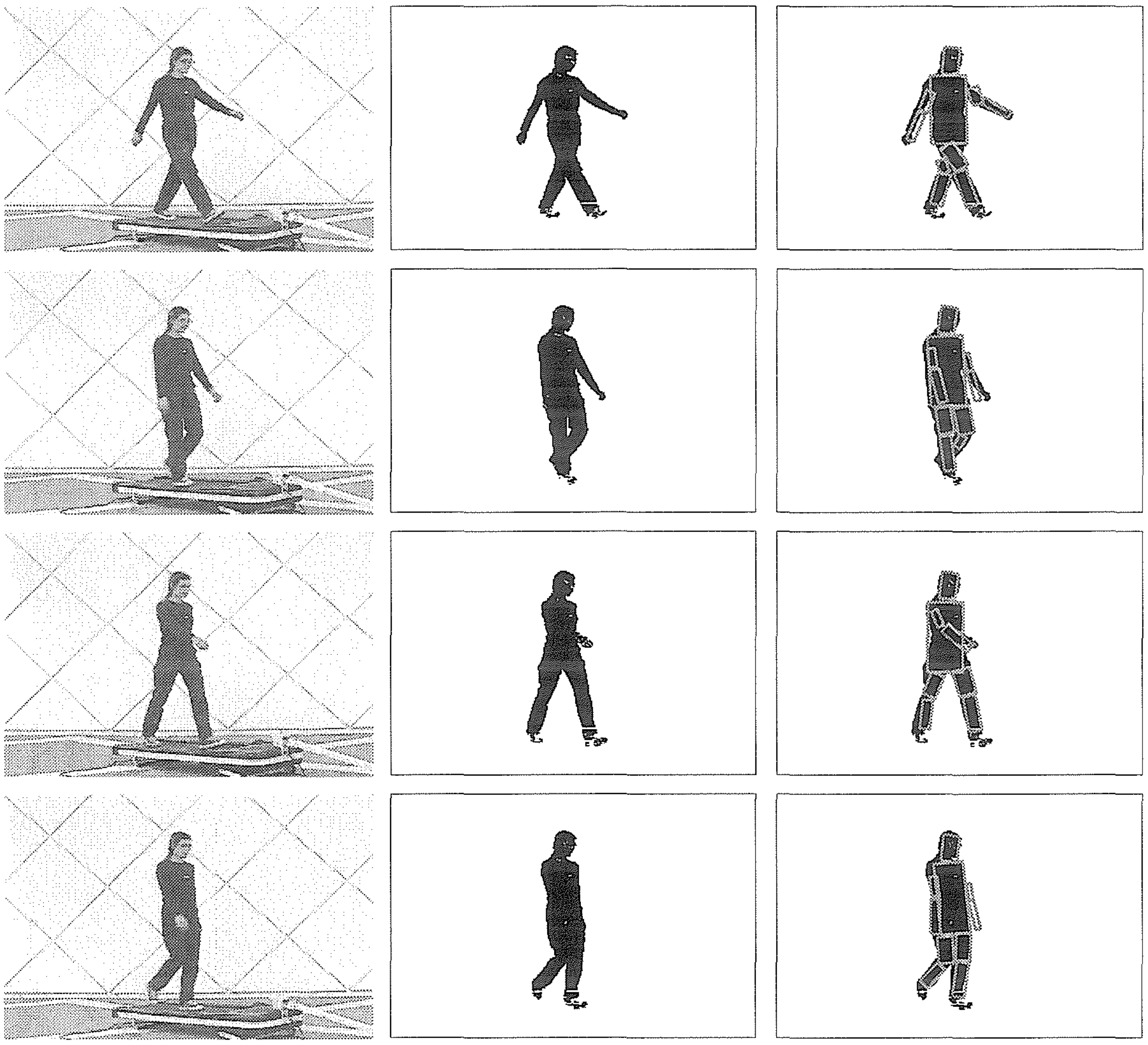


Figure 6.8: 45° **Walking**. (left): sample images. (middle): foreground data after background subtraction. (right): best fitting codewords.

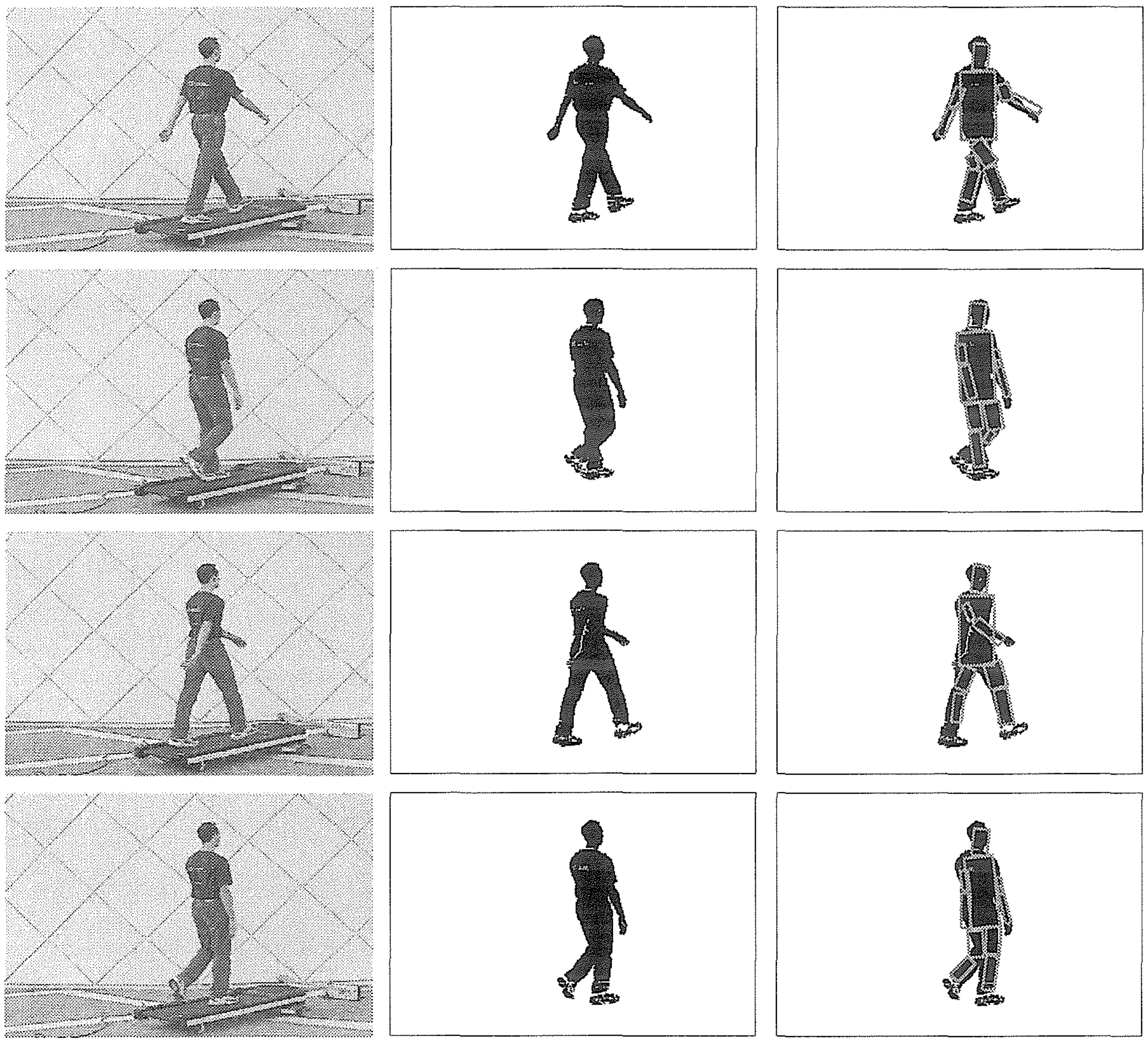


Figure 6.9: 135° **Walking**. (left): sample images. (middle): foreground data after background subtraction. (right): best fitting codewords.

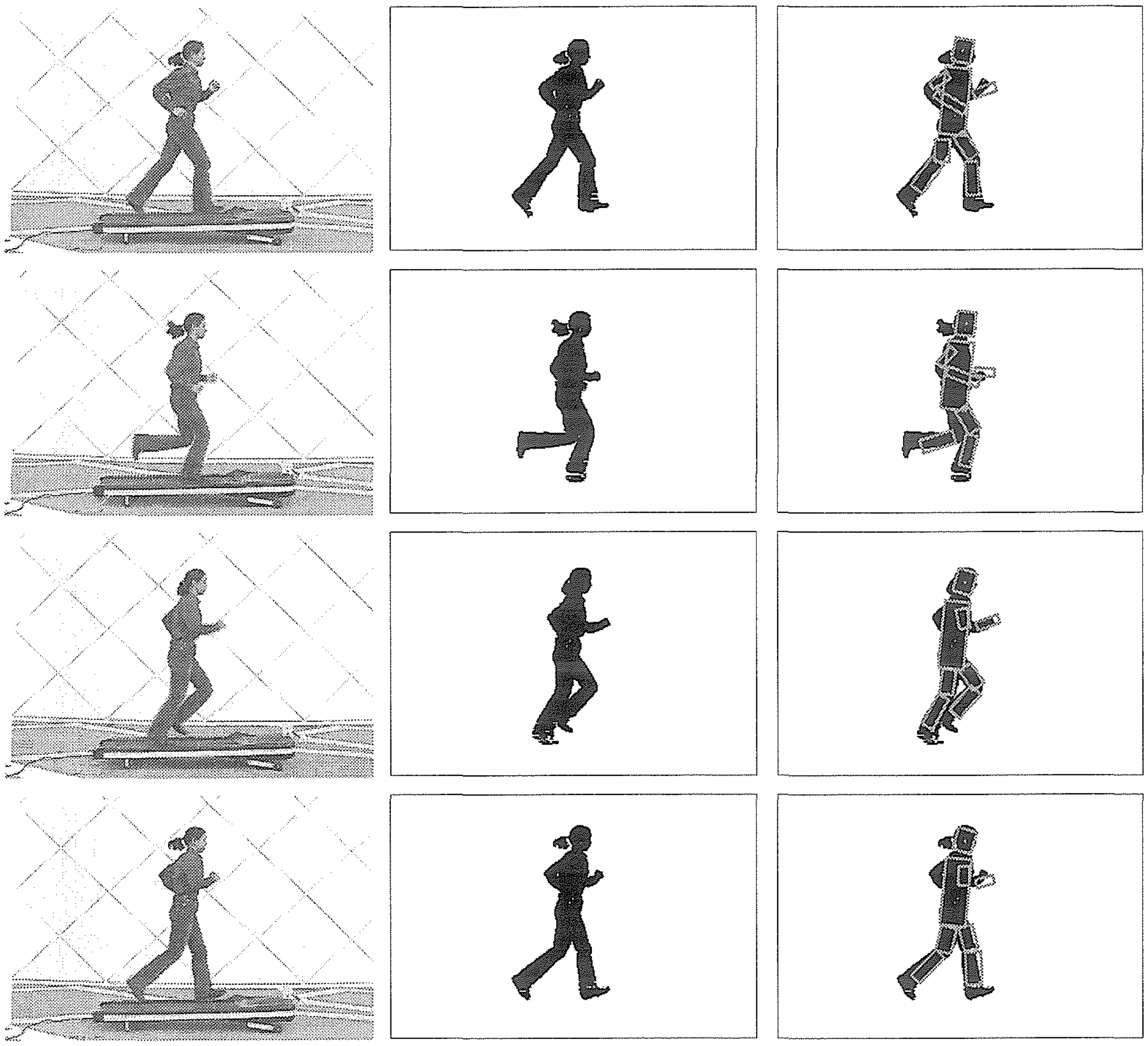


Figure 6.10: 90° **Running**. (left): sample images. (middle): foreground data after background subtraction. (right): best fitting codewords.

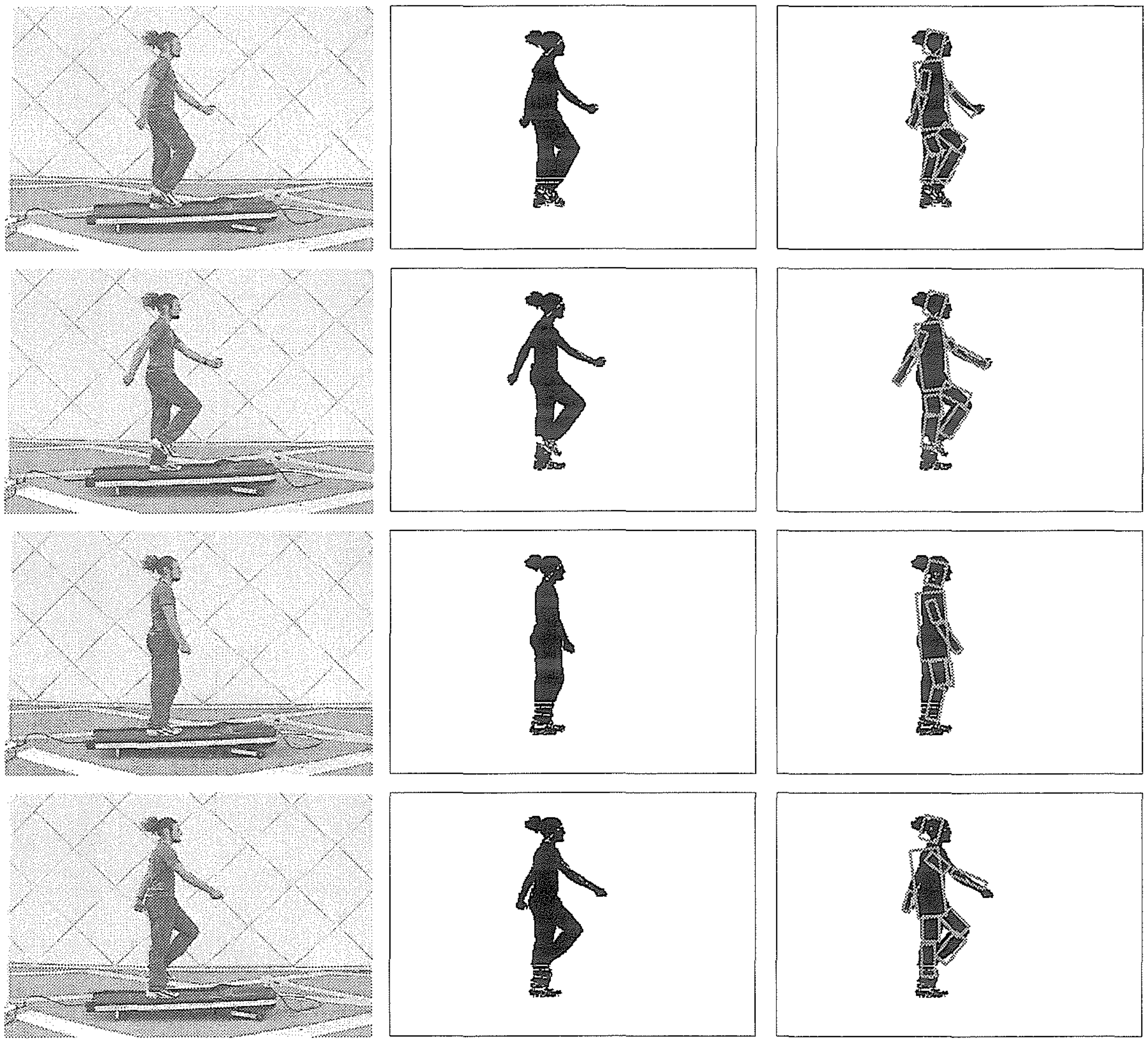


Figure 6.11: 90° **Stepping**. (left): sample images. (middle): foreground data after background subtraction. (right): best fitting codewords.

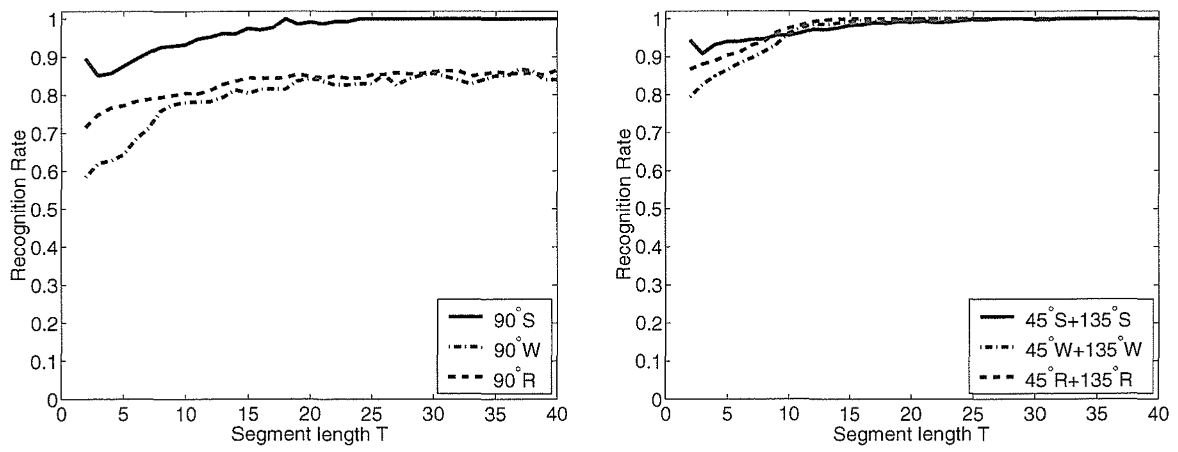


Figure 6.12: **Recognition Rate vs. Segment Length T.** The recognition is to recognize the actions in the observed segments of T frames long as one of the 9 learned periodic actions (stepping, walking and running viewed from 45°, 90°, and 135° angles). The recognition rate is the percentage of an action in all the containing segments being recognized correctly. A total of 270 codewords are trained and used for this recognition experiment. The chance level for recognizing the actions at 90° view angles shown in the left plot is 1/9, and that for recognizing the actions in the right plot is 2/9 since the actions of the same gait captured at 45° and 135° view angles are regarded as one action. Our recognition rates are much higher than the chance levels.

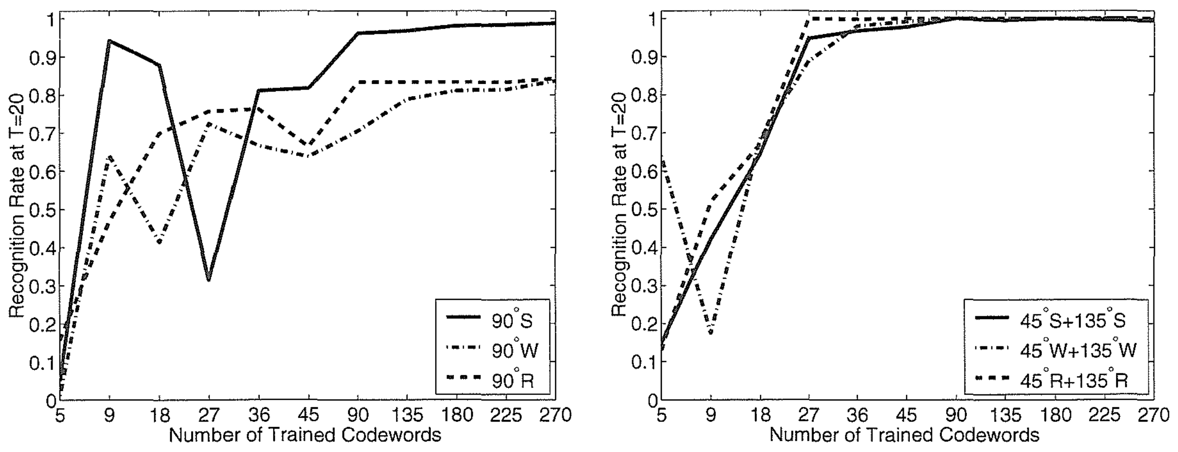


Figure 6.13: **Recognition Rate vs. Number of Trained Codewords for Periodic Actions.** The recognition of the observed actions as one of the 9 learned periodic actions (stepping, walking and running viewed from 45°, 90°, and 135° angles) is repeated with different numbers of trained codewords. The testing segment length is set at $T = 20$ for the reported recognition rate. The chance levels for recognizing these actions are the same as those in Figure 6.12.

6.5.4 Recognition of Nonperiodic Actions

In this section we describe the experiments in modeling and recognizing nonperiodic actions. The actions to reach 8 different directions using the right arm are captured from front view. Starting from the one reaching upward, we name the actions as $Rh0°$, $Rh45°$, $Rh90°$, $Rh135°$, $Rh180°$, $Rh225°$, $Rh270°$, and $Rh315°$, respectively, while the reaching angle increases counter clockwise.

Training Actions

The training actions are performed by 1 subject with the specially designed color-coded clothes. The training subject repeats each reaching action 20 times in 40 seconds. 240 codewords are learned from this set of actions first. The representative frames of the actions are shown in Figure 6.14.

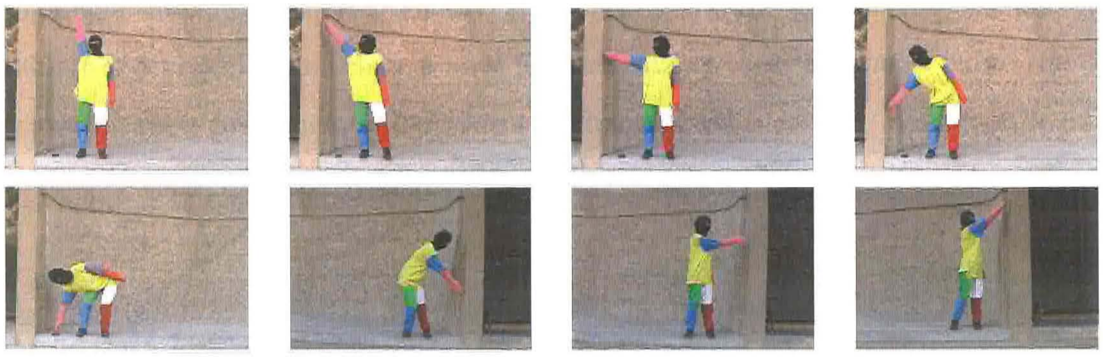


Figure 6.14: **Key Frames of the 8 Training Reaching Actions.**

Recognition of the Actions

A set of 400 sequences of different reaching actions done by 5 subjects are captured for testing. Each sequence is about 60 frames long (2 seconds of video). For a nonperiodic action, the recognition is done using all the frames that the sequence contains. The confusion matrix is shown in Table 6.3. Most of the actions are recognized with 100% success rate. The examples of images, foreground data and best fitting codewords for different testing actions are shown in Figures 6.16 and 6.17.

Different numbers $N = (4, 8, 16, 24, 32, 40, 80, 120, 160, 200, 240)$ of codewords are also trained for this set of actions, and the recognition experiments are repeated with

Reach	$Rh0°$	$Rh45°$	$Rh90°$	$Rh135°$	$Rh180°$	$Rh225°$	$Rh270°$	$Rh315°$
$Rh0°$	90%							
$Rh45°$		100%						
$Rh90°$			100%					
$Rh135°$				100%				
$Rh180°$					100%			
$Rh225°$						100%		
$Rh270°$							100%	
$Rh315°$	10%							100%

Table 6.3: **Confusion Matrix for Recognition of Nonperiodic Actions** with 240 trained codewords.

each group of trained codewords. Figure 6.15 shows how the recognition rate varies with respect to different number of trained codewords. For all the actions, above 80% recognition rates are achieved with $N \geq 120$ and 100% recognition rate are achieved for 6 out of 8 actions. The chance level for recognizing these actions is only $1/8$. Our recognition rates are much higher than the chance level.

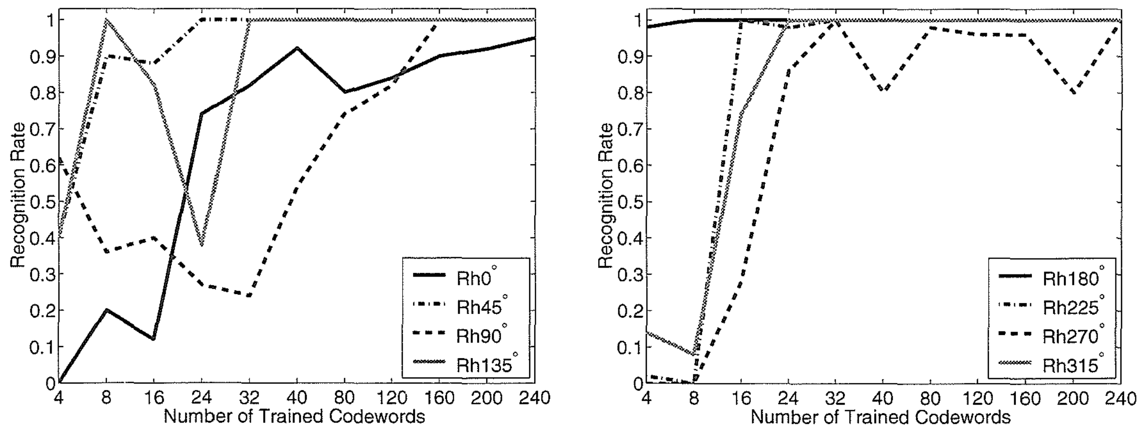


Figure 6.15: **Recognition Rate vs. Number of Trained Codewords for Nonperiodic Actions.** The recognition of the observed actions as one of the 8 learned nonperiodic reaching actions is repeated with different numbers of trained codewords. The recognition rate is the percentage of an action in all the containing sequences being recognized correctly. The chance level for recognizing these actions is $1/8$. Our recognition rates are much higher than the chance level when the number of trained codewords is larger than 120.

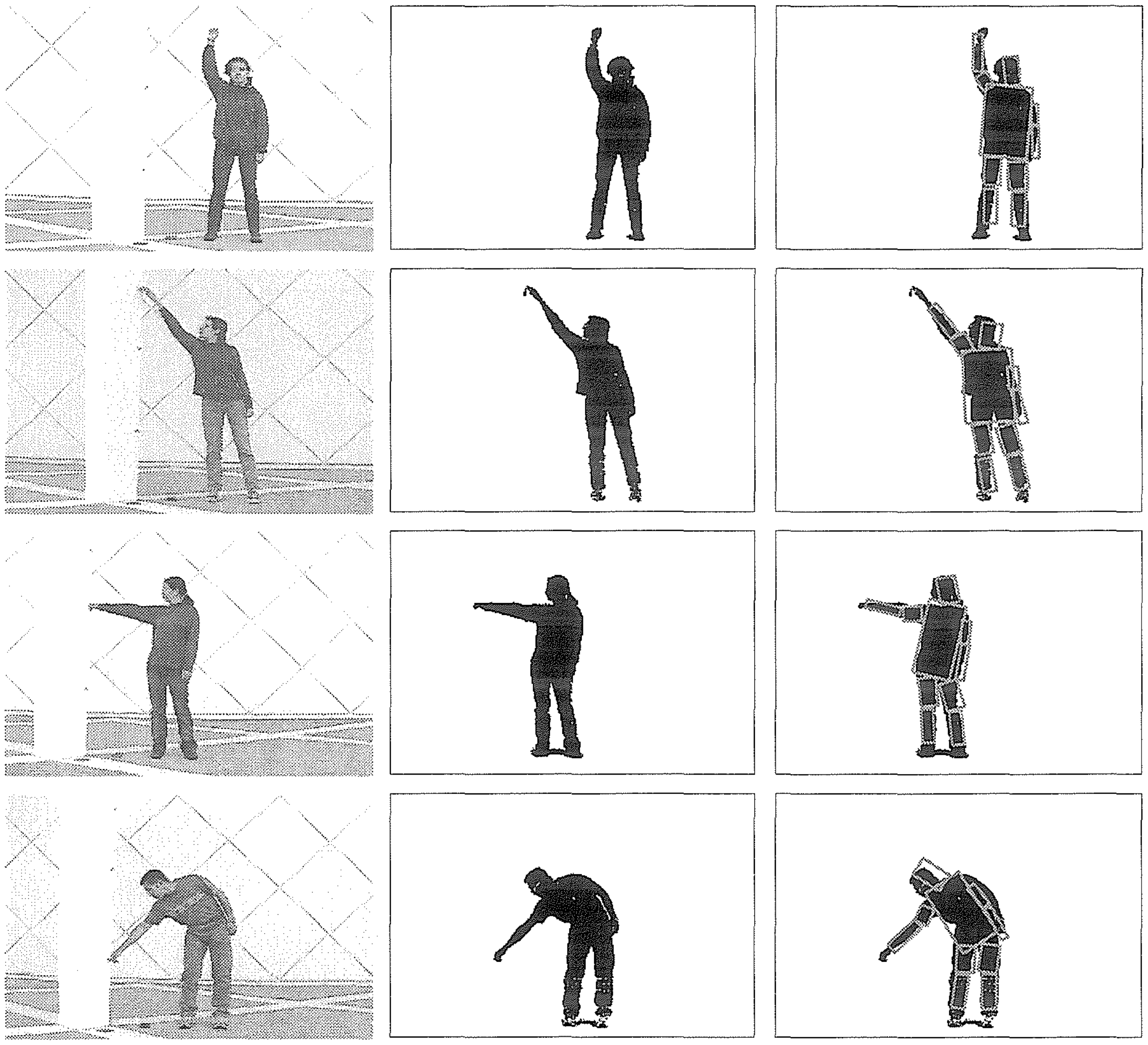


Figure 6.16: **Nonperiodic Actions Recognition I.** (left): sample images of reaching actions for testing (*Rh*0°, *Rh*45°, *Rh*90°, *Rh*135°). (middle): foreground data after background subtraction. (right): best fitting codewords.

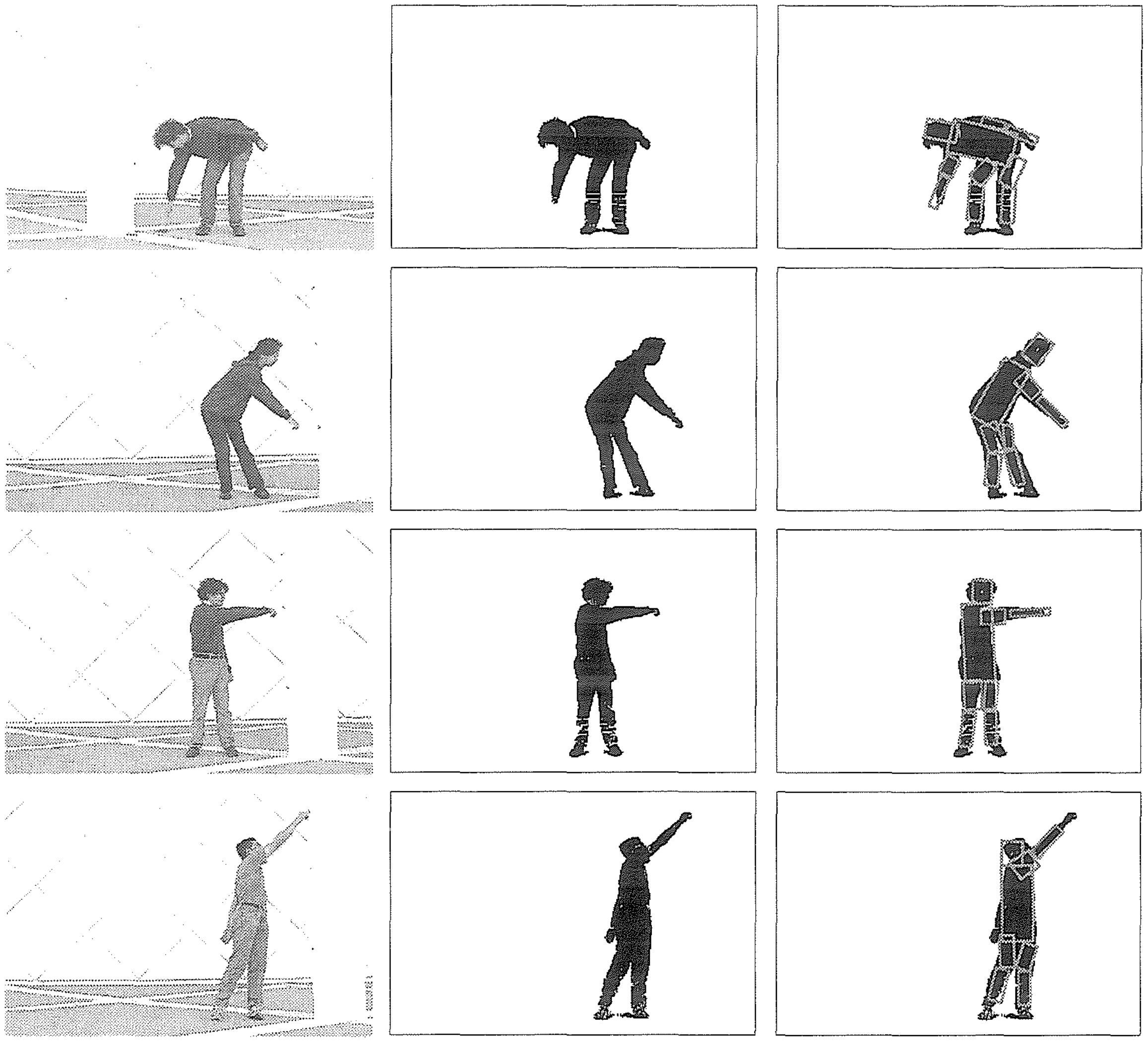


Figure 6.17: **Nonperiodic Actions Recognition II**. (left): sample images of reaching actions for testing ($Rh180°$, $Rh225°$, $Rh270°$, $Rh315°$). (middle): foreground data after background subtraction. (right): best fitting codewords.

6.5.5 Extension: Recognition of More Natural Periodic Actions

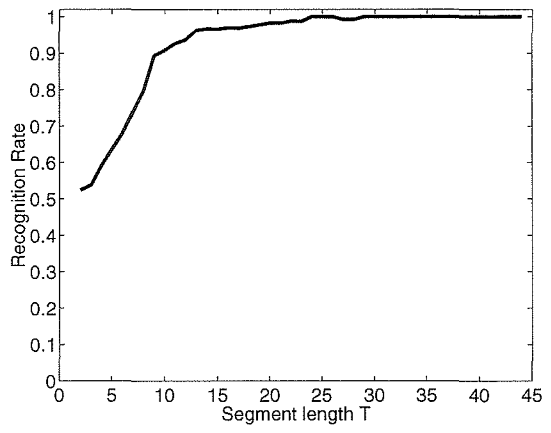
In section 6.5.3 we have demonstrated the system performance in recognizing periodic actions. The three actions of walking, stepping and running are performed on a treadmill. Such an experimental setting allows us to record an action as long as we want and thus simplifies the tasks of training and testing. Using the same set of trained codewords, we demonstrate how well some more natural periodic actions can be recognized in this section.

A set of 12 sequences of periodic actions of 3 different subjects are captured. They either walk or run from left to right in front of a fixed camera. Each action is repeated twice by all subjects and captured as two sequences, respectively. Foreground data are obtained by background subtraction as before. Some sample images and extracted foreground data are shown in the left and middle columns of Figures 6.19, 6.20 and 6.21. Notice that the foreground data after background subtraction are not as clean as data shown in previous experiment in section 6.5.3. In Figure 6.20, the background color is so close to that of the subject's arm skin that the arms are not detected as foreground at all. The translational speed of the subjects and the fuzzy foreground increase the difficulties in recognizing these actions.

We use the same training results obtained from training actions of $45°$, $90°$ and $135°$ walking, stepping and running with codeword number $N = 270$ as shown in section 6.5.3 to recognize these more natural actions. The best fitting codewords for the sequences are recognized and some samples are shown in the right columns of Figures 6.19, 6.20 and 6.21. Since the test sequences in this experiment are very short ($\approx$ 50 frames for walking and $\approx$ 30 for running), to demonstrate how the number of sequence frames influence the recognition rate, all possible smaller segments of T consecutive frames long in a sequence are generated for recognition. For the walking action, the maximum T is chosen as the shortest frame length of the 6 walking sequences captured. Same is done for the running action. Table 6.4 shows the confusion matrix of these two actions at $T = 20$. One can observe that the gaits (walking or

Act	walking from left to right	running from left to right
45°S		
90°S		
135°S		
45°W	2%	
90°W	98%	
135°W		
45°R		10%
90°R		90%
135°R		

Table 6.4: **Confusion Matrix for Recognition of More Natural Periodic Actions** ($T = 20$)



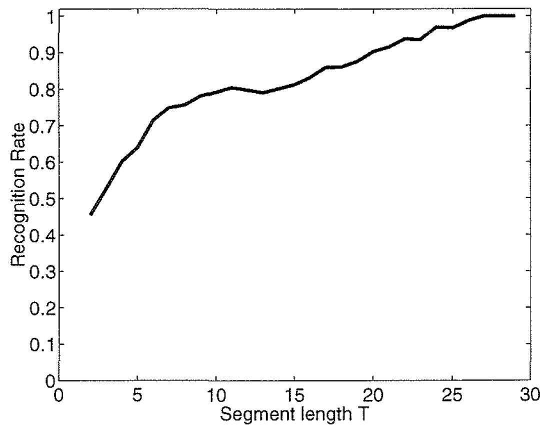


Figure 6.18: **Recognition Rate vs. Segment Length T for Natural Walking (left) and Running (right) from Left to Right** . The recognition is to recognize the actions in the observed segments with T frames long as one of the 9 learned periodic actions (stepping, walking and running viewed from 45°, 90°, and 135° angles). The recognition rate for natural walking or running is the percentage of the action in all the containing segments being recognized as 90° walking or 90° running, respectively. The chance level for recognizing the actions is 1/9. Our recognition rates are much higher than the chance level.

running) are correctly recognized, and that 98% of walking segments are recognized as 90° walking and 90% of running segments are recognized as 90° running. If we regard the walking and running from left to right as 90° actions respectively, Figure 6.18 shows how the correct recognition rate varies with respect to various segment length T for walking (left plot) and running (right plot). Above 98% correct recognition rates are achieved for both actions at $T \geq 20$ and $T \geq 25$, respectively. The chance level for recognizing these actions is only 1/8. It demonstrates that our recognition scheme can recognize very well the periodic actions of subjects with normal translational speed although the actions are trained by those without translational speed.

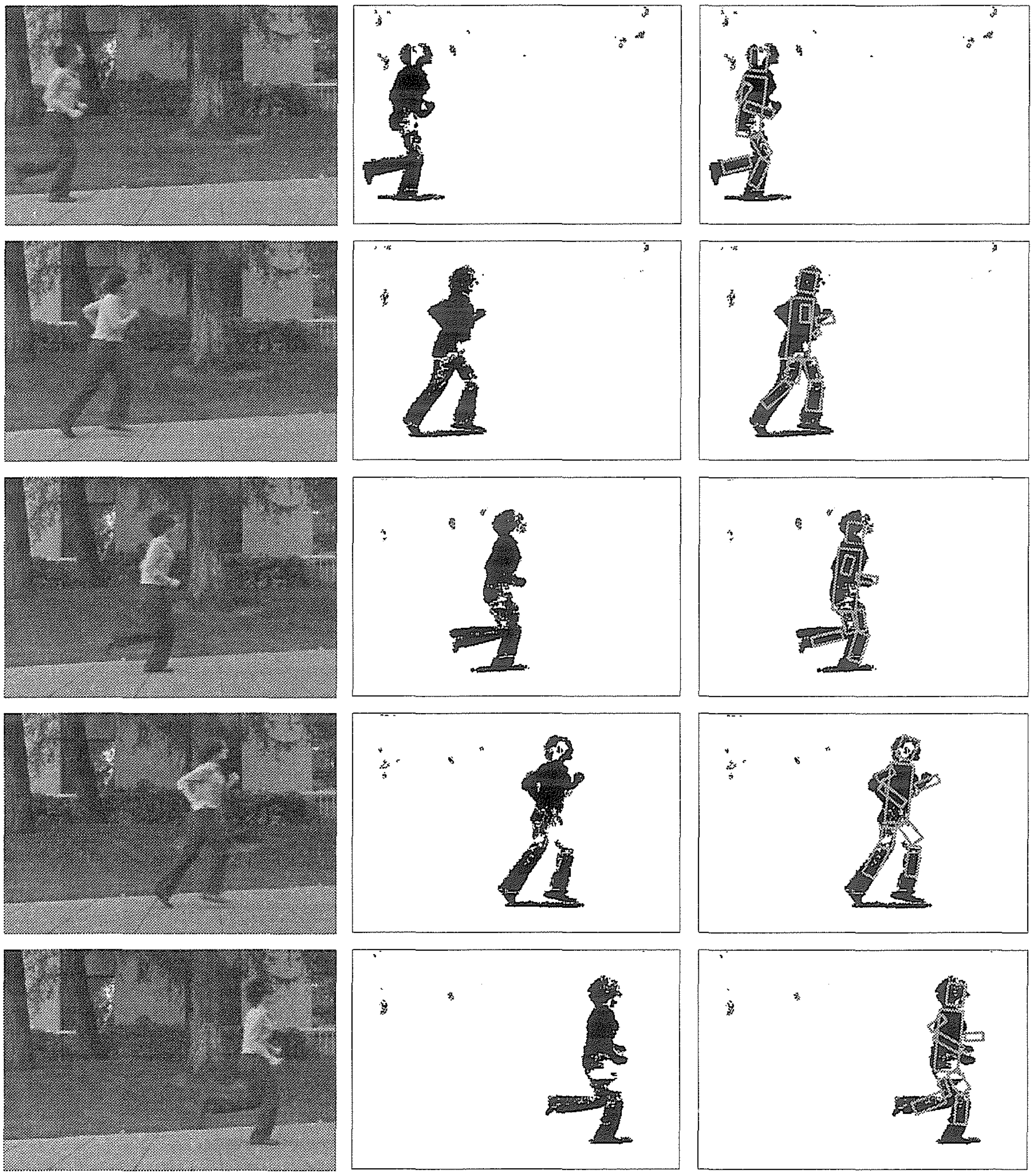


Figure 6.19: **A Female Subject Runs from Left to Right**. (left): sample images. (middle): foreground data after background subtraction. (right): best fitting codewords.

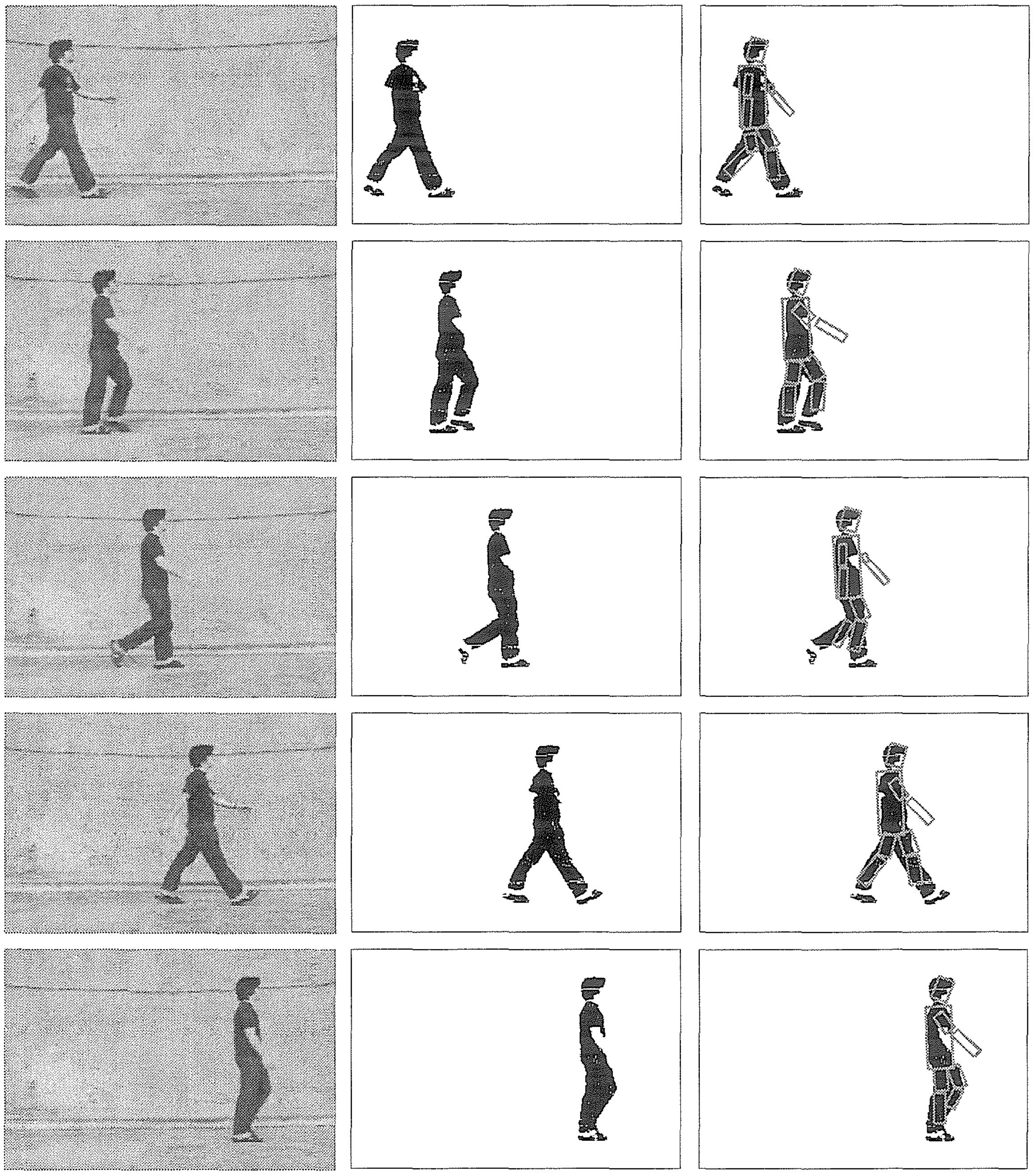


Figure 6.20: **A Male Subject Walks from Left to Right**. (left): sample images. (middle): foreground data after background subtraction. (right): best fitting codewords.

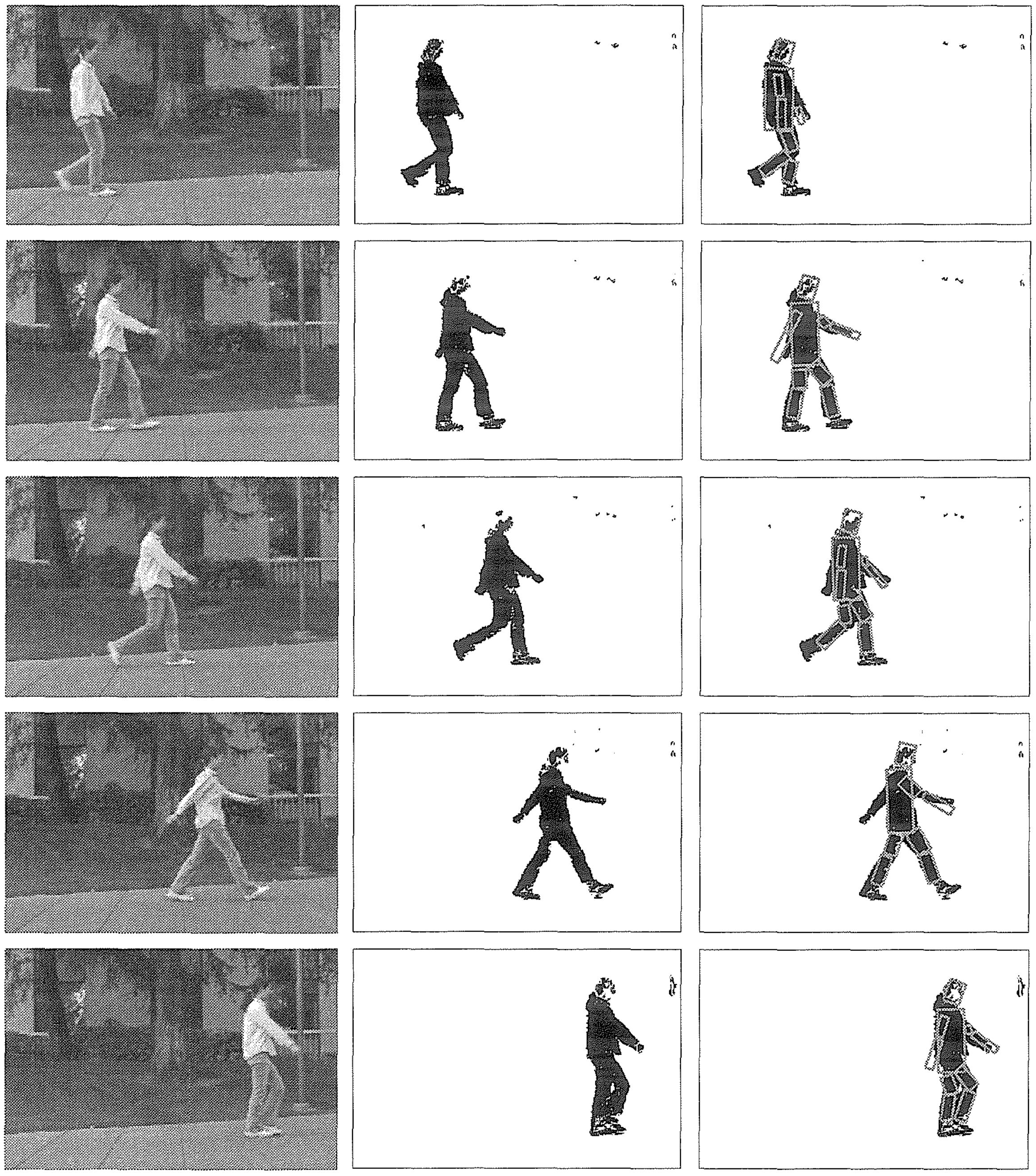


Figure 6.21: **A Female Subject Walks from Left to Right**. (left): sample images. (middle): foreground data after background subtraction. (right): best fitting codewords.

6.6 Conclusion

In this chapter we propose and test an algorithm for the recognition of both human poses and actions simultaneously from an image sequence. The experiments show that above 80% recognition rates are achieved for all the test actions while over half of the actions can be recognized with an accuracy rate higher than 98%. It also demonstrates that our recognition scheme can recognize very well the periodic actions of subjects with normal translational speed although the actions are trained by those without translational speed. Our fundamental idea can be extended to recognizing other types of actions, for example, gesture action, as long as the experimental setup for training is properly designed. A possible future work that we may pursue is to find a better probabilistic model other than the simple binomial one to fit a codeword to an observation. In addition, the background subtraction is assumed to be successful in our system in order to obtain the clear foreground region. We also ran some experiments on the sequences captured by a moving camera. The dominant background (camera) motion is estimated using the scheme proposed in [6] and the foreground region is obtained as the outliers to the background motion. However the resulted foreground region is very poor because only body edges can be reliably segmented as foreground and most internal regions are missed. Consequently our system is not able to reliably recognize the actions from such poor foreground data. How to recognize actions from image sequences without background subtraction remains an open and challenging problem.

Chapter 7 Conclusion and Future Direction

Motion analysis is one of the most important and active areas in computer vision. In this thesis we have presented our studies toward the problems of 3D general motion estimation and segmentation and 2D human motion detection and recognition.

It has been extensively studied to recover 3D motion from either 2-views or 3-views. In particular, there is a whole body of literature in the last decade addressing the 3-view motion analysis where the trilinear constraints and trifocal tensor arise. Trilinear constraints as simple linear functions of trifocal tensor give satisfactory estimates of $3D$ motion parameters that are close to the optimal maximum likelihood solutions. However, it is difficult to explain such good performance because so far before our work trilinear constraints have been treated only as algebraic equations without explicit physical meaning. In the beginning of this thesis, we presented our discovery that trilinear constraints are actually equivalent to the optimally weighted 3D depth matching constraints (Chapter 3). Therefore they have clear geometrical meaning. Minimizing the trilinear algebraic equations in a least square sense over all points is equivalent to minimizing a geometric depth matching error in 3D space with appropriate weight applied to every point. A geometrical interpretation to the weight functions of trilinear constraints is also given. These theoretical work fills in a gap in 3D motion estimation for thorough understanding of trilinear constraints. One additional contribution in this part is a robust scheme proposed to propagate scale information between different pairs of views which is significantly insensitive to noise in the measurement data.

Segmentation of multiple rigid motions is a prerequisite to motion estimation algorithms because the latter can only deal with one rigid motion. A method which combines modified separation matrix scheme and 3D motion based EM algorithm is proposed in Chapter 4 to address the segmentation problem from only two views. The main advantages of the modified separation matrix scheme are that it has excellent performance with spatially well separated motion groups and demands very low computational cost. The drawback is that the method is based on some hypothetic spurious motions and does not really characterize different true 3D motions between groups, thus it fails in segmenting complicated scenes with spatially overlapped groups. To overcome this shortcoming, we developed a 3D motion based EM algorithm which iteratively segments the groups and estimates the 3D motion parameters for each group starting from a random initialization. Its excellent performances are demonstrated in segmenting two complicated scenes: one contains two transparent cylinders rotating about the same axis but to opposite direction, and the other contains a forward-backward moving plane and a rotating cubic box hung in front. However, this EM algorithm is not perfect. It is time-consuming and sometimes global maximum is not guaranteed. We showed by a vast number of simulated experiments that to some extent the two algorithms are compensative to each other. Therefore, we propose to combine these two methods. Apply the efficient modified separation matrix scheme first to the scene. If it fails, apply the 3D motion based EM algorithm again. Such a combination algorithm can deal with the majority of segmentation problems of different difficult levels.

Compared with the well-done research in 3D motion estimation, 3D motion segmentation remains a difficult problem. Collaborating with Jin and Soatto in UCLA vision group, we have developed and implemented a real-time 3D motion estimation system which can reconstruct the camera trajectory and the observed scene structure in real time. We believe that this system is the best demonstration of the success of theoretical work so far in the vision field toward motion estimation. However, the current techniques for 3D motion segmentation can only be applied off line. It is still a long way to a unified and efficient framework to segment all possible scenes with guaranteed performance so that the ultimate segmentation algorithm can be embedded in a real-time motion recovery system.

It is very often that we are also interested in motions of some particular objects.

Human motion detection and recognition is very important to many applications, such as surveillance system, pedestrian detection for automatic vehicles, human-computer interface, etc. We have presented a human motion detection scheme which is able to detect human bodies by their motion patterns from a pair of frames in the presence of extraneous motions and self-occlusions between body parts (Chapter 5). By decomposing the human body into a triangulated graph, our method models human motion by an approximate joint density function of the positions and velocities of body joint features. If we know in advance the correspondence between the selected features in the image and the body joints (called feature labeling problem), then we can directly compute the likelihood that labeled configuration represents a human body by inserting the configuration into our model. However, we do not have this labeling information beforehand. Therefore, the detection is performed by setting a threshold to the summation of all the likelihoods each of which is the probability a labeling on the observed image features representing a human body. The likelihood summation is estimated efficiently by an algorithm similar to dynamic programming. Once the person is detected, its location is determined to be the best labels with the maximum labeling likelihood by applying dynamic programming. Our detection system reaches 90% detection rate from a pair of frames.

The main drawback of our approach is that, apart from self-occlusion, the body joints in our model may not be reliably observed as candidate features in the more natural image sequences. Depending on the different clothes people wear and the different lighting conditions, the candidate features selected on the human body in the images may be far away from the correct positions of body joints. It has been demonstrated that if a human body is not modeled by joints, but the middle positions of body parts and/or limbs, it is not easy to perceive its motions any more. Therefore, a question for future study is: can we infer the human body joints from the observed features which may locate anywhere on the human body as well as the background? This is crucial to the generalization of our detection system. Otherwise, the subject has to be dressed specially as in our experiment to make sure that most body joints are observable. Another issue arising from our system is the choice of the triangulated decomposition. Our current triangulation structure is only one among many possible choices. What is the best decomposition structure for detection? My colleague Song has further exploration in [83] to answer this question.

Detection is followed by recognition of human actions when a human body is observed long enough in an image sequence. In this thesis, we also proposed an algorithm for the recognition of human actions (Chapter 6). The algorithm consists of three stages: background subtraction, body pose classification, and action recognition. A new concept *movelet* is introduced to represent a human pose in space-time by the shapes and motions of image patches which correspond to the main body parts. The set of all possible movelets is very large. We discretize this set into a smaller collection of movelet codewords by vector quantization. Then human actions are modeled by HMM. However, unlike the usual HMM whose states do not exactly correspond to some physical terms, we predefine the HMM states of the actions as the movelet codewords. The different HMM transition matrices of different actions represent the different spatial-temporal paths in the codeword space that the actions can possibly take. Such an HMM framework enables us to simultaneously find the best action as recognition and the best codeword sequence as representation of the observed sequence. The algorithm is tested on both periodic and nonperiodic actions. Specially designed color-coded clothes are used for training the movelet codewords. Above 80% recognition rates are achieved for all actions. The influences of the sequence length and the number of codewords on algorithm performance are studied experimentally in this thesis. As an extension, it also demonstrates that our recognition scheme can recognize very well the periodic actions of subjects with normal translational speed although the actions are trained by those without translational speed.

Our recognition algorithm is limited by the requirement of background subtraction which is not always possible. When the background information is not available or the camera is not still, a practical substitute for background subtraction is to extract the foreground region by applying the independent motion segmentation [6]. However, compared with the background subtraction, the disadvantage of this segmentation technique is that it can not extract the non-textured foreground regions from the background. We have done some experiments on the sequences taken by a moving camera. The scheme proposed in [6] is applied to estimate the dominant camera motion and segment the foreground regions as the outliers to the camera motion. What we observe is that the body edges are clearly detected as foreground and most of internal regions of body are missed. Our recognition scheme cannot reliably recognize the actions from such poor foreground data. Therefore, an interesting and challenging future work is to reliably obtain the foreground regions from general image sequences. Another open issue is whether there is a better model than the simple binomial distribution that fits codewords to foreground configurations. If we consider the problem from the computational cost point of view, our current centroid searching scheme is not very efficient. There is a searching algorithm called "brent's searching", which might be applied to this problem. It is worthy exploring how efficient the brent's searching can be and how much the recognition performance using this searching algorithm might be lower than that using the current pixel to pixel exhaustive searching.

In addition, although we have presented both a human motion detection system and a human action recognition algorithm, they are kind of independent to each other. It will be an interesting future research to unify both into one framework so that the human body can be detected in the shortest time (2 frames or 60 ms), and then its action can be recognized after 15 to 20 frames.

We hope our work on motion analysis presented in this thesis will enlighten more interesting work in the future.

Bibliography

[1] Y. Abdel-Aziz and H. Karara, "Direct linear transformation into object space coordinates in close-range photogrammetry", In *Proc. ASP Symposium on Close-Range Photogrammetry, Urbana, Illinois, USA*, pages 1–18, 1971.

[2] Y. Amit and A. Kong, "Graphical templates for model registration", *IEEE Trans. Pattern Anal. Mach. Intell.*, 18:225–236, 1996.

[3] J. Barron, D. Fleet, and S. Beauchemin, "Performance of optical flow techniques", *Int. J. of Computer Vision*, 12(1):43–78, 1994.

[4] C. Bishop, *Neural networks for pattern recognition*, Oxford University Press, 1995.

[5] A. Bissacco, F. Nori, and S. Soatto, "Recognition of human gaits", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, volume II, pages 52–57, 2001.

[6] M. Black and P. Anandan, "The robust estimation of multiple motions: parametric and piecewise-smooth flow fields", *Computer Vision and Image Understanding*, 63(1):75–104, 1996.

[7] J. Bouguet, *Thesis: Visual methods for three-dimensional modeling*, California Institute of Technology, 1999.

[8] J. Bouguet and P. Perona, "Visual navigation using a single camera", In *Proc. 5th Int. Conf. Computer Vision*, pages 645–652, 1995.

[9] C. Bregler, "Learning and recognizing human dynamics in video sequences", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 568–574, 1997.

[10] C. Bregler and J. Malik, "Tracking people with twists and exponential maps", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 8–15, 1998.

[11] D. Brown, "Calibration of close range cameras", In *Proc. 12^{t}h Congress Int. Soc. Photogrammetry, Ottawa, Canada*, 1972.

[12] J. Burns, A. Hanson, and E. Riseman, "Extracting straight lines", *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(4):425–455, 1986.

[13] J. Davis and A. Bobick, "The representation and recognition of action using temporal templates", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 928–934, 1997.

[14] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the em algorithm", *J.R.Statist. Soc. B*, pages 39:1–38, 1977.

[15] J. Fang and T. Huang, "Some experiments on estimating the 3-d motion parameters of a rigid body from two consecutive image frames", *IEEE Trans. Pattern Anal. Mach. Intell.*, 6:547–554, 1984.

[16] O. Faugeras, "What can be seen in three dimensions with an uncalibrated stereo rig?", In *Proc. 2nd Europ. Conf. Comput. Vision, G. Sandini (Ed.), LNCS-Series Vol. 588, Springer-Verlag*, pages 563–578, 1992.

[17] O. Faugeras, *Three dimensional computer vision, a geometric viewpoint*, MIT Press, 1993.

[18] O. Faugeras, Q. Luong, and S. Maybank, "Camera self-calibration: theory and experiments", In *Proc. 2nd Europ. Conf. Comput. Vision, G. Sandini (Ed.), LNCS-Series Vol. 588, Springer-Verlag*, pages 321–334, 1992.

[19] O. Faugeras and B. Mourrain, "On the geometry and algebra of the point and line correspondences between n images", In *Proc. 5th Int. Conf. Computer Vision*, pages 951–956, 1995.

[20] O. Faugeras and T. Papadopoulo, "A nonlinear method for estimating the projective geometry of 3 views", In *Proc. 6th Int. Conf. Computer Vision*, pages 477–484, 1998.

[21] P. Felzenszwalb and D. Huttenlocher, "Efficient matching of pictorial structures", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, volume II, pages 66–73, 2000.

[22] X. Feng, J. Bouguet, and P. Perona, "A new geometrical interpretation of trilinear constraints", In *Proc. 4th Asian Conf. Computer Vision*, pages 389–396, 2000.

[23] X. Feng and P. Perona, "Scene segmentation from 3d motion", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 225–231, 1998.

[24] X. Feng and P. Perona, "Human action recognition by sequence of movelet codewords", *1st International Symposium on 3D Data Processing Visualization Transmission*, 2002.

[25] L. Goncalves, *Thesis: Automatic observation and synthesis of human motion*, California Institute of Technology, 2000.

[26] L. Goncalves, E. Bernardo, E. Ursella, and P. Perona, "Monocular tracking of the human arm in 3d", In *Proc. 5th Int. Conf. Computer Vision*, pages 764–770, 1995.

[27] I. Haritaoglu, D. Harwood, and L. Davis, "Ghost: a human body part labeling system using silhouettes", In *Proc. Int. Conf. Pattern Recognition*, pages 77–82, 1998.

[28] I. Haritaoglu, D. Harwood, and L. Davis, "Who, when, where, what: a real time system for detecting and tracking people", In *Proc. Int. Conf. on Automatic Face and Gesture Recognition*, pages 222–227, 1998.

[29] C. Harris and M. Stephens, "A combined corner and edge detector", In *Alvey88*, pages 147–152, 1988.

[30] R. Hartley, "Estimation of relative camera positions for uncalibrated cameras", In *Proc. 2nd Europ. Conf. Comput. Vision, G. Sandini (Ed.), LNCS-Series Vol. 588, Springer-Verlag*, 1992.

[31] R. Hartley, "A linear method for reconstruction from lines and points", In *Proc. 5th Int. Conf. Computer Vision*, pages 882–887, 1995.

[32] R. Hartley, "Lines and points in three views and the trifocal tensor", *Int. J. of Computer Vision*, 22(2):125–140, 1997.

[33] R. Hartley, "Minimizing algebraic error in geometric estimation problems", In *Proc. 6th Int. Conf. Computer Vision*, pages 469–476, 1998.

[34] R. Hartley, R. Gupta, and T. Chang, "Stereo from uncalibrated cameras", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 761–764, 1992.

[35] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, Cambridge University Press, 2000.

[36] R.I. Hartley, "Projective reconstruction from line correspondences", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 903–907, 1994.

[37] R.I. Hartley and P. Sturm, "Triangulation", *Computer Vision and Image Understanding*, 68(2):146–157, 1997.

[38] A. Heyden, "Reconstruction from image sequences by means of relative depths", In *Proc. 5th Int. Conf. Computer Vision*, pages 1058–1063, 1995.

[39] A. Heyden and K. Åström, "Algebraic properties of multilinear constraints", *Mathematical Methods in Applied Sciences*, 20(13):1135–1162, 1997.

[40] B. Horn, "Relative orientation", *Int. J. of Computer Vision*, 4:59–78, 1990.

[41] N. Howe, M. Leventon, and W. Freeman, "Bayesian reconstruction of 3d human motion from single-camera video", *Tech. Rep. TR-99-37, a Mitsubishi Electric Research Lab*, 1999.

[42] T. Huang and O. Faugeras, "Some properties of the e-matrix in two-view motion estimation", *IEEE Trans. Pattern Anal. Mach. Intell.*, 11:1310–1312, 1989.

[43] S. Ioffe and D. Forsyth, "Human tracking with mixtures of trees", In *Proc. 8^{th} Int. Conf. Computer Vision*, volume I, pages 690–695, 2001.

[44] A. Jepson and M.J. Black, "Mixture models for optical flow computation", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 760–761, 1993.

[45] G. Johansson, "Visual perception of biological motion and a model for its analysis", *Perception and Psychophysics*, 14:201–211, 1973.

[46] S. Ju, M. Black, and Y. Yacoob, "Cardboard people: a parameterized model of articulated image motion", In *Proc. Int. Conf. on Automatic Face and Gesture Recognition*, pages 38–44, 1996.

[47] K. Kanatani, *Statistical optimization for geometric computation: theory and practice*, Elsevier Science, 1996.

[48] H. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections", *Nature*, 293:133–135, 1981.

[49] B.D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision.", *Proc. 7th Int. Joinnt Conf. on Art. Intell.*, 1981.

[50] Q. Luong and O. Faugeras, "The fundamental matrix: theory, algorithms, and stability analysis", *Int. J. of Computer Vision*, 17(1):43–75, 1996.

[51] Y. Ma, J. Kosecka, and K. Huang, "Rank deficiency condition of the multiple view matrix for mixed point and line features", In *Proc. 5th Asian Conf. Computer Vision*, 2002.

[52] Y. Ma, J. Kosecka, and S. Sastry, "Optimization criteria and geometric algorithms for motion and structure estimation", *Int. J. of Computer Vision*, 44(3):219–249, 2001.

[53] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, "Euclidean reconstruction and reprojection up to subgroups", In *Proc. 7th Int. Conf. Computer Vision*, pages 773–780, 1999.

[54] J. MacLean, A. Jepson, and R. Frecker, "Recovery of egomotion and segmentation of independent object motion using the em algorithm", *British Machine Vision Conference*, pages 175–184, 1994.

[55] S. Maybank, *Theory of reconstruction from image motion*, volume 28 of *Information Sciences*, Springer-Verlag, 1993.

[56] J. McIntosh and K. Mutch, "Matching straight lines", *Computer Vision, Graphics and Image Processing*, 43(3):386–408, 1988.

[57] G. McLachlan and K. Basford, *Mixture Models inference and applications to clustering*, Marcel Dekker Inc, New York, 1988.

[58] P. McLauchlan and D. Murry, "A unifying framework for structure and motion recovery from image sequences", In *Proc. 5th Int. Conf. Computer Vision*, pages 314–320, 1995.

[59] C. Morimoto, Y. Yacoob, and L. Davis, "Recognition of head gestures using hidden markov models", In *Proc. Int. Conf. Pattern Recognition*, pages 461–465, 1996.

[60] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, 1994.

[61] P. Neri, M. Morrone, and D. Burr, "Seeing biological motion", *Nature*, 395:894–896, 1998.

[62] T. Papadopoulo and O. Faugeras, "A new characterization of the trifocal tensor", In *Proc. 5th Europ. Conf. Comput. Vision, LNCS-Series Vol. 1406-1407, Springer-Verlag*, pages 109–123, 1998.

[63] R. Polana and R. Nelson, "Detecting activities", *Journal of Visual Communication and Image Representation*, 5(2):172–180, 1994.

[64] R. Polana and R. Nelson, "Low level recognition of human motion", In *IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pages 83–88, 1994.

[65] L. Rabiner and B. Juang, "An introduction to hidden markov models", *IEEE ASSP Magazine*, pages 4–16, Jan 1986.

[66] J. Rehg and T. Kanade, "Digiteyes: vision-based hand tracking for human-computer interaction", In *Proceedings of the Workshop on Motion of Non-Rigid and Articulated Bodies*, pages 16–24, November 1994.

[67] K. Rohr, "Incremental recognition of pedestrians from image sequences", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 8–13, New York City, June, 1993.

[68] R. Rosales and S. Sclaroff, "3d trajectory recovery for tracking multiple objects and trajectory guided recognition of actions", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, volume II, pages 117–123, 1999.

[69] R. Rosales and S. Sclaroff, "Inferring body pose without tracking body parts", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, volume II, pages 721–727, 2000.

[70] A. Shashua, "Trilinearity in visual recognition by alignment", In *Proc. 3rd Europ. Conf. Comput. Vision, J.-O. Eklundh (Ed.), LNCS-Series Vol. 800-801, Springer-Verlag*, volume I, pages 479–484, 1994.

[71] A. Shashua, "Algebraic functions for recognition", *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(8):779–789, Aug 1995.

[72] A. Shashua and M. Werman, "Trilinearity of three perspective views and its associated tensor", In *Proc. 5th Int. Conf. Computer Vision*, pages 920–925, 1995.

[73] A. Shashua and L. Wolf, "On the structure and properties of the quadrifocal tensor", In *Proc. 6th Europ. Conf. Comput. Vision, LNCS-Series Vol. 1842-1843, Springer-Verlag*, volume I, pages 710–724, 2000.

[74] J. Shi and J. Malik, "Normalized cuts and image segmentation", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 731–737, 1997.

[75] D. Sinclair, "Motion segmentation and local structure", In *Proc. 4th Int. Conf. Computer Vision*, pages 366–373, 1993.

[76] S. Soatto and R. Brockett, "Optimal structure from motion: local ambiguities and global estimates", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 282–288, 1998.

[77] S. Soatto, R. Frezza, and P. Perona, "Recursive motion estimation on the essential manifold", In *Proc. 3rd Europ. Conf. Comput. Vision, J.-O. Eklundh (Ed.), LNCS-Series Vol. 800-801, Springer-Verlag*, pages 61–72, Stockholm, May 1994.

[78] S. Soatto, R. Frezza, and P. Perona, "Motion estimation via dynamic vision", *IEEE Trans. on Automatic Control*, 41(3):393–413, 1996.

[79] S. Soatto and P. Perona, "Three demensional transparent structure segmentation and multiple 3d motion estimation from monocular perspective image sequences", *IEEE Workshop on Motion of Nonrigid and Articulated Objects,Austin*, pages 228–235, November 1994.

[80] Y. Song, X. Feng, and P. Perona, "Towards the detection of human motion", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 810–817, 2000.

[81] Y. Song, L. Goncalves, E. Bernardo, and P. Perona, "Monocular perception of biological motion - detection and labeling", In *Proc. 7th Int. Conf. Computer Vision*, pages 805–812, 1999.

[82] Y. Song, L. Goncalves, and P. Perona, "Monocular perception of biological motion - clutter and partial occlusion", In *Proc. 6th Europ. Conf. Comput. Vision, LNCS-Series Vol. 1842-1843, Springer-Verlag*, volume II, pages 719–733, 2000.

[83] Y. Song, L. Goncalves, and P. Perona, "Learning probabilistic structure for human motion detection", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, volume 2, pages 771–777, 2001.

[84] M. Spetsakis and Y. Aloimonos, "Structure from motion using line correspondences", *Int. J. of Computer Vision*, 4:171–183, 1990.

[85] M. Spetsakis and Y. Aloimonos, "A multiframe approach to visual motion perception", *Int. J. of Computer Vision*, 6:245–255, 1991.

[86] T. Starner and A. Pentland, "Visual recognition of american sign language using hidden markov models", *International Workshop on Automatic Face and Gesture Recognition*, pages 189–194, 1995.

[87] R. Szeliski and S. Kang, "Recovering 3d shape and motion from images using nonlinear least squares", *"J. Visual Communication and Image Representation*, 5(1), 1994.

[88] C. Tomasi and T. Kanade, "Shape and motion from image streams: a factorization method – 3. detection and tracking of point features", CMU-CS 91-132, School of CS – CMU, April 1991.

[89] P. Torr and A. Zisserman, "Robust parameterization and computation of the trifocal tensor", *Image and Vision Computing*, pages 591–605, 1997.

[90] W. Triggs, "The geometry of projective reconstruction i: matching constraints and the joint image", In *Proc. 5th Int. Conf. Computer Vision*, pages 338–343, 1995.

[91] R. Tsai, "An efficient and accurate camera calibration technique for 3d machine vision", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 364–374, 1986.

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

[93] R.Y. Tsai and T.S. Huang, *Image understanding, S. Ullman and W. Richards (eds.)*, chapter Uniqueness and estimation of three-dimensional motion parameters of rigid objects, Ablex Publishing, Norwood NJ, 1984.

[94] S. Ullmann, "The interpretation of structure from motion", *Proc. R. Soc. London 203*, 1979.

[95] T. Vieville and Q. Luong, "Motion of points and lines in the uncalibrated case", *"Technical Report 2054, INRIA"*, 1993.

[96] J.Y.A. Wang and E.H. Adelson, "Representing moving images with layers", *IEEE Transactions on Image Processing Special Issue: Image Sequence Compression*, pages 625–638, 1994.

[97] Y. Weiss, "Smoothness in layers: Motion segmentation using nonparametric mixture estimation", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 520–526, 1997.

[98] Y. Weiss and E.H. Adelson, "A unified mixture framework for motion segmentation: incorporating spatial coherence and estimating the number of models",

In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 321–326, 1996.

[99] J. Weng, N. Ahuja, and T. Huang, "Closed-form solution + maximum likelihood: a robust approach to motion and structure estimation", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 381–386, 1988.

[100] J. Weng, N. Ahuja, and T. Huang, "Optimal motion and structure estimation", *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(9):864–884, 1993.

[101] J. Weng, T. Huang, and N. Ahuja, "Motion and structure from two perspective views: algorithms, error analysis and error estimation", *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(5):451–476, 1989.

[102] J. Weng, T.S. Huang, and N. Ahuja, "Motion and structure from line correspondences: closed-form solution, uniqueness and optimization", *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(3):318–336, 1992.

[103] Y. Yacoob and M. Black, "Parameterized modeling and recognition of activities", *Computer Vision and Image Understanding*, 73(2):232–247, 1999.

[104] J. Yamoto, J. Ohya, and K. Ishii, "Recognizing human action in time-sequential images using hidden markov model", In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 379–385, 1992.

[105] Y. Yasumoto and G. Medioni, "Robust estimation of three-dimensional motion parameters from sequence of image frames using regularization", *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(4):464–471, 1986.

[106] Z. Zhang and O. Faugeras, "Three dimensional motion computation and object segmentation in a long sequence of stereo frames", *Int. J. of Computer Vision*, 7(3):211–241, 1992.

[107] L. Zhao and C. Thorpe, "Recursive context reasoning for human detection and parts identification", *IEEE Workshop on Human Modeling, Analysis and Synthesis*, pages 18–25, 2000.

[108] X. Zhuang, T. Huang, and R. Haralick, "Two-view motion analysis: a unified algorithm", *J. Opt. Soc. Am. A*, 3(9):1492–1500, 1986.