ABSTRACT

Title of dissertation: OBJECT TRACKING AND MENSURATION IN SURVEILLANCE VIDEOS

Jie Shao
Doctor of Philosophy, 2009

Dissertation directed by: Professor Rama Chellappa
Department of Electrical Engineering

This thesis focuses on tracking and mensuration in surveillance videos. The first part of the thesis discusses several object tracking approaches based on the different properties of tracking targets. For airborne videos, where the targets are usually small and with low resolutions, an approach of building motion models for foreground/background proposed in which the foreground target is simplified as a rigid object. For relatively high resolution targets, the non-rigid models are applied. An active contour-based algorithm has been introduced. The algorithm is based on decomposing the tracking into three parts: estimate the affine transform parameters between successive frames using particle filters; detect the contour deformation using a probabilistic deformation map, and regulate the deformation by projecting the updated model onto a trained shape subspace. The active appearance Markov chain (AAMC). It integrates a statistical model of shape, appearance and motion. In the AAMC model, a Markov chain represents the switching of motion phases (poses), and several pairwise active appearance model (P-AAM) components characterize the shape, appearance and motion information for different motion phases. The second part of the thesis covers video mensuration, in which we have proposed a height-measuring algorithm with less human supervision, more flexibility and improved robustness. From videos acquired by an uncalibrated stationary camera, we first recover the vanishing line and the vertical point of the scene. We then apply a single view mensuration algorithm to each of the frames to obtain height measurements. Finally, using the LMedS as the cost function and the Robbins-Monro stochastic approximation (RMSA) technique to obtain the optimal estimate.

OBJECT TRACKING AND MENSURATION IN SURVEILLANCE VIDEOS

by

Jie Shao

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2009

Dedication

To Jiang, Audrey and my parents.

Acknowledgements

The dissertation not only holds what I have done during my study in the University of Maryland, it also reflects the relationships with many inspiring and generous people I have met during this period. The list is long, but I wish to express my sincere gratitude to them.

To my advisor, Professor Rama Chellappa, for his valuable guidance and encouragement on research, his sustained financial support, and his diligent and honest attitude toward life.

To my committee members, Professor David W. Jacobs, Min Wu, Peter Petrov, and Dr. Kevin S. Zhou for their encouraging words, thoughtful comments, and time and attention. I also thank Professor Larry S. Davis, who was on my proposal examination committee.

I want to specially thank Dr. Kevin S. Zhou for all the instructive suggestions, the many paper collaborations and for the summer intern opportunity in Siemens Corporate Research (SCR) in 2005. I enjoyed every fruitful discussion we had. I also express my appreciation to Dr. Fatih Porikli at Mitsubishi Electric Research Lab (MERL) for hosting me a summer intern in 2004. I enjoyed working with him.

To all my brilliant lab colleagues at the Center for Automation Research (CfAR) for sharing their enthusiasm for and comments on my work: Haiying Liu, Feng Guo, Jian Li, Zhanfeng Yue, Yang Ran, Namrata Vaswani, Naresh Cuntoor, Aravind Sundaresan, Hao Wu, Mahesh Ramachandran, and Ashwin Sankaranarayanan.

I take this special occasion to thank my parents and parents-in-law for their selfless support and never ending love. Last but not the least, I thank my husband, for his great encouragement and support, and his lifelong love.

Table of Contents

List of Tables viii

List of Figures x

1 Introduction 1
- 1.1 Video Tracking and Mensuration . . . 1
  - 1.1.1 Video Surveillance . . . 3
  - 1.1.2 Research Motivation . . . 4

2 Rigid Object Tracking: Using Foreground and Background Modelling 8
- 2.1 Introduction . . . 8
- 2.2 Related Literature . . . 12
- 2.3 Models . . . 14
  - 2.3.1 State Vector of The Model . . . 15
  - 2.3.2 Background Motion Model . . . 16
  - 2.3.3 Foreground Motion Model . . . 17
- 2.4 Algorithms . . . 18
  - 2.4.1 The Particle Filter . . . 18
  - 2.4.2 Iterative Optimization for Motion Estimation . . . 19
  - 2.4.3 Algorithm Implementation . . . 22
- 2.5 Measurements . . . 23
  - 2.5.1 Foreground Observation Likelihood . . . 24
    - 2.5.1.1 Motion-Cue Based Foreground Measurement Model 25
    - 2.5.1.2 Appearance-Cue Based Foreground Measurement Model 28
    - 2.5.1.3 Adaptive Fusion Weights . . . 29
  - 2.5.2 Background Observation Model . . . 32
- 2.6 Experiments . . . 32
  - 2.6.1 Experiment results . . . 32
  - 2.6.2 Comparison and Discussions . . . 36
- 2.7 Conclusion . . . 39

3 Non-Rigid Object Tracking: Motion and Deformation 40
- 3.1 Introduction . . . 40
  - 3.1.1 Related Work . . . 40
- 3.2 Preliminaries . . . 43
  - 3.2.1 B-spline Parametric Curves . . . 43
  - 3.2.2 Particle Filter . . . 44
- 3.3 Motion Estimation . . . 46
  - 3.3.1 Dynamic Motion Model . . . 46
  - 3.3.2 Estimate State Vector by Particle Filter . . . 46
    - 3.3.2.1 Prediction Step . . . 46
    - 3.3.2.2 Updating Step . . . 48
- 3.4 Deformation Estimation . . . 49

3.4.1 Set Normal Lines Adaptive 50
3.4.1.1 Lengths of normal lines 51
3.4.1.2 Centers of normal lines 51
3.4.2 Multi-cue Deformation Probability Map 53
3.4.2.1 Multi-cue Fusion 53
3.4.2.2 Gradient Magnitude Cues 54
3.4.2.3 Gradient Orientation Cues 57
3.4.2.4 Shape Template Cues 57
3.4.2.5 Foreground Cues 58
3.5 Regulation on Shape Deformation 59
3.5.1 Generic Shape Model 60
3.5.2 Adaptive Shape Model 61
3.5.3 Subspace Projection 62
3.5.4 Alignment 63
3.6 Implementation and Experiments 64
3.6.1 Algorithmic Implementation 64
3.6.2 Experimental Results 65
3.6.2.1 Experiment on Stationary Camera Data 67
3.6.2.2 Experiments on Moving Camera Data 67
3.6.2.3 Experiments on Occlusion Data 71
3.6.2.4 Experiments on medical sequence 71
3.6.2.5 Performance Evaluation 73
3.7 Discussion 73

4 Regulation in Object Tracking: Active Appearance Markov Chain (AAMC) Model 77
4.1 Introduction 77
4.1.1 Previous Work 78
4.2 Active Appearance Markov Chain Model 80
4.2.1 Markov chain: model switching 82
4.2.2 P-AAM components 82
4.3 Fusion Process Using AAMC Model 84
4.3.1 Component $\mathbf{d}^2_{t|\mathbf{y}_t,\mathbf{x}_{t-1}}$ 86
4.3.2 Component $\mathbf{d}^2_{t|\mathbf{z}_t,\mathbf{x}_{t-1},\mathbf{z}_{t-1}}$ 87
4.3.3 The best estimate of $\mathbf{x}_t$ with fixed $\mathbf{z}_t$ 89
4.4 Evaluation Methods 89
4.5 Experiments 91
4.5.1 Echocardiography tracking 91
4.5.2 Pedestrian Tracking 95
4.5.3 Face tracking 97
4.6 Conclusions 100

5 Video Mensuration Using Stationary Cameras 102
5.1 Introduction . . . 102
5.1.1 Related work . . . 104
5.2 Single view measurement from minimal calibration . . . 106
5.3 Self minimal calibration from moving objects . . . 108
5.3.1 The geometric properties of moving objects . . . 108
5.3.2 Recovering vanishing line and vertical vanishing point . . . 110
5.3.2.1 Trajectory refinement . . . 111
5.3.2.2 Simultaneous trajectory grouping and vanishing point estimation . . . 112
5.3.2.3 Vertical point estimation . . . 117
5.4 Multi-frame measurement fusion using stochastic approximation . . . 117
5.4.1 Single frame measurement . . . 118
5.4.2 Multi-frame fusion using least median estimator and Robbins-Monro stochastic approximation . . . 119
5.5 Uncertainty analysis of measurement . . . 122
5.5.1 Analytical estimation of uncertainty on height measurement . 122
5.5.2 Monte Carlo estimation of uncertainty on height measurement 123
5.5.3 Error estimation result . . . 125
5.6 Experiments and Discussion . . . 127
5.6.1 Vanishing line estimation . . . 127
5.6.2 Single-view height measurement . . . 130
5.6.3 Two-view object height measurement . . . 135
5.6.4 Discussion on GPCA vs. our algorithm . . . 136
5.7 Summary . . . 139

6 Conclusions 141
6.1 Summary of the Dissertation . . . 141
6.2 Future Direction of Research . . . 143

A Appendix 145
A.1 Robbins-Monro Optimization . . . 145
A.2 Proof of Theorem 1 . . . 146

List of Tables

2.1 Characteristic summary for experimental sequences 36

2.2 Number of successfully tracked frames using different tracking algorithms. BF-PF is our proposed algorithm. 37

3.1 Algorithm 1: set center of the normal line adaptive 52

3.2 Algorithm 2: Construction of Prior Shape Model 61

3.3 Algorithm 3: Active Contour Tracking 66

3.4 Comparison of tracking results in heavy-cluttered background sequences and non-rigid object sequences. MSSD is the Mean Sum of Squared Distance; VAR is the variance of the distances; ACT is the traditional active contour tracking algorithm. 75

4.1 Comparison of tracking performances based on the segmental Hausdorff distance and the surprisal vector distance: (1) Echocardiograph tracking results; (2) Face tracking results. "OF" is the original result from optical flow estimation. 95

4.2 Comparison of tracking results on heavily-cluttered background sequences and non-rigid object sequences in terms of segmental Hausdorff distance. 97

5.1 Monte Carlo Simulation 124

5.2 Error estimation results with the estimated height $h = 91inch$ using the analytical approach and Monte Carlo simulation. Relative errors are also computed. 125

5.3 Uncertainty analysis under different calibration errors. σ_n is the image noise, σ_c is the calibration noise, $\sigma_{n,c}$ is the entire uncertainty concerning both the image noise and calibration error. 126

5.4 Height measurement results under different scenarios. (M: moving; S: still; T: target; R: reference.) 135

5.5 Subspace number estimation using two algorithms. The GPCA algorithm has different estimates on subspace numbers with different threshold settings. "N/A" represents the case with no valid solution. 139

5.6 Comparison between single view metrology, two-view metrology and our method. *: Single view metrology can operate on video sequences, but is still based on individual frames. 140

List of Figures

2.1 Graphic model for iteratively determining θ^B and θ^F. 20

2.2 Examples of humans in low contrast videos. 26

2.3 Adaptive weights plot for different information cues. 31

2.4 Tracking of pedestrians running on the ground. 33

2.5 Pedestrian tracking in a ground based video sequence. 34

2.6 Multiple-target tracking on an airport runway. 34

2.7 Tracking of different persons running in the field. 35

2.8 Tracking results from different tracking algorithms. 38

3.1 Illustration of the proposed tracking system. 43

3.2 An illustration of estimating measurement model. 49

3.3 An example for making a normal line scanning adaptive. 53

3.4 An illustration of fusion from different visual sources. 55

3.5 Performance comparisons on a cluttered scene. 56

3.6 Illustrations of adaptive probability shape templates for pedestrians. . 58

3.7 Comparisons of tracking results between with and without regulation. 63

3.8 An example frame from an MRI sequence. 64

3.9 An example of starting tracking using a very rough initial contour. . . 65

3.10 Pedestrian tracking performance with a stationary camera. 68

3.11 The results of tracking a vehicle from the rear view. 68

3.12 A sequence with both background and object moving. 69

3.13 A sequence with both background and object moving. 70

3.14 An airborne sequence with a white truck moving on the ground. . . . 71

3.15 An example of an occluded contour being recovered by shape subspace projection. 72

3.16 Magnetic resonance imaging scans of human knees. 74

4.1 Illustrations of tracking system and AAMC model. 79

4.2 Graphical models of our algorithms and two distances being used in our system. 90

4.3 Examples of shape and appearance subspaces of P-AAM. 92

4.4 The snapshots of the tracking results of A4C and A2C sequences. . . 93

4.5 Two sequences in which both background and object move. 98

4.6 Graphic illustrations for ASMC and AAMC models. 99

4.7 Comparisons of difference tracking results on one example frame. . . . 100

4.8 The snapshots of tracking results from the testing face sequence. . . . 101

5.1 Illustration of the basic geometry and one example. 107

5.2 An example of refining trajectories by line fitting 112

5.3 Using the Fourier Descriptor to find outliers. 115

5.4 Using the EM algorithm to estimate the vanishing point. 116

5.5 The procedure for measuring objects heights from one frame. 120

5.6 Examples of vanishing lines being automatically estimated in video sequences. 128

5.7 Examples of multi-frame measurements. 129

5.8 An example of a moving object and a still reference object. 132

5.9 An example of a still target and a moving reference. 133

5.10 An example of a moving target and a moving reference. 134

5.11 The results of two-vide object height measurement. 137

Chapter 1

Introduction

1.1 Video Tracking and Mensuration

Video-based object tracking and recognition is an active research topic in computer vision. It concerns the location of a particular object at a given time and the recognition of the object. Conventional methods have considered the problem as a frame-to-frame operation [1] [2]. Various features, such as contours, corners [3], shape, and color [4] have been used. Features detected in one image are matched to corresponding features in the previous image. The shortcoming of feature-based methods results from the unreliability of feature detection due to noise and occlusion. An alternative approach was introduced in [5], which watches for activity in an environment and detects regions of change that might represent objects added to or removed from the scene after the activity has ceased locally. This approach views the problem as a discrete process, integrating all the disappearances and reappearances into one object location/history model. Probabilistic analysis has gained significant attention after the Kalman filter approach was proposed in [6], which works by predicting the location of the features being tracked in the next image, then using the error between the predicted and measured location to update the predictive model. The seminal work of Isard and Blake [7] extended the Kalman filter to more general problems. Their contribution introduced a time series state space model parameterized by a tracking motion vector, which may represent transformation parameters. The CONDENSATION algorithm, originally proposed in [8] in the signal processing literature, also known as the particle filter (PF), provides a numerical approximation to the posterior distribution of the motion vector at time t given the observations up to t. The algorithm has been used to solve many vision tasks as reported in [9], [10], [11].

The basic idea of this algorithm is: Given the state transition model characterized by the state transition probability $p(\theta_t|\theta_{t-1})$ and the observation model characterized by the likelihood function $p(Y_t|\theta_t)$, the PF is used to approximate the posterior distribution $p(\theta_t|Y_{1:t})$ by a set of weighted particles $S_t = \{\theta_t^{(j)}, \omega_t^{(j)}\}_{j=1}^{J}$ with $\Sigma_{j=1}^{J}\omega_t^{(j)} = 1$.

A new framework was proposed in [10], which can not only simultaneously perform tracking and recognition, but also improve the recognition rate compared to the conventional methods without any reduction in tracking accuracy. To make this algorithm more reliable, an adaptive approach uses the following strategies: (1) employing an adaptive appearance model and an adaptive velocity motion model [12] for representing inter-frame appearance changes. (2) constructing intra- and extra-personal spaces to model the appearance changes between video frames and gallery images. (3) exploiting the fact that the gallery images appear as frontal views. By embedding these abilities in a PF, enhancements in both stability and accuracy are achieved when confronted by pose and illumination variations.

1.1.1 Video Surveillance

Our tracking system applies to detecting, tracking and recognizing objects in surveillance video sequences. Therefore, a short review of relevant work from video surveillance literature is presented.

Automated video surveillance addresses the observation of people and vehicles in a busy environment, hopefully leading to an understanding of their actions and interactions. The technical issues involve moving target detection, tracking, classification, motion analysis, and activity understanding. In addition to obvious security applications, video surveillance techniques have been employed to measure traffic flow, detect accidents on highways, monitor pedestrian congestion in public spaces, and compile consumer demographics in shopping malls and amusement parks etc [13]. Typically, the basic techniques required for video surveillance are extraction of moving objects, continuous tracking of objects over time, and classification of objects of interest. MIT's PFinder system tracks a human's head, hands, torso, and feet. [4]. In [14], motion, color and stereo triangulation are used to detect humans in front of a smart kiosk. In [15], the VSAM project addressed the task of tracking multiple targets (people and vehicles) in cluttered scenes using multiple cameras. Simple classification into human, vehicle or clutter categories is performed. Other systems that can detect simple activities involving interactions with tracked objects, such as the one reported by Morris and Hogg [16], statistically model the interactions between people and cars. The W^4 system [17] detects and tracks multiple people, carried objects and exchanges. Irani and Anandan [18] describe a mosaic-based approach for representing a tracked object in a scene, while another work from the MIT AI lab uses an adaptive background model to track and classify moving objects [5]. Another task in a surveillance system is classifying human movement patterns into finer categories. Davis and Bobick use a view-based "temporal template" approach for classification [19]. [20] uses coupled Hidden Markov Models (HMM) to represent gestures involving both arms. The same framework is used in [21] to model human interactions.

1.1.2 Research Motivation

Generally, surveillance sequences have some special characteristics. A large portion of these videos suffer from poor resolutions. Most of the objects being tracked change directions. The objects are usually small, with low-contrast from the surroundings. Sometimes, it is even difficult to locate the target using human eyes. Furthermore, cameras capturing videos are not always fixed. Therefore, failures often occur when applying the traditional state-of-the-art methods because they have some limitations. This motivates us to find alternative approaches to overcome these limitations.

One of the limitations is introduced by the assumption of an affine transform model in algorithms. In our applications, the affine transform is not sufficient to describe the appearance changes in a 3D scene. Therefore, an extension of the algorithm suited to more general situations is initiated by representing the object motion using a 3D transformation. By doing this, the model assumption is generalized from an affine transformation to a perspective transformation. The core of the algorithm is a trilinear tensor operator that links point correspondences across three images to represent the 3D transformation among the different views [22]. We first employ the affine model to obtain sets of corresponding points, which then are used to estimate a trilinear tensor. By synthesizing a novel view from the derived tensor, we refine our motion prediction. Meanwhile, the desired virtual view is also used to update the appearance model. By repeatedly applying the tensor operations in a sequence of reference images in an estimate-refine cycle, we extend the previous tracking and recognition algorithm to a more general model. In the most significant aspect of our approach, we obtain our 3D transformation model without invoking the burdensome processes of explicitly recovering a depth map or camera parameters. This makes the entire system more robust and efficient.

Poor video quality and the dimness of the object cause other difficulties. Experiments show that the algorithms work well on large objects, but less so for small objects, especially in surveillance applications, in which a target may have only tens of pixels. Typically, tracking is done so that pixels belonging to the moving objects are different in intensity, chromaticity values and motion when compared to the background pixels. Therefore, background substraction is a much studied technique for object tracking. Typical background subtraction algorithms assume a static background and some prior knowledge of the background. These requirements limit the applications of the background-subtraction algorithm for object tracking. Such considerations lead us toward a new statistical method for tracking objects in a low-resolution surveillance video, which uses a time series state space model parameterized by a tracking motion vector, denoted by θ. In order to overcome the challenges from the differences between the intensities of background and foreground object not being significant or an insufficient number of target pixels, we integrate intensity and motion information. Motion information helps to discriminate the moving objects from the relatively still background. In the proposed algorithm, the difference images are used as measurements of observation to estimate the motion state of the model.

Typical classification systems for video surveillance perform either object detection without prior segmentation or object classification after detection. Systems of the latter type are part of larger systems which first perform tracking, and we focus on such systems in our research. After tracking the moving objects, features are extracted for classifications. These features include raw appearance, color, texture, shape, and motion. One critical step in all classification methods includes selecting suitable features from the whole feature set. In our study, we have explored several classification methods from different features. Our goal is to build an object classification system to distinguish vehicles and pedestrians based on tracked data and extracted features.

Popular video camera surveillance systems require us to gain an understanding of the event or identify a particular target. One factor for identification is the size of the target, for instance, a human's height or a vehicle's length. Several techniques have been developed for obtaining size information from video. Our goal is to measure an object in a video sequence, more particularly, in an airborne surveillance video sequence. We can extract several types of measurements. We will concentrate on measurements of: 1) object height; 2) relative size ratio between objects; and 3) the camera's position. Generally, the difficulties of measuring from video sequences result from: (1) low image quality, (2) little to no information concerning the camera parameters; and (3) the absence of stereo-view. Therefore, classic stereo algorithms do not really suit the task. Our research aims to develop a measurement system based on monocular videos and uncalibrated video cameras, in the sense that camera constant (focal length), principal point location and the image affinity parameters are irrelevant (radial lens distortion is not taken into account). The system employs a two phase approach. In the first phase, object tracking is performed using the active contour approach, which has the advantage that it can provide tighter object boundaries. This phase is used to segment the object from its surroundings. The second phase implements mensuration algorithms using a single view metrology approach based on 1) a reference height; 2) some vanishing points estimated from object motion to determine the horizon line of a reference plane; and 3) a vertical vanishing point.

Chapter 2

Rigid Object Tracking: Using Foreground and Background Modelling

2.1 Introduction

VISUAL TRACKING is an essential component of many applications such as intelligent robotics, ground and airborne video surveillance, diagnosis and monitoring of movements disorders, etc. While success has been attained when the video quality and content are good for tracking purpose, challenges exist when dealing with surveillance videos, in which targets are of low contrast with respect to the background or extremely small because the camera is too far away. In addition, motion blur, camera motion, non-stationary background, poor illumination, and non-uniform object motion compound the problem. Developing a visual tracking algorithm that achieves both robustness and accuracy under operational conditions remains an open problem.

Various approaches have been proposed to solve surveillance tracking problems. They can be roughly classified into four broad categories. Algorithms in each category perform well in solving specific tracking problems.

- **Foreground Tracking** [23, 4] These methods directly track the objects using either global transform estimation or feature correspondences. Such approaches work well when the object feature points are easy to detect and track from frame to frame. But when the object itself is dim, finding point correspondences across frames becomes difficult.

- **Background Subtraction** [17, 24] Algorithms in this category detect foreground objects using the difference between the current frame and an image of the scene's static background. When the camera is stationary, this approach usually provides good tracking results. When the camera is moving, the background estimation itself is difficult. Therefore, we seldom see this approach applied to moving-camera scenarios.

- **Motion-based Tracking** Approaches in this category [25, 26, 12] use motion information as a dominant cue. It groups consistent visual motions over time. This approach certainly performs well with a moving target, which provides adequate motion information for the system to detect or locate the target. However, when the target is in very slow motion or still, the approach fails because sufficient motion information is not available.

- **Appearance-based Tracking** These approaches [27, 28] construct 2D image-based templates to model changes explicitly in target regions. For example, the mean-shift algorithm [29] uses viewpoint-insensitive appearance models to track an object without any prior knowledge of camera motion or scene structure. For targets with high resolution or high contrast, appearance-based approaches can achieve precise results. Unfortunately, when the target only occupies 20-40 pixels in the image, the appearance-based approaches usually fail.

Therefore, the behaviors of tracking algorithms differ when the operate under different settings. Generally, most of these algorithms perform well for large objects, but less so for small objects, especially in surveillance applications with a small amount of pixels per target. In this chapter, we will concentrate on the moving-camera small-object tracking problem and improve two metrics that characterize the system performance: **robustness** and **accuracy**. The general questions that we address are: *How to represent the small object? How to incorporate both object motion and camera motion in the tracking system? How to define a robust observation model?* For example, both intensity-based and edge-based trackers are often distracted by other objects or low-contrast background clutter, which means that no single cue can perform efficiently in all kinds of situations. In the case of a moving camera, two different motions are caused by different sources, i.e., camera motion and object motion. We represent background and foreground motion parameters using two different sets of parameters.

Based on the above considerations, we present a robust method for tracking objects in surveillance videos characterized by a dynamic motion model. It consists of both background and foreground motions in the system state vector and the measurement model that fuses motion and appearance cues. The first feature aims to improve the tracking accuracy of the system, while the second one enhances the system robustness.

The *dynamic model* in the system is a time series state space model parameterized by a tracking motion vector, denoted as θ. To describe the two types of motions, the motion vector θ for the entire system consists of two subsets of parameters for background and foreground motions, respectively. The advantage of such a decomposition is that we can compensate for the camera motion by first stabilizing the image using the subset of the background motion parameters.

$$\theta = (\theta^F, \theta^B), \tag{2.1}$$

where θ^B represents the background parameter set, θ^F represents the foreground parameter set.

The *measurement model* fuses motion and appearance cues so that the appearance cue can distinguish objects from the background due to different intensity values, while the motion cue helps to discriminate moving objects from relatively still backgrounds. When used separately, neither is sufficiently robust to deal with all video sequences. So in the proposed algorithm, we integrate the two cues in the measurement model.

With the dynamic model and measurement model defined, we use a particle filter as our basic tracking algorithm. The algorithm provides a numerical approximation to the posterior distribution of the motion vector at time t, given the observation up to t, i.e., $p(\theta_t|Y_{1:t})$, where $Y_{1:t}$ is the observations of video frames up to time t. Since the system simultaneously estimates both background and foreground motion parameters in a single dynamic model, the efficiency of the algorithm is increased for: 1) segmenting background and foreground objects; 2) obtaining both motion and intensity information from background motion vector θ^B; and 3) tracking the foreground target based on the estimated foreground motion vector θ^F.

2.2 Related Literature

The foreground-based tracking method focuses on analyzing the foreground correspondence using specific object information over time, such as edges, or at a high level, object parts. Hogg [30] implemented a low level foreground tracking algorithm in the "Walker" system by extracting and matching object edges. The "Pfinder" system proposed by Wren *et al.* [4] is an example of high level approach, where the human body is represented by blobs and tracked from frame to frame. The correspondence analysis is often supported by prediction of object movements or regions-of-interest. An evolving model can accomplish prediction [31]. An alternative statistical approach involves learning probabilistic motion models prior to operation [32], using, say, a Kalman filter [33] or the CONDENSATION algorithm [23]. The latter extends the former and enables tracking in the presence of occlusion, non-Gaussian noise and cluttered backgrounds. It uses complex dynamics by sampling the posterior distribution estimated in the previous frame and propagates those samples to estimate the posterior for the current frame.

The background-based tracking approach relies on background estimation to detect and track the foreground objects. The objects are obtained by change detection using adaptive background subtraction, yielding a bounding box surrounding the target. Most of the methods in this category employ a single Gaussian [4] or a mixture of Gaussian background models [34]. The $W4$-system developed by Haritaoglu *et al.* [17] operates on monocular gray scale images and infrared images, detecting and tracking body parts. Other approaches include the three-stage (pixel/region/frame) Wallflower approach [35] and a two-stage color and gradient technique [24]. Further improvement is needed when handling dynamic backgrounds and non-trivial movement patterns.

A wide variety of tracking methods support the hypothesis that the performance of a tracker primarily depends on how distinguishable the object is from its surroundings. Several approaches can discriminate the object from the background. Collins and Liu [36] proposed an algorithm for on-line selection of discriminative features during tracking. The algorithm focuses on adaptively selecting the top-ranked discriminative appearance features in the entire feature space for tracking.

Other than the appearance cue, cues can also be derived from motion, color distribution, etc. Several real-time tracking algorithms reported in the literature fuse these cues. Viola *et al.* [37] proposed a pedestrian detection system that integrates intensity and motion information. The detector is trained (using *AdaBoost*) to take advantage of both motion and appearance information. Some approaches select the "optimal" cue for the entire sequence with other less reliable cues as supporting cues. A good example of cue selection is found in the layered hierarchy extraction algorithm proposed by Toyama and Hager [38]. ICONDENSATION, as an extension of CONDENSATION with importance sampling, proposed by Isard and Blake [9] also employed a multi-cue scheme by complementing the original intensity gradient cue with a supportive color cue. Sidenbladh *et al.* [39] incorporated motion measurements in the particle filter framework. Odobez *et al.* [40] improved the particle filter by using transition prior as the proposal distribution. Some approaches assume that each cue has the same reliability in all frames. For example, Birchfield [41] used intensity and color distribution of the target with equal confidences for robust head tracking. Democratic integration, proposed by Triesch *et al.* [42], introduced an adaptive multi-cue integration so the contribution of each cue varies according to its reliability in each frame. Such a strategy improves the robustness of the system in different visual situations. Vermaak *et al.* extended the idea for adaptation of multiple cues in the particle filter framework [43].

We present a comprehensive method that builds upon the reviewed methods. Utilizing the particle filter as a basic framework, the proposed approach combines both background and foreground motion parameters in a single state vector to take advantages of both background and foreground tracking. Two statistical features, motion and appearance information, are used to construct the observation model in an adaptive manner.

2.3 Models

Given the observations $Y_{1:t}$, with the first-order Markovian assumption $p(\theta_t|\theta_{1:t-1}) = p(\theta_t|\theta_{t-1})$, the problem of tracking can be formulated as a Bayesian filtering problem for estimating θ_t that maximizes

$$p(\theta_t|Y_{1:t}) \propto \mathcal{L}(Y_t|\theta_t) \int p(\theta_t|\theta_{t-1}) p(\theta_{t-1}|Y_{1:t-1})\, d\theta_{t-1} \tag{2.2}$$

The *motion transition probability* $p(\theta_t|\theta_{t-1})$ predicts θ_t given the previous state θ_{t-1}. The likelihood function $\mathcal{L}(Y_t|\theta_t)$ computed using the *measurement model*, reflects the probability of observing the measurement given the state θ_t. This section presents the dynamic motion model and motion transition. The measurement model is dicussed in Section 2.5.

2.3.1 State Vector of The Model

How to find what has changed between two successive frames? The most direct way is to check the absolute differences, based on the assumptions of a stationary camera and a noise-free image. Such assumptions are rarely valid in practice. We stabilize the image by compensating for the inter-frame differences caused by the camera motion, including the stabilization parameters as the background motion vector in the system state vector. The position of the target can be derived based on the foreground motion parameters. We use the 2D affine transform to represent the foreground motion. Therefore, we define the state vector of our tracking system as:

$$\theta = (\theta^F, \theta^B); \quad \theta^F = (\alpha^F, dx^F, dy^F, s^F); \quad \theta^B = (\alpha^B, dx^B, dy^B, s^B), \tag{2.3}$$

where the image rotation angle α^B, displacement (dx^B, dy^B) and scale s^B are four parameters in θ^B describing the background change caused by the moving camera. The pixel movement belonging to the background between two successive time instants t and $t-1$ is related by

$$\mathbf{x}_t = s_t^B \begin{bmatrix} \cos(\alpha_t^B) & \sin(\alpha_t^B) \\ -\sin(\alpha_t^B) & \cos(\alpha_t^B) \end{bmatrix} \mathbf{x}_{t-1} + d\mathbf{x}_t^B = \mathcal{T}\{\mathbf{x}_{t-1}; \theta_t^B\}, \tag{2.4}$$

where $\mathbf{x}_t = (x_t, y_t)^T$ is the pixel coordinate at time t, and $d\mathbf{x}_t^B = (dx_t^B, dy_t^B)^T$. The interpretation of the components in θ^F bears similarities to that of θ^B. Specifically, we concentrate on the problem of surveillance tracking, where the object usually occupies a small number of pixels and find that the translation of the object (dx^F, dy^F) is sufficient to describe the movements of the object. So, we simplify the foreground state vector as

$$\theta^F = (dx^F, dy^F). \tag{2.5}$$

2.3.2 Background Motion Model

The state transition is approximated by using a first-order Markov chain and a Gaussian noise model. One component of noise, denoted by μ_t, is modeled using the zero-mean Gaussian distribution accounting for sensor noise, digitization noise, etc; the other, denoted by ν_t, is modeled as a non-zero-mean Gaussian distribution caused by the camera motion. The background motion equation is written as:

$$\theta_t^B = \hat{\theta}_{t-1}^B + \nu_t^B + \mu_t^B \tag{2.6}$$

The estimation of ν_t is predicted from the previous particles and the assumption of brightness invariance [27], which means that there exists a θ_t^B such that $Y_t^B \simeq \mathcal{T}\{Y_{t-1}^B; \theta_t^B\}$, where Y_t^B is the image of the background with foreground patches cropped according to the estimated foreground parameters. Approximating $\mathcal{T}\{Y_{t-1}^B; \theta_t^B\}$ via a first-order Taylor series expansion around $\hat{\theta}_{t-1}^B$ yields:

$$\mathcal{T}\{Y_{t-1}^B; \theta_t^B\} \simeq \mathcal{T}\{Y_{t-1}^B; \hat{\theta}_{t-1}^B\} + \mathcal{C}_t(\theta_t^B - \hat{\theta}_{t-1}^B) = \mathcal{T}\{Y_{t-1}^B; \hat{\theta}_{t-1}^B\} + \mathcal{C}_t\nu_t \tag{2.7}$$

where $\mathcal{C}_t$ is the Jacobian matrix. After substituting Y_t^B into (2.7), we obtain:

$$\begin{aligned} Y_t^B &\simeq \mathcal{T}\{Y_{t-1}^B; \hat{\theta}_{t-1}^B\} + \mathcal{C}_t\nu_t \\ \nu_t &\simeq -\mathcal{B}_t(\mathcal{T}\{Y_{t-1}^B; \hat{\theta}_{t-1}^B\} - Y_t^B) \end{aligned} \tag{2.8}$$

where $\mathcal{B}_t$ is the pseudo-inverse of the $\mathcal{C}_t$. The estimation of $\mathcal{B}_t$ is performed based on the available data Θ^B_{t-1}, denoting the particle sample set of $\{\theta^{B(j)}_{t-1}\}^J_{j=1}$, and $\mathcal{Y}^{B\delta}_{t-1}$, denoting the difference between the set of all background samples and the image of the background at previous time, i.e., $\{Y^{B(j)}_{t-1} - Y^B_{t-1}\}^J_{j=1}$.

Using the differences in motion vectors and the observation matrix as inputs, we obtain a least square (LS) solution to $\mathcal{B}_t$ as:

$$\begin{aligned}\Theta^{B\delta}_{t-1} &= [\theta^{B(1)}_{t-1} - \hat{\theta}^B_{t-1}, \ldots, \theta^{B(J)}_{t-1} - \hat{\theta}^B_{t-1}] \\ \mathcal{B}_t &= (\Theta^{B\delta}_{t-1}\mathcal{Y}^{B\delta}_{t-1})(\mathcal{Y}^{B\delta}_{t-1}(\mathcal{Y}^{B\delta}_{t-1})^T)^{-1} \qquad (2.9)\end{aligned}$$

In practice, the matrix $\mathcal{Y}^{B\delta}_{t-1}(\mathcal{Y}^{B\delta}_{t-1})^T$ is often rank-deficient due to the large number of the particles being used. We perform a singular value decomposition (SVD) of $\mathcal{Y}^{B\delta}_{t-1}$, i.e., $\mathcal{Y}^{B\delta}_{t-1}$ as $\mathbf{USV}^T$. Then $\mathcal{B}_t = \Theta^{B\delta}_{t-1}\mathbf{VS}^{-1}\mathbf{U}^T$ or by retaining the top q components, we have $\mathcal{B}_t = \Theta^{B\delta}_{t-1}\mathbf{V}_q\mathbf{S}_q^{-1}\mathbf{U}_q^T$. ν_t can be computed using Eq.(2.8).

2.3.3 Foreground Motion Model

Based on the definition in Eq.(2.5), the foreground motion can be modeled as the speed of the object in x and y direction. We simply formulate the motion transition equation as:

$$\theta^F = \hat{\theta}^F_{t-1} + \mu^F_t \qquad (2.10)$$

where the random variable μ^F_t represents the changes in motion, represented as a zero-mean Gaussian distribution. When the complex foreground motion model with affine parameters is applied, the prediction process is similar to that of the background motion model illustrated in Section 2.3.2.

2.4 Algorithms

2.4.1 The Particle Filter

The optimal posterior density distribution of the Bayesian filtering problem in Eq.(2.2) can be analytically computed only in a restrictive set of cases. In general, an expression for the optimal density function cannot be determined analytically. The particle filter algorithm offers a numerical tool to approximate the optimal Bayesian solution. The *particle filter* was originally proposed as a probability propagation model in [8] in the signal processing literature and has been used to solve many tasks [23, 9] in computer vision. It approximates the current posterior distribution $p(\theta_t|Y_{1:t})$ by a set of weighted particles $\mathcal{S}_t = \{\theta_t^{(j)}, \omega_t^{(j)}\}_{j=1}^J$ with $\Sigma_{j=1}^J \omega_t^{(j)} = 1$, where J is the number of particles. To avoid *sample impoverishment* [44, 45], variations of particle filter, such as Bayesian bootstrap filters, control the distribution of particles by weights. The weights $\omega_t^{(j)}$ are updated recursively as follows [46]:

$$\omega_t^{(j)} \propto \frac{\mathcal{L}(Y_t|\theta_t^{(j)})p(\theta_t^{(j)}|\theta_{t-1}^{(j)})}{\pi(\theta_t^{(j)}|\theta_{t-1}^{(j)}, Y_{1:t})} \tag{2.11}$$

where $\pi(\theta_t^{(j)}|\theta_{t-1}^{(j)}, Y_{1:t})$ is the particle sampling proposal. Then, a resampling step is added to eliminate particles with lower weights and avoid the potential of the particle set collapsing into a single particle of higher weight. Selection of π is usually dependent on the application. In our approach, the proposal π is equal to the transition probability $p(\theta_t|\theta_{t-1})$, indicating that the new weights are updated as:

$$\omega_t^{(j)} \propto \mathcal{L}(Y_t|\theta_t^{(j)}) \tag{2.12}$$

Accordingly, the propagation is employed as follows: Given $\mathcal{S}_{t-1} = \{\theta_{t-1}^{(j)}, \omega_{t-1}^{(j)}\}_{j=1}^{J}$, which is weighted according to $\mathcal{L}(\theta_{t-1}|Y_{1:t-1})$, we first resample $\mathcal{S}_{t-1}$ to obtain a new set with equal weights $\{\theta_{t-1}^{'(j)}, 1\}_{j=1}^{J}$, then propagate $\theta_{t-1}^{'(j)}$ to $\theta_t^{(j)}$ using a prediction scheme discussed in Section 2.3. It is worth noting that although multiple cues are used later in measurement models, only the appearance cue is involved in predicting the motion models.

2.4.2 Iterative Optimization for Motion Estimation

According to the definition of the state vector in Eq.(2.3), the posterior distribution becomes a joint distribution of background and foreground motion parameters, which is $p(\theta_t^F, \theta_t^B|Y_{1:t})$. Our goal is to seek the optimal θ that maximizes the posterior probability, which requires solving θ_t^B and θ_t^F simultaneously.

$$(\hat{\theta}_t^F, \hat{\theta}_t^B) = \arg \max_{(\theta_t^F, \theta_t^B)} p(\theta_t^F, \theta_t^B|Y_{1:t}). \tag{2.13}$$

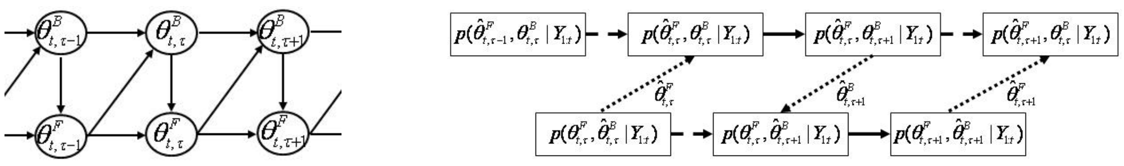
Because of the independence between the camera motion θ_t^B and the target motion θ_t^F, i.e.,

$$p(\theta_t^F, \theta_t^B|\theta_{t-1}^F, \theta_{t-1}^B) = p(\theta_t^F|\theta_{t-1}^F)p(\theta_t^B|\theta_{t-1}^B), \tag{2.14}$$

Eq.(2.2) can be expressed as

$$p(\theta_t^F, \theta_t^B|Y_{1:t}) \propto \mathcal{L}(Y_t|\theta_t^F, \theta_t^B) \int_{\theta_{t-1}^F} \int_{\theta_{t-1}^B} p(\theta_t^F|\theta_{t-1}^F)p(\theta_t^B|\theta_{t-1}^B)p(\theta_{t-1}^F, \theta_{t-1}^B|Y_{1:t-1})\, d\theta_{t-1}^F d\theta_{t-1}^B, \tag{2.15}$$

Given the high-dimensionality of the state vector θ, direct implementation of the particle filter is inefficient. To this end, we follow a divide-and-conquer strategy



(a) (b)

Figure 2.1: *(a) The graphic model for iteratively determining θ^B and θ^F at time t; (b) Progressively propagate the posteriors. The dashed lines represent the importance sampling to propagate the probabilities from the left side of the arrow to the right; the solid lines represent estimating the posteriors in Eqs.(2.18) and (2.19); the dotted lines refer to the optimal values of θ_t^B or θ_t^F determined by maximizing the posteriors in Eqs.(2.18) and (2.19).*

and develop an iterative algorithm similar to the expectation maximization (EM) [47] method to find a local solution by iteratively improving θ_t^B and θ_t^F. Let τ denote the index of iteration. With initial assumption of $\theta_{t,0}^F = \hat{\theta}_{t-1}^F$ and $\theta_{t,0}^B = \hat{\theta}_{t-1}^B$, we iteratively fix $\theta_{t,\tau}^F$ as $\hat{\theta}_{t,\tau-1}^F$ and $\theta_{t,\tau}^B$ as $\hat{\theta}_{t,\tau}^B$ and maximize the posteriors as follows when $\tau \geq 1$:

$$\hat{\theta}_{t,\tau}^B = \arg\max_{\theta_{t,\tau}^B} p(\hat{\theta}_{t,\tau-1}^F, \theta_{t,\tau}^B | Y_{1:t}). \tag{2.16}$$

$$\hat{\theta}_{t,\tau}^F = \arg\max_{\theta_{t,\tau}^F} p(\theta_{t,\tau}^F, \hat{\theta}_{t,\tau}^B | Y_{1:t}), \tag{2.17}$$

The strategy can be illustrated using a graphic model in Fig.2.1 (a). For each frame, multiple iterations are performed. However, based on experimental results, we have found that 3-6 iterations suffice.

At time t and iteration τ, we have acquired the optimal estimate $\hat{\theta}_{t,\tau}^B$ and $\hat{\theta}_{t,\tau}^F$. The $\tau + 1^{th}$ iteration is performed as follows:

- As an intermediate step, we first substitute θ_t^F in Eq.(2.15) by $\hat{\theta}_{t,\tau}^F$, leading to:

$$\begin{aligned}
p(\hat{\theta}_{t,\tau}^F, \theta_{t,\tau+1}^B | Y_{1:t}) &\propto \mathcal{L}(Y_t | \hat{\theta}_{t,\tau}^F, \theta_{t,\tau+1}^B) \int_{\theta_{t,\tau}^F} \int_{\theta_{t,\tau}^B} p(\hat{\theta}_{t,\tau}^F | \theta_{t,\tau}^F) p(\theta_{t,\tau+1}^B | \theta_{t,\tau}^B) p(\theta_{t,\tau}^F, \theta_{t,\tau}^B | Y_{1:t-1}) \, d\theta_{t,\tau}^F \, d\theta \\
&= \mathcal{L}(Y_t | \hat{\theta}_{t,\tau}^F, \theta_{t,\tau+1}^B) \int_{\theta_{t,\tau}^B} p(\theta_{t,\tau+1}^B | \theta_{t,\tau}^B) p(\hat{\theta}_{t,\tau}^F, \theta_{t,\tau}^B | Y_{1:t-1}) \, d\theta_{t,\tau}^B. \qquad (2.1
\end{aligned}$$

  Eq.(2.18) has the same form as in (2.2), thereby indicating that the particle filter algorithm can be applied to simulate $p(\theta_{t,\tau}^F, \theta_{t,\tau+1}^B | Y_{1:t})$ and obtain optimal estimate $\hat{\theta}_{t,\tau+1}^B$.

- Similarly, we have

$$p(\theta_{t,\tau+1}^F, \hat{\theta}_{t,\tau+1}^B | Y_{1:t}) \propto \mathcal{L}(Y_t | \theta_{t,\tau+1}^F, \hat{\theta}_{t,\tau+1}^B) \int_{\theta_{t,\tau}^F} p(\theta_{t,\tau+1}^F | \theta_{t,\tau}^F) p(\theta_{t,\tau}^F, \hat{\theta}_{t,\tau+1}^B | Y_{1:t-1}) \, d\theta_{t,\tau}^F. \tag{2.19}$$

We obtain the optimal estimate $\hat{\theta}_{t,\tau+1}^F$.

After a finite number (N_τ) of iterations, we estimate the optimal state vector $\hat{\theta}_t = \hat{\theta}_{t,N_\tau}$ at time t. Some issues need to be discussed further.

- Eqs.(2.18) and (2.19) show that two sets particles propagate in a progressively coupled manner. Accordingly, we have two separate weight sets throughout the iteration. We denote the weighted sample sets as $\{\theta_{t-1}^{F(j)}, \omega_{t-1}^{F(j)}\}_{j=1}^{J_F}$ and $\{\theta_{t-1}^{B(j)}, \omega_{t-1}^{B(j)}\}_{j=1}^{J_B}$, with J_B and J_F denoting the numbers of particles for background and foreground respectively.

- To compute the iterations, we need to know the proposal densities $p(\hat{\theta}_{t,\tau}^F, \theta_{t,\tau}^B | Y_{1:t-1})$ and $p(\theta_{t,\tau}^F, \hat{\theta}_{t,\tau+1}^B | Y_{1:t-1})$ in RHS of Eqs.(2.18) and (2.19). Invoking the importance sampling, the two proposal densities can be deducted from the posteriors at iteration τ, $p(\hat{\theta}_{t,\tau-1}^F, \theta_{t,\tau}^B | Y_{1:t})$ and $p(\theta_{t,\tau}^F, \hat{\theta}_{t,\tau}^B | Y_{1:t})$ in LHS of Eqs.(2.18) and

(2.19). For example, we have the weighted background sample set $\{\theta_{t,\tau}^{B(j)}, \omega_{t,\tau}^{B(j)}\}_{j=1}^{J_B}$,.

We update the weighted sample as

$$\begin{aligned}
\omega_{t,\tau}^{\prime B(j)} &= \omega_{t,\tau}^{B(j)} \frac{p(\hat{\theta}_{t,\tau}^{F}, \theta_{t,\tau}^{B(j)}|Y_{1:t-1})}{p(\hat{\theta}_{t,\tau-1}^{F}, \theta_{t,\tau}^{B(j)}|Y_{1:t-1})} && (2.20)\\
&= \omega_{t,\tau}^{B(j)} \frac{p(Y_{t-1}|\hat{\theta}_{t,\tau}^{F}, \theta_{t,\tau}^{B(j)})p(\hat{\theta}_{t,\tau}^{F}|\hat{\theta}_{t,\tau-2}^{F})}{p(Y_{t-1}|\hat{\theta}_{t,\tau-1}^{F}, \theta_{t,\tau}^{B(j)})p(\hat{\theta}_{t,\tau-1}^{F}|\hat{\theta}_{t,\tau-2}^{F})} && (2.21)
\end{aligned}$$

The updated sample set yields the posterior probability $p(\hat{\theta}_{t,\tau}^{F}, \theta_{t,\tau}^{B(j)}|Y_{1:t-1})$. Similar updating rule can be applied to obtain $p(\theta_{t,\tau}^{F(j)}, \hat{\theta}_{t,\tau+1}^{B}|Y_{1:t-1})$. Fig.2.1 (b) illustrates the progressive deduction for the proposals.

- The propagation probabilities differ for $\tau = 1$ and $\tau > 1$. When $\tau = 1$, the particles are drawn following the prediction rule for the transition motion model, discussed in Section 2.3, as well as the particle resampling and weight updating. For $\tau > 1$, both the old and new particles belong to the same time instant. Therefore, no prediction is involved.

In summary, we have decomposed one significant optimization problem that prohibits the direct use of the particle filter algorithm in the original state space into two small optimization problems operating iteratively, each of which can be solved using the particle filter algorithm. For brevity, in the following sections, we will continue to use notations θ_t^B and θ_t^F rather than $\theta_{t,\tau}^B$ and $\theta_{t,\tau}^F$ to derive the likelihood functions in Eqs.(2.18) and (2.19).

2.4.3 Algorithm Implementation

The tracking algorithm using the particle filter is briefly summarized as follows:

S.1 Initialization: assume that the system starts at time 0, with initial location of the target as $\mathbf{x}_0 = (x_0, y_0)^T$, and the initial sample set as $\{\theta_0^{B(j)}, \omega_{0,B}^{(j)}\}_{j=1}^{J_B}$ and $\{\theta_0^{F(j)}, \omega_{0,F}^{(j)}\}_{j=1}^{J_F}$. Set $t = 1$. At time $t \geq 1$,

S.2 Propagation: set $\{\theta_{t,0}^{B(j)} = \theta_{t-1}^{B(j)}, \omega_{t,0}^{B(j)} = \omega_{t-1}^{B(j)}\}_{j=1}^{J_B}$ and $\{\theta_{t,0}^{F(j)} = \theta_{t-1}^{F(j)}, \omega_{t,0}^{F(j)} = \omega_{t-1}^{F(j)}\}_{j=1}^{J_F}$.

For $\tau = 1 : N_\tau$:

S.2.1 $\forall_{j \in 1:J_B}$, resample $\{\theta_{t,\tau-1}^{B(j)}, \omega_{t,\tau-1}^{B(j)}\}$ to $\{\theta_{t,\tau-1}^{'B(j)}, 1\}$, draw particles, form the background motion sample $\theta_{t,\tau}^{B(j)}$ using (2.18), update the weights, compute the likelihood function $\mathcal{L}(Y_t|\hat{\theta}_{t,\tau-1}^F, \theta_{t,\tau}^{B(j)})$ and the posterior $p(\hat{\theta}_{t,\tau}^F, \theta_{t,\tau}^{B(j)}|Y_{1:t})$;

S.2.2 Select $\hat{\theta}_{t,\tau}^B = \arg\max_{\theta_{t,\tau}^{B(j)}} p(\hat{\theta}_{t,\tau-1}^F, \theta_{t,\tau}^{B(j)}|Y_{1:t})$;

S.2.3 $\forall_{j \in 1:J_F}$, resample $\{\theta_{t,\tau-1}^{F(j)}, \omega_{t,\tau-1}^{F(j)}\}$ to $\{\theta_{t,\tau-1}^{'F(j)}, 1\}$, draw samples and form the new foreground motion samples in Eq.(2.19), compute the likelihood function $\mathcal{L}(Y_t|\theta_{t,\tau}^{F(j)}, \hat{\theta}_{t,\tau}^B)$ and the posterior $p(\theta_{t,\tau}^{F(j)}, \hat{\theta}_{t,\tau}^B|Y_{1:t})$;

S.2.4 Select $\hat{\theta}_{t,\tau}^F = \arg\max_{\theta_{t,\tau}^{F(j)}} p(\theta_{t,\tau}^{F(j)}, \hat{\theta}_{t,\tau}^B|Y_{1:t})$;

S.3 Set $\hat{\theta}_t^B = \hat{\theta}_{t,\tau_{N_\tau}}^B$ and $\hat{\theta}_t^F = \hat{\theta}_{t,\tau_{N_\tau}}^F$. Update particle weights $\omega^{(j)}$. Update the current target location:

$$\hat{\mathbf{x}}_t = \hat{\mathbf{x}}_{t-1} + \hat{d\mathbf{x}}_t^F + \hat{d\mathbf{x}}_t^B = (x_{t-1}, y_{t-1})^T + (dx_t^F, dy_t^F)^T + (dx_t^B, dy_t^B)^T$$

S.4 Set $t \to t+1$ and goto ***S.2***.

2.5 Measurements

The posteriors defined in Eqs.(2.18) and (2.19) involve the computation of the foreground observation likelihood $\mathcal{L}(Y_t|\theta_t^F, \hat{\theta}_t^B)$ and the background observation likelihood $\mathcal{L}(Y_t|\hat{\theta}_t^F, \theta_t^B)$, which characterize the measurement model of our system. This section presents the measurement model and the computation of the observation likelihoods.

2.5.1 Foreground Observation Likelihood

To further improve the robustness of the tracking system in dynamically changing environments, we fuse multiple cues when measuring foreground observations. In our system, all visual cues contribute simultaneously to the overall observation model, and the relative relevances of cues are determined by the frame currently being processed, reflected by adaptive weights associated with the cues. This means no cue is ordained as "optimal", but the system itself weighs the contributions of different cues according to existing conditions. We design a measurement model that incorporates two principle cues, appearance and motion. Each cue generates a likelihood function. These functions are not entirely independent but are indirectly coupled by the result on which they agree [48]. We further assume that the measurements are also conditionally independent given the state, so that the entire likelihood can be factorized as

$$\mathcal{L}(Y_t|\theta_t^F, \hat{\theta}_t^B) = \prod_{i \in \{A,M\}} \mathcal{L}(Y_t^{(i)}|\theta_t^F, \hat{\theta}_t^B), \tag{2.22}$$

where $Y_t^{(A)}$ and $Y_t^{(M)}$ are the appearance and motion observations at t.

[49] reports an alternative integration technique, known as the "weighted voting" scheme that integrates the likelihoods derived for different cues as a weighted sum. This scheme and ours differ in that each cue makes an independent decision

before all the decisions are combined using a weighted sum. Such a strategy falls short because sometimes unreliable cues may still significantly influence the result.

2.5.1.1 Motion-Cue Based Foreground Measurement Model

The motion cue is acquired using the temporal difference image, denoted by Δ_t. It is estimated as:

$$\Delta_t = Y_t - \mathcal{T}\{Y_{t-1}; \hat{\theta}_t^B\}, \tag{2.23}$$

where Y_t and Y_{t-1} are the original frames, and $\mathcal{T}\{\bullet, \hat{\theta}_t^B\}$ is the stabilizing function determined by parameter $\hat{\theta}_t^B$. The advantage of using Δ_t is as follows: with respect to the background, most of objects of interests in surveillance videos are moving. In such cases, motion information appears to be a decisive measurement to separate the foreground object from the relatively static background, especially when the intensities of background and foreground do not differ greatly. Two extreme examples are shown in Fig.2.2, where it is difficult to separate visually the foreground from the background in the original data. But, if we use the difference images Δ_t shown in the second row, the targets are easily distinguished from the background. In the third row, we show the edge maps of Δ_t, in which the targets can be detected directly.

To extract the motion information of a foreground object, the edge gradient information of the Δ_t is used. The edge image E_t is generated as

$$E_t = \Delta_t \otimes DoG, \tag{2.24}$$

where $\otimes$ is a convolution operator and DoG is a 2D derivative of Gaussian (DoG)

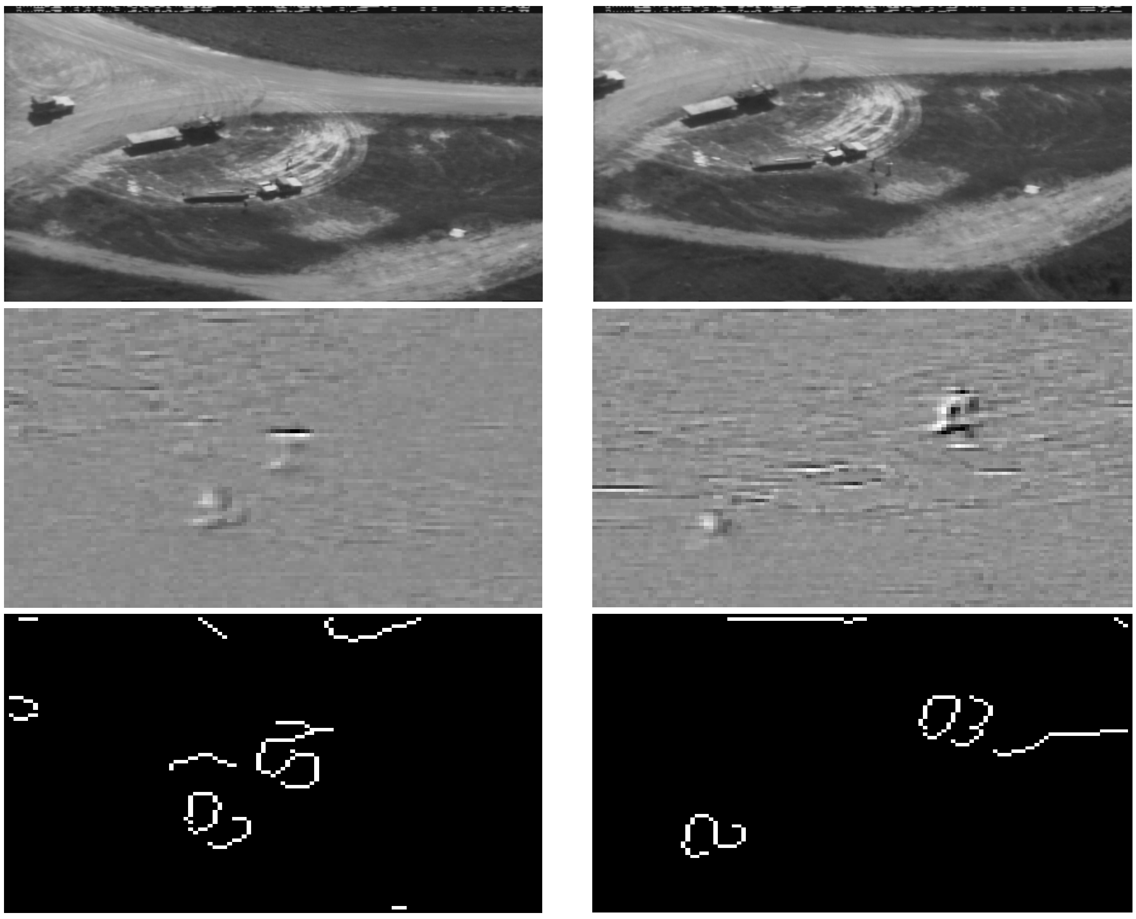


Figure 2.2: *Examples of humans in low contrast imagery passing through an open field. (upper) original images; (middle) part of* Δ *images containing the targets; (lower) part of the edge maps containing the targets.*

filter. Without loss of generality, we assume that the motions of all pixels belonging to the object are identical, because most objects in surveillance videos are small, and the local motion variances of the same object can be ignored. Therefore, we can use any feature pixels of the object, detected in E_t, to represent the entire object.

Generally, a particular pixel in Δ_{t-1} image translates by $\mathbf{dx} = (dx, dy)$ in Δ_t image due to motion continuity, which is defined as θ_t^F. The likelihood function of θ_t^F is estimated by introducing a cost function $D(\mathbf{x}; \mathbf{dx})$ that evaluates the discrepancy of two corresponding regions from two successive edge images, E_{t-1} and E_t. To minimize the risk of *drifting*, we add a static component inspection by matching a region between Δ_t and Δ_s. The latter is a static template obtained from the first two frames in the sequence and remains constant during tracking. Therefore, the cost function and the likelihood function both consist of two components, one dynamic and one static, defined as ($\theta_t^F \equiv \mathbf{dx}$):

$$\begin{aligned}
\mathcal{D}_t^{MD}(\mathbf{x}; \theta_t^F) &= \sum_{\mathbf{y} \in \mathbf{W_x}} \frac{|E_t(\mathbf{y} - \mathbf{dx}) - E_{t-1}(\mathbf{y})|^2}{|\mathbf{W_x}|} && (2.25)\\
\mathcal{L}(Y_t^{MD} | \theta_t^F, \hat{\theta}_t^B) &\propto \exp\,(-\frac{\mathcal{D}_t^{MD}(\mathbf{x}; \theta_t^F)}{2\sigma_{MD}^2}) && (2.26)\\
\mathcal{D}_t^{MS}(\mathbf{x}; \theta_t^F) &= \sum_{\mathbf{y} \in \mathbf{W_x}} \frac{|E_t(\mathbf{y} - \mathbf{dx}) - \Delta_s \otimes DoG|^2}{|\mathbf{W_x}|} && (2.27)\\
\mathcal{L}(Y_t^{MS} | \theta_t^F, \hat{\theta}_t^B) &\propto \exp\,(-\frac{\mathcal{D}_t^{MS}(\mathbf{x}; \theta_t^F)}{2\sigma_{MS}^2}) && (2.28)
\end{aligned}$$

where $\mathbf{W_x}$ is a support rectangle region with $\mathbf{x}$ as its center, $|\mathbf{W_x}|$ is the number of pixels in $\mathbf{W_x}$, MD and MS represent the dynamic and static components of the motion cue, respectively.

2.5.1.2 Appearance-Cue Based Foreground Measurement Model

Another important cue is related to local appearance. A reference template is generated for the object of interest. Candidate regions extracted from the current frame are compared to this reference template, and the smaller the discrepancy between the candidate and reference, the higher the probability that the object is located in the corresponding region. The localization performance hinges on the reference template selection. Similar to the motion cue, the appearance template also contains two components [27], one stable and one dynamic, which enhances the robustness of the system. The stable component is the object model manually cropped in the initializing stage, denoted by T_s. The dynamic one is the tracked object in time instant $t-1$, tuned to the same size of the stable template using scale transform. It is updated at each time instant. Accordingly, the cost functions and the likelihood functions are defined as:

$$\begin{aligned}
\mathcal{D}_t^{AD}(\mathbf{x};\theta_t^F) &= \sum_{\mathbf{y}\in\mathbf{W}_\mathbf{x}} \frac{|T_t(\mathbf{y}) - Y_t(\mathbf{y}-\mathbf{dx})|^2}{|\mathbf{W}_\mathbf{x}|} &(2.29)\\
\mathcal{L}(Y_t^{AD}|\theta_t^F,\hat{\theta}_t^B) &\propto \exp\,(-\frac{\mathcal{D}_t^{AD}(\mathbf{x};\theta_t^F)}{2\sigma_{AD}^2}) &(2.30)\\
\mathcal{D}_t^{AS}(\mathbf{x};\theta_t^F) &= \sum_{\mathbf{y}\in\mathbf{W}_\mathbf{x}} \frac{(|T_s(\mathbf{y}) - Y_t(\mathbf{y}-\mathbf{dx})|^2}{|\mathbf{W}_\mathbf{x}|} &(2.31)\\
\mathcal{L}(Y_t^{AS}|\theta_t^F,\hat{\theta}_t^B) &\propto \exp\,(-\frac{\mathcal{D}_t^{AS}(\mathbf{x};\theta_t^F)}{2\sigma_{AS}^2}) &(2.32)
\end{aligned}$$

where AS and AD represent the static and dynamic components of the appearance cue, Y_t is the original image at time t, T_t and T_s are the dynamic and static templates.

2.5.1.3 Adaptive Fusion Weights

One important factor that affects the effectiveness of multi-cue integration is the fusion weight set containing weights assigned to all cues. The values of weights reflect the contributions of the corresponding cues to the overall tracking system. Since each cue contains two components, we have four associated weights, denoted as $\{\beta_{c,t};\ c \in \mathcal{C} = \{AS, AD, MS, MD\}\}$. They are determined by the information reliability, which can be quantified based on the corresponding likelihoods. Hence, multi-cue integration can be implemented using a weighted product of the likelihood functions. The basic rule is that each cue associates with a score based on the error between the saliency of the individual's and the average's.

In each frame, we adjust the fusion weights according to the current visual context, allowing them to react to changing situation, and propagate them to the next frame as updated fusion weights. Thus, the cues with less reliable information are suppressed and those with reliable information contribute more to the fusion process. So Eq.(2.22) is reformulated as:

$$\begin{aligned} \mathcal{L}(Y_t|\theta_t^F, \hat{\theta}_t^B) &= \prod_{c\in\mathcal{C}} [\mathcal{L}(Y_t^c|\theta_t^F, \hat{\theta}_t^B)]^{\beta_{c,t}} \\ &\propto \exp\,(-\sum_{c\in\mathcal{C}} \frac{\beta_{c,t}\mathcal{D}_t^c(\mathbf{x};\theta_t^F)}{2\sigma_c^2}) \end{aligned} \tag{2.33}$$

The adaptive weights $\beta_{c,t}$ are determined using a mechanism similar to democratic integration [42]. Firstly, we measure the reliability in the form of the error associated with each cue at $t-1$:

$$\bar{E}_{c,t-1} = \frac{\mathcal{D}_{t-1}^c(\mathbf{x}_{t-1};\hat{\theta}_{t-1}^F)}{2\sigma_c} \tag{2.34}$$

where $\mathbf{x}_{t-1}$ is the previous estimate of the object position, and $\hat{\theta}^F_{t-1}$ is the foreground displacement estimated in previous frame. Then, a score $\gamma_{c,t}$ is approximated in Eq.(2.35) and normalized to a reliability measurement q_c in Eq.(2.36):

$$\gamma_{c,t} = \exp(-a\bar{E}_{c,t-1}) \tag{2.35}$$

$$q_{c,t} = \frac{\gamma_{c,t}}{\sum_{i\in\mathcal{C}} \gamma_{i,t}} \tag{2.36}$$

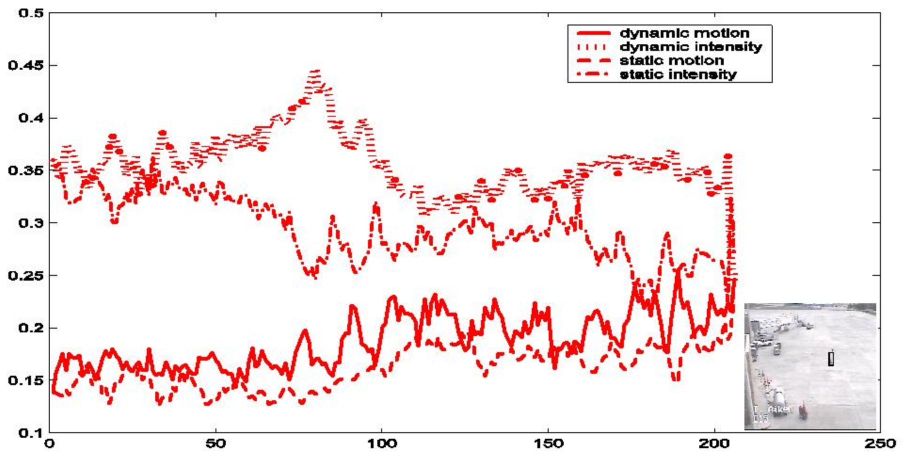
where a is a constant in the radial function. Finally, the weight $\beta_{c,t}$ is obtained as:

$$\beta_{c,t} = \frac{\xi}{\xi+1}\beta_{c,t-1} + \frac{1}{\xi+1}q_{c,t} \tag{2.37}$$

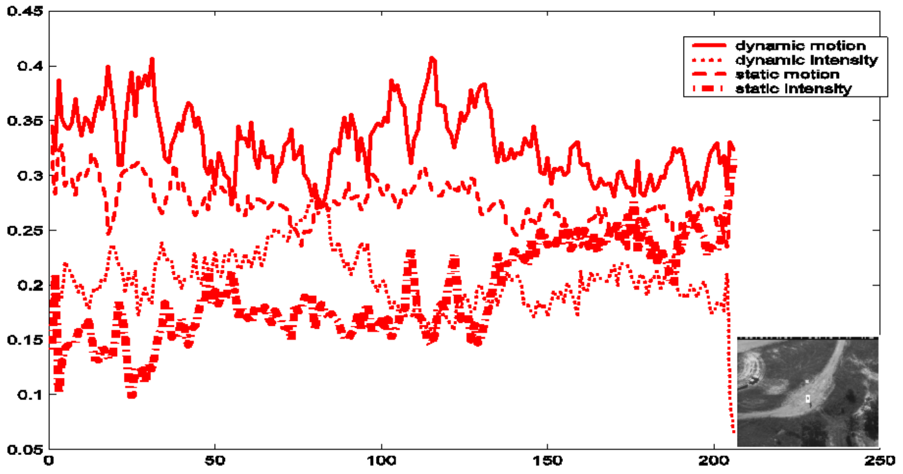
where ξ is a constant to control the adjusting speed of the system to the weight β_c, and q_c is a reliability measurement of cue c. The entire dynamics can be characterized by one equation:

$$\xi\dot{\beta}_{c,t} = q_{c,t} - \beta_{c,t} \tag{2.38}$$

In Fig.2.3 we illustrate how the adaptive weights change under different scenarios. Sequence (a) shows a slow-moving pedestrian in a high contrast video, therefore, the intensity information is dominant. On the contrary, sequence (b) shows a fast-moving pedestrian in a low-contrast video, and as a result, the weights for the motion information are greater than those of intensity information. We also notice that the dynamic components usually receive higher weights, because tracking results in previous frames help to generate reliable dynamic templates for likelihood evaluation.



(a)



(b)

Figure 2.3: *Adaptive weights plot for different information cues. Plot (a) shows a sequence containing slow-moving objects, where the weight plot shows that intensity cue (both dynamic and static components) is dominant in the observation model; Plot (b) shows a sequence containing a fast-moving object, where the weight plot shows that motion cue (both dynamic and static components) affects the observation model more than the intensity cue does. "Dynamic" means the likelihood function is estimated with respect to a dynamic template, while "static" means the likelihood function is estimated with respect to a static template.*

2.5.2 Background Observation Model

The likelihood function of the background observation model is much simpler: only the appearance cue is considered. This is reasonable because the background is large and usually contains more salient features. We define it as:

$$\mathcal{L}(Y_t|\hat{\theta}_t^F, \theta_t^B) \propto \exp\left(-\frac{\|Y_t^B - \mathcal{T}\{Y_{t-1}^B; \theta_t^B\}\|}{2\sigma_B^2}\right), \tag{2.39}$$

where Y_t^B represents the background of the video frame Y_t, with the foreground region specified by $\hat{\theta}_t^F$ excluded.

2.6 Experiments

2.6.1 Experiment results

We applied our algorithm to different sets of outdoor surveillance video sequences containing moving people. Most are captured by moving cameras. Generally, the pedestrians, occupy only 20-40 pixels, which is small. Some of the targets have weak intensity contrast with respect to the environment, on which the traditional trackers fail. In such cases, the weights of motion cues are expected to be higher than those of appearance cues. On the other hand, when the targets move slowly with a relatively stronger intensity contrast, the weights for appearance cues tend to be higher.

Fig.2.4 demonstrates an airborne sequence with a stationary background. We track two individuals separately. Their movements are slow, while the intensity contrasts are large. During tracking, the appearance information tends to be dominant.

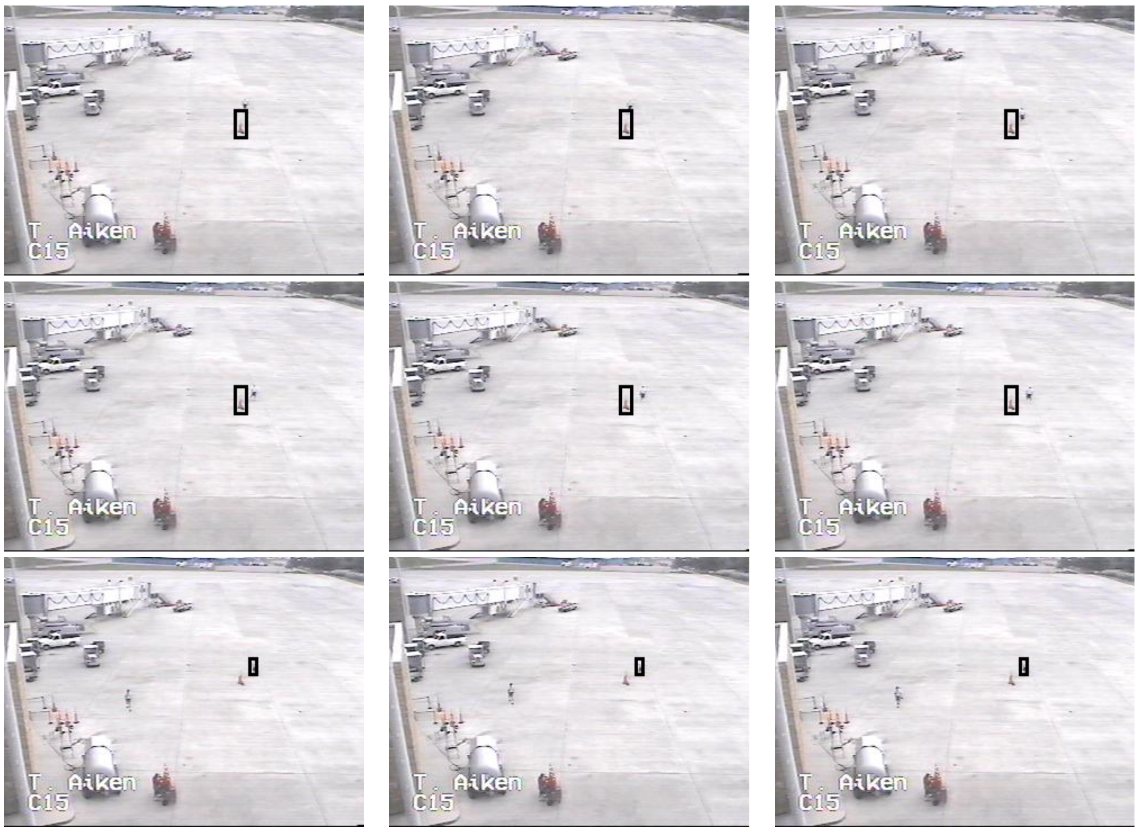


Figure 2.4: *Tracking of pedestrians running on the ground. The bounding box indicates the location of the targets.*

Fig.2.5 shows a ground-based sequence with a pedestrian crossing a road. The background is not static, requiring stabilization. From the frames, we find that the intensity difference between the target and the background is low. However, the pedestrian is moving prominently. Therefore, motion information becomes more dominant.

Fig.2.6 is an airborne sequence with multiple targets being tracked simultaneously, and the background is moving. Different targets also present different intensity contrasts with respect to the background. Consequently, the cue weights

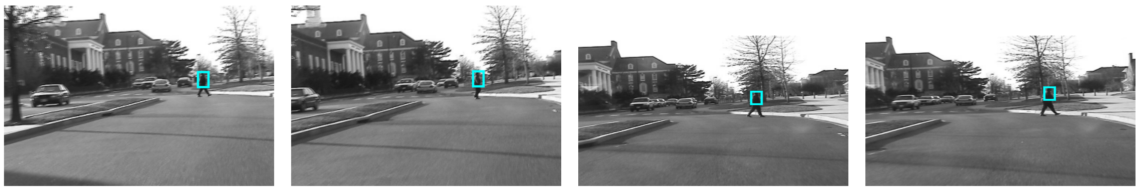


Figure 2.5: *Pedestrian tracking in a ground based video sequence. The bounding box indicates the location of the target.*

of different objects show different distributions. The tracking result is robust.

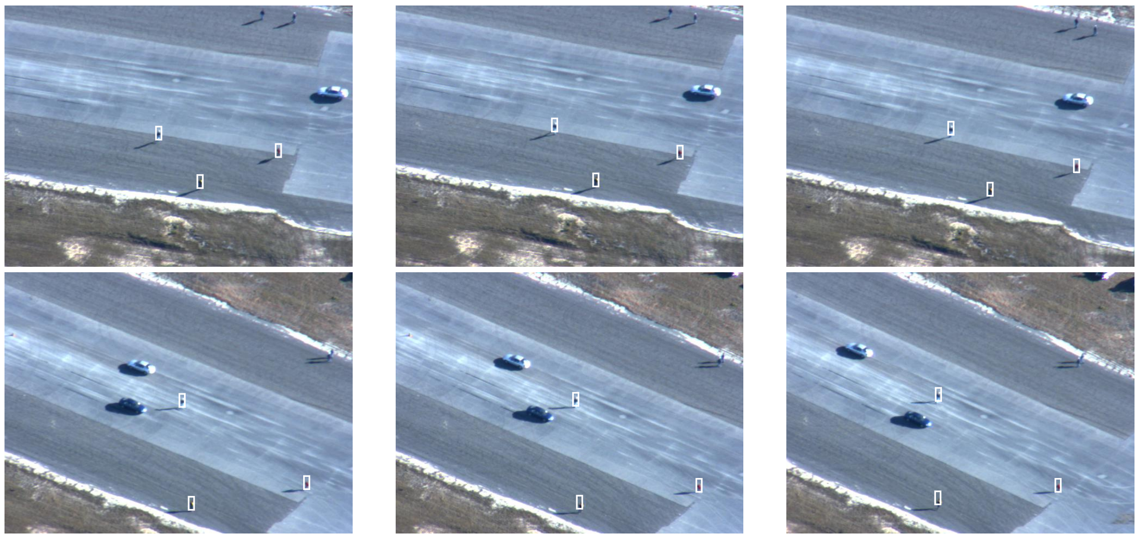
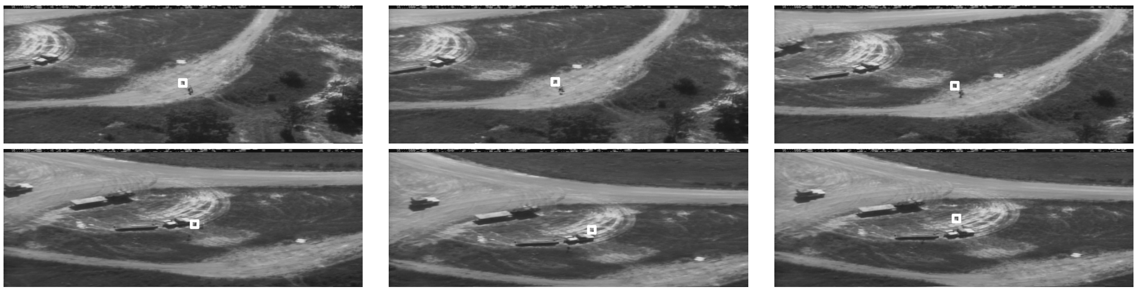
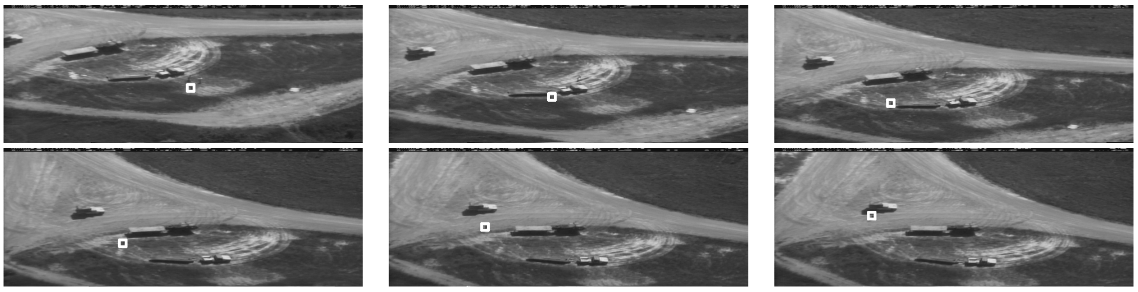


Figure 2.6: *Multiple-target tracking in an airport. In this sequence, as the intensities of the targets are different from the background, the appearance cues become dominant in the observation model.*

Fig.2.7 presents a fairly challenging airborne sequence with minimal visibility and a dynamic background. The two pedestrians in the sequence appear very small, and both have intensities similar to their surroundings in some frames. In these frames, motion information appears more reliable then intensity information, while in some other frames where motions of the targets are small, intensity information contributes more to the entire observation model.



(a)



(b)

Figure 2.7: *Tracking of different persons running in the field . The white box indicates the location of the target, (a) and (b) show two persons' moving, respectively. In this sequence, we can see that the objects are in a low-contrast background, therefore, the weights assigned to the motion cue are relatively higher than these assigned to the appearance cue.*

From the experimental results, we find that the proposed tracker performs well in most of the cases. We summarize the statistical characteristics of the above four sequences in Table 2.1. The background motion particles has 150 pixels with a Gaussian distribution used for μ_B. The foreground motion particles has 50 pixels with disturbance μ_F represented using a Gaussian distribution. The size of the DoG filter is 10×10, and the support region $\mathbf{W}_\mathbf{x}$ is 5×5. The practical values of all the variances σ_c involved in computation are set to the statistical values acquired from 10-20 training frames of corresponding sequences. Most values are between 1 and

	image size	target size	background	contrast	motion	target #
fig.2.4	320×240	5×12	stable	high	slow	1
fig.2.5	360×240	11×25	moving	low	fast	1
fig.2.6	640×480	8×20	moving	low	fast	3
fig.2.7	640×245	3×7	moving	very low	inconstant	2

Table 2.1: Characteristic summary for experimental sequences

3.5.

2.6.2 Comparison and Discussions

To provide some comparisons, we applied some standard tracking algorithms on the sequence shown in Fig.2.7. We used the algorithms provided from the VIVID Tracking Evaluation Web Site [50], which include: “Template Match” (TM) representing the normalized correlation template matching; “Basic Mean Shift” (MS) being the algorithm from [29]; “Variance Ratio” (VR) referring to the algorithm from [36]; “Fg/Bg Histogram Shift” (HIST) using two histogram models of the foreground and background to estimate the shift of the object; “Peak Difference” (PD) seeking the maximum peak difference to localize the object. To make the comparison fair, we start tracking from the same frame (frame 52) in each experiment and use the same initial bounding box. Fig.2.8 demonstrates the tracking results from these algorithms. All the algorithms tend to fail when applied to the low-resolution sequence in our experiments. We record the numbers of successfully tracked frames of all tested algorithms in Table 2.2. It is evident that the tracker introduced in this

Algorithm	HIST	MS	TM	VR	PD	BF-PF
Number of Frames	83	7	78	51	44	817

Table 2.2: Number of successfully tracked frames using different tracking algorithms. BF-PF is our proposed algorithm.

chapter tracks over a longer sequence (817 frames).

The success of our algorithm is due to the following reasons: 1) We use an adaptive multi-cue measurement model, which increases the reliability of the system. The foreground motion estimation is derived from both motion and appearance cues, thus overcoming the limitation of estimating using a single cue, which applies well to surveillance needs; 2) Cues are integrated using an adaptive mechanism that assigns weights to the cues in the current context, such that those with higher reliability receive larger weights and those with lower reliability have lower weights; 3) The edge information obtained from the derivative of Gaussian filter improves robustness. Otherwise, when the target enters the shadow area, tracking may fail due to environmental disturbance and image noise; 4) We simultaneously track both background and foreground motions. The background motion naturally provides parameters for camera stabilization, while the foreground information excludes the foreground pixels from the observation used for measuring the likelihoods of the background motions. Thus, the estimation error of the background motion parameters is reduced; 5) Stabilization is concerned with compensating for the errors caused by the moving camera; and 6) The cost function we used to estimate displacements is efficient for both non-rigid and rigid objects. We also discussed why the com-

(a) frame 128

(a) frame 132

(a) frame 135

(b) frame 54

(b) frame 56

(b) frame 59

(c) frame 126

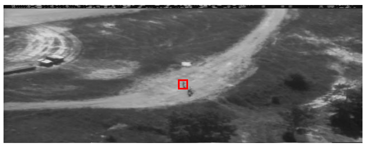
(c) frame 128

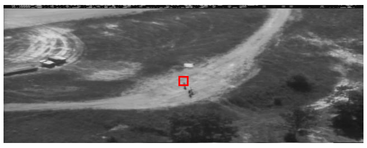
(c) frame 130

(d) frame 56

(d) frame 86

(d) frame 103


(e) frame 90


(e) frame 93

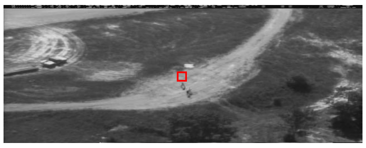

(e) frame 96

Figure 2.8: *Tracking results from different tracking algorithms. Each row represents a certain tracking algorithm we have used. Frames in the third column are snapshots of how the deviation occurs in each process. Trackers lost the target after these frames in our experiments. The algorithms we tested are: (a) HIST; (b) MS; (c) TM; (d) VR; and (e) PD.*

pared algorithms fail. Mainly, most algorithms use the appearance measurement to discriminate the foreground from the background. Therefore, if appearance features are not salient, the trackers perform poorly. Furthermore, these algorithms lack the ability to recover from tracker drift.

2.7 Conclusion

We have presented a surveillance tracking algorithm based on the particle filter framework. The approach builds a robust motion model over a state-space of multiple-hypotheses for the object. The algorithm can simultaneously track both background motion and foreground motion, which great ly improves the accuracy of the tracking result, especially for moving camera sequences; It also constructs a multi-cue observation model to enhance the robustness of the system. We applied our algorithm to several surveillance sequences with different visual conditions. The experimental results demonstrate that the tracker reliably tracks multiple hypotheses, even under challenging conditions due to low-contrast and fewer object pixels. The comparative experiments further validate that our algorithm significantly improves tracking performance. We are now investigating applications to several problems such as vehicle tracking and object classification.

Chapter 3

Non-Rigid Object Tracking: Motion and Deformation

3.1 Introduction

Visual tracking is an essential component of many applications from intelligent robotics to video surveillance. Basically, there are three groups of tracking methods: correspondence-based, transformation-based, and contour-based. The first group of methods is based on establishing correspondences between feature points. The second group tracks by estimating object motion, in which the objects are usually assumed to be made of planar shapes. The last group achieves tracking by finding the object contour in successive frames. It applies to cases when not only the location but also the deformation of a target is desired during tracking. These applications include surveillance tracking for recognition purpose and echocardiography tracking for computer aided diagnosis (CAD). The tracking approach proposed in this chapter belongs to the group of contour-based tracking methods.

3.1.1 Related Work

In this section we briefly introduce some existing work related to the contour tracking. A number of contour-based tracking methods have been proposed in the literature. As a milestone in contour-based tracking research, CONDENSATION, a parameterized B-spline contour tracking algorithm, was proposed by Isard and

Blake [9]. It uses a particle filter as the basic framework to track the global motion and deformation. The algorithm yields robust results when applied to rigid objects. However, it has no explicit criterion for extracting the exact boundary of a non-rigid object during tracking. In [51], Li *et al.* presented a particle filter for non-rigid object contour tracking. However the algorithm lacks the ability to discriminate a real boundary from all the detected edge points. Snake-based algorithms [52, 53] evolve an object boundary such that a weighted sum of external and internal energy terms is minimized. However, the methods are restricted to a relatively small range of scenarios because they assume the intensities inside objects to be fairly uniform. Besides, their computational complexity makes them less suitable for real-time applications. The level set approach, a powerful method that deals with topological changes of the moving level set function, uses partial differential equations (PDE) that describe the object motion, boundary and region-based information [54, 55, 56, 57]. They also prefer uniform intensity distributions inside objects.

Some other approaches closely related to non-rigid contour tracking include [58, 59, 60]. The concepts of motion and deformation were defined in [60]. ***Motion*** is parameterized by a finite dimensional group action, and ***deformation*** is the total deformation of the object contour (infinite dimensional group) modulo the finite dimensional motion group. By incorporating the prior information of the system dynamics in the deformation framework, [58] proposed a nonlinear dynamic model for tracking a slowly deforming and moving contour, with the contour represented implicitly as the infinite-dimensional locus of zeros of a given function. The algorithm suffers from expensive computational needs due to the joint minimization for the group action and the deformation. The work in [59] extended the ideas in [58]. It uses a particle filter to estimate the conditional probability distribution of motion and shape of the contour, which formalizes the incorporation of a prior system model along with an observation model. However, the algorithm also has limitations as it only relies on appearance cue in the observation model and lacks the ability to handle moving background cases.

Our approach shares some similarities with [59]. We claim that tracking non-rigid objects can be accomplished by estimating both translational (finite dimensional, ***Motion***) and non-translational movements (infinite dimensional, ***deformation***) of objects. Instead of estimating both motion and deformation in one step, we use a cascading framework. We propose to first estimate the motion, then the deformation. The estimation of deformation fulfills the operation of discriminating the real boundary from all edge points, the majority of which may be from the background. Robustness can be improved by constraining the deformation using a prior shape model. Therefore, we decompose the task of tracking non-rigid object contour into three components:

- 2D **Motion estimation** It estimates the object-wise spatial rigid-body motion, including translation and rotation parameters. Since the motion parameters are finite dimensional, we use a particle filter to estimate them.

- 2D **Shape deformation** It captures the pose changes of non-rigid objects. Each pixel on the boundary may have different but correlated deformations.

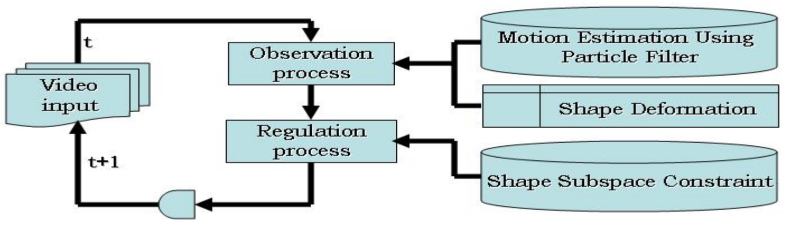


Figure 3.1: *Illustration of the proposed tracking system.*

We construct a deformation probability map using statistical analysis of different cues in each frame. The deformation occurs on the boundary pixels with higher deformation probabilities.

- **Shape regulation** It uses a trained shape subspace to restrict shape deformations. Regulation also reconstructs the occluded parts of the contours. Our method adaptively integrates the off-line trained, prior shape model with the object in the current video sequence.

Figure 3.1 presents a schematic illustration of the proposed system.

3.2 Preliminaries

3.2.1 B-spline Parametric Curves

In our contour tracking system, the tracking target is represented by a parametric B-spline curve. The visual 2D curves outlining the objects are represented in terms of parametric B-spline curves $\mathbf{r}(s) = [x(s), y(s)]^T$ [61]. The coordinates $[x(s), y(s)]$ are both spline functions of the curve parameter s. Furthermore, we use a set of control points $\mathbf{Q} = \{q_1, q_2, \dots, q_L\}$ to represent the B-spline curve, where each control point is defined as $q_l = (q_l^x, q_l^y)^T$, and L is the number of control points. One important reason for using the control point representation is because a set of $\mathbf{Q}$ can uniquely determine one B-spline curve. If we define the dynamic model that describes the contour motion as an affine transform, it is sufficient to apply the transform to the control points. Once the control points are transformed, the B-spline curve is transformed in the same manner. The property not only significantly improves the computational efficiency, but also helps to approximate the infinite deformation parameters using finite deformation parameters, i.e., the deformation of the curve is represented by the deformations of control points.

3.2.2 Particle Filter

The objective of tracking is to estimate the state of the dynamic model recursively, given some noisy visual observations, which allows us to formulate a Bayesian model:

$$p(\theta_t|Y_{1:t}) \propto p(Y_t|\theta_t) \int p(\theta_t|\theta_{t-1}) p(\theta_{t-1}|Y_{1:t-1}) \, d\theta_{t-1}, \tag{3.1}$$

where θ denotes the state vector, Y the observation, $p(Y_t|\theta_t)$ the likelihood function at time instant t. Observation Y is referred as images, or, visual cues in images. All inferences to the unknown state vector are based on the posterior probability in (3.1). The basic criterion is to find the vector with the maximum posterior probability. Many techniques can be used to achieve the goal, such as the Kalman filter [62] and the particle filter [46]. The Kalman filter can be used when the data are modeled by a linear Gaussian model. The latter one, also known as the sequential

Monte Carlo algorithm, presents a set of simulation-based methods proposed to handle more complex non-linear, high-dimensional data of probably non-Gaussian distribution. Many contour tracking algorithms use the particle filter algorithm given its flexibility, ease of implementation, parallelize and apply to general settings. We also use the particle filter to estimate the state vector of the dynamic model.

In the particle filter algorithm, the *prediction step* samples new particles based on the state transition probability $p(\theta_t|\theta_{t-1})$, and the previous posterior distribution $p(\theta_{t-1}|Y_{1:t-1})$, while the *update step* is controlled by particle weights, characterized by the likelihood function $p(Y_t|\theta_t)$:

$$\omega_t^{(j)} \propto p(Y_t|\theta_t^{(j)}), \tag{3.2}$$

The algorithm approximates the current posterior distribution $p(\theta_t|Y_{1:t})$ by a set of weighted particles $S_t = \{\theta_t^{(j)}, \omega_t^{(j)}\}_{j=1}^{J}$ with J representing the number of particles. To avoid the potential of the particles collapsing into a few particles with high weights, Sequential Importance Sampling (SIS) [63, 8] draws particles from a proposal distribution $g(\theta_t^{(j)}|\theta_{t-1}^{(j)}, Y_{1:t})$ and eliminates particles with lower weights. The weights are assigned as:

$$\omega_t^{(j)} \propto \frac{p(Y_t|\theta_t^{(j)})p(\theta_t^{j}|\theta_{t-1}^{(j)})}{g(\theta_t^{(j)}|\theta_{t-1}^{(j)}, Y_{1:t})}. \tag{3.3}$$

The selection of proposal distribution depends on the properties of different applications.

3.3 Motion Estimation

3.3.1 Dynamic Motion Model

We use the parameters of 2D affine transform to represent the finite dimensional motion of the object. The contour transform is given in terms of the homogeneous coordinates of the control points

$$\begin{bmatrix} q_{l,t-1} \\ 1 \end{bmatrix} = T \cdot \begin{bmatrix} q_{l,t} \\ 1 \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} q_{l,t} \\ 1 \end{bmatrix}, \qquad (3.4)$$

where $T \in \mathbb{R}^{3\times 3}$ represents an affine transform matrix with independent scale factors along both x and y axes. Accordingly, the state vector of the dynamic model is defined as

$$\theta_t = (T_{11} \;\; T_{12} \;\; T_{21} \;\; T_{22} \;\; T_{13} \;\; T_{23})^T. \qquad (3.5)$$

3.3.2 Estimate State Vector by Particle Filter

We use the particle filter to obtain the MAP (maximum a posterior) estimator of the state vector θ.

3.3.2.1 Prediction Step

Rather than use a proposal distribution for prediction, we predict the configuration of particles based on the following state transition model:

$$\theta_t = \hat{\theta}_{t-1} + \nu_t + U_t \qquad (3.6)$$

with $\hat{\theta}_{t-1}$ is the previous state estimate, ν_t as the predicted adaptive velocity in the motion vector, and U_t as the driving noise, assumed to be zero-mean Gaussian noise. The computation of ν_t entails the incorporation of the previous particle configuration in the prediction. Therefore, the diversity of particles is not compromised.

The prediction of ν_t is based on the assumption of brightness invariance [27], which means that there exists a θ_t such that the warping patch is similar to the previous image patch. We claim if $Z(\mathbf{Q}_t)$ is the intensities (colors) of the control point set $\mathbf{Q}_t$ (for robustness purpose, we use the corresponding intensities of Gaussian smoothed image frames), there exists θ_t that satisfies

$$Z_{t-1}(T(\theta_t) \cdot \mathbf{Q}_t) = Z_{t-1}(\hat{\mathbf{Q}}_{t-1})^1. \tag{3.7}$$

where $\hat{\mathbf{Q}}_{t-1}$ denotes the control point set estimated at $t-1$. We simplify (3.7) by $\mathcal{T}\{Z_t, \theta_t\} = \hat{Z}_{t-1}$, where $\hat{Z}_{t-1}$ is the corresponding intensities of $\hat{\mathbf{Q}}_{t-1}$. Approximating $\mathcal{T}\{Z_t; \theta_t\}$ via a first-order Taylor series expansion around $\hat{\theta}_{t-1}$ yields:

$$\mathcal{T}\{Z_t; \theta_t\} \simeq \mathcal{T}\{Z_t; \hat{\theta}_{t-1}\} + \mathcal{C}_t(\theta_t - \hat{\theta}_{t-1}) = \mathcal{T}\{Z_t; \hat{\theta}_{t-1}\} + \mathcal{C}_t \nu_t, \tag{3.8}$$

where $\mathcal{C}_t = \partial \mathcal{T} / \partial \theta$ is the Jacobian matrix. Substituting $\hat{Z}_{t-1}$ into (3.8), we obtain:

$$\hat{Z}_{t-1} \simeq \mathcal{T}\{Z_t; \hat{\theta}_{t-1}\} + \mathcal{C}_t \nu_t \tag{3.9}$$

$$\nu_t \simeq -\mathcal{B}_t(\mathcal{T}\{Z_t; \hat{\theta}_{t-1}\} - \hat{Z}_{t-1}) \tag{3.10}$$

where $\mathcal{B}_t$ is the pseudo-inverse of $\mathcal{C}_t$. Using the differences in motion vectors and

[1]More strictly, the affine transform on the LHS of (3.7) should be formulated as $T(\theta_t) \cdot \begin{bmatrix} \mathbf{Q}_t & \mathbf{1} \end{bmatrix}^T$.

the observation matrix as inputs, we obtain a least square (LS) solution to $\mathcal{B}_t$ as:

$$\Theta^{\delta}_{t-1} = [\theta^{(1)}_{t-1} - \hat{\theta}_{t-1}, \ldots, \theta^{(J)}_{t-1} - \hat{\theta}_{t-1}] \tag{3.11}$$

$$\mathcal{Z}^{\delta}_{t-1} = [Z^{(1)}_{t-1} - \hat{Z}_{t-1}, \ldots, Z^{(J)}_{t-1} - \hat{Z}_{t-1}] \tag{3.12}$$

$$\mathcal{B}_t = (\Theta^{\delta}_{t-1}\mathcal{Z}^{\delta T}_{t-1})(\mathcal{Z}^{\delta}_{t-1}\mathcal{Z}^{\delta T}_{t-1})^{-1} \tag{3.13}$$

where Θ^{δ}_{t-1} is the set of differences between all particle samples of $\{\theta^{(j)}_{t-1}\}^{J}_{j=1}$, and the optimal estimate $\hat{\theta}_{t-1}$, $Z^{(j)}_{t-1}$ is the j^{th} patch sample with state vector sample $\theta^{(j)}_{t-1}$, and $\mathcal{Z}^{\delta}_{t-1}$ is the set of all intensity differences between samples of $\{Z^{(j)}_{t-1}\}^{J}_{j=1}$ and $\hat{Z}_{t-1}$. Obviously, we incorporate the particle configuration at $t-1$ for prediction.

3.3.2.2 Updating Step

The weight updates are based on the likelihood function $p(Y_t|\theta_t)$. We follow the definition of observation model in [9]. We search the normal line $\mathbf{n}_l$ on each control point q_l, which is determined by corresponding θ, and detect feature points $\{z^{(l)}_j\}^{N_l}_{j=1}$, where N_l is the number of features detected along the normal. Multiple feature points appear due to the background clutter. Assuming that $\{z^{(l)}_j\}$ can be modeled as a spatial Poisson distribution along the normal lines and the true control point is a Gaussian distribution, the 1-D measurement density along $\mathbf{n}_l$ can be determined by the distances between the feature points to the corresponding control point, formulated as

$$p_l(z|q_l) \propto 1 + \frac{1}{\sqrt{2\pi}\sigma\psi\lambda}\sum_{j=1}^{N_l}\exp\left(-\frac{(z^{(l)}_j - q_l)^2}{2\sigma^2}\right), \tag{3.14}$$

where ψ is the probability of non-detection, λ is the density of clutter in the Poisson distribution, σ is the standard deviation of the Gaussian distribution. Figure 3.2

gives an intuitive illustration of evaluating the likelihood function for one control point. With the assumption that feature outputs on distinct control points are statistically independent, the overall likelihood becomes:

$$p(Y|\theta) = \prod_{l=1}^{L} p_l(z|q_l). \tag{3.15}$$

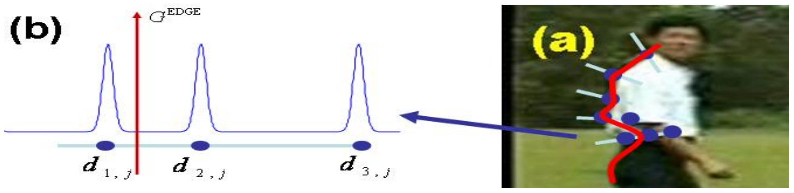


Figure 3.2: *(a) The red line is the contour determined by the control points. Light lines are the normal lines on the control points. The solid dots are feature points detected by the normal lines; (b) 1-Dimensional measurement density along the line normal on one control point q_j. $d_{1,j}$, $d_{2,j}$ and $d_{3,j}$ are three distances between the feature points to the corresponding control point. The vertical axis G^{EDGE} represents the gradient magnitude along the normal line $\mathbf{n}_j$.*

3.4 Deformation Estimation

After the MAP estimator $\hat{\theta}$ is obtained and the transformed control points $\hat{\mathbf{Q}}$ are acquired based on $\hat{\theta}$, the transformed curve $\tilde{\mathbf{r}}$ is determined. The next step is to enforce local deformation, i.e., find the real boundary points. [9] employed a simple strategy where the exact contour is determined by selecting feature points with maximum gradient magnitudes detected on corresponding normal lines. In other words, the contour estimation counts only on the gradient magnitudes. Unfortunately, this strategy does not always work, especially when the background is heavily cluttered or the object undergoes shape deformations between frames.

In our algorithm, we identify the correct feature points by detecting the deformation along the normal lines on transformed control points $\hat{\mathbf{Q}}$. Two elements are involved. One relates to the assumption that real boundary points of an object are detected along the orthogonal directions of the contour. It implies that the scanning range of normal lines influences the probability of the real boundary being detected. The other is that the gradient magnitudes are not sufficiently robust for exact contour delineation, especially when contaminated by background clutters and object textures. Accordingly, our strategy for deformation estimation contains two new features: (1) set normal lines adaptive; and (2) integrate several statistical cues into a deformation confidence map.

3.4.1 Set Normal Lines Adaptive

The scanning range of normal lines is determined by both searching lengths and centers. Earlier algorithms [9, 51, 64, 65, 57] set the search lengths and centers of normal lines identical and fixed, which may result in false detections due to inadequate modeling of shape variations. We enforce adaptability of the normal lines to reduce the probability of false detections.

3.4.1.1 Lengths of normal lines

Intuitively, a short normal line may miss the true boundary pixel, while a long one may intersect with edge points from background clutter. To reduce the possibility of a normal line intersecting with background clutter without sacrificing the chance of finding the actual boundary pixel, the lengths of normal lines are altered according to the pose variations of the corresponding contour control points in training sequences. For example, in sequences of walking humans, the relative positions of the head and trunk change slightly from frame to frame; while the sides, especially the legs and arms, change their relative positions more. Therefore, we should set the normal lines with large pose variations longer than those of small pose variations. The pose variations of pixels can be learned off-line (for example, walking pedestrian samples from USF dataset [66], consisting of one thousand 120×80 binary images with aligned pedestrian silhouettes.) as:

$$\begin{aligned} \xi(l) &= \mathbf{E}\|\mathrm{q}_l^k - \mathbf{E}(\mathrm{q}_l^k)\|^2 && (3.16) \\ u(l) &\propto L_{min} \log \frac{\xi(l)}{\min(\xi(l))} && (3.17) \end{aligned}$$

L_{min} is a constant representing the minimum length, and k denotes the index of training samples.

3.4.1.2 Centers of normal lines

Earlier algorithms set the centers of scanning normal lines as the control points on the estimated contour. This is not optimal. It is possible that the normal lines might intersect and therefor the estimated contour might be looped. Making the line centers adaptive by applying a distance transform (DT) [67] significantly reduces the probability of normal line intersections inside the object. The detailed steps are listed in Table 3.1. It is worth noting that we only concern closed contours in the algorithm. For open contours, we will force them to close by linking the first point and the last point. Figure 3.3 demonstrates the procedure for sketching adaptive normal lines along the estimated contour, with which we are able to search for the real contour pixels with more flexibility.

Table 3.1: Algorithm 1: set center of the normal line adaptive

1. Based on the transformed control points set $\hat{\mathbf{Q}}_t$ (the result from global motion estimation), construct a binary image $\mathcal{BI}$, set the region Ω circled by the contour $\tilde{\mathbf{r}}_t$ to 1;

2. Apply DT to $\mathcal{BI}$ to obtain a distance map $\mathcal{DI}$, which is defined as:

$$\mathcal{DI}(\mathbf{x}) = \begin{cases} \min_{\mathbf{y}\in\tilde{\mathbf{r}}} dist(\mathbf{x},\mathbf{y}), & \mathbf{x}\in\Omega; \\ 0, & \text{otherwise.} \end{cases} \tag{3.18}$$

3. On each control point $\hat{q}_l$, draw a normal line $\mathbf{n}_l$, find the maximum distance value satisfying $\mathcal{DI}_l = max_{\mathbf{x}\in\mathbf{n}_l}\mathcal{DI}(\mathbf{x})$. The lengths of the normal line on two sides of the control point are set as follows:

$$u(l)_{in} = min(u(l)/2, \mathcal{DI}_l - d_0) \tag{3.19}$$

$$u(l)_{out} = max(u(l)/2, u(l) - [\mathcal{DI}_l - d_0]) \tag{3.20}$$

where d_0 represents a minimum safe distance to avoid contour loops. Here $d_0 = 2$.

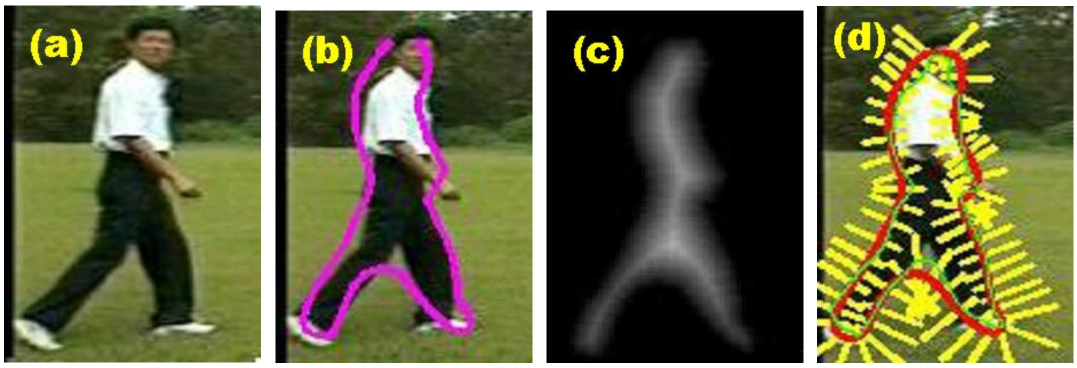


Figure 3.3: *An example of how to make a normal line scanning adaptive. (a) cropped original object; (b) estimated contour; (c) distance transformed object; (d) normal lines on control points.*

3.4.2 Multi-cue Deformation Probability Map

To extract the real contour, we define a posterior deformation probability map as $P_t(Y|\hat{\mathbf{Q}})$, based on the transformed control point set obtained by global motion estimation with Y representing related visual cues. In the map, a high probability implies that the corresponding pixel is more likely to rest on the real contour, and a low value implies a lower likelihood. Instead of using edge magnitudes as the only visual cue, we integrate several cues to evaluate the deformation probability.

3.4.2.1 Multi-cue Fusion

Fusion with respect to different cues can be interpreted as using multiple measurement sources. Assuming that we have M cues, the observation can be represented by $Y = (Y_1, \ldots, Y_M)$. We further assume that the observations are conditionally independent [68]. The deformation probability is therefore factorized as:

$$P_t(Y|\hat{\mathbf{Q}}) = \prod_{i=1}^{M} P_t(Y_i|\hat{\mathbf{Q}}) \tag{3.21}$$

Scanning for pixels with maximum probability values (ML estimator) on adaptive normal lines, we obtain the refined contour pixels, denoted as $\hat{\mathbf{r}}$. As an example, Fig.3.4 shows some probability maps from different cues on one processed frame, together with the fusion map $P_t(Y|\hat{\mathbf{Q}})$. Apparently, the fusion result suppresses noise from the gradient magnitude cues inside the contour, which is caused by object textures. We introduce the computation of probability maps from different cues in the rest of this section. It is worth noting that the scanning range for each pixel is one dimensional, so the two dimensional deformation probability map P can be further simplified to several one dimensional probability vectors associated with each control point. Therefore, the calculation of the deformation probabilities is limited to normal lines. This improves the computational efficiency.

3.4.2.2 Gradient Magnitude Cues

As shown in Figure 3.4.(a), gradient magnitude is an important feature for representing an object boundary. However, when the object is not homogenous in color or intensity, many edges are generated inside the object. We want to minimize the effect of inside edges. Anisotropic diffusion offers one possible approach to allow the entire image to be more uniform in color or texture, while still preserving the object boundaries [69]. It is highly possible that points with high gradient magnitudes after diffusion belong to the boundary of the target.

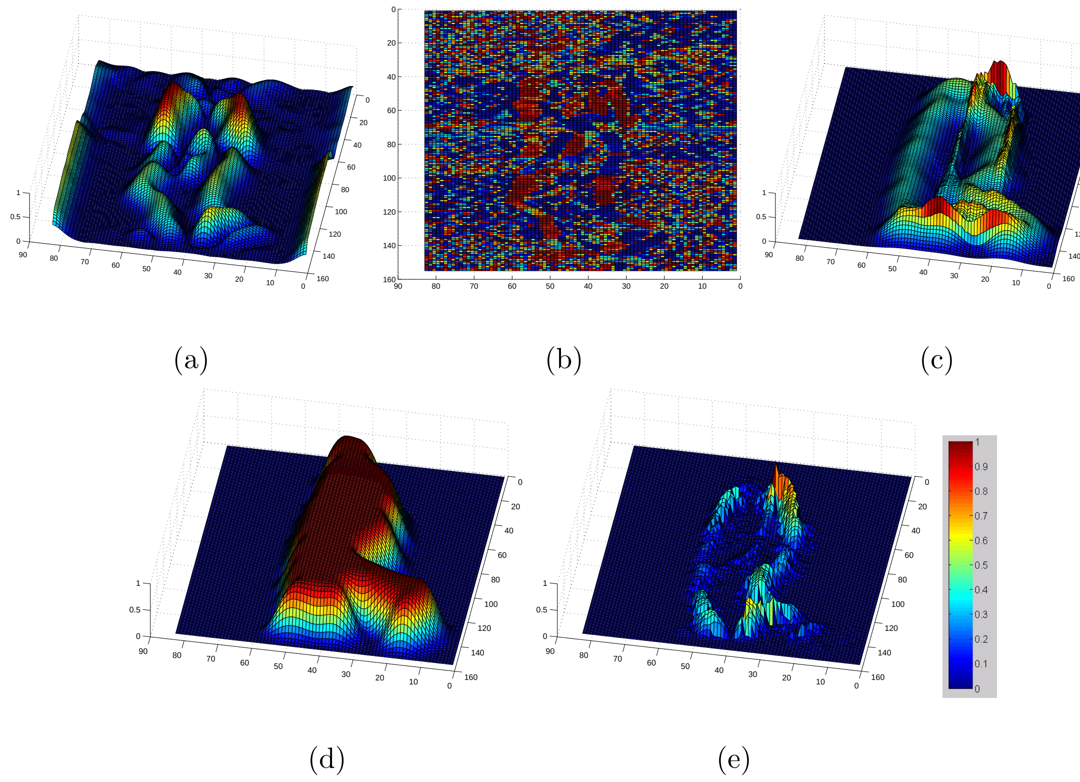
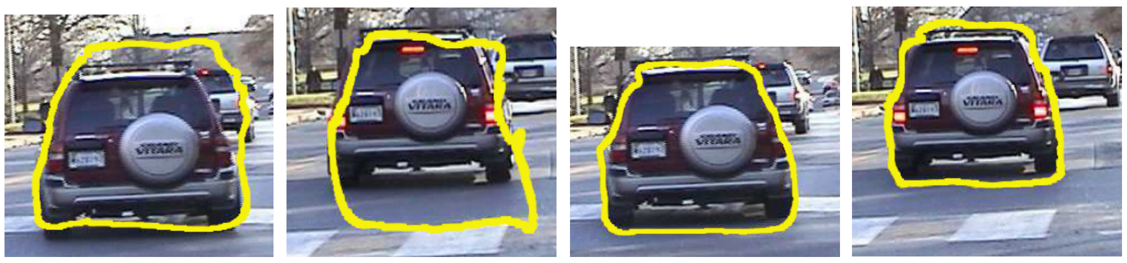


Figure 3.4: *An illustration of fusion from different visual sources, including the probability maps of (a)gradient Magnitude; (b) gradient Orientation; (c) shape Template; (d) foreground; (e) fusion.*



(1) Using the regular edge map *(2) Using the feature maps $\mathcal{FI}$*

Figure 3.5: *Performance comparisons on a cluttered scene. A stabilization step is applied to obtain the second set of results to obtain $\mathcal{FI}$ because the sequence was acquired by a moving camera.*

After the diffused feature map $\mathcal{EI}$ is extracted from the original image, a motion mask $\triangle\mathcal{I}$, indicating the possible area where motion could occur, is applied to $\mathcal{EI}$ to further suppress the background clutter. The masked map is then filtered by $\mathbf{G}$, a smoothing Gaussian filter. Eventually, we normalize the filtered map to a magnitude probability map P by:

$$P_t(Y_m|\hat{\mathbf{Q}}) = \alpha_m(\mathcal{EI} \bullet \triangle\mathcal{I}) * \mathbf{G} \tag{3.22}$$

where α_m is a normalizing coefficient, and $*$ denotes convolution. Figure 3.5 demonstrates performance improvement when we use the diffusion edge map convolved with the motion mask. Since the background and the tracking object move, a background stabilizing step [70] is applied to estimate the motion mask $\triangle\mathcal{I}$. The stabilization is based on the idea that the background movement can be modeled as a planar affine transform.

3.4.2.3 Gradient Orientation Cues

A gradient orientation map $\mathcal{OI}$ provides the orientations of edges. The gradient orientation presents a useful feature to discriminate the real object boundaries from all the detected edges, especially when the background clutter is present. If we denote the orientation of the normal on the true boundary as ϕ, we expect the local normal orientation to yield a Gaussian distribution with mean equal to ϕ [71]. At the meantime, the normal orientation distributions for pixels not on the boundary tend to have a uniform distribution between $[0, 2\pi)$. Figure 3.4.(b) illustrates the different distributions presented by the pixels on the contour and not on the contour. This leads to the definition of the orientation probability map as:

$$P_t(Y_o|\hat{\mathbf{Q}}) \propto \exp(-\frac{(\mathcal{OI}_t(\mathbf{x}) - \hat{\phi}_{t-1}(l))^2}{\sigma_o^2}) \quad \forall \mathbf{x} \in \mathcal{R}(\mathbf{n}_l), \tag{3.23}$$

where $\mathcal{R}(\mathbf{n}_l)$ defines a proximity region to $\mathbf{n}_l$:

$$\mathcal{R}(\mathbf{n}_l) \triangleq \{\mathbf{y} \in \mathcal{R}(\mathbf{n}_l) : dist(\mathbf{y}, \mathbf{n}_l) \leq dist(\mathbf{y}, \mathbf{n}_k), k \neq l\} \tag{3.24}$$

3.4.2.4 Shape Template Cues

The shape of a tracking object has its specific pattern. Therefore, the shape template could be used as one cue, indicating the probability that each image pixel belongs to the real object contour. Our shape template differs from the static shape energy proposed by Cremers *et al.* [72], which is pretrained and remains unchanged during tracking. We use an online model that incorporates a dynamic part that varies according to the observations (transformed contour by global motion estimation). Let Y_s denote the shape template observation; it contains the shape prior model $\mathcal{A}_{\mathbf{S}}$ and the dynamic template $\mathcal{A}_{\hat{\mathbf{Q}}}$. The former is a static template generated from the training data, and the latter yields a Gaussian distribution, the mean of which is set to the transformed contour $\hat{\mathbf{Q}}$. The probability is given by:

$$P_t(Y_s|\hat{\mathbf{Q}}) = a_t\mathcal{A}_{\mathbf{S}} + (1 - a_t)\mathcal{A}_{\hat{\mathbf{Q}}} \tag{3.25}$$

where $0 < a_t < 1$ is the weight that controls the integration of $\mathcal{A}_{\mathbf{S}}$ and $\mathcal{A}_{\hat{\mathbf{Q}}}$. The construction of $\mathcal{A}_{\mathbf{S}}$ is straightforward based on how frequently it appears as 1 in the training data. Figure 3.4.(c) provides an example of the probability map from the shape template cue. We show more examples of probability templates in Figure 3.6.

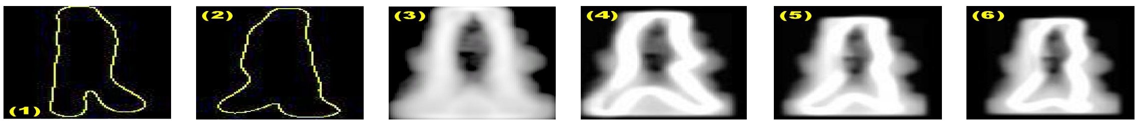


Figure 3.6: *Illustrations of adaptive probability shape templates for pedestrians. (1),(2): shape training samples; (3): shape prior model; (4),(5),(6): examples of probability shape templates in different frames.*

3.4.2.5 Foreground Cues

To suppress the contamination from the background clutter, we use a foreground probability map $P_t(Y_f|\hat{\mathbf{Q}})$ that estimates the likelihood of a pixel belonging to the tracked object. This map is calculated by comparing the current frame to a set of background models representing the static parts of the scene. The pixel-wise background models are adapted directly by the previous values for static camera setups. For moving cameras, these models can be fit after consecutive frames are aligned on a mosaic using globally estimated motion parameters. We define the background as layers of multivariate Gaussian functions $\{(\mu_t^i, \mathbf{\Sigma}_t^i, \kappa_t^i, \upsilon_t^i)\}_{i=1..\mathcal{K}}$ where μ_t^i is the posterior mean, $\mathbf{\Sigma}_t^i$ is the marginal posterior covariance, υ_t^i is the degrees of freedom, κ_t^i is the number of prior measurements of the i^{th} layer, and $\mathcal{K}$ is the number of layers in 3D color space. At each frame, we update the layer parameters using an online Bayesian estimation method as described in [73]. We order the layers according to confidence scores. Our confidence measure is inversely proportional to the determinant of the covariance. Then, we filter the layers with confidence values less than a specific threshold. For the remaining layers, we measure the Mahalanobis distance of observed color $I(\mathbf{x})$

$$d_i(\mathbf{x}) = (I(\mathbf{x}) - \mu_{t-1}^i)^T (\mathbf{\Sigma}_{t-1}^i)^{-1} (I(\mathbf{x}) - \mu_{t-1}^i) \tag{3.26}$$

and, update the parameters of the confident layers. Pixels outside of 99% confidence interval of all confident layers of the background are considered as foreground pixels. After the update, the foreground probability map at a pixel is determined as

$$P_t(Y_f|\hat{\mathbf{Q}}) = \alpha \exp(-\min_{i=1}^{\mathcal{K}} d_i(\mathbf{x})) \tag{3.27}$$

where α is a normalizing constant. Figure 3.4.(d) show an example of the probability map based on the foreground cue.

3.5 Regulation on Shape Deformation

Based on the estimated global motion and local deformation, we could track the contour from frame to frame. However, what if the deformation estimate is severely corrupted by noise? For example, the exact contour points may not always coincide with maximum probability pixels, but may appear at the second maximum pixel. In such cases, shape regulation serves as an important constraint to recover from detection errors. Intuitively, learning the prior shape knowledge of the object from the training set could help in delineation. The training samples should ideally cover all deformation variations. If an object in one frame exhibits a particular type of deformation not present in the training set, the system searches for the deformation in the subspace that is closest to the target, i.e. the system projects an arbitrary deformation onto the subspace, then achieving regulation.

3.5.1 Generic Shape Model

Several approaches exist for subspace construction. One of them was introduced by Cootes *et al.*, the active shape model (ASM) [74] with the points distribution model (PDM) described in [75], for obtaining the shape subspace. Our training method is based on the idea of PDM. A shape model is defined in terms of x and y coordinates of every "landmark" point lying on the outline of the target. The number of "landmark" points is fixed at equal intervals along the contours. The *control points* of B-splines are regarded as the "landmark" points. Table 3.2 gives the steps for training a prior model from a set of N samples, each represented by a set of columnized L control points ${\mathbf{Q_s}}^i = \{q_s^{(i,j)} | 1 \leq i \leq N, 1 \leq j \leq L\}$.

Table 3.2: Algorithm 2: Construction of Prior Shape Model

(a) Align the set of examples to a common frame of reference, $\mathtt{x}^i$=aligned($\mathbf{Q_s}^i$) [75];

(b) Calculate the mean of the aligned examples $\bar{\mathtt{x}}$, and the deviations $\delta\mathtt{x}^i = \mathtt{x}^i - \bar{\mathtt{x}}$;

(c) Calculate the eigensystem of the covariance matrix of the deviations, $\mathbf{C} = \frac{1}{L}\sum_{i=1}^{M}(\delta\mathtt{x}^i)(\delta\mathtt{x}^i)^T$.

(d) The first $\mathtt{t}$ principal eigenvectors of the eigensystem are used to generate $\mathtt{x} = \bar{\mathtt{x}} + \mathbf{Pb}$, where $\mathbf{b}$ is a $\mathtt{t}$-element vector of shape variation parameters and $\mathbf{P}$ is a $2L \times \mathtt{t}$ matrix of $\mathtt{t}$ eigenvectors, which composes the estimated shape subspace. We denote the eigenvalue diagonal matrix as $\Lambda_{\mathtt{t}}$, which is a $\mathtt{t} \times \mathtt{t}$ matrix.

3.5.2 Adaptive Shape Model

The constructed shape model is a generic model that can apply to all cases, as long as a target belongs to the corresponding object category. However, what we need is a deformation model that can represent more accurate shape variations. One solution is to update the existing PCA model with the initial contour of the current sequence (either manually marked or automatically detected) [76]. Denote $\mathtt{x}_0$ as the aligned initial contour, vector $\mathbf{b}_s = \mathbf{P}^T(\mathtt{x}_0 - \bar{\mathtt{x}})$ as the subspace component, the projection residue is obtained as:

$$\mathtt{x}_r = \mathtt{x}_0 - \bar{\mathtt{x}} - \mathbf{Pb}_s. \tag{3.28}$$

The residue part represents the shape variation not being covered in the prior model, so the generic subspace $(\bar{\mathtt{x}}, \mathbf{P}, \Lambda_{\mathtt{t}})$ is updated corresponding to the residue by the following equations:

$$\begin{aligned}
\bar{\mathtt{x}}^* &= \beta\bar{\mathtt{x}} + (1-\beta)\mathtt{x}_r && (3.29)\\
\mathsf{e}_r &= \frac{\mathtt{x}_r^T}{\|\mathtt{x}_r\|}(\mathtt{x}_0 - \bar{\mathtt{x}}) && (3.30)\\
\mathbf{C}^* &= \beta\begin{bmatrix}\Lambda_{\mathtt{t}} & \mathbf{0}\\ \mathbf{0}^T & 0\end{bmatrix} + \beta(1-\beta)\begin{bmatrix}\mathbf{b}_s\mathbf{b}_s^T & \mathsf{e}_r\mathbf{b}_s\\ \mathsf{e}_r\mathbf{b}_s^T & \mathsf{e}_r^2\end{bmatrix} && (3.31)
\end{aligned}$$

where β is the update weight. By applying SVD to (3.31), we obtain $(\mathbf{P}^*, \Lambda^*_{\mathtt{t+1}})$ satisfying $\mathbf{P}^*\Lambda^*_{\mathtt{t+1}}(\mathbf{P}^*)^T = \mathbf{C}^*$. For brevity, we still use $(\bar{\mathtt{x}}, \mathbf{P}, \Lambda_{\mathtt{t}})$ to denote the updated shape subspace in the rest of the chapter.

3.5.3 Subspace Projection

The projection of deformation can be described as representing the deformed contour by a linear combination of basis in the shape subspace. We first align the deformed contour point vector set $\hat{\mathbf{r}}_t$ to $\mathtt{x}_t$, and then apply

$$\mathtt{x}_{p,t} = \mathbf{P}\mathbf{P}^T(\mathtt{x}_t - \bar{\mathtt{x}}) + \bar{\mathtt{x}}, \qquad (3.32)$$

where $\mathtt{x}_{p,t}$ is a linear combination of subspace basis. It is possible that some control points on $\vec{\mathtt{r}}$ may be occluded, or not detected along the normal lines. Let us denote the index set of detected points as $I_d = \{i_1, i_2, \ldots\}$. We can recover a complete projected contour as follows:

$$\begin{aligned}
\mathtt{x}_{p,t} &= \mathbf{P}\mathbf{P}^{\dagger}_{I_d}(\mathtt{x}_{I_d,t} - \bar{\mathtt{x}}_{I_d}) + \bar{\mathtt{x}} && (3.33)\\
\mathbf{P}^{\dagger}_{I_d} &= (\mathbf{P}^T_{I_d}\mathbf{P}_{I_d})^{-1}\mathbf{P}^T_{I_d} && (3.34)
\end{aligned}$$

A projection example is demonstrated in Figure 3.7, with a comparison between tracking results with and without shape regulation. Evidently, using the subspace can preclude the contour from deforming to an irregular shape.

3.5.4 Alignment

Alignment is needed to normalize the contour, because the shape subspace is constructed from normalized training samples. We use the method for rigid shape matching proposed by Cootes *et al.* [74]. The basic idea is to find a transform matrix (containing rotation, translation and scale coefficients) and match the given contour to the mean of the shape model. An example is shown in Figure 3.8. The sequence is collected by magnetic resonance imaging (MRI) of human knees. Our primary interest lies in the articular cartilage layer. Fig.3.8.(1) depicts the detected contour pixels, where some pixels are too obscure to be detected. Fig.3.8.(2) shows the projected contour using the shape subspace without alignment. Fig.3.8.(3) depicts the contour pixels after alignment, and Fig.3.8.(4) gives the final result.

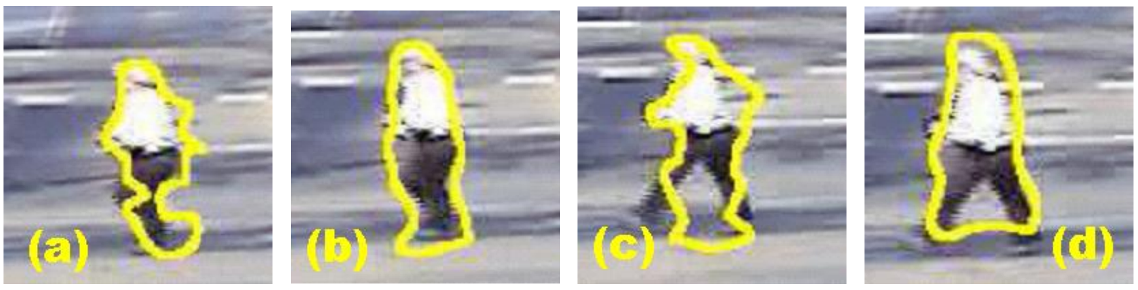


Figure 3.7: *Comparisons between tracking results with and without subspace regulation. (a) and (c) are without regulation, (b) and (d) are with regulation.*

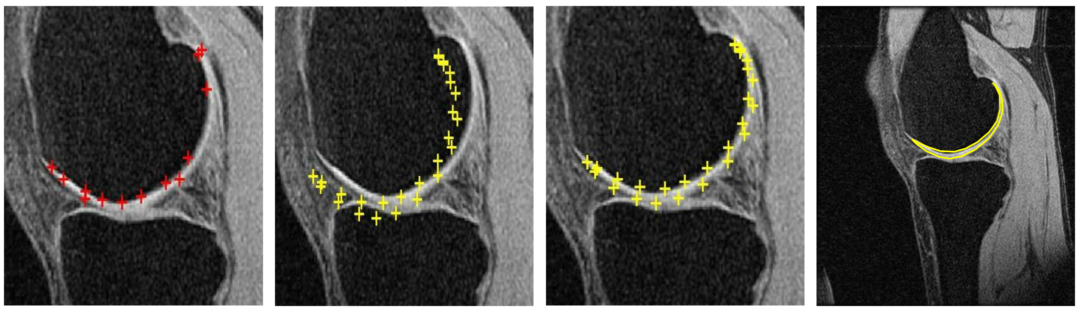


Figure 3.8: *One frame from an MRI sequence. The target of interest is the articular cartilage layer. (1) depicts the detected contour pixels. (2) shows the recovered contour pixels using the shape subspace projection. (3) depicts the contour pixels after alignment. (4) the final result.*

3.6 Implementation and Experiments

3.6.1 Algorithmic Implementation

A summary of the complete contour tracking algorithm is given in Table 3.3. We want to further discuss the initialization. The easiest way to acquire the initial contour in the beginning frame is to outline manually the object of interest in the first frame. The pixels along the contour are sorted in the clockwise order. The control points are then selected based on the uniform arc length rule. We can also apply automatic shape detection methods, such as the direct use of probability shape template to detect pedestrians [77]. Our tracking algorithm has the excellent quality of a high tolerance to localization errors in the initial contours. In our experiments, we observed that based on an approximate but reasonable initial guess, the tracking results of the object after three to six frames appear to be fairly accurate,

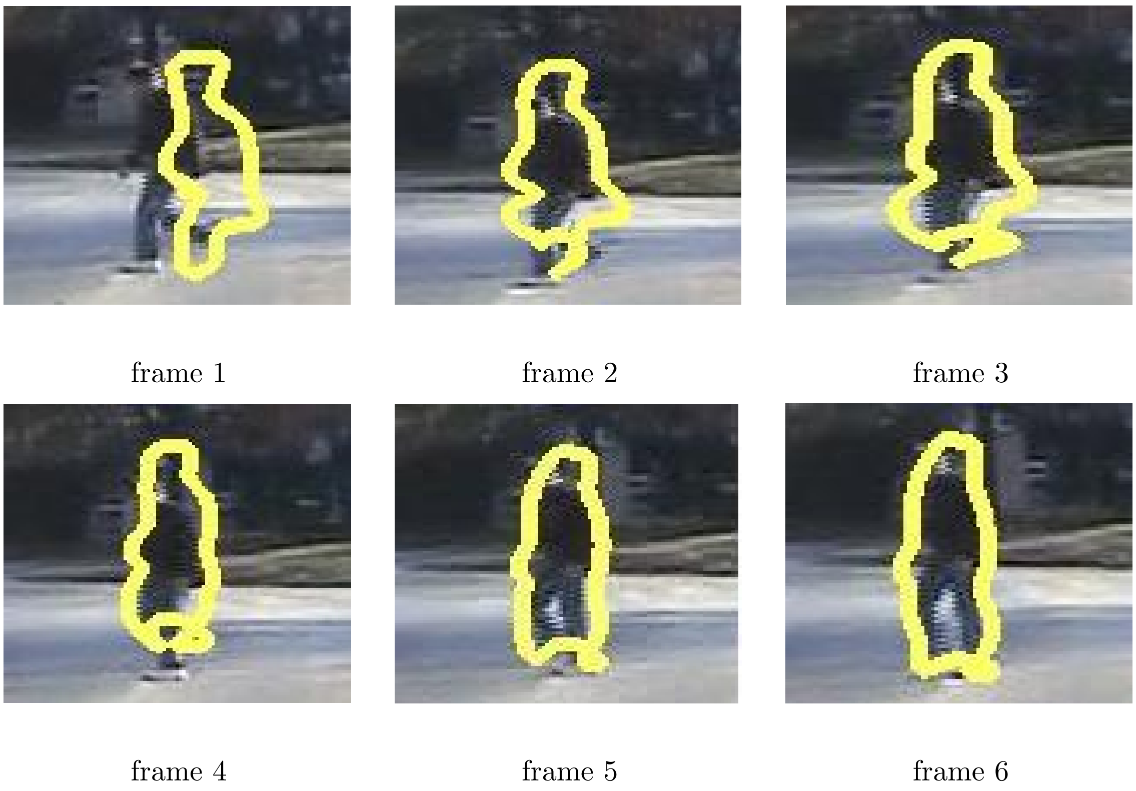


Figure 3.9: *An example of starting tracking using a very rough initial contour. After tracking for six frames, we find that the contour has been attached to the object fairly precisely.*

as demonstrated in Figure 3.9.

3.6.2 Experimental Results

We have applied the cascading contour tracker to different sets of outdoor surveillance video sequences, containing moving people and vehicles. All the objects of interests are assumed to be objects moving in non-rigid forms. In most cases, the backgrounds contain clutter that significantly affect the tracker's performance. Among them, four sets of sequences were captured by moving cameras,

Table 3.3: Algorithm 3: Active Contour Tracking

```
Step1. Initialization: Draw a set of particles from the prior $p(\theta_0)$ to obtain $\{\theta_0^{(j)}, \omega_0^{(j)}\}, j = 1, \ldots, J$, where $J$ is the number of the particles. Obtain the initial control point set $\{\mathbf{Q}_0^{(j)}\}$ from $\theta_0$. Set $t = 1$.
Step2. Global motion estimation:
Step2.1 Prediction: Estimate the state vector shift $\nu_t$, draw particles $\{\theta_t^{(j)}\}$ and
    accordingly the control point set samples $\{\mathbf{Q}_t^{(j)}\}, j = 1, \ldots, J$.
Step2.2 Update: Calculate the likelihood function $\mathcal{L}(Y_t|\theta_t^{(j)})$ and the posterior
    $\pi_t^{(j)} = p(\theta_t^{(j)}|Y_{1:t})$ for each sample, then normalize $\{\pi_t^{(j)}\}$ and update
    $\{\theta_t^{(j)}, \omega_t^{(j)}\}, j = 1, \ldots, J$. Find the MAP estimator of the global motion
    $\hat{\theta}_t = \theta_t^{\arg\max_{j \in \{1,\ldots,J\}} \pi_t^{(j)}}$ and the corresponding $\hat{\mathbf{Q}}_t$.
Step3. Local Deformation Estimation: Based on the estimated $\hat{\mathbf{Q}}_t$, generate the deformation probability map $P_t$. The deformed contour $\hat{\mathbf{r}}_t$ can be determined by scanning the adaptive normal lines $\mathbf{n}_{l,t}$ for pixels with maximum deformation probabilities.
Step4. Regulation: Project $\hat{\mathbf{r}}_t$ onto the shape subspace to acquire the final estimated contour.
Step5. $t \to t + 1$, go to step.2.
```

which means that the foreground cue is not involved when computing the probability maps. The other two sets of sequences were captured using static cameras, therefore the foreground cue is involved when generating the deformation probability map. The processing speed of the algorithm implemented in c++ is 6-10 fps (frames per second) on a 1.5GHz windows PC. In our experiments, most comparisons are made between our method and the traditional method described in [9].

3.6.2.1 Experiment on Stationary Camera Data

Figure 3.10 shows a sequence (the frame size is $437\times90\times3$) acquired by a stationary camera. It contains a pedestrian walking in the scene. The traditional active contour tracker works poorly due to two possible reasons: (1) the normal lines on some of the control points do not detect the edge points; and (2) in some frames, the contour becomes intertwined. Our proposed method takes advantage of the adaptive normal-line strategy to avoid intertwined contour, and subspace projection to recover missing edge points. The proposed method achieves satisfactory result throughout 159 frames in which the object is presented.

3.6.2.2 Experiments on Moving Camera Data

Figures 3.11, 3.12, 3.13, and 3.14 demonstrate some typical results for moving camera sequences. Figure 3.11 shows a sequence (the frame size is $543\times814\times3$) capturing a moving SUV by a camera from following vehicle. Although the rear view of the vehicle is a rigid object, the surrounding disturbances and small view

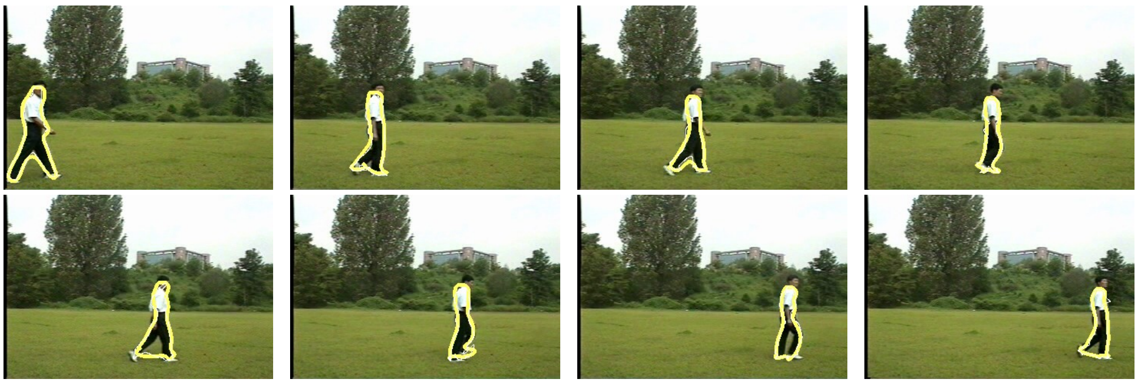


Figure 3.10: *Pedestrian tracking performance with a stationary camera. The thick yellow lines represent the final tracked contours.*

changes throughout the video lead to failures when using the traditional rigid-object trackers. Two major distractions derive from plain road marks and the sudden appearance of another vehicle in front of our target. Still, we obtain good results due to the use of a deformation map based on several statistical cues.

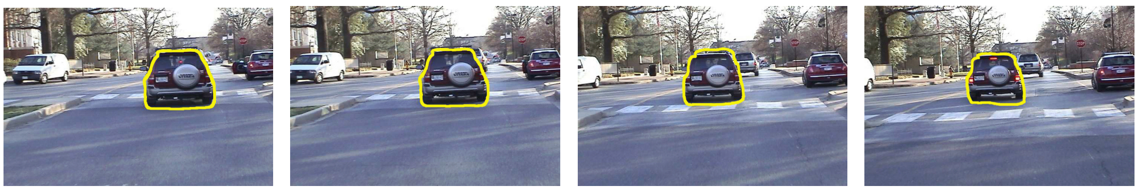


Figure 3.11: *The results of tracking a vehicle from the rear view. The comparison result is given in Figure 3.5.*

Figures 3.12 and 3.13 illustrate two challenging sequences with pedestrians crossing the road. The frame size in both sequences is 541×818×3. Tracking difficulties arise from the following facts: 1) the camera is moving forward rapidly and, therefore, the global motion of the pedestrian not only includes translation and rotation, but zoom as well; 2) the background is full of road marks and shadows. The traditional contour trackers become distracted by these background disturbances; 3) the pedestrian in Figure 3.12 is seen wearing a white shirt and black pants. The strong contrast between two parts usually leads to the tracking result shrinking to either the upper part or the lower part of the body; and (4) the object has similar coloring, with respect to the background (woods) in Figure 3.13. Our tracker produces satisfactory results. We notice that the poses of the pedestrians vary significantly from frame to frame in both sequences. However, the tracking remains robust.

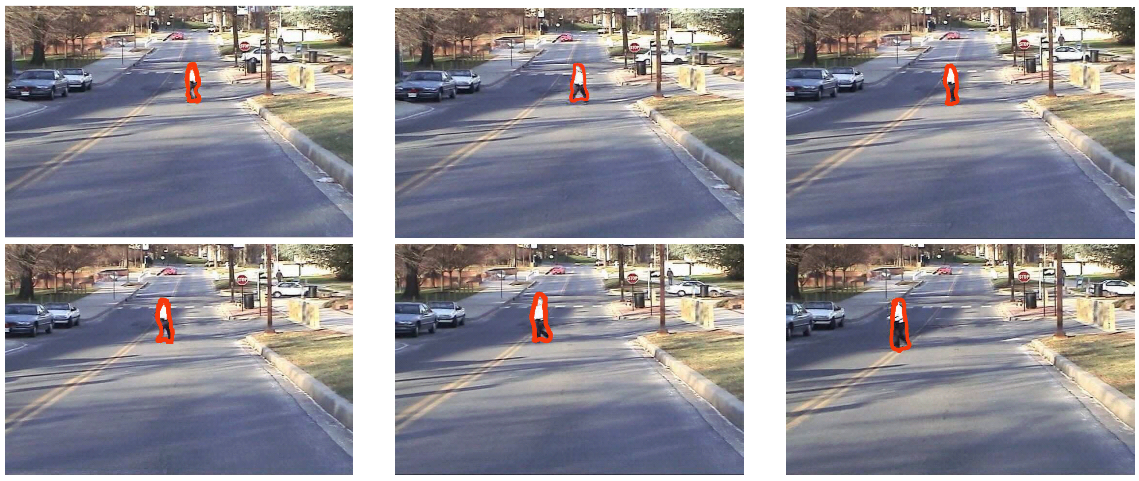
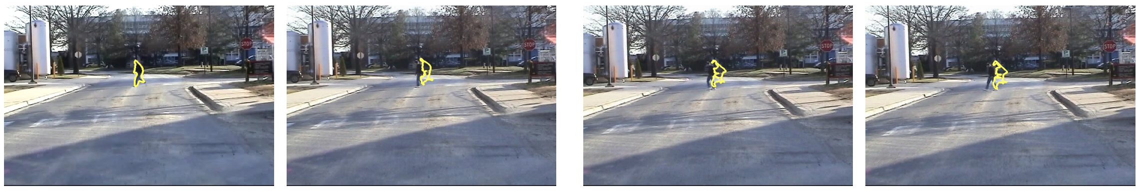
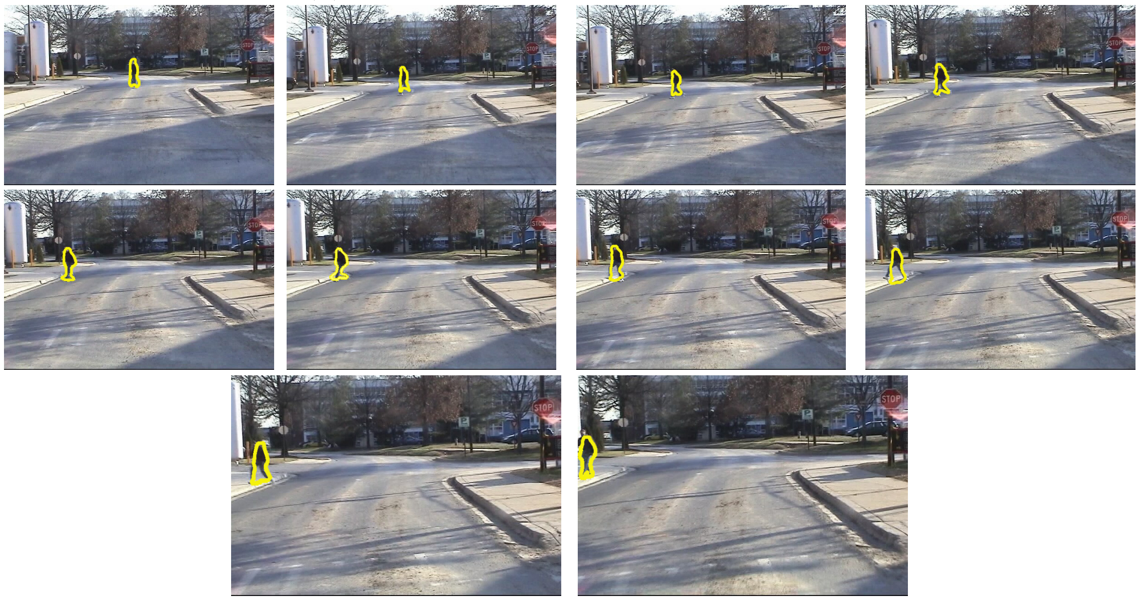


Figure 3.12: *A sequence with both background and object moving. The challenges of processing this sequence are due to: 1) camera motion; 2) background clutter; 3) the pedestrian is wearing a shirt and pants that vary strongly in color.*

Figure 3.14 provides an example with a sequence containing a moving truck (the frame size is 480×720×3). Although usually a truck cannot be treated as a non-rigid object, we still observe 2D shape deformation on the truck because of its 3D rotation. This sequence demonstrates the advantage of using contour-



(1) Results from the traditional algorithm



(2) Results from the proposed algorithm

Figure 3.13: *A sequence with both background and object moving. The background is heavily cluttered, and the intensity of the tracking pedestrian is very similar to the background color.*

based tracking. The truck in the sequence changes views from the back to the side, which means that some of the object's corresponding points will disappear. Such a view change will significantly undermine the result of a regular 2D appearance-based tracker. As mentioned in section 3.1, contour-based tracking can still produce robust results without using 3D models. The results appear promising even with the presence of obscure boundaries, low color contrast, nonstationary camera, and background clutter.

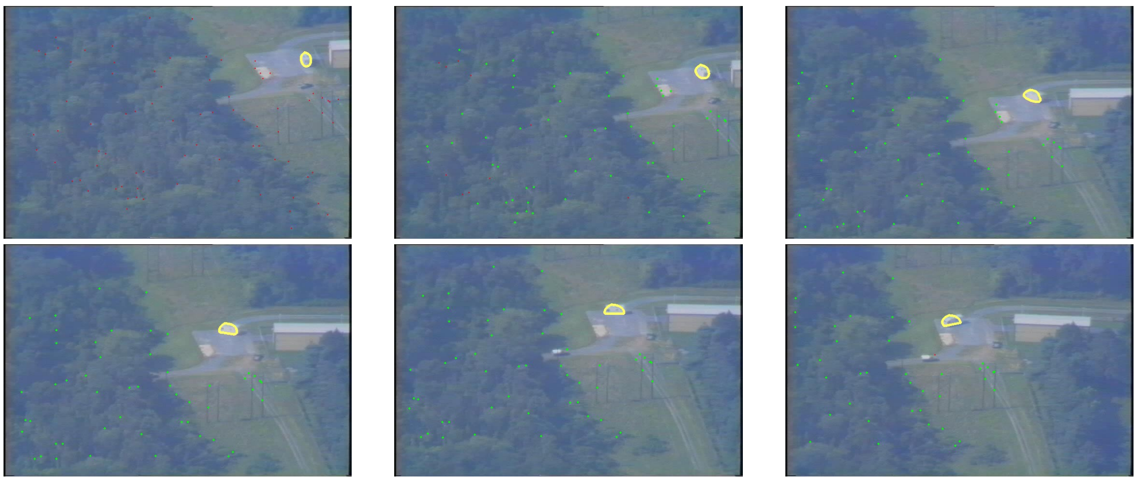


Figure 3.14: *An airborne sequence with a white truck moving on the ground. The truck shows a back view in frames (1) and (2), then shows the back-right view in frame (3), a side view in (4) and (5) and a front-right view in frame (6). This experiment demonstrates that the contour-based tracker can provide satisfactory results on sequences containing 3D object rotations.*

3.6.2.3 Experiments on Occlusion Data

We exploit the application of shape regulation to recover occluded contours, the results of which are shown in Figure 3.15. This sequence (the frame size is 576×768×3) contains walking people acquired by a stationary camera. In the last several frames, the pedestrian is partially occluded by trees. With subspace projection, we see that the occluded parts have been reconstructed. In this sequence, the clutter is heavy given the presence of parked cars and trees in the scene.

3.6.2.4 Experiments on medical sequence

The MRI sequence in our experiment consists of 2D image slices that form a 3D image cube for subsequent 3D visualization, in which we are particularly interested

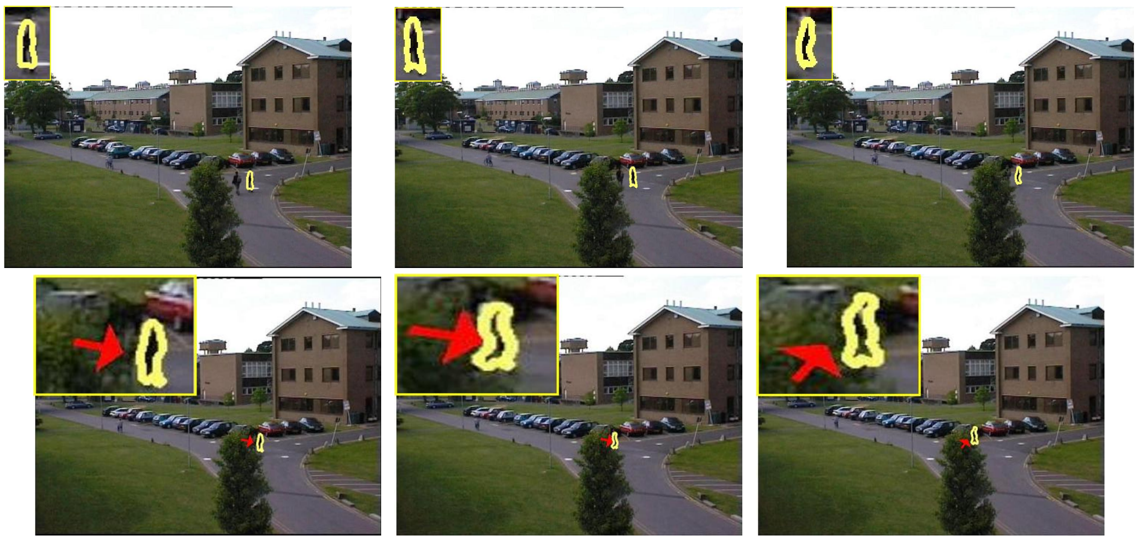


Figure 3.15: *An example of an occluded contour being recovered by shape subspace projection. In the last three frames, the pedestrian is partially occluded by surrounding trees. Our tracking result recovers the partially occluded contour. Red arrows indicate the occluded parts. We also note that parked cars contribute to background clutter.*

in the articular cartilage layer (a thin, white, crescent-like layer). The difficulties of tracking these layers are due to the following facts: the surrounding tissues often have similar intensity values, which leads to some boundary points becoming nearly undetectable; and the sequence is of low resolution because of the preprocessing step that enlarges the original image. The traditional active contour method is not effective in this case, because it lacks the means to handle edge point occlusion. The "snake" method also has difficulty in finding the correct boundary due to similar intensity values between the cartilage layer and the surrounding tissues. Our method works well in this scenario. Figure 3.16 demonstrates the tracking results of applying the proposed algorithm to the MRI sequence, in which the frame size is 584×584. As a comparison, we also provide a set of tracking results using the traditional contour tracker.

3.6.2.5 Performance Evaluation

We use the Mean Sum of Squared Distance (MSSD) [78] measure to evaluate the tracking performance of different algorithms. For a sequence with K frames, where contour $\mathbf{r}_k$ in each frame has L control points, $\{(x_{k,1}, y_{k,1}), \ldots, (x_{k,m}, y_{k,m})\}$, we define:

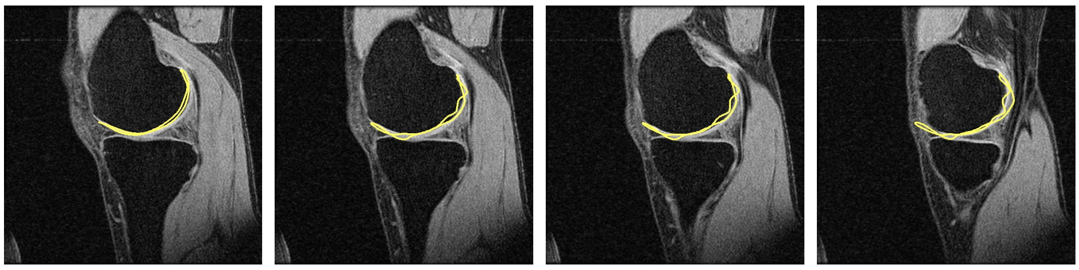
$$MSSD = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{L}\sum_{j=1}^{L}(x_{k,j} - x^0_{k,j})^2 + (y_{k,j} - y^0_{k,j})^2 \tag{3.35}$$

where $[x^0, y^0]$ represents the corresponding ground truth. We compare the proposed algorithm with the active contour tracking method for two cases: sequences with heavily-cluttered backgrounds and sequences with strong non-rigid movements. Table 3.4 shows the values of MSSD and variance of squared distance in these two cases, respectively. From all the experimental results and the comparison table, we conclude that the cascading tracker performs well in most cases.

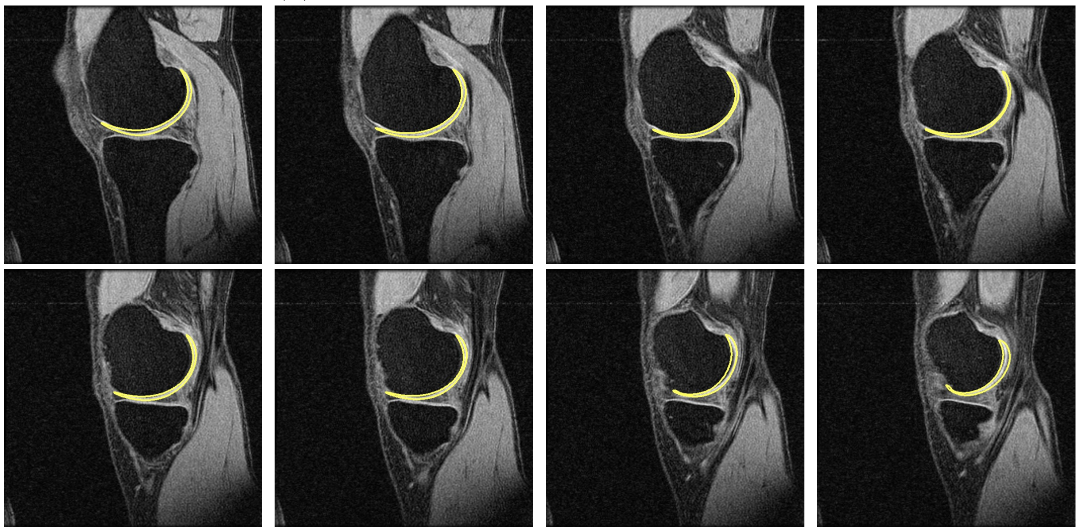
3.7 Discussion

The cascading contour tracker we have presented is motivated by the fact that the non-rigid movement can be decomposed as global motion and local deformation. The algorithm contains three major steps: motion estimation, deformation estimation and shape regulation. The following discussion covers the major aspects of the algorithm.

Multi-step vs. Single-step Compared with most methods that use single-



(1) Results from the traditional algorithm



(2) Results from the proposed algorithm

Figure 3.16: *Magnetic resonance imaging scans of human knees. The area of interest is the articular cartilage in each image. The upper row demonstrates the results by applying the traditional algorithm. The middle and bottom rows show the results by applying our proposed method.*

step estimation [9, 51, 59] to obtain the non-rigid contour movement with motion and deformation simultaneously, we choose a cascading framework to estimate motion and deformation separately. One-step estimation presents more systematic formulas, but it suffers from high dimensional computation and poor efficiency. Fortunately, the multi-step approach provides an efficient solution. This can be explained as follows: we use an affine transform to model the global motion. Therefore, the

Table 3.4: Comparison of tracking results in heavy-cluttered background sequences and non-rigid object sequences. MSSD is the Mean Sum of Squared Distance; VAR is the variance of the distances; ACT is the traditional active contour tracking algorithm.

	ACT		Cascading Tracker	
	MSSD	VAR	MSSD	VAR
heavy-cluttered background	14.7584	74.4052	7.9129	12.5602
non-rigid object	9.9741	24.6597	6.0571	8.5222

motion state vector is only six-dimensional, which eases the burden on the particle filter. We interpret shape deformation by the deformations that occurred on control points, which approximates the infinite dimensional space by a finite dimensional one. Since the processing occurs in a successive manner and a rough contour is obtained by motion estimation, the deformation is searched only in the region of the rough contour and does not involve numerical simulations. Thus, the entire computation complexity is reduced.

Multi-cue vs. Single-cue Although gradient magnitude provides a strong cue to estimate the boundary pixels, sometimes it is not robust. In our method, estimating deformation counts on more visual cues, with the aim to realize more robust contour detection. We could further improve the fusion method by associating adaptive fusion weights with different cues [48].

Adaptive vs. Non-adaptive Normal Line We adaptively set the scanning normal lines according to the prior shape pose variations and previous shape estimate. The advantages of adaptability are: (1) it reduces the probability of loop occurrence; and (2) setting appropriate lengths of normal lines increases the possibility of correct detection for real boundary pixels, while reducing the chance of false detection.

Regulation vs. Non-regulation Regulation is important for non-rigid contour tracking. It not only constrains shape deformation and corrects estimating errors, but recovers the occluding contour pixels as well. Therefore, the method applies more to scenarios with non-rigid object movements and heavily-cluttered backgrounds.

Our method can successfully track non-rigid objects and provide tight contours enclosing the changing shapes of the targets throughout the sequences. We are currently working on extensions of the algorithm to the multi-target tracking problem. Model regulation is also worth further exploration. We only use shape prior knowledge as a constraint in this chapter. However, training samples contain not only shape, but appearance and motion information as well. It is expected that constructing a prior model based on all information should improve the robustness of trackers, which is discussed in the next session.

Chapter 4

Regulation in Object Tracking: Active Appearance Markov Chain (AAMC) Model

4.1 Introduction

As discussed in the previous chapter, visual tracking is an essential component in many computer vision fields. Among them, many require accurate delineation of moving objects, such as in surveillance and medical sequence tracking. Therefore, contour tracking, i.e., localizing contours or boundaries of objects during tracking, has attracted significant interest. The main challenges in contour tracking can be attributed to handling all types of variations of a target object, which highly influences the system performance. As a solution, constraints represented by subspaces are usually applied for robustness. Most of the typical contour tracking algorithms [79, 80, 81, 82, 83, 76] consist of two parts: an observation process for estimation and a fusion process (regularization) for ensuring robustness. The observation process estimates the kinematic motion for individual landmark points and the fusion process regularizes the estimated contour using the observation process under certain constraints. Fig. 4.1(a) presents a schematic illustration of such a system.

Many efforts relate to regularization in tracking, using shape or appearance constraints, or a combination of both. However, few exist where the regularization method has integrated shape, appearance and motion knowledge. In this chapter, we introduce the active appearance Markov chain (AAMC) model to capture statistical regularities in shape, appearance and motion. The model has two-parts: the shape and appearance knowledge is learned from training samples using the technique inherited from the widely applied active appearance model (AAM); and the motion information between frames is modeled using a discrete Markov chain. The model is illustrated in 4.1 (b). The necessary terminology is given below:

Shape. The shape (or contour) of the object is parameterized by a set of $M_{\mathcal{S}}$ landmark points. We denote the shape by the vector $\mathcal{S} = [\mathtt{x}_1, \mathtt{x}_2, \ldots, \mathtt{x}_{M_{\mathcal{S}}}]^{\mathsf{T}}$, where $\mathtt{x}_m = (x_m, y_m)$ is the m^{th} landmark point. The shape is aligned to the mean shape using a similarity transformation to reduce variations caused by translation, rotation and scaling.

Appearance. For a video frame with shape $\mathcal{S}$, we generate a *shape-free* global appearance patch vector $\mathcal{G}$ by warping the video frame to a canonical template where the landmarks are located at fixed positions, such as in the mean shape.

Motion. There are two kinds of motions. The first is the motion of a moving platform. The second is the motion of the landmark points. The model is concerned with the latter motion.

4.1.1 Previous Work

Model constraints have been exploited as the prior knowledge in many contour-based tracking approaches. The active shape models (ASM) [84] have been used by

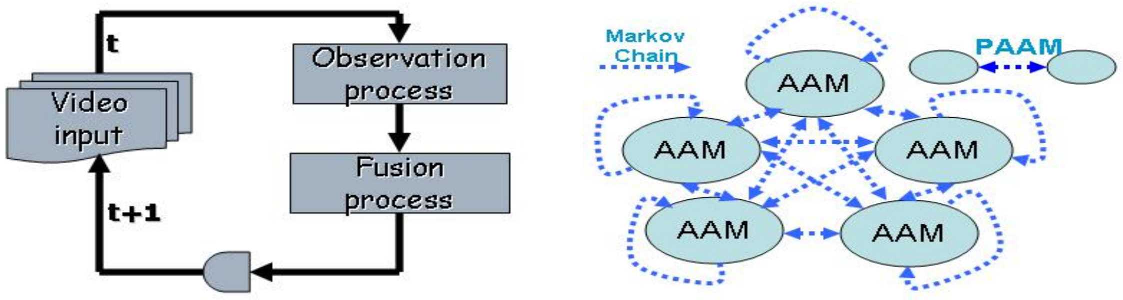


Figure 4.1: *(a) Illustration of a tracking system. (b) Illustration of AAMC model.*

Blake and Isard in [80] as shape subspace constraints in their particle filter algorithm to improve the robustness of the tracker. The ASM was used by Zhou *et al.* [76] in a fusion framework, enhanced by an online adaptation step to adapt the shape model offline trained to the current observation. As an extension of ASM, the AAM was introduced by Cootes *et al.* in [85]. It covers the variabilities in both image appearance and object shape exemplified by a set of training samples. AAM was further extended to 2D+3D models for non-rigid face tracking in [86]. Provided that the training data represents the variability in the population, AAM can generate any statistical intermediates that may occur during tracking, thus improving its robustness. An adaptive view-based appearance model was proposed by Morency *et al.* [87] for rigid object tracking. The view-based model is adjusted online by registering each frame against the views of appearance models. The performance of the algorithm on non-rigid object tracking is limited. All these approaches lack the ability to describe dynamic motion throughout the video sequences. When dealing with structured movements, such as the pedestrian and cardiac motion, it is crucial to characterize the inter-phase motion information.

The active appearance motion model (AAMM) [88] was proposed by Bosch *et al.* to characterize shape, appearance and motion for echocardiography segmentation, where a subspace model for an entire movement cycle (a full cardiac cycle) is learned. However, the AAMM suffers from three problems. First, it is suitable to segment a spatiotemporal target, but it is not amenable for online tracking tasks. Second, the AAMM cannot handle global movement between frames. Third, it lacks adaptability to different cases since it falls in the "observation explains model" category. Finally, the AAMM is in high-dimensional space. Consequently, it is difficult to collect sufficient data to cover desired variations, resulting in ineffective modeling. More expensive computation is inevitable in both training and testing. A more efficient P-AAM model, integrating shape, appearance and motion, was proposed in [89], targeting echocardiography tracking based on the periodicity of heart motions.

4.2 Active Appearance Markov Chain Model

Contour tracking algorithms locate object contours in video sequences. For certain type of objects, their shape, appearance and movement pattern bear some similarities. Clearly, a statistical model that captures the typical variations in shape, appearance and inter-frame motion from sample dataset helps increase the robustness of the tracking system. The AAMC model is generated for this purpose.

Definition 1. *Any movement can be divided into several stages. We define these stages as* ***motion phases****. If movements of similar objects share similar mo-*

tion patterns, the shape and appearance of these objects from same ***motion phases*** *and the inter-phase motions between* ***motion phases*** *bear some similarities.*

Given a set of sample sequences, we divide all sample frames into P motion phases, each of which is represented by shape and appearance models. A Markov chain model represents the dynamic transitions between these motion phases. Hence, any pair of motion phases has a transition probability in the Markov chain.

Definition 2. *If the transition probability between two motion phases is non-zero, we call the pair of motion phases a* ***connected pair****. Otherwise, we call the pair a* ***non-connected pair****.*

We then construct the joint shape and appearance models for all connected pairs. The joint model not only covers the shape and appearance variations, but also the inter-phase motion.

Definition 3. *The joint model for shape, appearance and inter-phase motion variations is called the* ***pairwise AAM (P-AAM) component****.*

Therefore, the AAMC model includes two parts: (1) the switching of motion phases is represented by a Markov chain; and (2) the shape and appearance information and the inter-phase motion are represented by subspaces of several P-AAM components. Some necessary notations include: z_t, denoting the index of one motion phase; $\mathbf{x}_t$, denoting the final estimate of shape and appearance at time t, $\mathbf{x}_t = [\mathbf{x}_{t,\mathcal{S}}^{\mathsf{T}}, \mathbf{x}_{t,\mathcal{G}}^{\mathsf{T}}]^{\mathsf{T}}$; and y_t, denoting the output from the observation process, $\mathsf{y}_t = [\mathsf{y}_{t,\mathcal{S}}^{\mathsf{T}}, \mathsf{y}_{t,\mathcal{G}}^{\mathsf{T}}]^{\mathsf{T}}$.

4.2.1 Markov chain: model switching

The dynamic motion between motion phases is modeled as a discrete Markov chain [90]. A discrete Markov chain is a probabilistic model of a random process with discrete states. In a first order Markov model, the state of the system at time t can be predicted knowing only the state at time $t-1$. We assume that the Markov chain is time-homogeneous, i.e., the transition probability $P(\mathsf{z}_t = j|\mathsf{z}_{t-1} = i) = P_{ij}$ does not depend on time and can be learned off-line from training samples. The phase switching probability is defined as $Pr_{\mathcal{MC}}(\mathsf{z}_t) = P(\mathsf{z}_t|\mathsf{y}_t, \mathsf{z}_{t-1}, \mathbf{x}_{t-1})$, which measures the probability of one AAM model being associated with the current frame. We assume the following:

$$P(\mathsf{z}_t|\mathsf{y}_t, \mathbf{x}_{t-1}, \mathsf{z}_{t-1}) = P(\mathsf{z}_t|\mathsf{z}_{t-1}), \tag{4.1}$$

which implies that the current observation y_t and the previous shape and appearance state $\mathbf{x}_{t-1}$ do not influence the transition probability.

4.2.2 P-AAM components

The P-AAM component is a joint model describing the shape, appearance and motion variations between two connected motion phases. For each connected pair of motion phases, we extract all pairwise samples from training datasets that consist of two consecutive frames, each belonging to one of motion phases in the connected pair, respectively. We then generate two sets of data for shape and appearance, respectively. All the shapes are represented by $M_{\mathcal{S}}$ landmark points, forming a $2M_{\mathcal{S}}$-dimensional vector $\mathcal{S}$. The appearance is represented by an $M_{\mathcal{G}}$-dimensional vector $\mathcal{G}$. We concatenate the shape and appearance vectors from two consecutive motion phases to form a paired data:

$$\mathcal{S}_{ij} = [\mathcal{S}_i^{\mathsf{T}} \mid \mathcal{S}_j^{\mathsf{T}}]^{\mathsf{T}}, \;\; \mathcal{G}_{ij} = [\mathcal{G}_i^{\mathsf{T}} \mid \mathcal{G}_j^{\mathsf{T}}]^{\mathsf{T}}, \tag{4.2}$$

where $i, j \in \{1, 2, \ldots, P\}$ is the phase index.

Then the joint model is constructed using the standard approach for AAMs [85]:

(i) Construct the shape subspace based on $\mathbf{s_{ij}}$ using the principal component analysis (PCA). The subspace can be represented by:

$$\mathcal{S}_{ij} \approx \bar{\mathcal{S}}_{ij} + \mathbf{P}_{ij}^{\mathcal{S}} \mathbf{b}_{ij}^{\mathcal{S}}, \tag{4.3}$$

where $\mathbf{P}^{\mathcal{S}}$ is a subspace matrix (eigenvectors) describing a sufficient fraction of the total shape variation, and $\mathbf{b}^{\mathcal{S}}$ is a vector containing the combination coefficients for each of the eigenvectors. (ii) In a similar way, construct the appearance subspace based on $\mathcal{G}_{ij}$ using the PCA. The appearance patches are cropped from sample frames according to the corresponding shapes and aligned to a mean shape using the thin-plate splines warping algorithm [91]. This process is necessary to compensate for inter-frame translations, because it guarantees that the created model contains no motion introduced by external factors, such as translations and rotations of objects or movements of sensors.

$$\mathcal{G}_{ij} \approx \bar{\mathcal{G}}_{ij} + \mathbf{P}_{ij}^{\mathcal{G}} \mathbf{b}_{ij}^{\mathcal{G}}. \tag{4.4}$$

(iii) Apply a third PCA to the combination of the shape and appearance:

$$\mathbf{b}_{ij} = \begin{bmatrix} \mathbf{b}_{ij}^{\mathcal{S}} \\ \mathbf{w}_{ij}^{\mathcal{G}} \mathbf{b}_{ij}^{\mathcal{G}} \end{bmatrix} \approx \mathbf{Q}_{ij} \mathbf{c}_{ij} = \begin{bmatrix} \mathbf{Q}_{ij}^{\mathcal{S}} \\ \mathbf{Q}_{ij}^{\mathcal{G}} \end{bmatrix} \mathbf{c}_{ij}, \tag{4.5}$$

where $\mathbf{w}_{ij}^{\mathcal{G}}$ is a diagonal matrix that balances the energy discrepancy between the shape and appearance models, $\mathbf{Q}$ is the eigenvector matrix, and $\mathbf{c}$ is a latent vector that controls both shape and appearance models. Substituting Eq. (4.5) into Eqs. (4.3) and (4.4) yields

$$\mathcal{S}_{ij} \approx \bar{\mathcal{S}}_{ij} + \mathbf{P}_{ij}^{\mathcal{S}} \mathbf{Q}_{ij}^{\mathcal{S}} \mathbf{c}, \tag{4.6}$$

$$\mathcal{G}_{ij} \approx \bar{\mathcal{G}}_{ij} + \mathbf{P}_{ij}^{\mathcal{G}} (\mathbf{w}_{ij}^{\mathcal{G}})^{-1} \mathbf{Q}_{ij}^{\mathcal{G}} \mathbf{c}. \tag{4.7}$$

The number of P-AAM components is identical to the number of connected pairs in the Markov chain model.

4.3 Fusion Process Using AAMC Model

The system of contour tracking can be interpreted statistical by: the state vectors of the system include the shape and appearance estimates $\mathbf{x} = [\mathbf{x}_{\mathcal{S}}^{\mathsf{T}}, \mathbf{x}_{\mathcal{G}}^{\mathsf{T}}]^{\mathsf{T}}$ and the Markov chain index in the AAMC model z. At time instant t, we want to find the current estimate that maximizes:

$$P(\mathbf{x}_t, \mathsf{z}_t | \mathbf{x}_{t-1}, \mathsf{z}_{t-1}) = P(\mathbf{x}_{t,\mathcal{S}}, \mathbf{x}_{t,\mathcal{G}}, \mathsf{z}_t | \mathbf{x}_{t-1,\mathcal{S}}, \mathbf{x}_{t-1,\mathcal{G}}, \mathsf{z}_{t-1}) \tag{4.8}$$

The fusion process has a goal of combining observation (output from the observation process) and prior knowledge (the trained AAMC model) to obtain an optimal estimate. Therefore, we have one additional condition to our formula, the observation y_t. Accordingly, Eq. (4.8) can be reformulated as:

$$\begin{aligned} &P(\mathbf{x}_t, \mathsf{z}_t | \mathsf{y}_t, \mathbf{x}_{t-1}, \mathsf{z}_{t-1}) \\ &\quad = P(\mathbf{x}_t | \mathsf{y}_t, \mathsf{z}_t, \mathbf{x}_{t-1}, \mathsf{z}_{t-1}) P(\mathsf{z}_t | \mathsf{y}_t, \mathbf{x}_{t-1}, \mathsf{z}_{t-1}) \end{aligned} \tag{4.9}$$

Eq.(4.9) implies that the fusion process actually includes two steps: (1) select the optimal Markov chain index $\mathbf{z}$ in the AAMC model; and (2) estimate the optimal $\mathbf{x}$ based on the observation and the AAMC model. The optimization problem is reduce to

$$\begin{aligned}[\mathbf{x}_t^*, \mathbf{z}_t^*] &= \arg\max_{\mathbf{x},\mathbf{z}} P(\mathbf{x}_t, \mathbf{z}_t | \mathrm{y}_t, \mathbf{x}_{t-1}, \mathbf{z}_{t-1}) \\ &= \arg\max_{\mathbf{x},\mathbf{z}} \log P(\mathbf{x}_t | \mathrm{y}_t, \mathbf{z}_t, \mathbf{x}_{t-1}, \mathbf{z}_{t-1}) + \log Pr_{\mathcal{MC}}(\mathbf{z}_t) \qquad (4.10)\end{aligned}$$

The second item in Eq. (4.9) is the model switching probability in the Markov chain model, as discussed in Sec.4.2.1. The first item in Eq.(4.10) is measured following the work by Zhou *et al.* [76]. A fusion cost is introduced for estimating the probability, which is composed as the sum of two Mahalanobis distance squares from the observation and prior model:

$$d^2_{t|\mathrm{y}_t, \mathbf{z}_t, \mathbf{x}_{t-1}, \mathbf{z}_{t-1}} = d^2_{t|\mathrm{y}_t, \mathbf{x}_{t-1}} + d^2_{t|\mathbf{z}_t, \mathbf{x}_{t-1}, \mathbf{z}_{t-1}} \qquad (4.11)$$

where

$$d^2_{t|\bullet} = (\mathbf{x}_t - \mathbf{x}_{t|\bullet})^{\mathsf{T}} \mathbb{C}^{-1}_{t|\bullet} (\mathbf{x}_t - \mathbf{x}_{t|\bullet}) \qquad (4.12)$$

Using this mechanism has the advantage that the best estimate is generated from the uncertainties of the two sources (the observation and prior model), which are incorporated into the system in the form of their covariances.

$P(\mathbf{x}_t | \mathrm{y}_t, \mathbf{z}_t, \mathbf{x}_{t-1}, \mathbf{z}_{t-1})$ can then be evaluated depending on the fusion cost function defined in Eqs.(4.11) and (4.12) in an inverse proportional form. Eq.(4.10) becomes:

$$\begin{aligned}[\mathbf{x}_t^*, \mathbf{z}_t^*] &= \arg\max_{\mathbf{x},\mathbf{z}} \ \log Pr_{\mathcal{MC}}(\mathbf{z}_t) - d^2_{t|\mathrm{y}_t, \mathbf{z}_t, \mathbf{x}_{t-1}, \mathbf{z}_{t-1}} \\ &= \arg\min_{\mathbf{x},\mathbf{z}} \ d^2_{t|\mathrm{y}_t, \mathbf{z}_t, \mathbf{x}_{t-1}, \mathbf{z}_{t-1}} - \log Pr_{\mathcal{MC}}(\mathbf{z}_t) \qquad (4.13)\end{aligned}$$

We note that $\mathtt{z}_t$ is a discrete random variable with limited states. Therefore, we can achieve the minimum of Eq. (4.13) in two steps: (1) with each fixed $\mathtt{z}_t$, estimate the minimum values of the objective functions and the corresponding $\mathbf{x}_t^*(\mathtt{z}_t)$; (2) select the global optimal estimate $\mathbf{x}_t^*$ and $\mathtt{z}_t^*$ with the global minimum value. The entire model can be illustrated as a graphic model in Fig.4.2 (a).

4.3.1 Component $\mathbf{d}^2_{t|\mathtt{y}_t,\mathbf{x}_{t-1}}$

The first distance arises from the observation source. The mean vector $\mathbf{x}_{t|\mathtt{y}_t,\mathbf{x}_{t-1}} = [\mathtt{y}_{t,\mathcal{S}}, \mathtt{y}_{t,\mathcal{G}}]$ is directly derived from the observation process at the current frame. The covariance matrix $\mathbb{C}_{t|\mathtt{y}_t,\mathbf{x}_{t-1}}$, which reflects the uncertainty of the observation, is given as:

$$\mathbb{C}_{t|\mathtt{y}_t,\mathbf{x}_{t-1}} = \begin{bmatrix} \mathtt{R}_{\mathcal{S}} & \mathbf{0} \\ \mathbf{0} & \mathtt{R}_{\mathcal{G}} \end{bmatrix} \tag{4.14}$$

$$\mathtt{R}_{\mathcal{S}} = \eta \begin{bmatrix} \mathtt{C}_1 & & \\ & \dots & \\ & & \mathtt{C}_{M_{\mathcal{S}}} \end{bmatrix} \qquad \mathtt{R}_{\mathcal{G}} = \begin{bmatrix} \mathtt{B}_1 & & \\ & \dots & \\ & & \mathtt{B}_{M_{\mathcal{G}}} \end{bmatrix}$$

where η balances the energy between the shape and appearance parts in the observation process, and the 2×2 matrix $\mathtt{C}_i$ is the covariance matrix associated with the corresponding landmark point $\mathtt{x}_i$:

$$\mathtt{C}_i = \sigma^2 \left(\sum_{(x_m,y_n)\in W(x_i,y_i)} \begin{bmatrix} \frac{\partial^2 \mathbf{I}(x_m,y_n)}{\partial^2 x} & \frac{\partial^2 \mathbf{I}(x_m,y_n)}{\partial x \partial y} \\ \frac{\partial^2 \mathbf{I}(x_m,y_n)}{\partial x \partial y} & \frac{\partial^2 \mathbf{I}(x_m,y_n)}{\partial^2 y} \end{bmatrix} \right)^{-1}. \tag{4.15}$$

$\mathbf{I}$ is the image intensity function, and $W(x_i, y_i)$ represents a local area centered on $W(x_i, y_i)$. The variance for the appearance observation $R_{\mathcal{G}}$, is a diagonal matrix with the diagonal entries set to be the square of the appearance difference between the current observation and the previous one, we have

$$\mathtt{B}_i = \mathcal{F} \otimes \|\mathtt{y}^i_{t,\mathcal{G}} - \mathbf{x}^i_{t-1,\mathcal{G}}\|^2 \tag{4.16}$$

where $\mathcal{F}$ is a smoothing filter for robustness.

4.3.2 Component $\mathbf{d}^2_{t|\mathbf{z}_t,\mathbf{x}_{t-1},\mathbf{z}_{t-1}}$

The second distance relates to the P-AAM components in the AAMC model, in the sense that the mean vector $\mathbf{x}_{t|\mathbf{z}_t,\mathbf{x}_{t-1},\mathbf{z}_{t-1}}$ and the covariance matrix $\mathbb{C}_{t|\mathbf{z}_t,\mathbf{x}_{t-1},\mathbf{z}_{t-1}}$ are derived from the P-AAM component. For a P-AAM component of motion phases i and j, its joint distribution $p(\mathcal{S}_i, \mathcal{S}_j, \mathcal{G}_i, \mathcal{G}_j)$ is Gaussian, whose mean and covariance matrix are listed as:

$$\begin{aligned}
\mu_{ij} &= \begin{bmatrix} \mu^{\mathsf{T}}_{\mathcal{S}_i} & \mu^{\mathsf{T}}_{\mathcal{S}_j} & \mu^{\mathsf{T}}_{\mathcal{G}_i} & \mu^{\mathsf{T}}_{\mathcal{G}_j} \end{bmatrix}^{\mathsf{T}}, \\
\mathbf{\Phi}_{ij} &= \begin{bmatrix} \mathbf{\Phi}^{<\mathcal{S}_i,\mathcal{S}_i>} & \mathbf{\Phi}^{<\mathcal{S}_i,\mathcal{S}_j>} & \mathbf{\Phi}^{<\mathcal{S}_i,\mathcal{G}_i>} & \mathbf{\Phi}^{<\mathcal{S}_i,\mathcal{G}_j>} \\ \mathbf{\Phi}^{<\mathcal{S}_j,\mathcal{S}_i>} & \mathbf{\Phi}^{<\mathcal{S}_j,\mathcal{S}_j>} & \mathbf{\Phi}^{<\mathcal{S}_j,\mathcal{G}_i>} & \mathbf{\Phi}^{<\mathcal{S}_j,\mathcal{G}_j>} \\ \mathbf{\Phi}^{<\mathcal{G}_i,\mathcal{S}_i>} & \mathbf{\Phi}^{<\mathcal{G}_i,\mathcal{S}_j>} & \mathbf{\Phi}^{<\mathcal{G}_i,\mathcal{G}_j>} & \mathbf{\Phi}^{<\mathcal{G}_i,\mathcal{G}_j>} \\ \mathbf{\Phi}^{<\mathcal{G}_j,\mathcal{S}_i>} & \mathbf{\Phi}^{<\mathcal{G}_j,\mathcal{S}_j>} & \mathbf{\Phi}^{<\mathcal{G}_j,\mathcal{G}_i>} & \mathbf{\Phi}^{<\mathcal{G}_j,\mathcal{G}_j>} \end{bmatrix}.
\end{aligned} \tag{4.17}$$

Two different cases should be considered: (1) for $i \neq j$: $p(\mathcal{S}_i, \mathcal{G}_i | \mathcal{S}_j, \mathcal{G}_j)$ is the conditional probability and yields a Gaussian distribution, since the joint distribution is Gaussian. We derive the mean vector and covariance matrix as:

$$\begin{aligned}
\mathbf{x}_{t|\mathbf{z}_t,\mathbf{x}_{t-1},\mathbf{z}_{t-1}} &= \mu_{\mathbf{z}_t|\mathbf{z}_{t-1},\mathbf{x}_{t-1}} = \mu_{i|j} \\
&= \mu_i + \Sigma_{ij}\Sigma^{-1}_{jj}(\mathbf{x}_{t-1} - \mu_j) \qquad (4.18) \\
\mathbb{C}_{t|\mathbf{z}_t,\mathbf{x}_{t-1},\mathbf{z}_{t-1}} &= \Sigma_{\mathbf{z}_t|\mathbf{z}_{t-1}} = \Sigma_{i|j} = \Sigma_{ii} - \Sigma_{ij}\Sigma^{-1}_{jj}\Sigma^{\mathsf{T}}_{ji} \qquad (4.19)
\end{aligned}$$

where

$$\Sigma_{ij} = \begin{bmatrix} \mathbf{\Phi}^{<\mathcal{S}_i,\mathcal{S}_j>} & \mathbf{\Phi}^{<\mathcal{S}_i,\mathcal{G}_j>} \\ \mathbf{\Phi}^{<\mathcal{G}_i,\mathcal{S}_j>} & \mathbf{\Phi}^{<\mathcal{G}_i,\mathcal{G}_j>} \end{bmatrix}, \quad \mu_i = \begin{bmatrix} \mu_{\mathcal{S}_i} \\ \mu_{\mathcal{G}_i} \end{bmatrix}. \tag{4.20}$$

(2) for $i = j$: the model updating is similar to the prediction step of the Kalman filter. Denoting $\delta_{i,\mathbf{x}_{t-1}} = \mathbf{x}_{t-1} - \mu_i$, we have

$$\Sigma_{i|i,\mathbf{x}_{t-1}} = \Sigma_{ii} - \Sigma_{ii}\Sigma^{-1}_{\mathbf{x}_{t-1}|i}\Sigma_{ii} \tag{4.21}$$

$$\mu_{i|i,\mathbf{x}_{t-1}} = \mu_i + \Sigma_{ii}\Sigma^{-1}_{\mathbf{x}_{t-1}|i}(\delta_{i,\mathbf{x}_{t-1}}) \tag{4.22}$$

where $\Sigma_{\mathbf{x}_{t-1}|i}$ can be written as [87]:

$$\Sigma_{\mathbf{x}_{t-1}|i} = \frac{1}{\epsilon(\delta_{i,\mathbf{x}_{t-1}})}\frac{\partial^2\epsilon(\delta_{i,\mathbf{x}_{t-1}})}{\partial^2\delta_{i,\mathbf{x}_{t-1}}} \tag{4.23}$$

$$\epsilon(\delta_{i,\mathbf{x}_{t-1}}) = ||\mathbf{x}_{t-1} - \mu_i||^2 \tag{4.24}$$

From Eqs.(4.18) and (4.22), we observe that using the conditional probability $p(\mathcal{S}_i, \mathcal{G}_i|\mathcal{S}_j, \mathcal{G}_j)$ is beneficial because $\mathbf{x}_{t|\mathbf{z}_t,\mathbf{x}_{t-1},\mathbf{z}_{t-1}}$ always updates during the iterations, so is adaptive to previous observation $\mathbf{x}_{t-1}$. Meanwhile, Eqs. (4.19) and (4.21) show that the updated model subspaces lower the uncertainty in estimates with respect to the original model subspaces. The proof is straightforward: The elements along the variance matrix Σ are the marginal variances for the shape and appearance of one motion phase and their uncertainties. Because Σ_{ii}, $\Sigma_{ij}\Sigma^{-1}_{jj}\Sigma^{\mathsf{T}}_{ji}$ and $\Sigma_{ii}\Sigma^{-1}_{\mathbf{x}_{t-1}|i}\Sigma_{ii}$ are both positive definite matrices, the diagonal entries of $\Sigma_{\mathbf{z}_t|\mathbf{z}_{t-1}}$ are no larger than those of Σ_{ii} under both circumstances. Therefore, the uncertainty is non-decreasing and bounded during tracking.

4.3.3 The best estimate of $\mathbf{x}_t$ with fixed $\mathtt{z}_t$

With each fixed $\mathtt{z}_t$, the minimization of $d^2_{t|\mathtt{y}_t,\mathtt{z}_t,\mathbf{x}_{t-1},\mathtt{z}_{t-1}}$ can be solved by the best linear unbiased estimator (BLUE). We notice that in most of cases, $\mathbb{C}_{t|\mathtt{z}_t,\mathbf{x}_{t-1},\mathtt{z}_{t-1}}$ tends to be singular due to the high dimensionality of the shape and appearance vectors. Suppose the rank of the covariance matrix is $\mathtt{q}$ and its rank-$\mathtt{q}$ SVD is $\mathtt{U}_\mathtt{q}\Lambda_\mathtt{q}\mathtt{U}_\mathtt{q}^\mathsf{T}$, the original BLUE equation is altered to:

$$\mathbf{x}^*(\mathtt{z}_t) \quad = \quad \mathtt{U}_\mathtt{q}(\mathtt{U}_\mathtt{q}^\mathsf{T}\mathbb{C}_1^{-1}\mathtt{U}_\mathtt{q} + \Lambda_\mathtt{q}^{-1})^{-1}(\mathtt{U}_\mathtt{q}^\mathsf{T}\mathbb{C}_1^{-1}\mathbf{x}_1 + \Lambda_\mathtt{q}^{-1}\mathtt{U}_\mathtt{q}^\mathsf{T}\mathbf{x}_2). \tag{4.25}$$

where $\mathbf{x}_1$ denotes $\mathbf{x}(t|\mathtt{y}_t, \mathbf{x}_{t-1})$, and $\mathbf{x}_2$ denotes $\mathbf{x}(t|\mathtt{z}_t, \mathbf{x}_{t-1}, \mathtt{z}_{t-1})$. In fact, the equation represents a BLUE fusion operation of two subspaces, one being $\mathcal{N}(U_\mathtt{q}^\mathsf{T}\mathbf{x}_2, \Lambda_\mathtt{q})$, and the other being the nonorthogonal projection of $\mathcal{N}(\mathbf{x}_1, \mathbb{C}_1)$ in the subspace, which is $\mathcal{N}((\mathtt{U}_\mathtt{q}^\mathsf{T}\mathbb{C}_1^{-1}\mathtt{U}_\mathtt{q})^{-1}\mathtt{U}_\mathtt{q}^\mathsf{T}\mathbb{C}_1^{-1}\mathbf{x}_1, (\mathtt{U}_\mathtt{q}^\mathsf{T}\mathbb{C}_1^{-1}\mathtt{U}_\mathtt{q})^{-1})$.

4.4 Evaluation Methods

To evaluate tracking performance, we need to measure the proximity between two contours. We propose a *segmental Hausdorff distance* (segHD) that allows a certain degree of non-rigidity. As illustrated in Fig.4.2 (b), the segHD between two corresponding landmark points $\mathtt{x}$ and $\mathtt{x}'$ on the two curves $\mathcal{C}$ and $\mathcal{C}'$, respectively, is defined as the Hausdorff distance (HD) between two segments $\omega(\mathtt{x})$ and $\omega(\mathtt{x}')$, where $\omega(\mathtt{x})$ defines a segment around $\mathtt{x}$ on the curve $\mathcal{C}$.

$$segHD(\mathtt{x}, \mathtt{x}') \quad = \quad HD(\omega(\mathtt{x}), \omega(\mathtt{x}')); \tag{4.26}$$

$$d_{segHD}(\mathcal{C}, \mathcal{C}') \quad = \quad \frac{\int_\mathtt{x} segHD(\mathtt{x}, \mathtt{x}')d\mathcal{C}}{\int_\mathtt{x} d\mathcal{C}}. \tag{4.27}$$

We further take the mean of the segHD of all landmarks as the distance between $\mathcal{C}$ and $\mathcal{C}'$, denoted by $d_{segHD}(\mathcal{C}, \mathcal{C}')$.

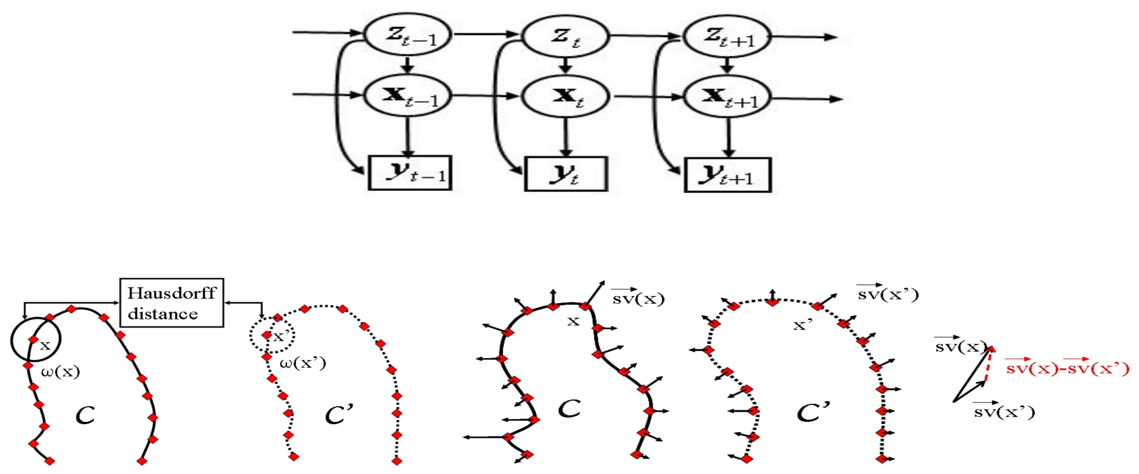


Figure 4.2: *(a) The observations ($\boldsymbol{y}_t$) are jointly produced by the motion phases ($\boldsymbol{z}_t$) and the vectors of shape and appearance ($\mathbf{x}_t$), $\mathbf{x}_t$ and $\boldsymbol{z}_t$ are determined by selecting the optimal estimate on all possible values of $\boldsymbol{z}_t$. (b) Segmental Hausdorff distance, and (c) Surprisal vector distance.*

The segHD measures only the 'physical distance' between two contours, ignoring their curvedness. The psychological term introduced by Feldman and Singh [92], the *surprisal vector* $\overrightarrow{sv}$, which quantifies how the curve is perceived, is favored to reflect the similarity on curvedness. Fig.4.2 (c) illustrates the surprisal vector. The direction of $\overrightarrow{sv}$ is the same as the outward normal direction, and the magnitude $|\overrightarrow{sv}|$ is a function of curvature. At a highly-curved portion of the contour, $|\overrightarrow{sv}|$ is large; at the flat portion, it is small. Using the surprisal vector, we compute a *surprisal vector distance* $d_{surp}(\mathcal{C}, \mathcal{C}')$ given below to characterize the proximity of two contours in their curvedness.

$$surp(\mathrm{x}, \mathrm{x}') = ||\overrightarrow{sv}(\mathrm{x}) - \overrightarrow{sv}(\mathrm{x}')||^2; \tag{4.28}$$

$$d_{surp}(\mathcal{C}, \mathcal{C}') = \frac{\int_{\mathrm{x}} surp(\mathrm{x}, \mathrm{x}') d\mathcal{C}}{\int_{\mathrm{x}} d\mathcal{C}}. \tag{4.29}$$

4.5 Experiments

4.5.1 Echocardiography tracking

Ultrasound sensing is a popular imaging modality due to its non-invasiveness, low cost, and that it has little or no side effects. One of the main applications of ultrasound imaging occurs in 2D cardiac ultrasonography (or echocardiography), which images the heart and surrounding structures. An echocardiography tracking system provides an *automatic* tool to delineate accurately the border of left ventricle (LV). The observation process computes optical flow for individual control points [93], and the fusion process regularizes the whole contour. Training data includes $\tilde{1}1000$ frames for A4C (apical four-chamber) view and $\tilde{9}200$ frames for A2C (apical two-chamber) view. Each frame needs to be aligned using the thin-plate splines warping algorithm and cropped to a 50×40 patch. Because echocardiograms have highly non-Gaussian intensity histograms, we use a nonlinear ultrasound-specific normalization method [88] to transform the non-Gaussian intensity histogram to have a Gaussian distribution.

Rhythmic heart movements have the special properties of periodicity and invariant occurrence orders for motion phases in cardiac cycles. Consequently, as long as we have prior knowledge of heart rate of an echocardiography sequence and time for end of diastole (ED) and end of systole (ES), the motion phase z_t of the current frame can be directly determined by the following equation:

$$\mathsf{z}_t = \left\lceil \frac{t - t_{ED}P}{T} \right\rceil \tag{4.30}$$

where T is the heart cycle, t_{ED} is the time of the end of diastole, and P is the total

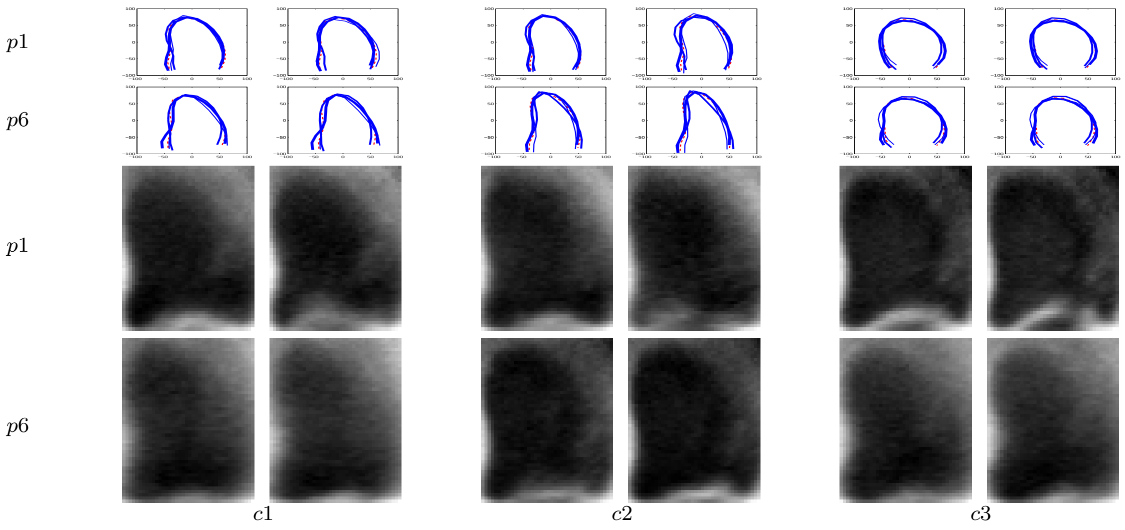


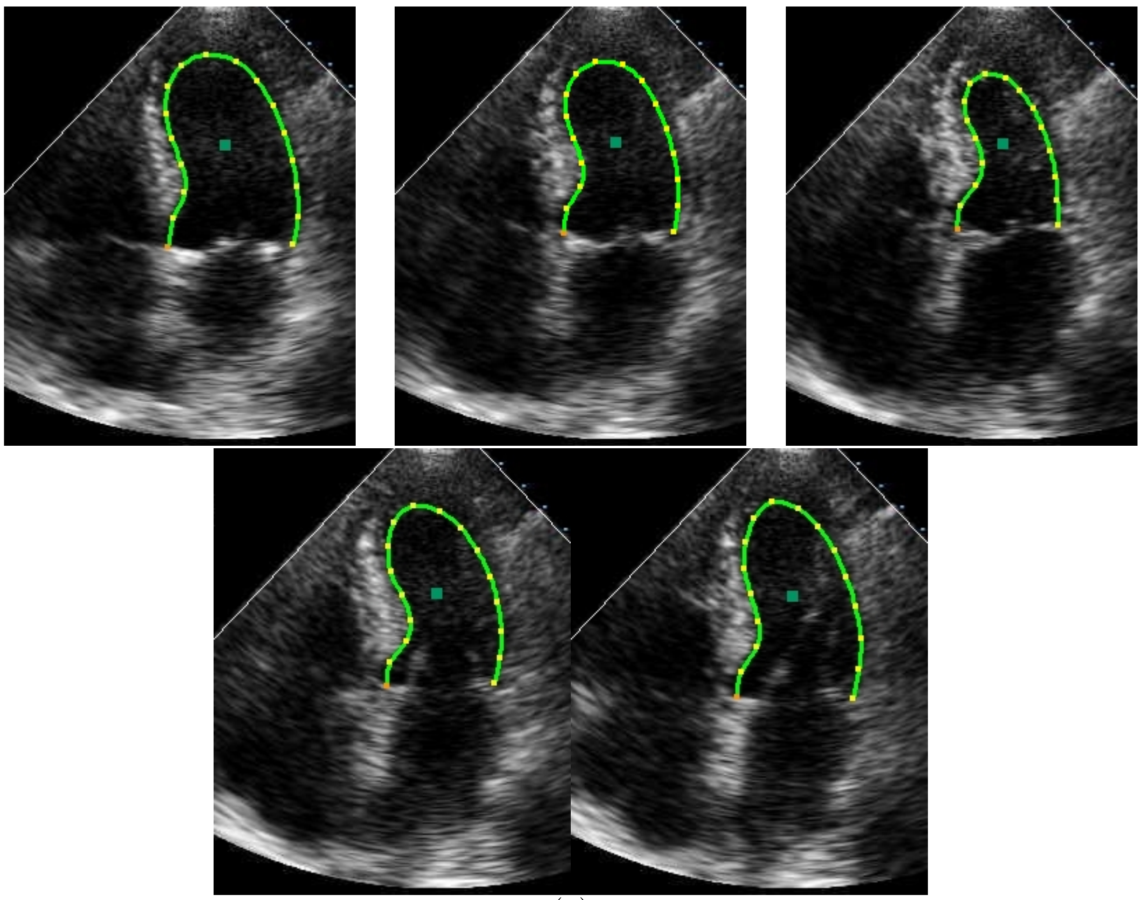
Figure 4.3: *Example of shape and appearance subspaces of the trained P-AAM. In our experiments, we use an N-cluster model (N = 3). c: cluster index, p: phase index. Rows in the figure correspond to clusters; columns correspond to phases. In the shape model, the red dotted lines represent the means of the subspaces, while the three blue solid lines in each plot represent three eigenvectors associated with the top three eigenvalues in the corresponding subspaces.*

number of motion phases. In our case, $P = 9$. The AAMC model is now specialized as a model with several P-AAM components [89]. Accordingly, given $\mathbf{z}_t = p$, the posterior function becomes $p(\mathbf{x}_p|\mathbf{x}_{p-1})$. The problem in Eq. (4.13) degenerates to:

$$\mathbf{x}_t^* = \arg\min_{\mathbf{x}} d^2_{p|\mathbf{y}_t,p-1,\mathbf{x}_{p-1}} \tag{4.31}$$

and it can be directly solved with the BLUE method. In practice, for each motion phase, since the joint shape and appearance space is hardly linear, we group the data into several clusters and learn one P-AAM for each cluster to handle the data.

For testing, we used 400 A4C sequences and 320 A2C sequences, whose ground truth contours were generated by experts. Fig. 4.4 shows the tracking contours on


(a)

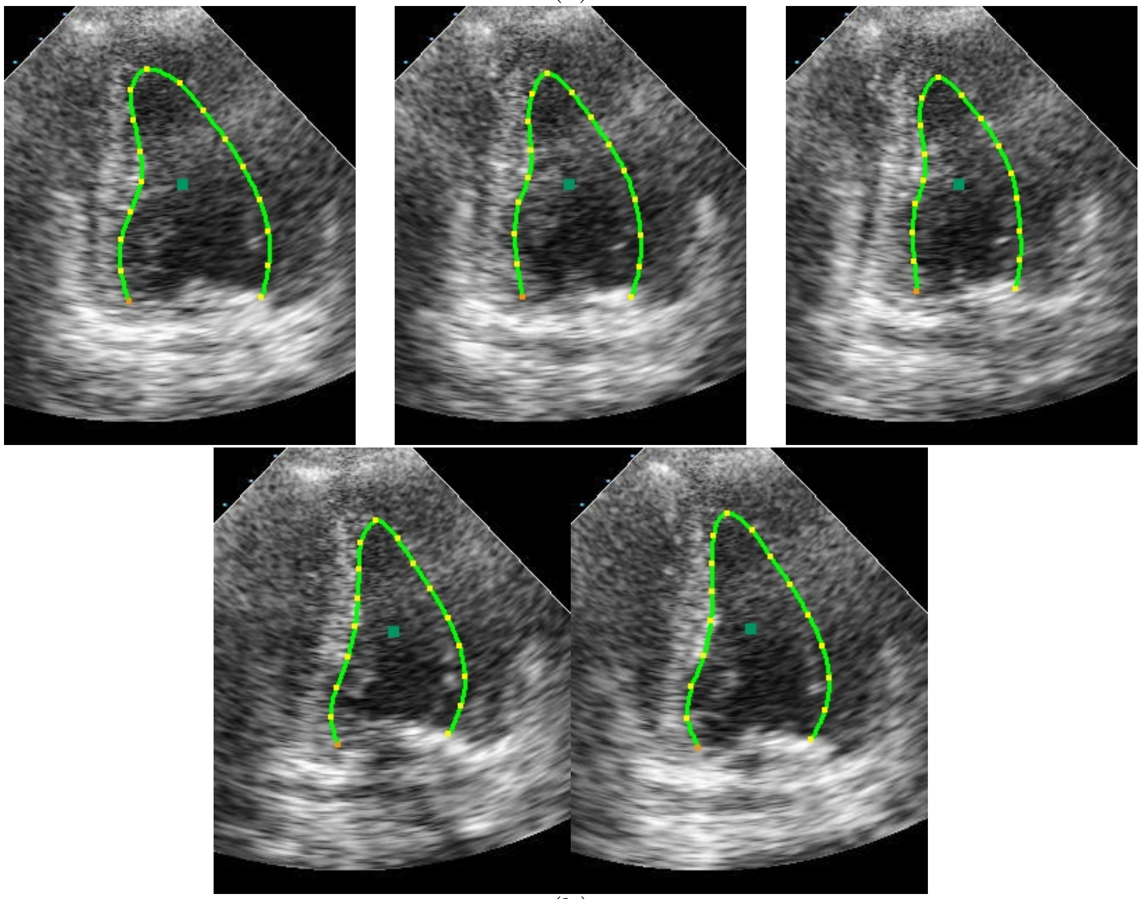

(b)

Figure 4.4: *The snapshots of the tracking results of (a) an A4C sequence and (b) an A2C sequence.*

sample frames of an A4C sequence and an A2C sequence. Several methods have been compared. The results, reported in Table 4.1 (1), use the median and standard deviation of the two distances introduced in sec.4.4. The "SSD" means the general optical flow method using the SSD similarity function [94]. The "CD_2" represents the optical flow method using the similarity function in [95], which considers a simplified ultrasound image formation. The "NPAM" refers to the optical flow method proposed in [89]. The "P-AAM" means regularizing the NPAM results using the P-AAM. From Table 4.1(1)(a), we observe that NPAM improves the tracking results significantly in terms of segHD, compared to "SSD" and "CD_2". Further, using the P-AAM decreases segHD. The advantage of using P-AAM is further highlighted when the surprisal vector distance is used, as shown in Table 4.1(1)(c). The effectiveness of shape, appearance and motion information when used as prior knowledge is shown in Table 4.1(1)(b), in which four kinds of prior knowledge are compared. The "ASM" shows results obtained using the *phase-separate* ASM only, without involving the pairwise model. No motion and appearance information is integrated in the model. The "AAM" model uses the *phase-separate* AAM only, which takes into account shape and appearance. The third model uses the P-ASM (pairwise ASM) model, with shape and motion but does not use appearance information. The last model is the P-AAM model that jointly considers shape, appearance and motion. Table 4.1(1)(b) suggests that using more prior information decreases tracking error. It also indicates the order of importance of the three elements: shape, appearance and motion. For example, the fact that the AAM provides better performance than the P-ASM suggests that the appearance information contributes more to the entire

(a) sequences	Segmental Hausdorff distance d_{segHD} (pixels)			
	SSD	CD$_2$	NPAM	P-AAM
A2C	10.8612 ± 2.2621	7.9392 ± 1.5645	2.7042 ± 0.6732	2.6275 ± 0.6623
A4C	11.0310 ± 2.5927	7.3640 ± 2.3561	2.5291 ± 0.6076	2.4588 ± 0.5550

(b) sequences	Segmental Hausdorff distance d_{segHD} (pixels)			
	ASM	AAM	P-ASM	P-AAM
A2C	2.6901 ± 0.6611	2.6844 ± 0.6881	2.6849 ± 0.6951	2.6275 ± 0.6623
A4C	2.5191 ± 0.5915	2.4776 ± 0.5614	2.5059 ± 0.5930	2.4588 ± 0.5550

(c) sequences	Surprisal vector distance d_{surp}			
	SSD	CD$_2$	NPAM	P-AAM
A2C	0.3204 ± 0.1256	0.0957 ± 0.1197	0.0352 ± 0.0514	0.0098 ± 0.0110
A4C	0.3024 ± 0.1147	0.0995 ± 0.1006	0.0345 ± 0.0586	0.0096 ± 0.0097

(1) *Echocardiography tracking*

	OF	ASM	AAM	ASMC	AAMC
Segmental Hausdorff distance	8.2345±13.3848	5.0825± 9.0518	4.8817± 9.1915	4.3162±8.1642	4.0686±7.4780
Surprisal vector distance	0.3167±0.0539	0.0692±0.0417	0.0693±0.0403	0.0648±0.0258	0.0611±0.0294

(2) *Face tracking*

Table 4.1: Comparison of tracking performances based on the segmental Hausdorff distance and the surprisal vector distance: (1) Echocardiograph tracking results; (2) Face tracking results. "OF" is the original result from optical flow estimation.

system than the motion information.

4.5.2 Pedestrian Tracking

Contour-based pedestrian tracking is an important topic in surveillance systems. Our approach is based on the active contour tracking algorithm [80], to which the particle filter [8] is applied. During walking, a human body undergoes both rigid and non-rigid movements. Accordingly, the observation process is decomposed into two steps: (1) **2D motion estimation** that yields object-wise spatial rigid-body motion, including translation and rotation parameters. We use a particle filter based on an affine model to estimate all the parameters; and (2) **2D shape deformation** that captures the pose changes of non-rigid objects by using a posterior deformation probability map based on statistical analysis of the corresponding frame.

In the fusion process, we observe that appearances of pedestrians vary largely due to different textures and colors of clothes people wear, which makes modeling an appearance subspace highly unreliable. Therefore, we apply a special case of the AAMC model, the ASMC (active shape Markov chain) model in the fusion process, i.e., we use only the prior shape and motion knowledge. Accordingly, the subspace of one P-AAM component includes only the shape subspace. The state vector in this system includes only $[\mathbf{x}_{t,\mathcal{G}}, \mathbf{z}_t]$. The training data for the ASMC model on walking pedestrians consists of approximately 2000 80×120 binary frames with aligned pedestrian silhouettes (from the USF dataset) from 23 sequences. The shape vector has 120 dimensional (60 control points). We divided one complete movement cycle of people walking into five motion phases. The elements of the trained transition matrix $\mathbb{T} \in \mathbb{R}^5 \times \mathbb{R}^5$ are all non-zero. Therefore, we have 20 pairs of P-AAM components. Fig.4.6 (a) shows the trained ASMC model. We tested our AAMC model on several sequences captured by moving cameras. Fig.4.5 shows results on two sequences. We compare three algorithms in Table 4.2 with scenarios of heavily-cluttered backgrounds and non-rigid targets. The ground truth was generated manually. We evaluate the tracking results by the segmental Hausdorff distance. The "ACT" refers the traditional active contour tracking method pro-

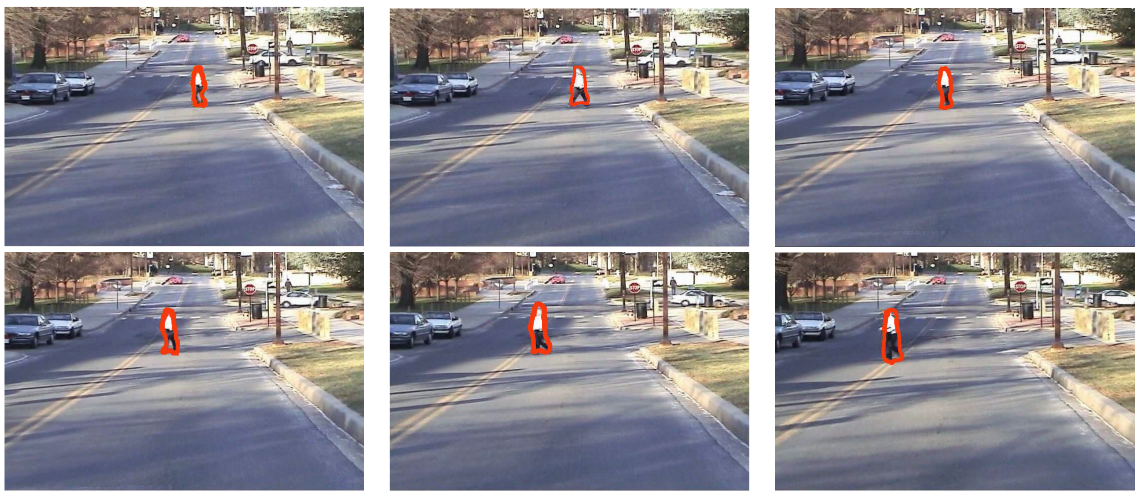
	ACT	ASM	ASMC
heavy-cluttered background	14.7584 ± 74.4052	7.9129 ± 12.5602	6.5297 ± 10.3378
non-rigid object	9.9741 ±24.6597	6.0571 ± 8.5222	5.2188 ± 6.2265

Table 4.2: Comparison of tracking results on heavily-cluttered background sequences and non-rigid object sequences in terms of segmental Hausdorff distance.

posed in [80]. The "ASM" refers the contour tracking using the ASM constraints. The "ASMC" is our proposed algorithm. Both "ASM" and "ASMC" algorithms use the two-step approach in the observation process. We observe that: (1) using a two-step operation in the observation process significantly improves the performance because the operation concerns both rigid and non-rigid movements; and (2) use of "ASMC" further improves the robustness of the tracking system because it integrates shape, and motion prior knowledge, and makes the regularizing model adaptive during tracking, while the "ASM" model does not.

4.5.3 Face tracking

Linear subspace methods have been widely used in several face tracking systems. We also evaluate the use of the AAMC model in face tracking scenarios. To illustrate the contribution of the regularization part clearly, we use a simple non-parametric optical flow approach in the observation process to estimate the dynamic motion [94]; the fusion process includes the regularization by the AAMC model. The model is generated based on the IMM face database [96] with 240 still grayscale images of 40 different people. Some interior parameters are: the number


(a)

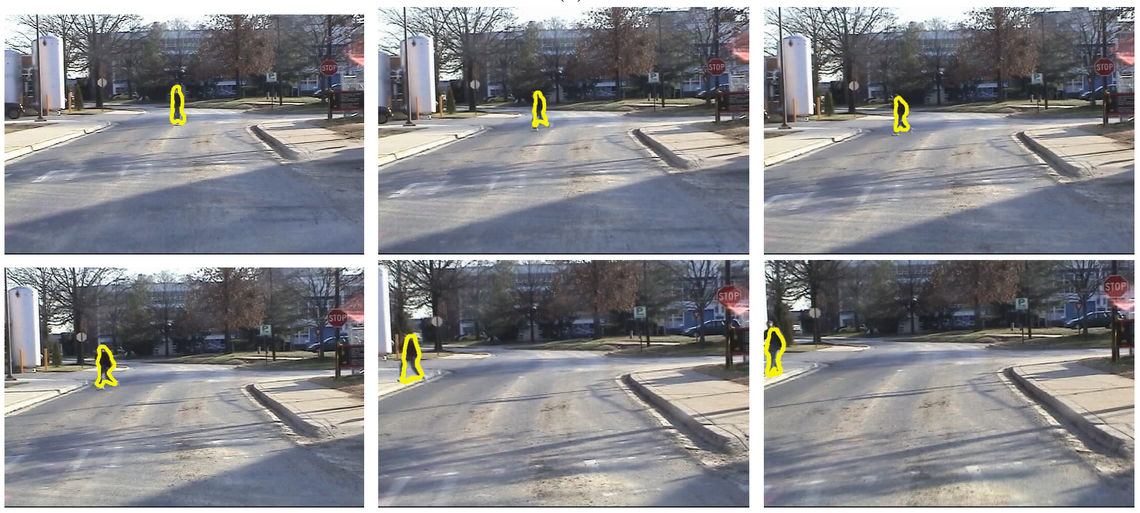

(b)

Figure 4.5: *Two sequences in which both background and object move.*

of motion phases is 4 (two frontal views with different poses and two side views); the shape of the facial structure was manually annotated using 58 landmark points, i.e., the dimension of the shape subspace is 116 (the landmark points are distributed around eyebrows, eyes, nose, mouth, and jaw). The appearance subspace of the face is trained from aligning the original region of interest to 30×20 patch. Therefore, the dimension of appearance subspace is 600. The testing sequence was captured

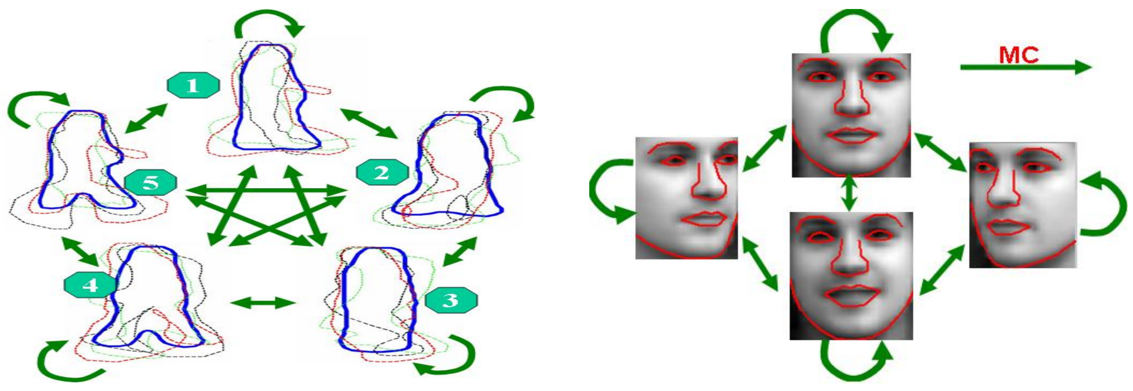


Figure 4.6: *(a) The graphic illustration of the ASMC model for pedestrian tracking. The contours with heaviest weights are mean shapes of motion phases. The thinner contours in each shape model component represent the three eigenvectors associated with the top three eigenvalues in the corresponding subspace; and (b) The graphic illustration of the AAMC model for face tracking. The red solid lines are the means of shape subspaces, the patches are the means of appearance subspaces. In both plots, the arrowed lines between two ASM (AAM) models represent the Markov chain connections.*

by a moving camera in a subway, with resolution of 180×240. Fig.4.6(b) shows the trained AAMC model for face tracking. Fig.4.7 compares several tracking results on one frame in the testing clip when using different approaches, including the original optical flow estimate and the fusing estimates after applying the ASM, AAM, ASMC, and AAMC models. The AAMC model detects that a motion phase switch (from side-view to frontal) occurs in this frame, while the ASMC model does not detect it. Fig.4.8 shows some tracking results on the testing sequence. We notice that the original optical flow estimation tends to drift, and which the AAMC model significantly reduces. Table 4.1(2) reports the results using different models in terms of segHD and surprisal distance. The AAMC model always gives the best

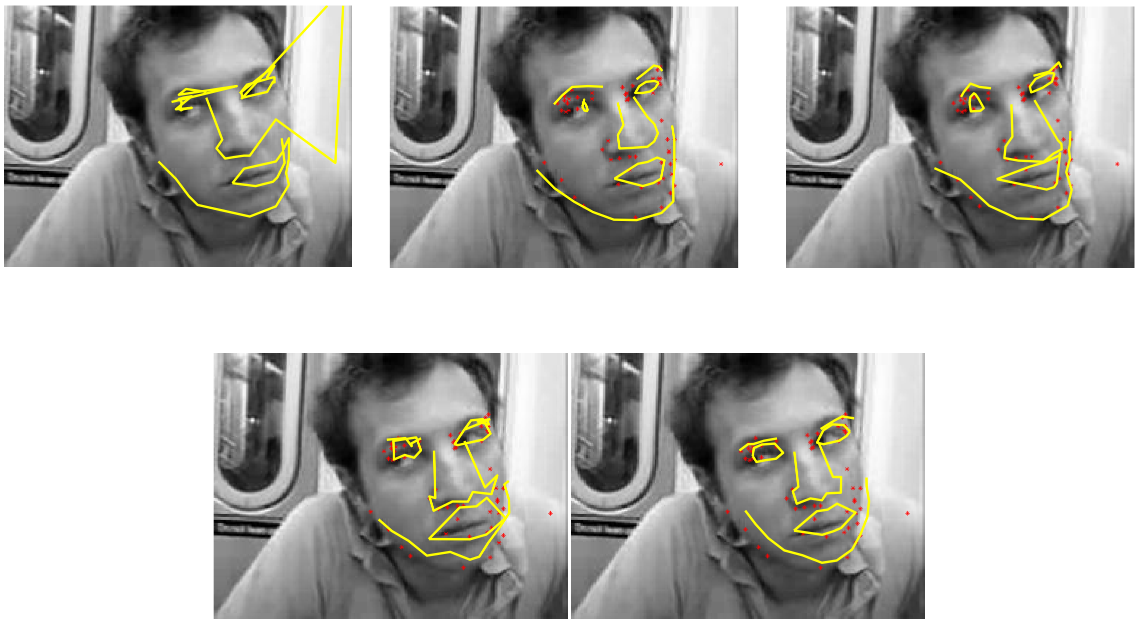


Figure 4.7: *Comparison of different tracking results on one frame (frame220), where a switching of motion phases occurs. The red dots represent the original location of landmark points. (a) the original estimate from observation process; (b) fused with ASM model; (c) fused with ASMC model; (d) fused with AAM model; (e) fused with AAMC model.*

performance among all the models we have tried in our experiments.

4.6 Conclusions

We have proposed a general model to represent shape, appearance, and motion information. The proposed AAMC model has two parts: the Markov chain model controls the transition between motion phases; the P-AAM component models the shape and appearance information using subspaces and the inter-phase motion by paired data in each component. The model is used for contour tracking as a prior model to fuse with the observation and generate an optimal estimate. The nature of

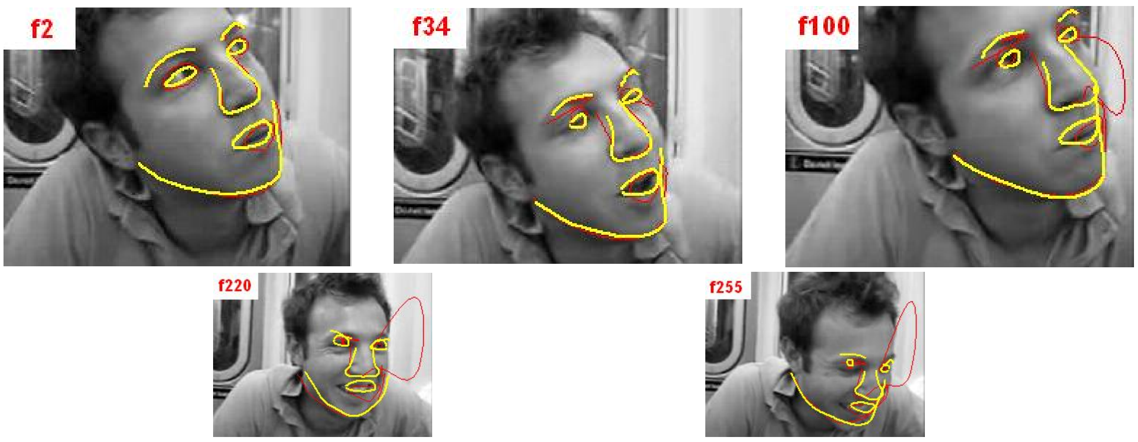


Figure 4.8: *The snapshots of tracking results from the testing face sequence. The red lines denote the original observation output by optical flow method, and the yellow lines represent the optimal estimates after fusing with the AAMC model.*

the AAMC model also enables its adaptability throughout tracking. We tested the algorithm on several applications, including echocardiography tracking, pedestrian and face tracking. The tracking results evaluated by the segmental Hausdorff distance and surprisal vector distance demonstrate improved robustness and accuracy.

Chapter 5

Video Mensuration Using Stationary Cameras

5.1 Introduction

In photogrammetry applications, it is required to make measurements on digital images. A major difficulty happens with the distortion of geometric scene properties due to imaging transformation in a perspective camera. The most influenced properties include the size of an object, the length ratio and parallelism between line segments, all of which are not preserved in a projected image. An ideal way to handle all the uncertainties is by recovering the real 3D world from a 2D projected image, which, in most cases, needs significantly more information than a single image can provide with unknown camera calibration. A semi-automatic measuring method using image data from a single camera has been proposed in [97], which uses minimal calibration information and the perspective geometry approach for mensuration. Motivated by the wide availability of video surveillance systems, we propose to extend single image metrology techniques to video data acquired by stationary cameras, which is referred to as video metrology.

Video metrology enables improved automation, flexibility and accuracy. (i) Achieving *automation* is critical for handling the large volume of video data. Traditional photogrammetric systems involve some human interactions, which are unpleasantly burdensome and often unreliable. Geometric features, such as orthogonal lines, parallel lines and vanishing points are usually manually labeled, and even objects need to be manually selected from surroundings. Video sequences enable an automatic solution: Objects can be first segmented using tracking or segmentation algorithms, then motion information can estimate the minimal calibration of scenes. The need for human interactions is reduced greatly. (ii) Exploitation of motion information offers more *flexibility.* Photogrammetry relies on man-made environments characterized by plenty of solid parallel and orthogonal lines to infer the vanishing line/point. These strong cues are not always present. However, the tracked moving objects in video sequences provide a way to estimating the vanishing line of ground planes and the vertical vanishing point. In some cases, mensuration from a single image is not reliable, especially when occlusions or instant disturbances occur. (iii) Since video sequences provide additional temporal information, fusing multi-frame measurements smoothes the measuring errors of individual frames, improving the *accuracy* of final estimate of the measurement.

In this chapter, we develop a minimal supervised algorithm based on monocular videos and uncalibrated stationary cameras, which is of increasing interest in many applications. "Uncalibrated" means that the camera focal length, principal point location and image affinity parameters are not assumed to be available. Since object motion is a highly critical factor in our algorithm, when there are no moving objects appear in a scene, the problem degenerates to one of single view metrology problems. The proposed algorithm has the following key steps:

- By tracking the motion of objects in an uncalibrated video sequence, we automatically recover useful geometric properties of the scene. These geometric properties include the vanishing line of the ground plane and the vertical vanishing point.

- We apply the Expectation Maximization (EM) algorithm to cluster tracked trajectories of feature points simultaneously into groups and estimate the corresponding vanishing points belonging to each group. Outliers of trajectories are also detected.

- With estimated minimal calibration information, we apply a single-view metrology algorithm [97] to estimate the height of a target object in each frame. We also derive an uncertainty analysis for the height measurement.

- Finally, we fuse all the measurements for frames using a well-known stochastic approximation technique for an optimal estimate.

5.1.1 Related work

Several techniques have been developed for metrology. The vanishing point of parallel lines has been proven useful in recovering 3D scenes for decades [98, 99]. Criminisi *et al.*, [97] outline a method for recovering an affine scene structure from a single perspective view using vanishing lines and points. Without the knowledge of a camera's internal calibration, they can compute the area and length ratios of any planes parallel to the reference plane. Gurdjos *et al.*, [100] show how to estimate the heights of soccer players from videos given vanishing points/lines. These approaches are limited in the dependence on manual supervision. As an extension, Liang *et al.*,

[101] introduce two view metrology which estimate affine height measurements by recovering planar homography of the reference plane between two views separated by a (near) pure translation.

By taking advantage of tracking results from video sequences, several automatic mensuration algorithms have been developed. The calibration technique proposed by Lv *et al.*, [102] uses certain constraints, that walking pedestrians are essentially perpendicular to the ground plane and their heights remain constant, to determine the vanishing line. Renno *et al.*, [103] use projected sizes of pedestrians to estimate the vanishing line of a ground plane. Stauffer etc. propose [104] a self-calibration method by linearly approximating measurable projected properties. Bose and Crimson [105] propose a method using constant velocity trajectories of objects to derive vanishing lines for recovering reference plane and rectification. Their algorithm uses an additional constraint brought by the constant-velocity assumption, which is not always available in surveillance sequences. Guo and Chellappa [106] present an algorithm that detects and tracks wheels of an automobile for wheel base mensuration, with sufficient results to determine the vehicle classes. We also leverage object motion in videos to acquire calibration information for measurement. Furthermore, the proposed algorithm incorporates all the measurements from individual video frames to improve the accuracy of the final measurement.

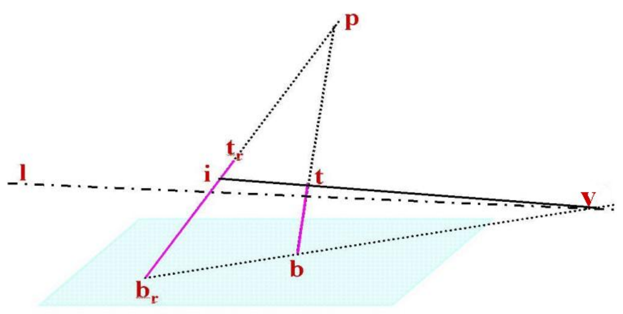
5.2 Single view measurement from minimal calibration

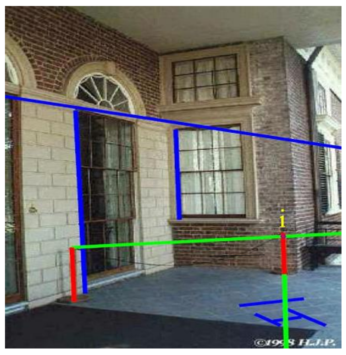
Perspective projection introduces challenges in video mensuration applications. One can easily conduct mensuration on images of objects, but little can be inferred from the mensuration without understanding how the objects have been mapped from world coordinates to image coordinates. Given a typical pinhole camera, Criminisi et al. [97] illustrate the following geometric fact: Given the vanishing line of a *reference plane* (in our case we refer to the ground plane), a *vanishing point* for another *reference direction* (not parallel to the plane, in our case the vertical direction) and a *reference height*, we can estimate any object's height if the object and the reference object are both on the reference plane. We call this a **minimal calibration condition**.

Denote the vanishing line of a reference plane (ground plane) as l , the vertical vanishing point as p and the length of a reference line segment $\overline{t_r b_r}$ as h_r. We want to measure the height h of segment $\overline{tb}$. Intersection v can be obtained by $v = \overline{b_r b} \times l$; intersection i is determined by $i = \overline{tv} \times \overline{pb_r}$. The four points p, i, t_r, b_r define a cross ratio [107]. $\overline{b_r i}$ is the projected segment of the segment $\overline{tb}$ onto $\overline{t_r b_r}$. Therefore, the world height ratio h/h_r can be derived using the cross ratio invariance and 1D homography [97] as:

$$\frac{h}{h_r} = \frac{d(p,i)d(t_r,b_r)}{d(i,b_r)d(t_r,p)} \tag{5.1}$$

where $i = (p \times b_r) \times (t \times (l \times (b_r \times b)))$. The height of any object can be estimated in a similar manner. Figure 5.1 illustrates the computation of the height ratio using the above geometric method. A real image application is also demonstrated.



(a)


(b)

Figure 5.1: *(a) The basic geometry of the scene. l is the vanishing line of the reference plane, line $\overline{b_r t_r}$ and $\overline{bt}$ are two parallel lines on the reference plane. p is the vanishing point on direction of $\overline{b_r t_r}$ and $\overline{bt}$. (b) An example of applying the algorithm to a real image, the red lines represent the reference height and the target height respectively, "i" is the same position illustrated in (a).*

5.3 Self minimal calibration from moving objects

5.3.1 The geometric properties of moving objects

The minimal calibration condition includes three parts, *the vanishing line of a ground plane*, *the vertical point* and *the reference height(s)*. Traditional approaches rely on manually labeled parallel lines to obtain vanishing points, which are based on the following well-known geometric properties: (i) The vanishing line (the line at infinity), $l_\infty = (l_1, l_2, l_3)^T$, of a ground plane has two degrees of freedom and can be determined as the line through two or more vanishing points of the plane. (ii) In a perspective view, a set of parallel lines pass through a common point in the image plane, which is called their vanishing point. The vertical point is the vanishing point of all the vertical world lines on the image plane. Therefore, two sets of parallel lines (non-parallel to each other) on the ground plane and one set of vertical lines are the minimum requirements to determine the minimal calibration.

Instead of detecting parallel line sets from man-made structures [108, 109, 110], our approach focuses on automatically extracting parallel lines through motion information obtained from video. This is feasible because when an object is moving in a scene, it holds some properties:

1. Most of the moving objects are on the ground plane. For a rigid object, the trajectory of each point on the object over time is parallel to the ground plane. For a non-rigid object, like a human, some points on the object have their trajectories parallel to the ground plane, while the others have theirs not

parallel to the ground plane due to deformation.

2. Since the vanishing line can also be defined as the line of intersection between the image plane and the plane passing through the projection center that is parallel to the ground plane, an object's projected size should be zero at the vanishing line of the ground plane, where the object is at infinite distance from the camera. Consequently, when an object moves in a direction with decreasing projected size in an image plane, all the points on the same object finally converge into one point, which belongs to the vanishing line of the ground plane.

3. In short term, the most common moving pattern of objects is a straight line. Therefore, it is not difficult to collect at least two sets of trajectory segments in straight lines, each set containing the trajectories of feature points belonging to the same moving object. We denote each trajectory by l_i, a homogeneous three-dimensional vector associated with feature point i. These sets of trajectory segments need not be from different moving objects; they can be collected from the same object during different time intervals. The only requirement is that these sets should be of different world orientations. Each set can determine one vanishing point on the ground plane, which satisfies:

$$l_i \cdot v_j = 0 \quad \forall i \in \mathbf{U}^j \tag{5.2}$$

where v_j is the 2D homogeneous coordinate of the vanishing point, $(v_j^x, v_j^y, 1)^T$, associated with trajectory set $\mathbf{U}^j$. With at least two vanishing points, we can estimate the vanishing line of the ground plane.

In an ideal situation of no-tracking-error and rigid-motion, by using the above properties, one can accurately recover the vanishing line and the vertical point. Unfortunately, such conditions do not apply to most surveillance applications. It is often the case that tracking trajectories are contaminated by errors, and a set of trajectories from one object do not intersect at one common point. Therefore, robust techniques for selecting good trajectories (inliers) for vanishing point estimation are needed.

5.3.2 Recovering vanishing line and vertical vanishing point

Our tracking method is based mainly on feature points obtained through the optical flow method [111], under the assumption that objects are moving on the ground plane and each world trajectory of tracked feature point is parallel to the ground plane. The error in estimating vanishing points results from two factors: the correspondence error from tracking data and the effect of non-rigid motion on straightness and parallelism of trajectories. Both result in a set of supposedly parallel lines that do not intersect precisely in one point. As a solution, mutual trajectory similarities are evaluated with an auxiliary blob tracking module, which significantly suppresses these errors. With the trajectories from both feature points and motion blobs available, the next steps consist of refining trajectories with the constraint that the corresponding motion spaces to be one dimensional, applying the EM algorithm to cluster the trajectories into different groups, excluding outlier trajectories, and estimating the vanishing points and hence the vanishing line, and estimating the vertical vanishing point.

5.3.2.1 Trajectory refinement

Since lines parallel to the ground plane project to the image plane under a projective transformation, we collect all trajectory segments (a trajectory contains a sequence of image coordinates of one feature point in successive frames) that can be fit by straight lines. We parameterize a piece of trajectory segment by an angle θ (between the line and the x axis) and the perpendicular distance ρ from the origin to the line using equation:

$$x \sin\theta - y \cos\theta + \rho = 0 \tag{5.3}$$

The parameters can be directly determined by the eigenvalues λ_1, λ_2 and eigenvector v (associated with the largest eigenvalue) of the matrix D associated with the normalized trajectory support region:

$$\mathbf{D} = \begin{pmatrix} \sum_j (x_j - \bar{x})^2 & \sum_j (x_j - \bar{x})(y_j - \bar{y}) \\ \sum_j (x_j - \bar{x})(y_j - \bar{y}) & \sum_j (y_j - \bar{y})^2 \end{pmatrix} \tag{5.4}$$

where $\bar{x}$ and $\bar{y}$ are the means of x and y coordinates. Therefore, $\theta = \arctan(-v(1), v(2))$ and $\rho = \bar{y}\cos\theta - \bar{x}\sin\theta$. We can thereby define l_i in equation (5.2) as $l_i = (\sin\theta, -\cos\theta, \rho)^T$. Such a notation allows the perpendicular distance from any point $\mathbf{x} = (x, y, 1)^T$ to the line to be calculated simply as $d(l_i, \mathbf{x}) = |l_i \cdot \mathbf{x}|$.

Fitting a straight line to any trajectory acts as a constraint to limit the feature point motion space to be one dimensional [112]. It can efficiently reduce the vibration of feature point movement, while retaining its parallelism to the ground. Figure 5.2

shows an example on trajectory refinement.

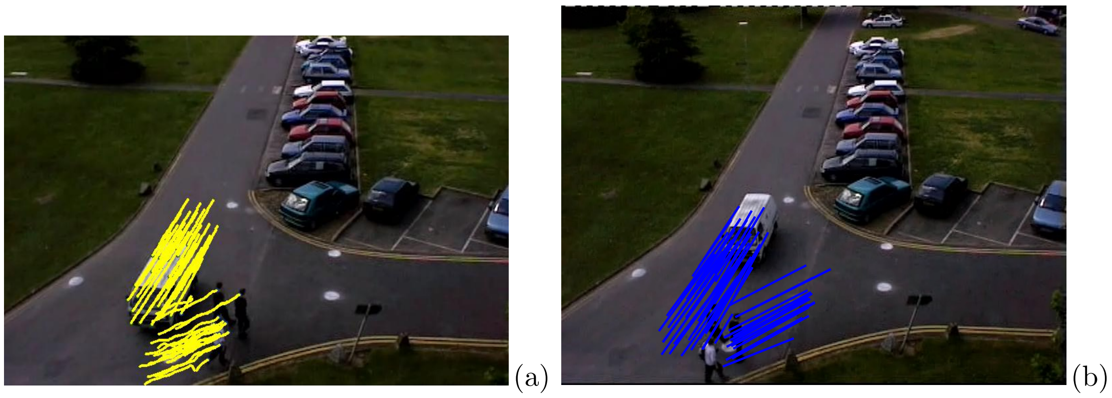


Figure 5.2: *An example of refining trajectories by line fitting. The left image depicts the trajectories from tracking data, and the right image shows the refined trajectories.*

5.3.2.2 Simultaneous trajectory grouping and vanishing point estimation

The problem now can be addressed as: given a set of trajectory segments $L = \{l_i, i = 1, \ldots, N\}$, we want to find a set of vanishing points, $V = v_k, k = 1, \ldots, K$, which minimizes the following objective function:

$$J(V) = \sum_k \sum_{i \in \mathbf{U}^k} |l_i \cdot v_k| \quad s.t. \ \forall m \neq n, \ \mathbf{U}^m \cap \mathbf{U}^n = \phi \tag{5.5}$$

where $\mathbf{U}^k$ denotes the trajectory index subset associated with a particular vanishing point v_k. The objective function implies two operations: grouping observed trajectories and estimating corresponding vanishing points. We formulate it as a problem of finding the most likely estimates of vanishing points, as well as probabilities of each trajectory belonging to a particular vanishing direction, as suggested in [113]. We use the EM to estimate iteratively the vanishing points and cluster the trajectories.

In the EM algorithm, the objective function is derived from the expected log likelihood with respect to each of unknown parameters v_k:

$$J(v_k) = \sum_i w_{i,k} \log p(l_i|v_k) \tag{5.6}$$

where $w_{i,k}$ is a membership function that measures the posterior probability of a vanishing point v_k that a trajectory segment l_i belongs to $w_{i,k} = p(v_k|l_i)$.

The moving objects not only provide the trajectories of all feature points belonging to the objects, but also the trajectories of blob centroids [5, 27]. We denote the trajectories of the blob centroids as $O_j, j = 1, \ldots, M$, with same parameterizations as l_i. Using the Bayesian rule, we have:

$$\begin{aligned} p(v_k|l_i) &= \sum_j p(v_k, O_j|l_i) = \sum_j p(v_k|O_j, l_i)p(O_j|l_i) \\ &= \sum_j p(v_k|O_j, l_i)\frac{p(l_i|O_j)p(O_j)}{\sum_m p(l_i|O_m)p(O_m)} = \frac{\sum_j p(v_k|O_j, l_i)p(l_i|O_j)}{\sum_m p(l_i|O_m)} \end{aligned} \tag{5.7}$$

where we assume that the probability of any O_j appearing is equal, i.e. $p(O_j)$ is equal.

The term $p(l_i|O_j)$ determines the likelihood of l_i belonging to the object O_j. Denote by $R^t_{O_j}$ the support region of j^{th} blob at t frame and $X^t_{l_i}$ the position of i^{th} feature point at t frame . We have:

$$p(l_i|O_j) = \begin{cases} \exp(-\frac{d(O_j,l_i)^2}{2\sigma^2}), & \sum_{t=t_i^0}^{t_i^T} \mathbf{I}(X^t_{l_i}) > T_i^{thresh}; \\ 0, & \text{otherwise.} \end{cases} \tag{5.8}$$

where t_i^0 and t_i^T are the starting time and disappearing time of i^{th} trajectory, and T_i^{thresh} is equal to (0.8 0.9) of the existing time of i^{th} trajectory. $\mathbf{I}$ is an indicator function:

$$\mathbf{I}(X_{l_i}^t \in R_{O_j}^t) = \begin{cases} 1, & X_{l_i}^t \in R_{O_j}^t; \\ 0, & \text{otherwise.} \end{cases} \tag{5.9}$$

$d(O_j, l_i)$ measures the distance between the trajectory l_i and the blob distance O_j. Instead of using the standard angle difference between two lines to measure the distance, we introduce the Kullback-Leibler (KL) distance [114]. We want to select "good" feature point trajectories, which means we count the trajectories with higher probabilities to be world-parallel, while dropping outliers resulting from either tracking failure or non-rigid object motion. The KL distance has the advantage of being able to evaluate the relative similarities between two trajectories and select preferred trajectories. Therefore, $d(O_j, l_i)$ is computed as:

$$d(O_j, l_i) = \sum_{p=1}^{\mathbf{P}} f_{l_i}^p \log(\frac{f_{l_i}^p}{f_{O_j}^p}), \;\; \sum_p f_{l_i}^p = \sum_p f_{O_j}^p = 1.0 \tag{5.10}$$

where f is computed using the normalized Fourier descriptors over points of each trajectory (before refinement). For one series of Fourier descriptors from the corresponding trajectory, neglecting the first coefficients (DC component) of the Fourier descriptors, we smooth the magnitudes f_{mag} and phases f_{pha} of the Fourier descriptors, then normalize them to achieve a probability distribution. So $f = f_{mag} \cdot f_{pha}$. Figure 5.3 illustrates how the KL distance removes outliers from the observation data.

The term $p(v_k|O_j, l_i)$ evaluates the likelihood of vanishing point v_k belonging to both trajectories O_j and l_i. The likelihood values are determined by the distance between points and lines. We assume that it takes the following form:

$$p(v_k|O_j, l_i) \propto \exp(-\frac{|O_j \cdot v_k|^2}{2\sigma_1^2}) \cdot \exp(-\frac{|l_i \cdot v_k|^2}{2\sigma_2^2}) \qquad (5.11)$$

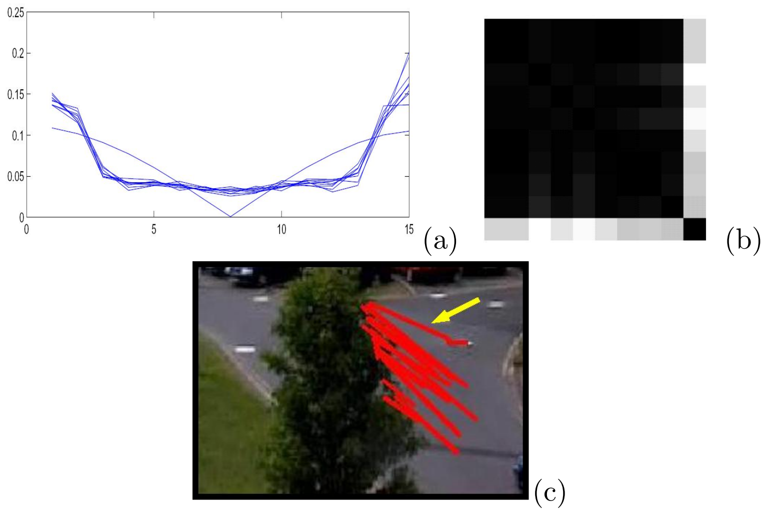


Figure 5.3: *Using the Fourier Descriptor to find outliers. (a) shows the magnitude of the Fourier descriptors (excluding DC component), with each line representing a piece of observed trajectory. (b) the mutual KL distance between trajectories. The lighter color indicates larger distance. Obviously the last trajectory appears farther from other data, which implies that it is a possible outlier. (c) The red lines are the related feature point trajectories; and the yellow arrow points to the outlier trajectory.*

The initialization of the EM algorithm is based on the observation that one object moving along one direction can determine one vanishing point. Therefore, using LMS, each initial vanishing point is computed with all the feature point trajectories related to the corresponding moving object. In the E-step, we update the current membership function defined by (5.7) with the vanishing point estimate from the last iteration, $w_{i,k} = p(v_{k-1}|l_i)$. The M-step involves maximization of the objective function in (5.6). We have:

$$J(v_k) = \max_{v_k} \sum_i w_{i,k} \log p(l_i|v_k) = \min_{v_k} \sum_i w_{i,k} |l_i \cdot v_k|^2 \tag{5.12}$$

which can be solved by the LMS method using a linear approximation. Iteratively alternating the E and M steps, we finally achieve an optimal estimate of the position of vanishing points. Figure 5.4 shows an example of applying the EM algorithm for estimating vanishing line. The vanishing line is obtained as the LMS solution of $\mathcal{L} \cdot \mathbf{V} = 0$, where $\mathbf{V} = [v_1, v_2, ..., v_K]$.

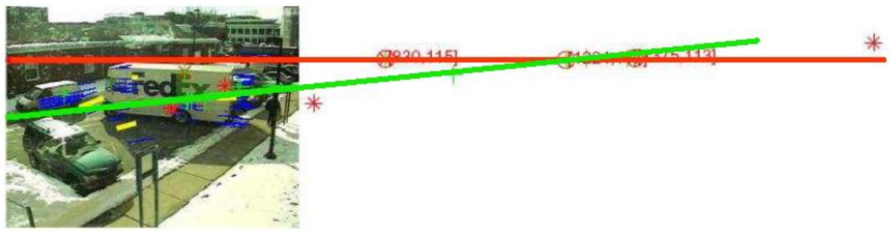


Figure 5.4: *Using the EM algorithm to estimate the vanishing point. The "+" depicts the iteration of the algorithm, "o" is the final value. The green line is the initial vanishing line, and the red is the final vanishing line of the ground.*

5.3.2.3 Vertical point estimation

The vertical vanishing point can be estimated by the aid of a geometric property that the orthocenter of three orthogonal vanishing points (x, y and z axis) as vertices is the principal point P [98]. We further assume that P in the image plane is close enough to the image center. Therefore, when we find the vanishing line of the ground plane $\mathcal{L}$, the vertical point p satisfies: $\mathcal{L} \cdot (p \times P) = 0$, which can be further formulated as

$$p \cdot \mathcal{L}_{\perp} = 0, \quad \mathcal{L}_{\perp} = (\frac{\mathcal{L}^{(2)}}{\mathcal{L}^{(1)}}, -1, \frac{\mathcal{L}^{(2)}}{\mathcal{L}^{(1)}} P^{(1)} + P^{(2)})^T. \tag{5.13}$$

where $\mathcal{L}^{(i)}$ represents the i^{th} element of $\mathcal{L}$. Another constraint results from the principal axis of motion blob in each frames, which is defined as the eigenvector with smaller angle to y image axis. (We did not define the principal axis as the eigenvector associated with the larger eigenvalue because when the object is a vehicle, then the largest eigenvalue of the motion blob is related to the width of the vehicle instead of the height.) Those principal axes, denoted by $E = [e_1, e_2, ...e_M]$, satisfy:

$$p^T E = \mathbf{0}_M. \tag{5.14}$$

The RANSAC method is applied to estimate the vertical point p with the above two constraints.

5.4 Multi-frame measurement fusion using stochastic approximation

One advantage of video metrology is that the result does not count on any single frame in the video. That means, the incorrectness measurement derived from one frame due to bad tracking locations, occlusions or pose variations can be compensated by fusing multiple-frame measurements. Therefore, after we obtain the minimal calibration, we apply measurement in each frame of the sequence and fuse them to achieve an optimal estimation.

5.4.1 Single frame measurement

Given a stationary reference height h_r, two image points b_r and t_r identified to represent the reference object, combined with the estimated vertical point $\hat{p}$ and the estimated vanishing line $\hat{\mathcal{L}}$, we apply the following operations in each frame k:

- **Step 1.** Locate two image points t_k and b_k determined by the principal axis of the motion blob derived from tracking data, which form a line segment representing the object height h_k;
- **Step 2.** Locally tune t_k and b_k subject to the collinear constraint with two end points and the estimated vertical point $\hat{p}$. The two tuned end points are denoted by t'_k and b'_k;
- **Step 3.** Compute the projected point $t'^k_\perp$ from t'_k to line segment $\overline{(b_r, t_r)}$ by $t'^k_\perp = (\hat{p} \times b_r) \times (t'_k \times (\hat{\mathcal{L}} \times (b'_k \times b_r)))$;
- **Step 4.** Compute the object height as:

$$h_k = h_r \cdot \frac{d(t_r, \hat{p}) d(t'^k_\perp, b_r)}{d(\hat{p}, t'^k_\perp) d(t_r, b_r)}. \tag{5.15}$$

In step 2, we apply the constraint that two end points and the vertical point are collinear. This helps to correct inaccurate tracking data and principal axis estimation by pose variance, improving the precision of the measurement. The step can be formulated to estimate the maximum likelihood estimates (MLE) t'_k and b'_k of t_k and b_k, by minimizing the sum of *Mahalanobis distance* between t_k, b_k and their MLE estimates t'_k, b'_k:

$$\begin{aligned} \min_{t',b'} \;\; & (t'_k - t_k)^T \Lambda_t (t'_k - t_k) + (b'_k - b_k)^T \Lambda_b (b'_k - b_k) \\ & \qquad\qquad\qquad s.t. \;\; (t'_k \times b'_k) \cdot \hat{p} = 0, \end{aligned} \tag{5.16}$$

where the covariance of image points t, b is given by:

$$\Lambda_\kappa = \begin{pmatrix} \frac{\partial^2 I(\kappa_x,\kappa_y)}{\partial x^2} & \frac{\partial^2 I(\kappa_x,\kappa_y)}{\partial x \partial y} \\ \frac{\partial^2 I(\kappa_x,\kappa_y)}{\partial x \partial y} & \frac{\partial^2 I(\kappa_x,\kappa_y)}{\partial y^2} \end{pmatrix}^{-1}, \quad \kappa = \{t, b\}, \tag{5.17}$$

and $I(\kappa_x, \kappa_y)$ is the intensity at the point (κ_x, κ_y). The solution to the above optimization is first derived in [97] and is included in Appendix A. The entire procedure is illustrated by an example of walking pedestrians in Figure 5.5, in which the vertical point is invisible in the view due to the limit in image size.

5.4.2 Multi-frame fusion using least median estimator and Robbins-Monro stochastic approximation

The measurements from multiple frames include outlier observations due to bad tracking errors, articulated motion, and occlusions. It makes the process of using mean of the multi-frame estimates not robust. The least median of squares estimation (LMedS) [115] has the well-known property of being less sensitive to outliers. In the LMedS method, the cost function is defined as:

$$\theta^* \;\; = \;\; \arg\min_\theta median_k \; w_k^{-1}(h_k - \theta)^2. \tag{5.18}$$

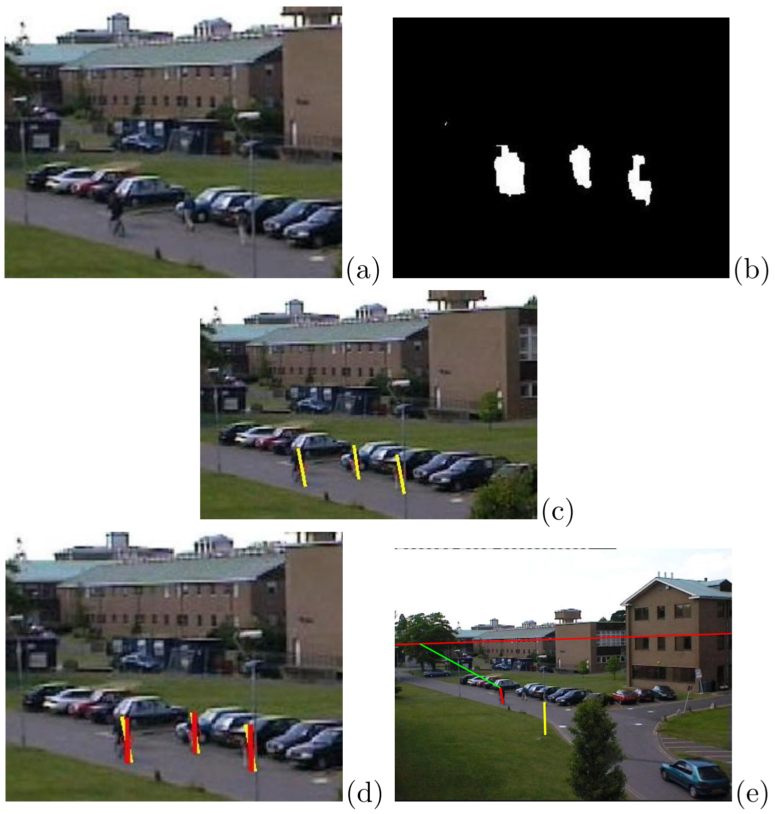


Figure 5.5: *The procedure for measuring objects heights from one frame. (a) The original image. (b) The obtained motion blobs indicating the moving objects in (a). (c) The estimated principal axes in yellow of the moving objects (step 1). (d) The principal axes in red after local tuning (step 2). (e) The thick red line depicts the vanishing line of the ground plane, the thin green line is the line segment of* $t \times (l \times (b \times b_r))$ *in equation (5.1), the yellow segment is the reference object with a tiny black circle on it depicting the projected position of the object.*

where w_k represents the covariance of h_k, computed by using the implicit function theorem. We will discuss the computation of w_k in section 5.5.

Since the cost function is a non-linear estimator without a closed form solution and the distribution of the noise is unknown because of the complexity of the error sources, the Robbins-Monro stochastic approximation (RMSA) [116, 117] algorithm provides an elegant tool to solve the optimization problem. We outline the specific LMedS problem using the standard RMSA notations: Let $\mathcal{H}$ be set of observations of size K, $\mathcal{W}$ be set of corresponding covariances, and $A(\theta) = \{f(\theta, h, w), h \in \mathcal{H}, w \in \mathcal{W}\}$ be the ordered set of the values of f at a fixed point θ. If $f(\theta, h_{(med)}, w_{(med)})$ is the median value, equation (5.18) becomes:

$$\theta^* = \arg\min_{\theta} f(\theta, h_{(med)}, w_{(med)}) = \arg\min_{\theta} F(\theta) \tag{5.19}$$

where $F(\theta)$ is differentiable w.r.t. θ. Initializing $\theta = \theta_0$, we calculate the gradient $grad(F(\theta))$ at the point. The recursion in the step i is given by $\theta_{i+1} = \theta_i - a_i * grad(F(\theta_i))$, where a_i is the gain sequence defined by $a_i = 0.1/(i+1)^{0.501}$, $f(\theta, h, w) = w^{-1}(h-\theta)^2$.

The iterative optimization algorithm is implemented using a fixed number of iterations. In the i^{th} step, it computes the following:

- Randomly chooses a subset $\mathcal{H}_s \in \mathcal{H}$ with S observations ($L < 0.5 \cdot K$) and the corresponding subset $\mathcal{W}_s \in \mathcal{W}$, and calculates the value set of $f(\theta_i, \mathcal{H}_s, \mathcal{W}_s)$ and sorts the values.

- Chooses the value corresponding to the median and the corresponding observation $h_{(med)}$ and covariance $w_{(med)}$, and calculates the gradient $\mathcal{D}(f)$ at

$(\theta_i, h_{(med)}, w_{(med)})$;

- Updates $\theta_{i+1} = \theta_i - a_i \cdot \mathcal{D}(f)$. Sets $i \leftarrow i + 1$.

5.5 Uncertainty analysis of measurement

Any metrology algorithm, whether manual or automatic, can achieve only a certain finite accuracy. In this section we will study how errors propagate through the measuring procedure. We focus on the uncertainty of the height measurement h_k, which is modeled by its covariance w. Two methods for estimating the uncertainty are exploited: an analytical method and a Monte Carlo method.

5.5.1 Analytical estimation of uncertainty on height measurement

The height measurement h_k derived from Eq. (5.15) depends directly on two MLE end points t'_k and b'_k, one vertical vanishing point $\hat{p}$ and one vanishing line $\hat{\mathcal{L}}$. Further, as illustrated in section 5.4 (e.g. step 2.), the two MLE end points are obtained through optimization approachs based on the tracking points t_k, b_k and the vanishing point $\hat{p}$. Therefore, h_k is dependent on $\{t_k, b_k, \hat{p}, \hat{\mathcal{L}}\}$ through $\{t'_k, b'_k\}$. In other words, the uncertainty propagates from $\{t_k, b_k, \hat{p}, \hat{\mathcal{L}}\}$ to $\{t'_k, b'_k\}$, and finally to h_k.

The computation of the covariance of h_k, denoted by w, involves two folds. The first fold is to derive the covariance of $\{t'_k, b'_k\}$ from parameters $\{b_k, t_k, \hat{p}\}$, in which the implicit function theorem has been used because of optimization function. The second fold is to compute to the final covariance w. With the covariance of $\{t'_k, b'_k, \hat{p}, \hat{\mathcal{L}}\}$, the problem is regarded as an explicit case and can be solved by applying the Jacobian matrix.

Define $\mathbf{x} \in \mathbb{R}^{10}$ as vector $\mathbf{x} = [b_k, t_k, \hat{p}, \hat{\mathcal{L}}]^T$, where $b_k \in \mathbb{R}^2$, $t_k \in \mathbb{R}^2$, $\hat{p} \in \mathbb{R}^3$ and $\hat{\mathcal{L}} \in \mathbb{R}^3$, with the assumption that the four components are statistically independent, the covariance matrix of $\mathbf{x}$ can be written as:

$$\mathbf{\Lambda_x} = \begin{bmatrix} \mathbf{\Lambda}_b & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Lambda}_t & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Lambda}_{\hat{p}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Lambda}_{\hat{\mathcal{L}}} \end{bmatrix} \tag{5.20}$$

Following the computation presented in Appendix B, the covariance w can be obtained as:

$$w = \nabla_h \mathbf{A} \mathbf{\Lambda_x} \mathbf{A}^T \nabla_h^T. \tag{5.21}$$

5.5.2 Monte Carlo estimation of uncertainty on height measurement

The Monte Carlo method estimates the uncertainty of the height measurement by exhaustive simulation [107]. It can validate the results of the analytical method. The inputs in each experiment include the plane vanishing line, the vertical vanishing point and the two end points of the object of interest, each of which is associated with a covariance. The simulation used 10,000 samples. The implementation steps are described in Table 5.1.

- For $j = 1$ to N (N is the number of simulation samples)
  - Generate a random vanishing line according to its covariance $\Lambda_{\hat{\mathcal{L}}}$;
  - Generate a random vertical vanishing point according to its covariance $\Lambda_{\hat{p}}$;
  - Generate two random end points b and t of the object of interest according to corresponding covariances Λ_b and Λ_t;
  - Estimate the MLE end points $\hat{b}$ and $\hat{t}$;
  - Compute the object height h_j by applying (5.15)
- The standard deviation of height estimation is computed as $\sigma_h^2 = \sum_{j=1}^{N}(h_j - \overline{h})^2/(N-1)$.

Table 5.1: Monte Carlo Simulation

5.5.3 Error estimation result

The noises in the experiments are basically from the image noises, which correspond to the estimates of the end-points of the object, the vanishing lines and the vertical point. To focus on exploring how the noises are propagated during the procedure of height estimation, we assume all the noises are Gaussian distributions with zero means and uniform standard deviation values. We have tried five noise levels in the experiments. Both the analytical estimations and Monte Carlo simulation results are reported in Table 5.2 with corresponding relative errors. The two sets of results are coincident.

σ (pixel)	Analytical result		Monte Carlo result		Relative error
	σ_h	$\frac{\sigma_h}{h}$	σ'_h	$\frac{\sigma'_h}{h}$	$\frac{\|\sigma_h - \sigma'_h\|}{\sigma'_h}$
0.2	0.9290	1.02%	1.0163	1.12%	8.59%
0.5	1.4573	1.62%	1.5499	1.71%	5.98%
1	2.0443	2.25%	2.1086	2.32%	3.05%
1.5	2.4969	2.75%	2.5507	2.81%	2.11%
2	2.8816	3.17%	2.9099	3.20%	0.97%

Table 5.2: Error estimation results with the estimated height $h = 91inch$ using the analytical approach and Monte Carlo simulation. Relative errors are also computed.

In our algorithm, the measurement uncertainty results from two sources, the image noise, which affects the estimate of the ending points of the object, the vanishing line and the vertical vanishing point; and the calibration error, which affects the estimate of the vanishing line and the vertical vanishing point. It is also worth mentioning here that the former occurs in both the manual-calibrated metrology and

$\sigma_n = 1$	Calibration error $\sigma_c = \sigma_n \times \gamma$				
$\sigma_n^2 = 4.1380$	$\gamma = 2$	$\gamma = 5$	$\gamma = 10$	$\gamma = 20$	$\gamma = 30$
$\sigma_{n,c}^2$	4.1653	4.2473	4.3839	4.6572	4.9304
$\frac{\|\sigma_c^2 - \sigma_n^2\|}{\sigma_n^2}$	0.66%	2.64%	5.94%	12.55%	19.15%

Table 5.3: Uncertainty analysis under different calibration errors. σ_n is the image noise, σ_c is the calibration noise, $\sigma_{n,c}$ is the entire uncertainty concerning both the image noise and calibration error.

the auto-calibrated metrology, while the latter is only introduced in auto-calibrated metrology. Therefore, it is useful to study how these two sources of errors propagate, with respect to the entire error propagation, to evaluate the robustness of the proposed algorithm. Table 5.3 reports another experiment we conducted in which we simulated extra calibration errors and added them to the height estimate procedure with the image noises. Basically, the extra calibration errors were added to the estimate error of the vanishing line and the vertical vanishing point. We computed the proportions of the calibration related uncertainties with respect to the image noise related uncertainty. From the reported results, we conclude that the calibration error contributes only a relative small portion to the entire measurement uncertainty, i.e., automatic calibration does not vastly affect the accuracy of final results, though it introduces extra errors. This happens mainly because the automatic calibration errors affect only the vanishing points and the vertical points. When computing the height, the contributions of the vanishing points and the vertical points are limited compared to those of the ending points of the object. The result validates that our algorithm with self-calibration can achieve reasonable performance.

5.6 Experiments and Discussion

5.6.1 Vanishing line estimation

We have applied our method to different sets of outdoor surveillance video sequences taken from stationary cameras. Moving objects include vehicles and humans; tracking data contain both feature point based and motion blob based data for subsequent clustering. We perform morphological operations on motion blobs for smoothing and denoising. After initialization, the EM algorithm usually takes 3 to 10 iterations to converge. In one possible reason for fast convergence, the motion blob information provides a strong clue to trajectory clustering. Some results appear in Figure 5.6. The first sequence has two major sets of trajectories: one from a vehicle, the other from a pedestrian. The second sequence explores three sets of trajectories: one from a car, the other two from riding and walking humans, respectively. The third sequence has one human running in the scene. We use two sets of trajectories from the same human, but during different time intervals and in different directions. The fourth sequence contains trajectories from two vehicles and two pedestrians, both along different directions and during different time intervals. A common characteristic shared by all these four scenes is a lack of solid background parallel lines, which makes manual labeling of vanishing lines a highly unreliable task.

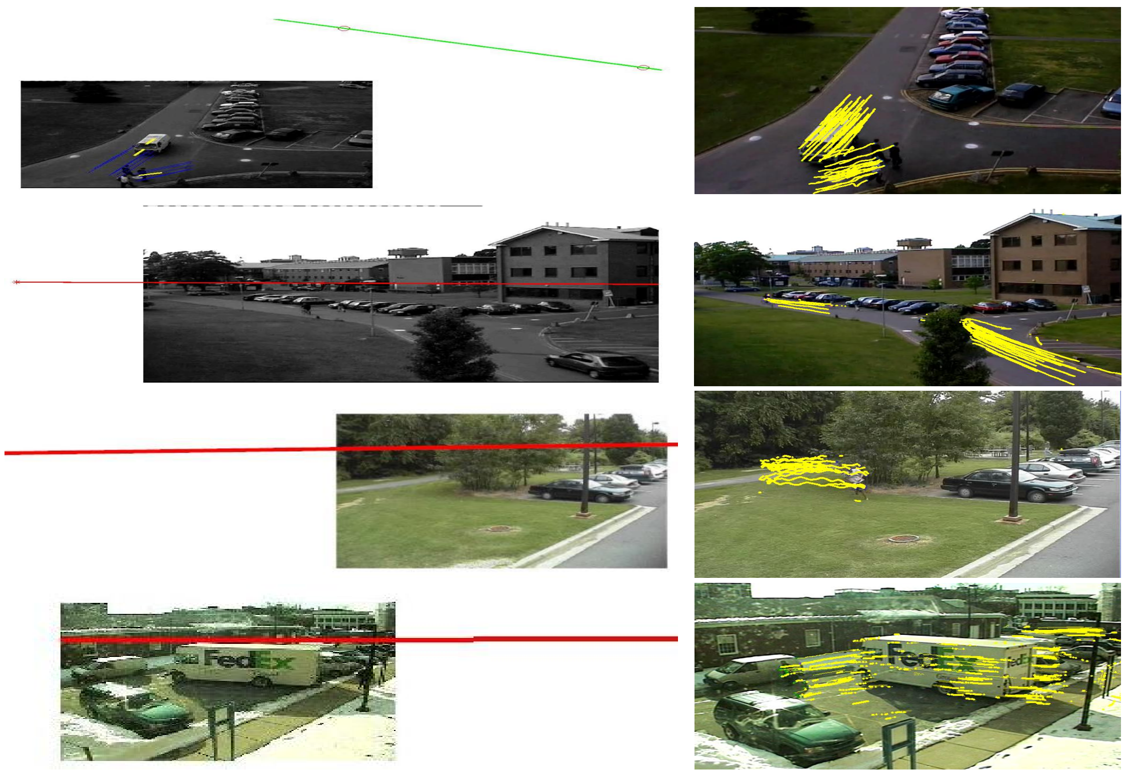


Figure 5.6: *Examples of vanishing lines being automatically estimated in video sequences. Left column: scenes with vanishing lines; Right column: trajectory observations. We observe many discontinuous and short trajectories in the last sequence, which are caused by instant motions of objects. Trajectories lasting no more than 20 frames are not counted in estimation.*

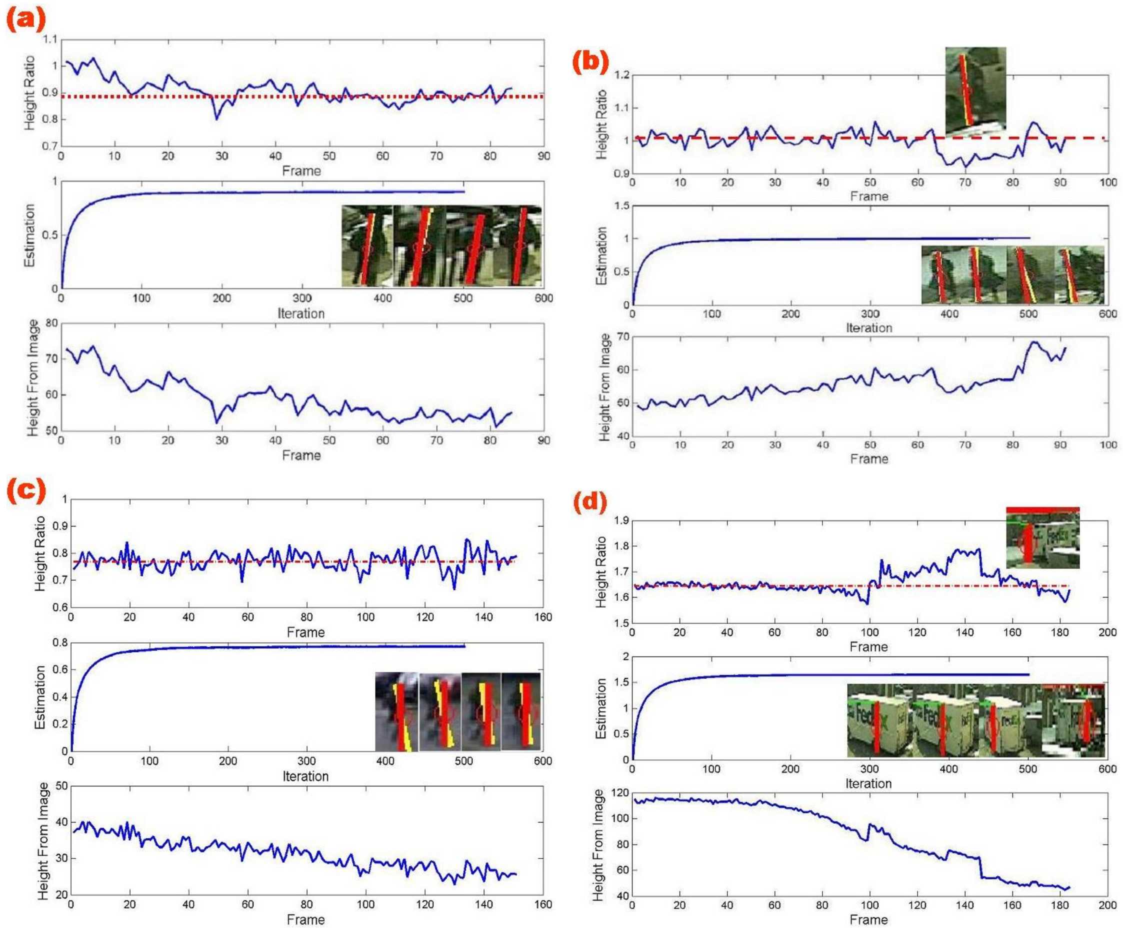


Figure 5.7: *Examples of multi-frame measurements. The first plot in each set of examples is a "height ratio vs frame" plot, the second plot dedpicts the "estimation convergence by RMSA", the third plot gives "image height vs frame" as a comparison. The red dashed lines in "ratio vs frame" plot give the optimal estimates obtained using the RMSA algorithm. The small cropped images show how the objects appear in different frames, with red lines representing the constrained heights. The small images in the* 1st *plot of example (b) and (d) explain the underlying reason of perturbations in certain intervals. (b) shows the pedestrian is occluded and (d) shows that the bottom point of the object is undetectable. Additional discussions in the text.*

5.6.2 Single-view height measurement

To measure the height of an object, we need to select an object as a reference in the same sequence, then compute the height ratio of the object of target w.r.t the reference object in each frame. Therefore, the height of the target is determined once the height of the reference object is available. In fusion, we set 10 as the number of random-chosen subset members S, 500 as the iteration times. Figure 5.7 shows estimations of object heights from multi-frame observations. For each set of examples, we show three plots: "ratio vs frame" and "image height vs frame" plot the worldly height ratio (w.r.t. the reference height) and the image height (image distance) of observed object in each frame, respectively; "estimation convergence by RMSA" records the converging procedure of applying the RMSA method to multi-frame measurements. From the descending (ascending) height curves in "image height vs frame" plots, we infer that the first, third and fourth objects are moving away from the camera, and the second object is moving toward to the camera. Meanwhile, in "ratio vs frame" plots the ratio curves remain even, indicating that the ratio is not influenced by the distance of object to camera. Therefore, the perspective projection is recovered. We observe slight periodic perturbations existing in all measurement results, incurred by the periodic walking property of people that affects the measurement. The "convergence" plot shows how the estimation procedure converges. The curves show abrupt drops in both "ratio" and "image height" plots in example (b) from frame 65 to 82, due to occlusion in these frames. However, this part of the data does not influence the final estimate. We estimate the optimal value as 1.0050, while the mean of the data is 0.9560. The optimal value excluding the error part is 1.0028. In example (d), we observe a hump from frame 110 to 150, which occurs when the truck wheels are both invisible and location of the bottom point of truck height is hard to detect. Using the RMSA method, we find that the best estimate is 1.6401 while the mean is 1.6601. Again as a comparison, the best estimate of data excluding the error part is 1.6374. Therefore, in such cases, the LMedS can generate more robust results.

We have explored three different scenarios in our experiments: (1) moving target, still reference object; (2) moving reference object, still target; (3) moving reference object and moving target. Figure 5.8 illustrate an example that estimates the height of a moving target (a FedEx truck) with a still UPS truck as a reference. The upper images demonstrate the metrology procedure. In the left bottom image, the red line and the blue line are the estimated vanishing line and the ground truth, respectively. Figure 5.9 shows a moving reference object (a FedEx truck) and a still target (a pole). The upper images interpret the measurement procedures. The bottom diagrams record the estimated height ratios and the image heights of the pole in successive frames. Figure 5.10 shows an example with a moving target (a pedestrian) and a moving reference (a FedEx truck). The measurement results and corresponding ground truths are listed in Table 5.4. The major deviation results from tracking errors. The "MT+MR" case yields the most inaccurate results because errors arise from tracking both the target and the reference object.

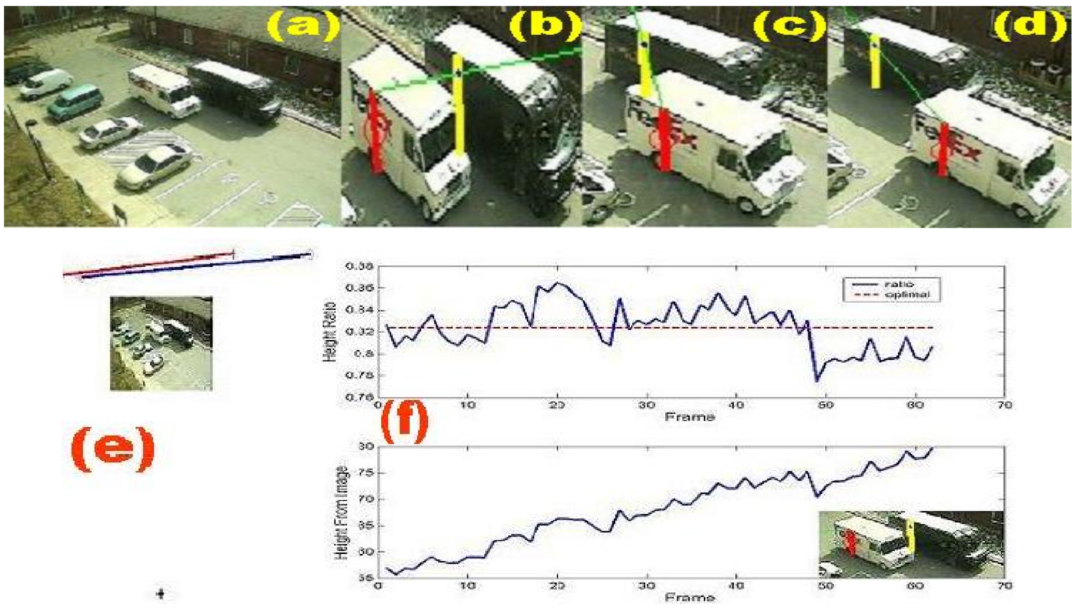
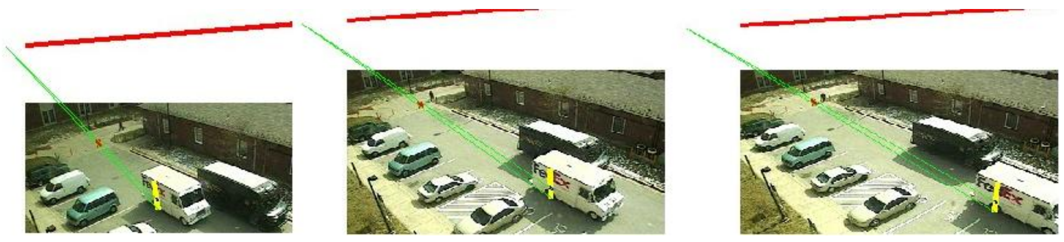
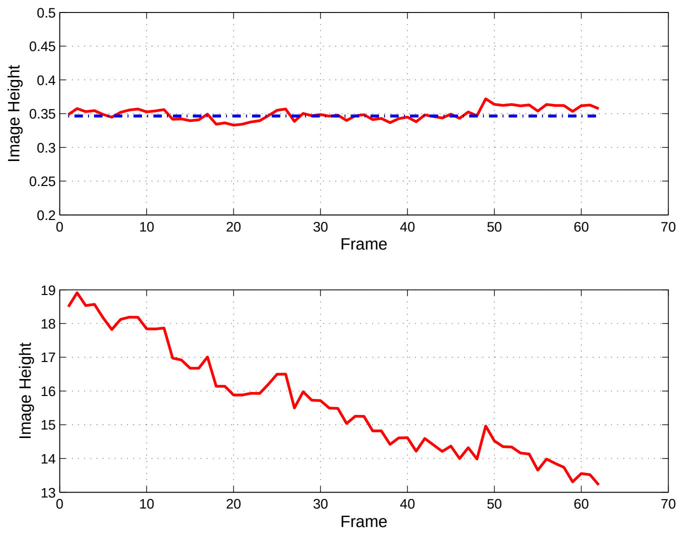


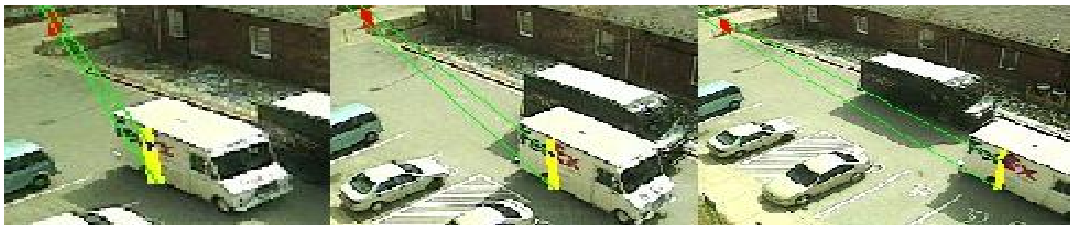
Figure 5.8: *An example of a moving object and a still reference object (MT/SR). (a) is an original frame in the sequence. (b)(c)(d) show some mensuration images. The red lines represent the height of the FedEx truck, the yellow lines represent the height of the UPS truck. (e) shows the vanishing line and the vertical point (marked by "+"). The red line is the estimated line, and the blue line the ground truth. The upper plot in (f) shows the height ratio of a moving FedEx truck to a still UPS truck, and the bottom plot in (f) is the image height of the FedEx truck throughout frames. The ground truth of the ratio of a FedEx truck height to a UPS truck height is* 0.8381, *our estimation is* 0.8233 ± 0.0218.

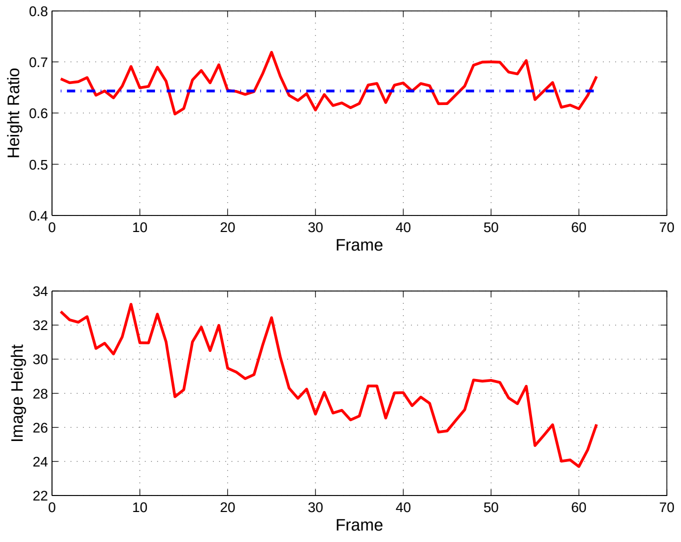

(a)



(b)

Figure 5.9: *An example of a still target and a moving reference (ST/MR). (a) illustrates some mensuration images in the sequence. The red line inside each frame depicts a still pole we want to measure, and the yellow line depicts the reference object, a moving FedEx truck. The red lines outside the frames are the estimated vanishing lines. Green lines are used for geometry deductions. In the upper diagram of (b), the red solid curve records the relative ratios of the target with respect to the reference in each frame, the dashed blue line is the final estimate. The red solid curve in the bottom diagram of (b) represents the image heights of the target given the reference height.*


(a)



(b)

Figure 5.10: *An example of a moving target and a moving reference (MT/MR). (a) illustrates some mensuration images in the sequence. The red line inside each frame depicts a moving pedestrian whose height we want to measure, and the yellow line depicts the reference height of a moving FedEx truck. In the upper diagram of (b), the red solid curve records the relative ratios of the target with respect to the reference, the dashed blue line is the final estimate. The red solid curve in the bottom diagram of (b) plots the image heights of the target given the reference height.*

	truth (inch)	estimate (inch)	covariance (inch)	error (inch)	relative error
MT/SR	88	90.5630	2.3180	2.5630	2.91%
ST/MR	34.10	32.7459	0.7788	1.3541	3.97%
MT/MR	64.36	60.7008	2.6176	3.6592	5.74%

Table 5.4: Height measurement results under different scenarios. (M: moving; S: still; T: target; R: reference.)

5.6.3 Two-view object height measurement

In this section, we applied the height estimation method to the Honeywell dataset [1]. Two static cameras with different view angles are used to acquire a pair of videos of the subject walking the same path in a hallway, as illustrated in Fig. 5.11. There are nine subjects, with each subject appearing between one and four times (maximum). The subject changes part of his/her clothing in different acquisition sessions. In total, 30 pairs of video sequences are acquired, all with different clothing.

Recognizing nonrigid object, such as a person, across different views and different clothing presents a challenging research topic. Conventional holistic appearance-based approaches [118, 119, 120, 121], like template matching and PCA, have difficulty in handling the large appearance variation. Part-specific appearance-based approaches, for example the Geometric Transform (GeT) in [122, 123], showed promising results. The GeT characterizes certain invariance under nonrigid-deformation

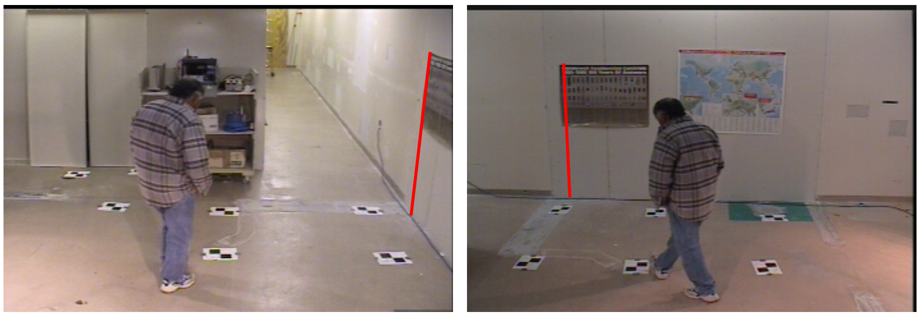
[1]The video sequences in this dataset were acquired by Honeywell Corporation under the HSARPA contract 433829 monitored by the Office of Navy Research (ONR).

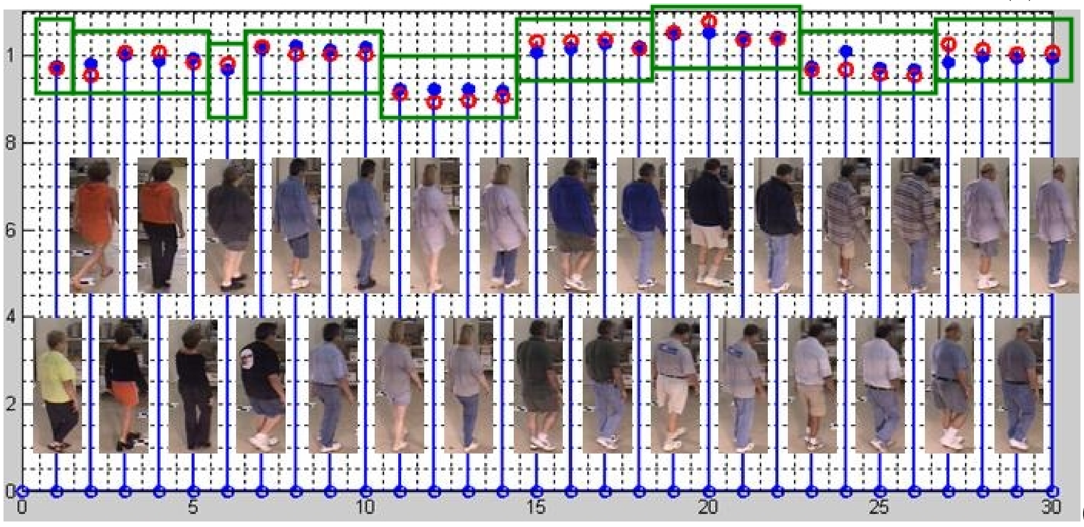
and view change, and uses the silhouette information to combat the appearance variation caused by clothing difference. A person's height is independent of view and clothing. If estimated correctly, it provides an invariant feature. When this invariant feature is combined with recent advances in nonrigid object recognition, such as [123], the object recognition accuracy should be improved. However, in this chapter, we do not delve into this particular avenue, leaving it for future work.

The two cameras in use share a common reference object, which is the black board marked by the red lines in Fig. 5.11(a). However, we do not know its exact height, hence all our height estimation is subject to an unknown scale factor. Fig. 5.11(b) presents the height estimation results. Its x-axis uses the cropped image of the subject in one view to represent each walking session, and its y-axis shows the estimated height values with two dots, one red dot for view 1 and one blue dot for view 2. It is clear that for almost all 30 walking sessions, the subject's heights are measured consistently across two views. Even for the same subject in difference sessions, as marked out by the green boxes, his/her estimated height values are quite consistent, regardless of the view and the clothing.

5.6.4 Discussion on GPCA vs. our algorithm

Generalized principal component analysis (GPCA) [124] is an algebraic, geometric approach to the problem of estimating a mixture of linear subspaces from sample data points. In other words, the problems lie in identifying each subspace from samples without knowing the number of subspaces and which sample point


(a)



(b)

Figure 5.11: *(a) The two views of one walking session in the Honeywell dataset. (b) The estimated height values.*

belong to which subspace. One of its applications is to detect vanishing points [125] given N sets of perspective projections of parallel lines. In our problem, it provides an alternative way to estimate the vanishing points by samples of feature point based trajectory data. Therefore, it is interesting to compare the performances of the GPCA and our proposed EM-based approach.

In the GPCA method, the problem of estimating n vanishing points from the set of N lines, without knowing which subsets of lines intersect in the same point, is equivalent to estimating a collection of n planes in $\mathbb{R}^3$ with normal vectors $v_i \in \mathbb{R}^2$ from sample lines $l_j \in \mathbb{R}^2$. We set the sample lines as the trajectories of all feature points obtained directly from tracking results. A polynomial factorization algorithm (PFA) is applied to estimate the number of subspaces. Unlike the plain lines extracted from man-made environments, the motion trajectory lines extracted from tracking results tend to be more noise-corrupted. In such a degenerate case, we have to use a pre-specified threshold for matrix rank estimation. Consequently, the results are fairly sensitive to threshold settings. We report experiment results in Table 5.5 with various thresholds. The results show that, in the presence of noise, the estimate of the number of subspaces by the GPCA method varies when using different threshold setting. We also notice that even after we have fixed the number of the subspaces, degenerate data samples may still not achieve valid solutions when estimating each subspace.

The possible reasons that our proposed algorithm outperforms the plain GPCA method on robustness and efficiency are: 1) The trajectory data are noise-corrupted. Our EM-based algorithm uses statistical method to suppress the influences from

	GPCA(pre-defined threshold)				EM-based
	0.0002	0.0015	0.015	0.035	algorithm
Experiment 1	N/A	4	2	N/A	2
Experiment 2	5	N/A	N/A	4	3

Table 5.5: Subspace number estimation using two algorithms. The GPCA algorithm has different estimates on subspace numbers with different threshold settings. "N/A" represents the case with no valid solution.

outliers. On the contrary, GPCA without any denoising processes hardly generates satisfactory results when working on noisy data; 2) The setting of the threshold in GPCA is data dependent; while our algorithm takes the blob motion into account, which guarantees a relative robust estimation.

5.7 Summary

We have presented an automatic method for robustly measuring object heights from video sequences in this chapter. We first recover the minimal calibration of the scene by tracking moving objects (blob based and feature point based), then applying mensuration algorithm to each of the frames, and finally fusing the multi-frame measurements, using the LMedS as the cost function and the RMSA as the optimization algorithm. To clarify the contribution of our algorithm, we make a comparison in Table 5.6 between single video metrology, two-view metrology and video metrology. The two-view metrology algorithm attempts to solve metrology problems with stationary objects and moving cameras. Consequently, it self-calibrates in an entirely new way, which has many similarities to approaches in the structure from motion (SfM) problem. In contrast, our algorithm concentrates on stationary cameras and moving objects.

	Single View Metrology [97]	Two-View Metrology [101]	Video Metrol
camera parameters	uncalibrated	uncalibrated	uncalibrate
calibration type	minimal	minimal	minimal
calibration method	human interactive	computer automatic	computer autor
data type	single image *	two frames	video
sequence feature	N/A	stationary target, moving camera	moving target, statio
object location	manual	tracker	tracker
measurement	single frame	single frame	multiple fran
occlusion handling	N/A	N/A	RMSA
error source	human operation	tracker	tracker, non-rigid

Table 5.6: Comparison between single view metrology, two-view metrology and our method. *: Single view metrology can operate on video sequences, but is still based on individual frames.

The proposed method can be used in surveillance system and forensic investigation. We plan to generalize the method to moving camera scenarios, where a stabilization process is needed to compensate for the position change of related geometry features, such as the vanishing line and the vertical point. It is also worth noting that automatically estimating the minimal calibration has significant usages not only in mensuration, but also in many other interesting applications, such as normalizing object sizes in perspective images for classification, rectifying perspective images and reconstructing 3D object structures.

Chapter 6

Conclusions

6.1 Summary of the Dissertation

This doctoral dissertation focused mainly on tracking and mensuration in surveillance videos. The first part of the thesis discussed several object tracking approaches, based on the different properties of tracking targets. Two tracking algorithms have been proposed: one for rigid objects and the other for non-rigid objects. A new generic regulation model has been introduced as a module for the generalized tracking framework. The second part of the thesis covered the video mensuration. We discussed a video-based height measuring algorithm.

Here are some the thesis' key contributions made:

- For airborne videos, where the targets are usually small and have low resolutions, an approach of building motion models for foreground/background has been proposed, in which the foreground target is simplified as a rigid object. To make the tracker more robust and accurate, a motion model, consisting of both background and foreground motion parameters, is built. Multiple cues are adaptively integrated in a system observation model when estimating the likelihood functions.

- For non-rigid models, we proposed an active contour based algorithm. We decompose the estimation of non-rigid movements into three parts; transition estimation, deformation detection, and shape regulation. First, we employ a particle filter to estimate the affine transform parameters between successive frames. Second, by using a dynamic object model, we generate a probabilistic map of deformation to reshape its contour. Finally, we project the updated model onto a trained shape subspace to constrain deformations within possible object appearances. This enables us to reconstruct the occluded parts of the active contour and accurately track it, while allowing changes specific to the given object.

- We proposed a new regulation model to add constraints to tracking frameworks: the active appearance Markov chain (AAMC) model. It integrates a statistical model of shape, appearance and motion. In the AAMC model, a Markov chain represents the switching of motion phases (poses), and several pairwise active appearance model (P-AAM) components characterize the shape, appearance and motion information for different motion phases. Additionally, the configuration of the proposed model lends itself to adaptability during tracking.

- We proposed an automatic method of using video motions to recover the minimal calibration information and use it to extract object heights. From videos acquired by an uncalibrated stationary camera, we first recover the vanishing line and the vertical point of the scene based on tracking moving objects that lie primarily on a ground plane. Using geometric properties of moving objects, a probabilistic model is constructed for simultaneously grouping trajectories and estimating vanishing points. Then, we apply a single view mensuration algorithm to each of the frames to obtain height measurements. We finally fuse the multi-frame measurements using the LMedS as the cost function and the Robbins-Monro stochastic approximation (RMSA) technique. The method requires less human supervision, has increased flexibility and improve robustness.

6.2 Future Direction of Research

Both video tracking and mensuration discussed in this dissertation could be expanded in multiple ways. The following lists some potential directions to explore in the near future.

Model Regulation We have proposed a regulation model based on shape, appearance and motion information for tracking frameworks. It is straightforward to consider extending the usage of the model for object detection and recognition.

Video Mensuration for Multiple View Recognition We conducted a simple experiment on multiple view video mensuration. The preliminary result seems promising in its use of height information to recognize objects with different appearances and views. We could try some optimization approaches we could try to improve the measurement results with multi-view videos. For example, we can apply bundle-adjustment to obtain more accurate estimates of vanishing lines. The recognition results should show improvement.

Generalized Video Mensuration The dissertation is focused on one single problem on video mensuration: measuring object height. We would like to extend the method to measure other metrics, such as object volumes, object widths, etc. These metrics are useful in several applications, including the medical field, object recognition, and robotics.

Radian Distortion At this time, radian distortion is not a concern in the video mensuration approach because we concentrate on self-minimal-calibration of the scene and stochastic approximation of the multi-frame estimates. However, radian distortion cannot be ignored. It would be interesting to apply our method on videos after radian distortion corrections.

Chapter A

Appendix

A.1 Robbins-Monro Optimization

Using RMSA [116], the problem of seeking the LMedS can be presented as: Let $\mathcal{H}$ be the set of observations of size K, $\mathbb{C}$ be set of corresponding variances, $A(\theta) = \{f(\theta, h, c), h \in \mathcal{H}, c \in \mathbb{C}\}$ be the ordered set of the values of f at a fixed point θ. Eqn.(5.18) becomes:

$$\theta^* = \arg\min_{\theta} f(\theta, h_{(med)}, c_{(med)}) = \arg\min_{\theta} F(\theta) \tag{A.1}$$

where $F(\theta)$ is differentiable with respect to θ. Initializing $\theta = \theta_0$, we calculate the gradient $grad(F(\theta))$ at the point. The recursion in the step i is given by:

$$\theta_{i+1} = \theta_i - a_i * grad(F(\theta_i)), \tag{A.2}$$

where a_i is the gain sequence defined by $a_i = 0.1/(i+1)^{0.501}$, $f(\theta, h, c) = c^{-1}(h-\theta)^2$. After substituting the gradient in (A.2), we have

$$\theta_{i+1} = \theta_i + 2\frac{a_i(h - \theta_i)}{c}. \tag{A.3}$$

The optimal estimate θ^* is achieved after a number of recursions.

A.2 Proof of Theorem 1

Given $\xi \in \mathbb{R}^{10}$ as vector $\xi = [b, t, \hat{p}, \hat{\mathcal{L}}]^T$, where $b \in \mathbb{R}^2$, $t \in \mathbb{R}^2$, $\hat{p} \in \mathbb{R}^3$ and $\hat{\mathcal{L}} \in \mathbb{R}^3$, and with the assumption that the four components are statistically independent, the covariance matrix of ξ can be written as:

$$\mathbf{\Lambda}_\xi = \begin{bmatrix} \mathbf{\Lambda}_b & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Lambda}_t & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Lambda}_{\hat{p}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Lambda}_{\hat{\mathcal{L}}} \end{bmatrix} \tag{A.4}$$

(1) We first derive the covariance of the two MLE end points. As claimed, we must use the implicit function [126] because the two MLE end points are obtained through minimization. The implicit function theorem states that if $\mathbf{x} \in \mathbb{R}^m$, $z \in \mathbb{R}^n$, $\mathcal{C} : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable mapping, $\mathcal{C}(\mathbf{x}, z) = 0$, where $z = \varphi(\mathbf{x})$ with a mapping $\varphi : \mathbb{R}^n \to \mathbb{R}^m$. The local minimum of $\mathcal{C}(\mathbf{x}, z)$ with respect to z can be solved uniquely as z_0, and the covariance matrix of z_0 has a closed form.

The function $\mathcal{C}$ in our problem is the objective function in MLE points optimization. Therefore, we rewrite the MLE objective function as $\mathcal{C} : \mathbb{R}^7 \times \mathbb{R}^4 \to \mathbb{R}$, as follows:

$$\begin{aligned}
\Gamma &= \begin{pmatrix} \bar{b} - b \\ \\ \bar{t} - t \end{pmatrix} && \text{(A.5)} \\
\mathbf{x} = \xi_7 &= [b^1, b^2, t^1, t^2, \hat{p}^1, \hat{p}^2, \hat{p}^3] && \text{(A.6)} \\
z &= [\bar{b}^1, \bar{b}^2, \bar{t}^1, \bar{t}^2] && \text{(A.7)} \\
\mathcal{C}(\xi_7, z) &= \Gamma^T \begin{pmatrix} \Lambda_b & \mathbf{0} \\ \\ \mathbf{0} & \Lambda_t \end{pmatrix} \Gamma + \lambda(\bar{t} \times \bar{b}) \cdot \hat{p} && \text{(A.8)}
\end{aligned}$$

where z is a variable dependent on ξ_7. We define the MLE points as $\hat{\zeta} \in \mathbb{R}^4$ such that $\hat{\zeta} = [b', t']$. $\hat{\zeta}$ is a local minimum of $\mathcal{C}(\xi_7, z)$ with respect to z. Furthermore, the covariance matrix of the MLE end points $\mathbf{\Lambda}_{\hat{\zeta}}$ can be derived following the implicit function theorem as:

$$\mathbf{\Lambda}_{\hat{\zeta}} = \mathbf{H}^{-1} \frac{\partial \mathbf{\Phi}}{\partial \xi_7} \mathbf{\Lambda}_{\xi_7} \frac{\partial \mathbf{\Phi}}{\partial \xi_7}^T \mathbf{H}^{-T}, \tag{A.9}$$

where

$$\begin{aligned}
\mathbf{\Lambda}_{\xi_7} &= \begin{bmatrix} \mathbf{\Lambda}_b & \mathbf{0} & \mathbf{0} \\ \\ \mathbf{0} & \mathbf{\Lambda}_t & \mathbf{0} \\ \\ \mathbf{0} & \mathbf{0} & \mathbf{\Lambda}_{\hat{p}} \end{bmatrix}, && \text{(A.10)} \\
\mathbf{\Phi} &= \left(\frac{\partial \mathcal{C}}{\partial z} \right)^T, && \text{(A.11)} \\
\mathbf{H} &= \frac{\partial \mathbf{\Phi}}{\partial z}, && \text{(A.12)}
\end{aligned}$$

$\mathbf{H}$ is the Hessian matrix of $\mathcal{C}$.

(2) The second step is quite straightforward. We use a Jacobian matrix to compute c as:

$$c = \nabla_h \mathbf{\Lambda}_{\hat{\xi}} \nabla_h^T \tag{A.13}$$

where $\mathbf{\Lambda}_{\hat{\xi}}$ is the covariance matrix of parameters $\hat{\xi} = (\hat{\zeta}, \hat{p}, \hat{\mathcal{L}})$:

$$\mathbf{\Lambda}_{\hat{\xi}} = \begin{bmatrix} \mathbf{H}^{-1}\frac{\partial \mathbf{\Phi}}{\partial \xi_7}\mathbf{\Lambda}_{\xi_7}\frac{\partial \mathbf{\Phi}}{\partial \xi_7}^T\mathbf{H}^{-T} & \mathbf{H}^{-1}\frac{\partial \mathbf{\Phi}}{\partial \hat{p}}\mathbf{\Lambda}_{\hat{p}} & \mathbf{0} \\ \mathbf{\Lambda}_{\hat{p}}\frac{\partial \mathbf{\Phi}}{\partial \hat{p}}^T\mathbf{H}^{-T} & \mathbf{\Lambda}_{\hat{p}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Lambda}_{\hat{\mathcal{L}}} \end{bmatrix} \tag{A.14}$$

Combined with (A.4) and (A.9), the above equation can be simplified as:

$$\mathbf{\Lambda}_{\hat{\xi}} = \mathbf{A} \bullet \mathbf{\Lambda}_{\xi} \bullet \mathbf{A}^T \tag{A.15}$$

$$\mathbf{A} = \begin{bmatrix} \multicolumn{2}{c}{\mathbf{H}^{-1} \cdot \frac{\partial \Phi}{\partial \xi_7}} & \mathbf{0}_{4\times 3} \\ \mathbf{0}_{3\times 4} & \mathbf{I}_3 & \mathbf{0}_{3\times 3} \\ \multicolumn{2}{c}{\mathbf{0}_{3\times 7}} & \mathbf{I}_3 \end{bmatrix} \tag{A.16}$$

Finally, the entire computation of the covariance c can be compactly rewritten as:

$$c = \nabla_h \mathbf{A}\mathbf{\Lambda}_{\xi}\mathbf{A}^T\nabla_h^T \tag{A.17}$$

If interested, refer to [127, 128], for the derivations of the matrix $\mathbf{H}$, $\frac{\partial \Phi}{\partial \xi_7}$, $\mathbf{\Lambda}_{\hat{p}}$, $\mathbf{\Lambda}_{\hat{\mathcal{L}}}$ and ∇_h in (A.14) and (A.17).

Bibliography

[1] Y. Bar-Shalom and T.F. Fortman, *Tracking and Data Association*, Academic Press, 1988.

[2] G. Verghese, K.L. Gale, and C.R. Dyer, "Real-time motion tracking of three-dimensional object," *Proc. IEEE Robotics Automat. Conf.*, pp. 1998–2003, 1990.

[3] J. Shi and C. Tomasi, "Good features to track," *Proceeding of IEEE Computer Vision and Pattern Recognition 1994*, pp. 593–600, June 1994.

[4] C. Wren, A. Azarbayejanni, T. Darrel, and A. Pentland, "Pfinder: Real-time tracing of the human body," *IEEE Trans. on Pattern Analysis Machine Intelligence*, vol. 19, no. 7, pp. 780–785, July 1997.

[5] C. Stauffer and E. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, August 2000.

[6] A. Azarbayejani and A. Pentland, "Recursive estimation of motion, structure, and focal length," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 562–575, June 1995.

[7] M. Isard and A. Blake, "Contour tracking by stocastic propagation of conditional density," in *Proceeding of Proc. of European Conference on Computer Vision*, Cambridge, England, 1996, pp. 343–356.

[8] N.J. Gordon, D.J. Salmond, and A. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEEE Proceedings on Radar and Signal Processing*, vol. 140, pp. 107–113, 1993.

[9] M. Isard and A. Blake, "Condensation: conditional density propagation for visual tracking," *International Journal on Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[10] S. Zhou, V. Kruger, and R. Chellappa, "Probabilistic recognition of human faces from video," *Computer Vision and Image Understanding*, vol. 91, pp. 214–245, July/August 2003.

[11] G. Qian and R. Chellappa, "Structure from motion using sequential monte carlo methods," *Intl. Jl. Computer Vision*, vol. 59, no. 1, pp. 5–31, August 2004.

[12] T. Jebara and A. Pentland, "Parameterized structure from motion for 3d adaptive feedback tracking of faces," *Proc. of IEEE Computer Society Conf. on Computer Vision Pattern Recognition*, pp. 144–150, June, 1997.

[13] R. T. Collins, A.J. Lipton, and T. Kanade, "Introduction to the special section on video surveillance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 745–746, August 2000.

[14] J.M. Rehg, M. Loughlin, and K. Waters, "Vision for a smart kiosk," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition 97*, San Juan, Puerto Rico, June 17-19 1997, pp. 690–696.

[15] R. Collins, A. Lipton, and T. Kanade, “A system for video surveillance and monitoring,” Tech. Rep., 1999.

[16] R.J. Morris and D.C. Hogg, “Statistical models of object interaction,” *International Journal of Computer Vision*, vol. 37, no. 2, pp. 209–215, 2000.

[17] I. Haritaoglu, D. Harwood, and L.S. Davis, “w^4: Real-time surveillance of people and their activities,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809–830, August 2000.

[18] M. Irani and P. Anandan, “Video indexing based on mosaic representation,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 86, no. 5, pp. 905–921, May 1998.

[19] J.W. Davis and A.F. Bobick, “The representation and recognition of action using temporal templates,” in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition 97*, San Juan, Puerto Rico, June 17-19 1997, IEEE Computer Society Press.

[20] M. Brand, N. Oliver, and A. Pentland, “Coupled hidden markov models for complex action recognition,” in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition 97*, San Juan, Puerto Rico, June 17-19 1997, IEEE Computer Society Press.

[21] N.M. Oliver, B. Rosario, and A.P. Pentland, “A bayesian computer vision system for modeling human interactions,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, August 2000.

[22] S. Avidan and A. Shashua, “Novel view synthesis by cascading trilinear tensor,” *IEEE Trans. Visualization and Computer Graphics*, vol. 4, pp. 293–306, 1998.

[23] M. Isard and A. Blake, “Icondensation: unifying low-level and high-level tracking in a stochastic framework,” *Proc. of European Conference on Computer Vision*, pp. 893–908, June 2-6 1998.

[24] Omar Javed, K. Shafique, and Mubarak Shah, “A hierarchical approach to robust background subtraction using color and gradient information,” in *Proc. of IEEE Workshop on Motion and Video Computing*, Orlando, USA, December 5-6 2002.

[25] M.J. Black and D.J. Fleet, “Probabilistic detection and tracking of motion discontinuities,” in *Proc. of IEEE International Conference on Computer vision*, Corfu, Greece, September 2000, pp. 551–558.

[26] M.J. Black and A.D. Jepson, “Eigentracking: robust matching and tracking of articulated objects using a view-based representation,” in *Proc. of 4th European Conference on Computer Vision*, Cambridge, UK, April 13-14 1996, vol. 1.

[27] S. Zhou, R. Chellappa, and B. Moghaddam, “Visual tracking and recognition using appearance-adaptive models in particle filters,” *IEEE trans. on Image Processing*, vol. 13, no. 11, pp. 1491–1506, November 2004.

[28] C. Rasmussen and G.D. Hager, “Probabilistic data association methods for tracking complex visual objects,” *IEEE Trans. on Pattern Analysis Machine Intelligence*, vol. 23, no. 6, pp. 560–576, June 2001.

[29] D. Comaniciu, V. Ramesh, and P. Meer, “Real-time tracking of non-rigid objects using mean shift,” in *IEEE Computer Vision and Pattern Recognition*, Hilton Head, SC, June 13-15 2000, pp. 142–149.

[30] D.C. Hogg, *Interpreting images of a known moving object*, Ph.D. thesis, University of Sussex, UK, 1984.

[31] K. Rohr, *Human movement analysis based on explicit motion models*, chapter 8, pp. 171–198, Kluwer Academic Publishers, Dordrechit Boston, 1997.

[32] V. Pavlović, J.M. Rehg, T.J. Cham, and K.P. Murphy, “A dynamic bayesian network approach to figure tracking using learned dynamic models,” in *Proc. of International Conference on Computer Vision*, Corfu, Greece, September 1999.

[33] S. Blackman and R. Popoli, *Design and analysis of modern tracking systems*, Artech House, 1999.

[34] C. Stauffer and W.E.L. Grimson, “Adaptive background mixture models for real-time tracking,” in *Proc. of Computer Vision and Pattern Recognition*, Ft. Collins, CO, USA, June 23-25 1999, pp. 246–252.

[35] K. Toyama, B. Brumitt, J. Krumm, and B. Meyers, “Wallflower: Principals and practice of background maintenance,” in *Proc. of International Conference on Computer Vision*, Corfu, Greece, September 1999, pp. 49–54.

[36] R.T. Collins and Y Liu, “On-line selection of discrimination tracking features,” *Proc. of the 9th IEEE International Conf. on Computer Vision*, pp. 346–352, October 14-17 2003.

[37] P. Viola, M.J. Jones, and D. Snow, “Detecting pedestrians using patterns of motion and appearance,” *Proc. of the 9th IEEE International Conf. on Computer Vision*, vol. 2, pp. 734–741, October 14-17 2003.

[38] K. Toyama and G. Hager, “Incremental focus of attention for robust vision-based tracking,” *Int. J. Computer Vision*, vol. 35, no. 1, pp. 45–63, January 1999.

[39] H. Sidenbladh, M.J. Black, and D.J. Fleet, “Stochastic tracking of 3d human figures using 2d image motion,” in *Proc. of European Conference on Computer Vision 2000*, London, UK, June/July 2000, vol. 2, pp. 702–718.

[40] J.M. Odobez and D. Gatica-Perez, “Embedding motion in model-based stochastic tracking,” in *ICPR 2004*, Cambridge, UK, 2004, vol. 2, pp. 815–818.

[41] B. Birchfield, “Elliptical head tracking using intensity gradients and color histogram,” in *Proc. of Computer Vision and Pattern Recognition*, Santa Barbara, CA, June 23-25 1998, pp. 232–237.

[42] J. Triesch and vod der Malsburg, “Democratic integration: Self-organized integration of adaptive cues,” *Neural Computation*, vol. 13, pp. 2049–2074, 2001.

[43] J. Vermaak, P. Pérez, M. Gangnet, and A. Blake, “Towards improved observation models for visual tracking: selective adaptation,” in *Proc. of European Conference on Computer Vision 2002*, Copenhagen, Denmark, 2002, vol. 1, pp. 645–660.

[44] C. Berzuini, N.G. Best, W.R. Gilks, and C. Larizza, “Dynamic conditional independence models and markov chain monte carlo methods,” *J. Am. Stat. Assoc.*, vol. 92, no. 440, pp. 1403–1412, December 1997.

[45] N. Gordon, D. Salmond, and C. Ewing, “Bayesiand state estimation for tracking and guidance using the boostrap filter,” *J. Guidance, Control and Dynamics*, vol. 18, pp. 1434–1443, 1995.

[46] A. Doucet and N. de Freitas, *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, 2001.

[47] A.P. Dempster, M.N. Laird, and D.B. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *J. R. Stat. Soc.*, vol. B 39, pp. 1–38, 1977.

[48] M. Spengler and B. Schiele, “Towards robust multi-cue integration for visual tracking,” *Machine Vision and Application*, vol. 14, pp. 50–58, 2003.

[49] C. Bräutigam, J.-O. Eklundh, and H.I. Christensen, “A model-free voting approach for integrating multiple cues,” in *Proc. of 5th European Conference on Computer Vision*, Freiburg, Germany, June 2-6 1998, pp. 734–750, Springer-Verlag.

[50] Robert T. Collins, X. Zhou, and S.K. Teh, “An open source tracking testbed and evaluation web site,” in *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS 2005)*, January 2005.

[51] P. Li, T. Zhang, and A.E.C. Pece, “Visual contour tracking based on particle filters.,” *Image Vision Comput.*, vol. 21, no. 1, pp. 111–123, January 2003.

[52] M. Kass, A. Witkins, and D. Terzopoulos, “Snakes: active contour models,” *International journal on computer vision*, vol. 1, no. 4, pp. 321–331, January 1988.

[53] F. Leymarie and M.D. Levine, “Tracking deformable objects in the plane using an active contour model,” *IEEE Trans. on Pattern Analysis Machine Intelligence*, vol. 15, no. 6, pp. 617–634, June 1993.

[54] K. Paragios Nikos and D. Rachid, “A pde-based level-set approach for detection and tracking of moving objects,” in *Proceedings of the sixth ICCV*. 1998, p. 1139, IEEE computer society.

[55] M. Rousson and N. Paragios, “Shape priors for level set representations,” in *European Conference on Computer Vision*, Copenhagen, Denmark, May/June 2002, pp. 78–92.

[56] G. Sapiro, *Geometric partial differential equations and image analysis*, Cambridge University Press, New York, NY, USA, 2001.

[57] A. Yilmaz, X. Li, and M. Shah, "Contour-based object tracking with occlusion handling in video acquired using mobile cameras," *IEEE Trans. on Pattern Analysis Machine Intelligence*, vol. 26, no. 11, pp. 1531–1536, November 2004.

[58] J. Jackson, A.J. Yezzi, and S. Soatto, "Tracking deformable moving objects under severe occlusions," in *IEEE Conf. on Decision and Control*, Atlantis, Paradise Island, Bahamas, December 2004, vol. 3, pp. 2990–2995.

[59] Y. Rathi, N. Vaswani, A. Tannenbaum, and A. Yezzi, "Particle filtering for geometric active contours with application to tracking moving and deforming objects," in *IEEE Conf. on Computer Vision and Pattern Recognition*, San Diego, CA, USA, June 2005, vol. 2, pp. 2–9.

[60] A. Yezzi and S. Soatto, "Deformation: Deforming motion, shape average and the joint registration and approximation of structures in images," *International Journal of Computer Vision*, vol. 53, no. 2, pp. 153–167, Febuary 2003.

[61] J. Foley, A. van Dam, S. Feiner, and J. Hughes, *Computer Graphics Principles and Practice*, chapter 13, pp. 563–604, Addison-Wesley, 1990.

[62] R.E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[63] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Trans. on Signal Processing*, vol. 50, no. 2, pp. 174–188, Febuary 2002.

[64] D. Metaxas and D. Terzopouilos, "Shape and nonrigid motion estimation through physics-based synthesis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 6, pp. 580–591, 1993.

[65] G. Xu, E. Segawa, and S. Tsuji, "A robust active contour model with insensitive parameters," in *International Conference on Computer Vision 1993*, Berlin, Germany, 1993, pp. 562–566.

[66] P.J. Phillips, S. Sarkar, I. Robledo, P. Grother, and K.W. Bowyer, "The gait identification challenge problem: Data sets and baseline algorithm," in *Intl. Conf. on Pattern Recognition*, Washington, DC, USA, 2002, p. 10385.

[67] A Jain, *Fundamentals of Digital Image Processing*, chapter 9, Prentice-Hall, 1989.

[68] P. Pérez, J. Vermaak, and A. Blake, "Data fusion for visual tracking with particles," in *IEEE (issue on State Estimation)*, 2004.

[69] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Trans. on Pattern Analysis Machine Intelligence*, vol. 12, no. 7, pp. 629–639, July 1990.

[70] Q. Zheng and R. Chellappa, "A computational vision approach to image registration," *IEEE Trans. Image Processing*, vol. 2, pp. 311–326, July 1993.

[71] J.M. Coughlan and S.J. Ferreira, "Finding deformable shapes using loopy belief propagation," in *European Conf. Computer Vision*, Copenhagen, Denmark, 2002, vol. 3, pp. 453–468, Springer-verlag.

[72] D. Cremers, T. Kohlberger, and C. Schnörr, "Nonlinear shape statistics in mumford-shah based segmentation," in *European Conference on Computer Vision*, Copenhagen, May 28-31 2002, Spinger.

[73] F. Porikli and O. Tuzel, "Bayesian background modeling for foreground detection," in *Proc. of ACM Visual Surveillance and Sensor Network*, New York, NY, USA, 2005, pp. 55–58.

[74] T.F. Cootes and C.J. Taylor, "Active shape models," in *3rd British Machine Vision Conference*, D. Hogg and R. Boyle, Eds. September 1992, pp. 266–275, Spinger-verlag.

[75] T.F. Cootes and C.J. Taylor, "Combining point distribution models with shape models based on finite-element analysis," in *5th British Machine Vision Conference*, E. Hancock, Ed., York, England, September 1994, pp. 419–428.

[76] X.S. Zhou, D. Comaniciu, and Alok Gupta, "An information fusion framework for robust shape tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 1, pp. 115–129, 2005.

[77] H. Nanda and L. Davis, "Probabilistic template based pedestrian detection in infrared videos," in *Proceeding of IEEE intelligent vehicle symposium*, Versailles, France, June 18-20, 2002.

[78] Y. Akgul and C. Kambhamettu, "A coarse-to-fine deformable contour optimization framework," *IEEE trans. pattern analysis and machine intelligence*, vol. 25, no. 2, pp. 174–186, Febuary 2003.

[79] M. Black and A. Jepson, "Eigentracking: robust matching and tracking of articulated objects using a view-based representation," in *Proc. of European Conference on Computer Vision 96*, Cambridge, UK, 1996, pp. 610–619.

[80] A. Blake and M. Isard, *Acive Contours*, Springer Verlag, 1998.

[81] D. Cremers, "Dynamical statistical shape priors for level set-based tracking," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, pp. 1262–1273, August 2006.

[82] Y. Rathi, N. Vaswani, A. Tannenbaum, and A. Yezzi, "Particle filtering for geometric active contours and application to tracking deforming objects," in *IEEE Computer Society Conference of Computer Vision and Pattern Recognition 2005*, 2005, vol. 2, pp. 2–9.

[83] Y. Wu, G. Hua, and T Yu, "Switching observation models for contour tracking in clutter," in *Proceeding of Computer Vision and Pattern Recognition '03*, Madison, Wisconsin, USA, June 2003, vol. 1, pp. 295–302.

[84] T. Cootes and C. Taylor, “Active shape models - 'smart snakes',” in *British Machine Vision Conf.*, 1992, pp. 266–275.

[85] T.F. Cootes, G.J. Edwards, and C.J. Taylor, “Active appearance models,” *IEEE trans. on pattern analysis and machine intelligence*, vol. 23, no. 6, pp. 681–685, June 2001.

[86] S. Baker, I. Matthews, J. Xiao, R. Gross, T. Kanade, and T. Ishikawa, “Real-time non-rigid driver head tracking for driver mental state estimation,” in *11th World Congress on Intelligent Transportation Systems*, October 2004.

[87] L.P. Morency, A. Rahimi, and T. Darrell, “Adaptive view-based appearance models,” in *Proceeding of Computer Vision and Pattern Recognition '03*, Madison, Wisconsin, 2003, vol. 1, pp. 803–810.

[88] J.G. Bosch, S.C. Mitchell, B.P.F. Lelieveldt, F. Nijland, and O. Kamp, “Automatic segmentation of echocardiographic sequences by active appearance motion models,” *IEEE Trans. on Medical Imaging*, vol. 21, no. 11, pp. 1374–1383, November 2002.

[89] S. Zhou, J. Shao, B. Georgescu, and D. Comaniciu, “Echocardiography tracking via modeling of shape, appearance and motion,” in *International Conference on Medical Image Computing and Computer Assisted Intervention*, Copenhagen, Denmark, October 2006.

[90] L. Rabiner and B. Juang, “An introduction to hidden markov models,” *ASSP Magazine, IEEE*, vol. 3, no. 1, pp. 4–16, January 1986.

[91] Fred L. Bookstein, “Principal warps: Thin-plate splines and the decomposition of deformations,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 11, no. 6, pp. 567–585, June 1989.

[92] J. Feldman and M. Singh, “Information along contours and object boundaries,” *Psychological Review*, vol. 112, pp. 243–252, 2005.

[93] D. Comaniciu, “Nonparametric information fusion for motion estimation,” in *Computer Vision and Pattern Recognition*, 2003, pp. 59–66.

[94] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *Proc. DARPA IU Workshop*, 1981, pp. 121–130.

[95] D. Boukerroui, J.N. Alison, and M. Brady, “Velocity estimation in ultrasound images: A block matching approach,” in *Information Processing in Medical Imaging*, 2003, pp. 586–598.

[96] M.B. Stegmann, B.K. Ersbøll, and R. Larsen, “FAME – a flexible appearance modelling environment,” *IEEE Trans. on Medical Imaging*, vol. 22, no. 10, pp. 1319–1331, October 2003.

[97] A. Criminisi, I. Reid, and A. Zisserman, “Single view metrology,” *Intl. J. of Comp. Vis.*, vol. 40, pp. 123–148, 2000.

[98] B. Caprile and V. Torre, "Using vanishing points for camera calibration," *Int'l Journal of Computer Vision*, vol. 4, no. 2, pp. 127–140, March 1990.

[99] Robert Collins and J.R. Beveridge, "Matching perspective views of coplanar structures using projective unwarping and similarity matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, New York, NY, June 1993, pp. 240–245.

[100] P. Gurdjos and R. Payrissat, "About conditions for recovering the metric structure of perpendicular planes from the single ground plane to image homography," in *15th International Conference on Pattern Recognition*, Barcelona, September 2000, vol. 1, pp. 358–361.

[101] B. Liang, Z. Chen, and N. Pears, "Uncalibrated two-view metrology," in *17th International Conference on Pattern Recognition*, Cambridge, UK, August 2004, pp. 96–99.

[102] F. Lv, T. Zhao, and R. Nevatia, "Self-calibration of a camera from video of a walking human," in *Proc. of 16th International Conference on Pattern Recognition*, Québec, Canada, August 2002, vol. 1, pp. 562–567.

[103] J.R. Renno, J. Orwell, and G.A. Jones, "Learning surveillance tracking models for the self-calibrated ground plane," in *Proc. of British machine vision conference*, Cardiff, UK, September 2002, pp. 607–616.

[104] C. Stauffer, K. Tieu, and L. Lee, "Robust automated planar normalization of tracking data," in *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, Nice, France, October 2003.

[105] B. Bose and E. Grimson, "Ground plane rectification by tracking moving objects," in *Proc. of joint IEEE International Workshop on Visual Surveillance and Performence Evaluation of Tracking and Surveillance*, 2003.

[106] F. Guo and R. Chellappa, "Video mensuration using a stationary camera," in *European conf. on comp. vision*, 2006.

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

[108] C. Rother, "A new approach for vanishing point detection in architectural environments," in *Proc. of 11th British machine vision conference*, University of Bristol, September 2000, pp. 382–391.

[109] J. Kosecka and W. Zhang, "Video compass," in *Proc. of European Conference on Computer Vision*, Norwich, UK, September 2002, pp. 476–491.

[110] M. Bosse, R. Rikoski, J. Leonard, and S. Teller, "Vanishing points and 3d lines from omnidirectional video," in *Proceedings of the International Conference on Image Processing*, Rochester, New York, September 2002, vol. III, pp. 513–516.

[111] B. Lucas and T. Kanade, "An iterative image registation technique with an application to stereo vision," *International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

[112] M. Han and T. Kanade, "Reconstruction of a scene with multiple linearly moving objects," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition 2000*, June 2000, pp. 542–549.

[113] M. Antone and S. Teller, "Automatic recovery of relative camera rotations for urban scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition 2000*, Hilton Head, NC, June 2000, pp. 282–289.

[114] S. Kullback and R.A. Leibler, "On information and sufficiency," *Ann. of math. stat.*, vol. 22, pp. 79–86, January 1951.

[115] Z. Zhang, "Determining the epipolar geometry and its uncertainty: a review," *International Journal of Computer Vision*, vol. 27, no. 2, pp. 161–195, 1998.

[116] H. Robbins and S. Monro, "A stochastic approxiamation method," *Ann. of math. stat.*, vol. 22, pp. 400–407, 1951.

[117] J.C. Spall, *Introduction to stochastic search and optimization*, Wiley, 2000.

[118] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[119] A. Pentland, B. Moghaddam, and T. Starner, "View-based and modular eigenspaces for face recognition," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, July 1994.

[120] G. Wolberg, *Digital Image Warping*, IEEE Computer Society Press, 1994.

[121] T. Cootes and C. Taylor, "Statistical models of appearance for medical image analysis and computer vision," in *In Proc. SPIE Medical Imaging*, 2001, pp. 236–248.

[122] M. Lades, C. Vorbrueggen, J. Buhmann, J. Lange, C. von der Malsburg, R. Wuertz, and W. Konen, "Distortion invariant object recognition in the dynamic link architecture," *IEEE Trans. on Computers*, vol. 42, 1993.

[123] J. Li, S. Zhou, and R. Chellappa, "Appearance modeling using a geometric transformation," *IEEE trans. on image processing*, vol. 18, no. 4, pp. 889–902, 2009.

[124] R. Vidal, Y. Ma, and S. Sastry, "Generalized principal component analysis (gpca)," in *IEEE Conference on Computer Vision and Pattern Recognition*, Madison, Wisconsin, June 2003, vol. 1, pp. 621–628.

[125] R. Vidal, *Generalized principal component analysis (GPCA): an algebraic geometric approach to subsapce clustering and motion segmentation*, Ph.D. thesis, University of California at Berkeley, 2003.

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

[127] J.C. Clarke, "Modelling uncertainty: a primer," Technical report 216198, University of Oxford, dept. engineering science, 1998.

[128] A. Criminisi, *Accurate visual metrology from single and multiple uncalibrated images*, Distinguished dissertation. Springer-Verlag London Ltd., September 2001.