Abstract

Title of dissertation	Robust Methods for Visual Tracking and Model Alignment
	Hao Wu, Doctor of Philosophy, 2009
Directed by	Professor Rama Chellappa Department of Electrical and Computer Engineering

The ubiquitous presence of cameras and camera networks needs the development of robust visual analytics algorithms. As the building block of many video surveillance tasks, a robust visual tracking algorithm plays an important role in achieving the goal of automatic and robust surveillance. In order to maintain a persistent tracking of objects, it is critical to know when and where the tracking algorithm fails so that remedial measures can be taken to resume tracking. However, most present evaluation methods are ogg-line and based on manually labeled ground truth data. Online evaluation methods in the absence of ground truth are of urgent need. We propose a novel performance evaluation strategy for tracking systems based on particle filter using a time-reversed Markov chain. The posterior density of the time-reversed chain is computed and the distance between the prior density used to initialize the tracking algorithm and the time-reversed posterior density function forms the decision statistic for evaluation. This backward tracking-based performance evaluation strategy is also general enough to be applied to many other tracking algorithms.

In this dissertation, we also present a new bidirectional tracking strategy to

achieve better performance. Instead of looking only forward in the time domain, we incorporate both forward and backward processing of video frames using a time-reversibility constraint. This leads to a new minimization criterion that combines the forward and backward similarity functions and the distances of the state vectors between the forward and backward states of the tracker. The bidirectional track strategy significantly improves tracking robustness and accuracy. We illustrate the improvements due to the proposed approach for the popular KLT tracker and a search-based tracker.

Some objects of interest in surveillance applications like faces have relatively stable structures, which allows us to build parameterized shape models to localize the objects more precisely. There are some widely used algorithms for model alignment; however, most of them suffer from the problem of converging to local extrema when used in practice. In this dissertation, we present a machine learning method to learn a scoring function without local extrema to guide the gradient descent/accent algorithm and find the optimal parameters of the shape model. The method is called Boosted Ranking Model (BRM). By arranging the training samples in some special structure, we feed them pairwise into the rank training algorithm and learn a strong ranking function from a pool of weak features. Theoretically, this method can learn a function with arbitrarily few local extrema as long as the training samples are dense and the representation ability of the features are good enough. The extensive experimental results show that our proposed algorithm, BRM, outperforms existing algorithms.

Robust Methods for Visual Tracking and Model Alignment

by

Hao Wu

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

Advisory Committee:

Professor Rama Chellappa, Chair
Professor David W. Jacobs, Dean's Representative
Professor P.S. Krishnaprasad
Professor Min Wu
Dr. Qinfen Zheng

To my parents,
For their love, support and sacrifice
over the years.

Acknowledgments

I am especially thankful to my advisor, Prof. Rama Chellappa, for supervising and supporting me over these years. It was a great oppotunity to work with him at the Center for Automation Research in University of Maryland. I am particularly grateful to Prof. P. S. Krishnaprasad, Prof. David W. Jacobs, Prof. Min Wu and Dr. Qinfen Zheng for serving on my dissertation committee and their insightful questions and advises to my research work; I also want to thank Prof. Adrian Papamarcou for serving on my proposal committee.

I am very thankful to my intern mentors Dr. Yunqiang Chen, Dr. Lisa Brown, Dr. Xiaoming Liu, Dr. Gianfranco Doretto and university collaboratores Dr. Qinfen Zheng, Prof. Ankur Srivastava and Dr. Volkan Cevher for their helpful instructions. I learned a lot from them during the collaborations.

It is hard to thank all my friends and colleagues individually, but in particular, I would like to thank colleagues and friends at CfAR: Aswin Sankaranarayan, S. Kevin Zhou, Ashok Veeraraghavan, Feng Guo, Haibin Ling, Zhanfeng Yue, Jie Shao, Jian Li, Daozheng Cheng, Jing Cheng, Ruonan Li, Ming Du, Yang Ran and many others. They were always there when I needed help.

I would also like to thank my new manegers, colleagues and friends at Kodak Research Lab, Nancy Ferris, Majid Rabbani, Jiebo Luo, Alexander Loui, Charles Parker, Andrew Gallagher, Jerry Yu, Dhiraj Joshi and many others, for their considerations, suggestions and support for my thesis work.

Finally, I am greatly indebted to my parents for their consistent encouragement, love and sacrifice in all my endeavors, academic and non-academic, over the years. I dedicate this dissertation to them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Computer vision, as a scientific discipline, has been actively studied since the late 1970s when computers could process large amount of data, such as images or videos. In general, the ultimate goal of computer vision is to design and implement artificial vision system that can process images like human visual systems. At its current stage, computer vision still remains as a collection of diverse studies motivated by various applications. In this sense, it lacks a universal formulation and existing methods are often very task specific and seldom can be generalized to a wide range of applications. Although humans can perceive the information contained in the 3D world and 2D images without much effort, it is nontrivial for computers to achieve the same level of performance. Researchers have already presented many intelligent algorithms to solve a diverse set of problems in computer vision, however, most of them experience difficulties when used in the real world, limiting their usefulness. Therefore, it is critical to make computer vision algorithms perform robustly in real environments.

In this dissertation, we investigated three vision tasks: visual tracking, performance evaluation for visual tracking algorithms and the model alignment problem, especially for face alignment. In the following, we provide a brief summary of these

problems and present motivations for the work reported in this dissertation.

1.1 Visual Tracking

In recent times, the need for visual surveillance and automated processing of surveillance information has become important. Further, systems employing a large number of cameras are becoming increasingly common; for example, an average casino has between 2000 and 3000 surveillance cameras capturing data all the time; in Britain, there are about 4.2 million CCTV cameras. Usually only a small team of security personnel are available to monitor the cameras in video surveillance systems. This poses immense challenges to the personnel in paying attention to every camera and react quickly to all the emergencies. To overcome the disadvantages of human-monitored surveillance, computer vision technologies have been applied to enable computers to process data and initiate appropriate actions.

The main purpose of visual tracking is to consistently and automatically locate objects or other features of interest in video or image sequences. Visual tracking is one of the core technologies used in automatic video surveillance. It estimates the motion of objects for further analysis, such as camera positioning control, activity analysis, abnormality detection, object identification and recognition. The importance of robust and accurate tracking can not be overstated in surveillance applications.

1.1.1 Challenges

Currently there are lots of tracking algorithms and systems developed by researchers; however, in translating the laboratory solutions into commercial systems, most of the algorithms encounter unexpected failure modes in practice and performance is far below the expectation. There are many factors that could make tracking algorithms fail in practice. While it is difficult to list all of these factors, we present some of the most common challenges one faces while designing robust trackers:

- Ambiguous Appearance information. Because the image formation is a mapping from 3D to 2D, we can only see the projection of part of the object at a given time. Therefore, for a moving 3D object, its appearance in image depends on its pose and position with respect to the camera. Variations in appearance as the object moves contributes to track failures. Mostly, all the available information about appearance comes from the initialization step of the system, either manually extracted or from some detection algorithms. This initialization contains the appearance information only about part of the object. Other factors causing appearance variation include shadows, lighting variations, clothing, etc. These degradations are usually unexpected and hard to be incorporated into the object models in the system.

- The object dynamics is nonlinear and complex in practical systems. As it is hard to specify an accurate model for general object motion, approximations in the specification of model could introduce unwanted uncertainties in the

system. The accumulated uncertainties could eventually make the tracker drift away. Therefore, the visual tracking problem can be viewed as an ill-posed inverse perception problem.

- Occlusions may happen often in the visual system, where the distant surfaces are obscured by closer ones. This is a common challenge for a tracking system and also a major reason that causes the loss of track.

- Due to the projection from 3D world to 2D image plane, the depth information of the objects and the background has been lost during the imaging process. Hence, the background objects and clutter are very "close" to the objects of interest on image plane, which can easily cause distractions to the trackers.

- Image measurements are corrupted by noise and blur. In a static optical visual system with high quality cameras, this problem may not be a big issue, but in other applications involving infrared cameras or airborne cameras, this would be a very important problem. To increase the noise variance in the model could make the tracking algorithm easily diverge.

1.1.2 Online Performance Evaluation for Visual Tracking Algorithms

Although many sophisticated algorithms exist for tracking, each of them has failure modes or scenarios when the performance will be poor. Typically, this will happen when the data (the frames of the video) does not obey the modeling assumptions. Most algorithms fail to track a target (or targets) through crowded environments, in urban clutter or when changes in illumination and self-occlusion

are present. This often leads to a loss of track and affects the performance of subsequent stages of processing. Instead of designing advanced algorithms to avoid all possible failure modes, which is almost impossible in practice, we handle this problem by detecting the tracking failures first and then re-initializing the system to continue the tracking. In this context, performance evaluation plays an important role in practical visual tracking systems. However, existing performance evaluation algorithms concentrate on off-line statistical comparisons with manually created ground truth data. While comparison with ground truth can inform which tracking algorithm has better overall performance on a specific sequence, it does not extend gracefully for testing on new sequences without additional ground truth. In the absence of ground truth, off-line performance evaluation can not help to detect the loss of track and/or improve the robustness of tracking systems.

To monitor how well a tracker is working, online evaluation of performance is desired. Here, online means that the evaluation is automatic, without use of any ground truth, and that it is also an online and causal evaluation method. Hence, we can re-initialize the tracker procedure right after the failure is detected. In this way, even a poorly designed tracking algorithm can handle complex scenarios and achieve reasonably good overall tracking performance. However, this is a very challenging task and has received limited attention.

We propose an online performance evaluation method by constructing a time-reversed Markov chain and continue the reverse tracking to the starting point where the tracker was initialized [99]. Estimates obtained after reversed tracking

are then compared to the initial and forward tracking estimates to infer the tracking performance at the current time. The proposed evaluation algorithm is more flexible and can be applied to almost all existing tracking algorithms. By integrating this online performance evaluation strategy with the tracking algorithms, we can maintain a persistent tracking of objects over a long time period.

1.1.3 Tracking Algorithms using the Time-Reversibility Constraint

The robustness and accuracy of tracking algorithms are very important. In some applications, even small error in tracking can lead to bad results. For example, in structure from motion algorithms, the performance of the structure reconstruction is directly dependent on the accuracy of feature points detected by some feature tracking algorithms, since the reconstruction is an ill-posed problem and very sensitive to errors in establishing the point correspondences.

Since all the targets/feature points to be tracked are macroscopic solid objects in the physical world and the physical laws of classical mechanics are time-symmetric, the motion of the objects should be time-reversible, which means that the time-reversed process satisfies the same dynamical equations as well as the original process. From the information perspective, if the tracker can not go back to the previous state, it often implies there is information leak during the tracking procedure; on the contrary, a perfect tracker like human eyes can always be expected to give the correct status of objects no matter if it uses forward or backward tracking.

However, most of the existing tracking algorithms only look forward in the

time domain instead of looking bidirectionally during tracking. Instead of just looking forward in the time domain, we simultaneously perform both forward and backward tracking using the time-reversibility constraint. The bidirectional tracker reduces the possibility of the tracker getting stuck in local minima and significantly improves the tracking robustness and accuracy.

1.2 Model Alignment

In some situations, people are more interested in detailed component positions of the object in the image other than an overall rough location. This detailed information about the object makes it easier for many high-level analysis tasks, like face recognition, expression recognition, activity recognition, etc. To localize the detailed component of a general object is very hard, but if the interested object has a relatively stable shape, we can learn a parameterized shape model to represent various shapes belonging to the same category. Model alignment is to deform this parametrized shape model to best fit the image instance.

Face alignment/fitting is one of the most studied model alignment problems in computer vision, where a face model needs to be deformed to match the image of a face, so that the natural facial features are aligned with the model. The dramatic variations of facial appearance due to shape, pose, illumination, expression, occlusions, and image resolution make this a challenging problem. Due to its importance in a wide range of applications, there is a sizable literature on face alignment. The Active Shape Model (ASM) [19] is one of the early approaches

that attempts to fit the data with a model that can deform in ways consistent with a training set. The Active Appearance Model (AAM) [5] [18] is a popular extension of the ASM. Boosted Appearance Model (BAM) [62] is a recently proposed method to handle face alignment problem in a different framework.

In general, since model alignment algorithms need to find the best solutions in usually very high-dimensional spaces, they often experience the local extrema problem in practice. There are many sophisticated techniques proposed to handle the local extrema problem for a given objective function, but these algorithms are mostly focused on how to avoid getting stuck in local extrema. These kind of strategies make the search procedure for the global extrema more and more complicated. In this dissertation, we propose a novel approach to handle the local extrema problem, by learning a local-extrema-free objective function whose global extrema can be easily found by the basic gradient accent/descent algorithm. This is achieved by carefully arranging the training samples before feeding them into the modified classifiers.

1.3 Key Contributions

Computer vision research is a highly cross-disciplinary field, intersecting with many other related fields, like machine learning, signal processing, mathematics, control, physics, neurobiology, etc. In this dissertation, we develop robust visual tracking algorithm and an online performance evaluation method by using the notion of time-reversibility. We also developed a novel training method to learn

a local-extrema-free score function for model alignment problems. The details of the key contributions are as follows:

- We investigate the seldom used property of object motion - time reversibility - in visual tracking systems. Like another widely used regularization term called smoothness constraint in computer vision problems, using the time-reversibility constraint could help vision algorithms to be more robust in practice.

- Specifically, we provide an online performance evaluation algorithm for visual tracking systems in the absence of ground truth data. This strategy is based on the idea of time-reversal and can automatically detect tracking failures. Online performance evaluation makes it possible for tracking algorithms to detect failures soon after they occur and take actions to recover from those failures. In this way, the algorithms are able to maintain a consistent tracking performance over a long period. This idea works with most tracking algorithms.

- To further improve the robustness and accuracy of tracking algorithms, a new bi-directional Kanade-Lucas-Tomasi (KLT) for tracking algorithms using the time-reversibility constraint is proposed. We show significant improvements over the original KLT tracker in the experiments. This bidirectional tracking strategy is promissing enough to be generalized to other tracking algorithms.

- A training method based on rank learning is proposed to learn a local-extrema-free score function. Many computer vision problems boil down to optimization problems. Gradient descent/accent algorithms are widely used for such prob-

lems, but they easily get stuck in the local extrema. With the proposed strategy, we can dramatically reduce the local extrema of the object function. We illustrate the success of the proposed algorithm on model alignment problems in this dissertation. It is promising to use this idea to handle other vision problems that need to deal with the problem of local extrema.

1.4 Dissertation Outline

The rest of this dissertation is organized as below:

In Chapter 2, we briefly review various visual tracking algorithms and describe three representative algorithms used in our experiments, namely, stochastic visual tracking based on particle filters [44] [110], Kanade-Lucas-Tomasi (KLT) feature tracker [64] [89] and the mean-shift tracker [16] [17].

Chapter 3 presents the work on online performance evaluation for visual tracking algorithms. We review prior work on performance evaluation in related domains and propose our strategy for this problem in detail. Extensive experiments are performed to show the effectiveness of the proposed algorithm. Other helpful discussions on various issues and extensions to different tracking algorithms can also be found in this chapter.

Chapter 4 addresses the idea on bi-directional tracking using the time-reversibility constraint. The main experiments are done using the KLT feature tracking algorithm. We compare the bidirectional KLT tracker developed under the proposed framework and the original KLT tracker in different situations. Preliminary ex-

tension of this idea to large object tracking is also discussed.

In Chapter 5, we review some widely used model alignment algorithms including ASM, AAM and BAM algorithms. These algorithms provide the basis for many model alignment algorithms. We describe these algorithms in detail and analyze the advantages and deficiencies of these algorithms and then propose a novel training strategy to learn a local-extrema-free score function for model alignment. We use face alignment to illustrate the effectiveness of this training idea. Comprehensive comparisons to the state-of-art algorithm are conducted and the results show the superiority of the proposed strategy.

Finally, some conclusions and suggestions for future research directions are summarized in Chapter 6.

Chapter 2

Background on Visual Tracking and Time Reversibility

Over the last two decades, researchers have developed various algorithms for object tracking in video data [106] [11]. However, time-reversibility, as an intrinsic property of moving objects, has not received adequate attentions in the past when designing visual tracking algorithms. In Chapter 3 and 4, we will explore the applications of the time-reversibility constraint for developing visual tracking algorithms and online performance evaluation algorithms. In this chaper, we will first summarize three representative methods widely used in visual tracking applications. These algorithms are also adopted in our experiments presented in Chapter 3 and 4. Subsequently, we will discuss the concept of time-reversibility in some related areas: Markov chains, Brownian motion and optimal filtering.

2.1 Visual Tracking

We introduce three representative algorithms for tracking in this section. These algorithms are used as the building blocks for a host of other algorithms, and hence warrant special attention.

- **Bayesian filtering methods [25] [6] [49]:** This category includes both Kalman and particle filters, which have broad applications in many areas. When used in visual tracking, these methods still view the visual tracking problem as a classic stochastic filtering problem. With explicit state transition and observation models, the position of the targets can be sequentially estimated frame by frame.

- **Kanade-Lucas-Tomasi (KLT)tracker [64] [89] [85]:** This optical flow like algorithm is mainly designed for feature tracking scenarios. Good features are first located by examining the minimum eigenvalue of the gradient matrix of each image patch, and then tracked using a Newton-Raphson method based on the assumption of constant intensity. The KLT tracker is similar to optical flow algorithms in spirit.

- **Mean-Shift Tracker [16] [17]:** This method is also called kernel-based object tracking. The feature histogram-based target representations are regularized by spatial masking using an isotropic kernel, which enables locating the target through a mean shift optimization process.

In the following, we provide a detailed summary of these three methods in order to make this document self-contained.

2.1.1 Bayesian Filtering / Tracking

The generic discrete-time stochastic filtering problem can be expressed using a dynamic state-space form:

$$\begin{aligned} x_{n+1} &= f(x_n, d_n) \\ y_n &= g(x_n, v_n) \end{aligned} \tag{2.1}$$

where y_n is the measurement vector, x_n represents the state vector. d_n and v_n are random noise sequences. The stochastic filtering problem can be defined as: given initial density $p(x_0)$, transition density $p(x_n|x_{n-1})$ which is characterized by the state equation, and the likelihood $p(y_n|x_n)$ which is described by the measurement or observation equation, how to estimate the optimal current state at time n based on the measurements or observations up to time n.

In this section, we address the problem of Bayesian inference for generic nonlinear dynamical systems. Of the number of methods available for nonlinear filtering problems, we focus on the sequential Monte Carlo approach for sequential state estimation, which is also know as particle filtering.

In particle filtering [25], Let $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}^p$ denote the state space and the observation space of the system respectively. Let $x_t \in \mathbb{R}^d$ denote the state at time t, and $y_t \in \mathbb{R}^p$ the noisy observation at time t. We model the state sequence $\{x_t\}$ as a Markovian random process. Further we assume that the observations $\{y_t\}$ to be conditionally independent given the state sequence. Under these assumptions, the models defining the system are given as follows: 1) $p(x_t|x_{t-1})$: The state transition probability density function, describing the

evolution of the system from time $t-1$ to t. Alternatively, the same could be described with a *state transition model* of the form $x_t = h(x_{t-1}, n_t)$, where n_t is a noise process. 2) $p(y_t|x_t)$: the observation likelihood function, describing the conditional likelihood of observation given state. As before, this relationship could be in the form of an *observation model* $y_t = f(x_t, \omega_t)$ where ω_t is a noise process independent of n_t. and 3) $p(x_0)$: The prior state probability at $t = 0$.

Given the statistical descriptions of the models and noisy observations till time t, we are interested in making inferences about the state of the system at current time. Specifically, given the observations till time $t, \mathcal{Y}_t = \{y_1, \ldots, y_t\}$, we would like to estimate the posterior density function $\pi_t = p(x_t|\mathcal{Y}_t)$. Using the posterior density function, we aim to make inferences $I(f_t)$ of the form,

$$I(f_t) = \mathbf{E}_{\pi_t}[f_t(x_t)] = \int f_t(x_t)p(x_t|y_{1:t})dx_t \tag{2.2}$$

where f_t is some function of interest. An example of such an inference could be the conditional mean, where $f_t(x_t) = x_t$. Under Markovian assumption on the state space dynamics and conditional independence assumption on the observation model, the posterior probability is estimated recursively using the *Bayes Theorem* as:

$$\pi_t(x_t) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{p(y_t|y_{1:t-1})} = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{\int p(y_t|x_t)p(x_t|y_{1:t-1})dx_{t-1}} \tag{2.3}$$

Computation of $p(x_t|y_{1:t-1})$ is accomplished using the *prediction* step,

$$p(x_t|y_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1} \tag{2.4}$$

Equation (2.4) sets up the recursive step for estimation of the posterior at time

t, $\pi_t(x_t)$ from that at time $t-1$, $\pi_{t-1}(x_{t-1})$.

$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)\int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1}}{p(y_t|y_{1:t-1})} \tag{2.5}$$

Note that, there are no unknowns in (2.5) since all terms are either specified or computable from the posterior at the previous time step. The problem is that this computation need not have an analytical representation. However, foregoing the requirement for an analytic solution, the particle filter approximates the posterior π_t with a discrete set of particles or samples $\{x_t^{(i)}\}_{i=1}^N$ with associated weights $\{w_t^{(i)}\}_{i=1}^N$ suitably normalized. The approximation for the posterior density is given by

$$\hat{\pi}_t(x_t) = \sum_{i=1}^{N} w_t^{(i)} \delta_{x_t}\left(x_t^{(i)}\right) \tag{2.6}$$

where $\delta_{x_t}(\cdot)$ is the Dirac Delta function centered at x_t. The set $S_t = \{x_t^{(i)}, w_t^{(i)}\}_{i=1}^N$ is the weighted particle set that represents the posterior density at time t, and is estimated recursively from S_{t-1}. The initial particle set S_0 is obtained from sampling the prior density $\pi_0 = p(x_0)$.

2.1.2 Kanade-Lucas-Tomasi Tracker

The basic idea of KLT feature tracker first appeared in Lucas and Kanade's paper [64] in 1981; it was fully developed by Tomasi and Kanade [89] in 1991. In 1994, Shi and Tomasi [85] presented a KLT-based method to select good features to track. In the past decade, KLT has emerged as the most widely used feature tracker in many applications, such as structure from motion, computation of optical flow, etc.

The original KLT algorithm assumes that the intensity of the features remains constant when a camera moves, that is, $I(x, y) = J(x + \xi, y + \eta)$ assuming that the motion between two consecutive frames can be described as pure translation. This leads to the following objective function:

$$(\hat{\xi}, \hat{\eta}) = \arg\min_{\xi,\eta} \int\int_W [J(x + \xi, y + \eta) - I(x, y)]^2 * w(x, y)dxdy \tag{2.7}$$

Later, a symmetric expression is used to derive the solution [7]:

$$(\hat{\xi}, \hat{\eta}) = \arg\min_{\xi,\eta} \int\int_W [J(\mathbf{p} + \frac{\mathbf{d}}{2}) - I(\mathbf{p} - \frac{\mathbf{d}}{2})]^2 * w(\mathbf{p})d\mathbf{p} \tag{2.8}$$

where $\mathbf{p} = (x, y)^T$ and $\mathbf{d} = (\xi, \eta)^T$ and the weighting function $w(\mathbf{p})$ is usually set to 1. For simplicity we will omit all the function variables and $w(\mathbf{p})$ in the following. Also we will use the discrete form for the integrals involved in the derivations. Using a first order Taylor expansion to linearize the above nonlinear objective function and setting the derivative with respect to $\mathbf{d}$ to zero, we have:

$$\sum \sum_W (J - I + \mathbf{g}^T\mathbf{d})\mathbf{g} = 0 \tag{2.9}$$

$$\mathbf{g} = (\nabla \frac{I + J}{2})^T \tag{2.10}$$

This can be rearranged as:

$$\mathbf{Zd} = \mathbf{e} \tag{2.11}$$

where

$$\mathbf{Z} = \sum \sum_{W} \mathbf{g}\mathbf{g}^T$$
$$\mathbf{e} = \sum \sum_{W} (I - J)\mathbf{g} \tag{2.12}$$

2.1.3 Mean-Shift Tracker

The mean-shift tracker or the kernel-based tracker was presented by Comaniciu et al [16] [17]. It is a bottom-up process and contains two major components: target representation and localization.

- Target Model

  The target model can be considered as centered at the spatial location 0 and represented by its pdf q, which can be approximated by its m-bin histograms as below:

  $$\text{target model:} \qquad \hat{\mathbf{q}} = \hat{q}_{u u=1...m} \qquad \sum_{u=1}^{m} \hat{q}_u = 1 \tag{2.13}$$

  In practice, a target is represented by an ellipsoidal region in the image and all targets are normalized to a unit circle to eliminate the influence of different dimensions. After normalization, the pixel locations of the target region can be denoted by $\mathbf{x}^{\star}_{i\,\{i=1...n\}}$. Weights to different pixels can be evaluated by an isotropic kernel $k(x)$ to increase the robustness of the density estimation. Then the probability of the feature $u = 1...m$ in the target model can be computed as:

  $$\hat{q}_u = C \sum_{i=1}^{n} k(\| \mathbf{x}^{\star}_i \|^2) \delta[b(\mathbf{x}^{\star}_i) - u] \tag{2.14}$$

where

$$b : R^2 \rightarrow 1...m$$

$$C = \frac{1}{\sum_{i=1}^{n} k(\| \mathbf{x}_i^\star) \|^2)} \tag{2.15}$$

The function b maps the pixel at location $\mathbf{x}_i^\star$ to the index $b(\mathbf{x}_i^\star)$ of its bin in the quantized feature space.

- Target Candidate

In the subsequent frame a target candidate defined at location $\mathbf{y}$ is characterized by the pdf $p(\mathbf{y})$:

$$\text{target candidate:} \qquad \hat{\mathbf{p}}(\mathbf{y}) = \hat{p_u}(\mathbf{y})_{u=1...m} \qquad \sum_{u=1}^{m} \hat{p_u} = 1 \tag{2.16}$$

Denote the normalized pixel locations of the target candidate as $\mathbf{x}^\star_{i\,\{i=1...n_h\}}$, which is centered at y in the current frame. Similarly, we can compute the probability of the feature $u = 1...m$ in the target candidate as follows:

$$\hat{p_u}(\mathbf{y}) = C_h \sum_{i=1}^{n_h} k(\| \frac{\mathbf{y} - \mathbf{x}_i}{h} \|^2) \delta[b(\mathbf{x}_i) - u] \tag{2.17}$$

where

$$C_h = \frac{1}{\sum_{i=1}^{n_h} k(\| \frac{\mathbf{y}-\mathbf{x}_i}{h} \|^2)} \tag{2.18}$$

The bandwidth h defines the scale of the target candidates to handle the scale change in the process.

- Similarity Function The similarity function between $\hat{\mathbf{p}}$ and $\hat{\mathbf{q}}$ is denoted by:

$$\hat{\rho}(\mathbf{y}) \equiv \rho[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}] \tag{2.19}$$

The adopted target representation does not restrict the way similarity should be measured and various functions can be used for ρ. The Bhattacharyya coefficient is adopted here:

$$\hat{\rho}(\mathbf{y}) \equiv \rho[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\mathbf{y})\hat{q}_u} \tag{2.20}$$

And we can define the distance between the target model and the candidates as:

$$d(\mathbf{y}) = \sqrt{1 - \rho[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}]} \tag{2.21}$$

- Distance Minimization

To find the location of the target in the current frame, one minimizes the distance measure with respect to $\mathbf{y}$. To solve this minimization problem, we first linearize the Bhattacharyya coefficient using the Taylor expansion around the location $\hat{\mathbf{y}}_0$ of the target in the previous frame:

$$\begin{aligned} \rho[\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}] &\approx \frac{1}{2}\sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{\mathbf{y}}_0)\hat{q}_u} + \frac{1}{2}\sum_{u=1}^{m} \hat{p}_u(\mathbf{y}) \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\mathbf{y}}_0)}} \\ &\approx \frac{1}{2}\sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{\mathbf{y}}_0)\hat{q}_u} + \frac{C_h}{2}\sum_{i=1}^{n_h} w_i k(\| \frac{\mathbf{y} - \mathbf{x}_i}{h} \|^2) \\ \text{where} \qquad w_i &= \sum_{u=1}^{m} \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\mathbf{y}}_0)}} \delta[b(\mathbf{x}_i) - u] \end{aligned} \tag{2.22}$$

The above approximation holds when the target candidate $\hat{p}_u(\mathbf{y})_{u=1\ldots m}$ does not change drastically from the initial $\hat{p}_u(\hat{\mathbf{y}}_0)_{u=1\ldots m}$, which is often valid between consecutive frames.

The maximum in the local neighborhood can be solved by the mean shift procedure. The algorithm is summarized as below:

1. Initialize the location of the target with $\hat{\mathbf{y}}_0$ and compute $\{\hat{p}_u(\hat{\mathbf{y}}_0)\}_{u=1\ldots m}$, $\rho[\hat{\mathbf{p}}(\mathbf{y}_0), \hat{\mathbf{q}}] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{\mathbf{y}}_0)\hat{q}_u}$ and $\{w_i\}_{i=1\ldots n_h}$.

2. Find the next location of the target candidate by:

$$\hat{\mathbf{y}}_1 = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i g(\| \frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h} \|^2)}{\sum_{i=1}^{n_h} w_i g(\| \frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h} \|^2)} \tag{2.23}$$

where $g(x) = -k'(x)$.

3. While $\rho[\hat{\mathbf{p}}(\mathbf{y}_1), \hat{\mathbf{q}}] < \rho[\hat{\mathbf{p}}(\mathbf{y}_0), \hat{\mathbf{q}}]$, do $\hat{\mathbf{y}}_1 \leftarrow \frac{1}{2}(\hat{\mathbf{y}}_0 + \hat{\mathbf{y}}_1)$.

4. If $\| \hat{\mathbf{y}}_1 - \hat{\mathbf{y}}_0 \| < \epsilon$ Stop; otherwise, set $\hat{\mathbf{y}}_0 \leftarrow \hat{\mathbf{y}}_1$ and go to Step 1.

2.1.4 Summary

In general, visual tracking algorithms can be grouped into two categories: deterministic tracking and stochastic tracking [110]. Deterministic motion tracking algorithms like optical flow or KLT trackers usually implicitly assume the object appearance is corrupted by the Gaussian noise. However, the Gaussian noise model is often invalid in practice which may lead to a local optimal solution. Deterministic methods usually only keep the estimated location information about the object (center coordinates or bounding box) during tracking, which causes the tracker to drift away easily when it is in local extrema. Stochastic tracking frameworks, such as extended Kalman filtering, the CONDENSATION algorithm, particle filter algorithms, etc., often maintain a probability distribution of the ob-

ject location, hence are more robust than the deterministic algorithms due to their capability to escape local extrema.

2.2 Time Reversibility

Time-reversibility, also called as time-reversal symmetry or T-symmetry [42], is an important concept in many disciplines. As an important concept used in this dissertation, it has not been defined clearly in the field of visual tracking before. Therefore, it is important to give a clear description of this concept before we propose our tracking and performance evaluation algorithms that use the time reversibility constraint. In this section, we will review the definitions of time-reversibility in some fields closely related to visual tracking and briefly discuss it as it applies to visual tracking.

2.2.1 Time Reversible Markov Chains

Time-reversibility is an interesting topic in stochastic process. Here we will review the definition of time reversibility in one of the most important classes of stochastic processes - discrete time Markov chain [78].

Suppose at time n, an ergodic Markov chain has already reached its stationary status for a long time with transition probabilities P_{ij} and stationary probabilities π_i, if we reverse the sequence of states to $X_n, X_{n-1}, X_{n-2}, \cdots$, we can easily verify that this reversed sequence is also a Markov chain. Since independence is a symmetric relationship and the conditional distribution of the future states

$X_{m+2}, X_{m+3}, \cdots$ given the present state X_{m+1} is independent of the past state X_m, X_{m-1}, the conditional distribution of $X_m, X_{m-1}, \cdots$ given X_{m+1} is also independent of $X_{m+2}, X_{m+3}, \cdots$. Its transition probabilities, denoted by Q_{ij}, can be computed as:

$$\begin{aligned} Q_{ij} &= P\{X_m = j | X_{m+1} = i\} \\ &= \frac{\pi_j P_{ij}}{\pi_i} \end{aligned} \tag{2.24}$$

If $Q_{ij} = P_{ij}$ for all i, j, that is, the reversed Markov chain has the same transition probability matrix as the original Markov chain, then the Markov chain is called time reversible. The time-reversibility condition can also be expressed as:

$$\pi_i P_{ij} = \pi_j P_{ji} \forall ij \tag{2.25}$$

This condition is also known as the detailed balance condition. It means the probability of seeing a transition from j to i is the same as seeing a transition from i to j. Note that the detailed balance condition is stronger than that required merely for a stationary distribution. Detailed balance also implies that around any closed cycle of states, there is no net flow of probability.

$$P_{ik} P_{kj} P_{ji} = P_{ij} P_{jk} P_{ki} \forall i, j, k \tag{2.26}$$

Detailed balance is a weaker condition than requiring the transition matrix to be symmetric, $P_{ij} = P_{ji}$. That would imply that the uniform distribution over the states would automatically be stationary distribution [9].

2.2.2 Brownian Motion

Since many stochastic visual tracking algorithms adopt Brownian motion models for the object, here we discuss its time-reversibility separately. Brownian motion is a special stochastic process. It lies in the intersection of several important classes of processes. It is a Gaussian Markov process, has continuous paths, a process with stationary independent increments (a Levy process), and is also a martingale [15] [78].

It was widely believed that it was the Scottish botanist Robert Brown who first discovered this kind of motion in 1827 when he was studying pollen particles floating in water under the microscope. He observed the pollen grains or particles of dust exhibiting a jittery motion in the water, but he could not explain the origin of the motion. Many scientists tried to provide explanations for Brownian motion after its discovery. The mathematics behind Brownian motion was first described by Thorvald N. Thiele in 1880 [58]. By studying the stock and option markets using stochastic analysis, Louis Bachelier presented the first theory of Brownian motion in 1900 in his PhD dissertation "The theory of speculation" [68].

The first concise definition of the Brownian motion process was given by Wiener in 1918, therefore, it is sometimes called the Wiener process. A continuous-time stochastic process $B(t), t \geq 0$ is said to be a Brownian motion process if [78]:

- $B(0) = 0$;
- $B(t), t \geq 0$ has stationary and independent increments;

- For every $t > 0$, $B(t)$ is normally distributed with mean 0 and variance $\sigma^2 t$.

Consider$(B_t, 0 \leq t \leq 1)$, under the time reversal transformation, define $(X_t, 0 \leq t \leq 1)$ by $X_t = B_{1-t} - B_1$, then the time reversed process $(X_t, 0 \leq t \leq 1) \stackrel{d}{=} (B_t, 0 \leq t \leq 1)$, in this sense, Brownian motion is time-reversible.

Physically, it seems that Brownian motion apparently violates the Second Law of Thermodynamics, because the motion of a Brownian particle never slows down. At the end of the 19th century, Louis Georges Gouy suggested that "Brownian motion might offer a 'natural laboratory' in which to directly examine how kinetic theory and thermodynamics could be reconciled" [39]. This mystery was solved by Einstein [27] [39]. In 1905, Einstein developed a statistical molecular theory of liquids. When he applied the molecular theory of heat to liquids, he predicted that the random motions of molecules in a liquid impacting on larger suspended particles would result in irregular, random motions of the particles. His prediction corresponded to Brownian motion precisely. Einstein's work on Brownian motion indirectly confirmed the existence of atoms and molecules, and in 1908 Perrin and his students conducted an exhaustive set of experiments and confirmed Einstein's theory [39].

2.2.3 Time Reversibility in Optimal Filtering Process

Statistical mechanics, also known as statistical thermodynamics, provides macroscopic predictions based on microscopic properties of the system. Under certain conditions, the evolution of the observable dynamical variables of a thermody-

namics system can be captured in a Markov process formulation, in particular, through Kolmogorov's forward equation as the evolution equation for the statistical state as a probability distribution or density function [76]. A considerable amount of work has been done on this subject.

Recently, Mitter and Newton studied the information and entropy flow in optimal filtering algorithms [80]. They found that the information flows for such filters can be explained by the entropy flows of non-equilibrium statistical mechanics. Usually, this kind of statistical mechanical system is held away from its equilibrium state by an interaction with an exogenous system.

By introducing some thermodynamics concepts, the average energy, entropy and free energy, into the optimal filtering process, they analyzed the signal process, observation process, the filtering process and their interactions using principles from statistical mechanics. Their work provides helpful intuitive insights into the optimal filtering process from a statistical mechanics perspective. They also addressed the time reversal property in such filtering processes. Their results are developed for stable, time-homogeneous systems. By appropriate initializations for both the signal and observation, they investigated the evolution of both linear and nonlinear filters in finite time interval. We summarize their main conclusions below [80] [69] [70] [71]:

- The signal X can be thought of as an abstract statistical mechanical system interacting with a heat bath. This interaction drives the signal towards a stationary state - its invariant distribution which minimizes its free energy. During

this convergence, it can be shown the entropy of the "universe" containing the signal and the heat bath is non-decreasing and the abstract system obeys a law similar to the Second Law of Thermodynamics. Here using statistical mechanics terminologies, a system in its invariant distribution can be in its stationary equilibrium states or stationary non-equilibrium states according to the net flow of entropy. The authors also defined the entropy production rate of the signal process and showed if the signal is self-adjoint [80], then this flow is zero in the stationary state, which is also an equilibrium state; otherwise, it is called a non-equilibrium state. When the signal is in its stationary state there is on average no flow of energy between the heat bath and the signal. However, for individual outcomes of the signal there is a continuous exchange of energy back and forth between the components.

- The existence of the observation process Y can reduce the entropy of the universe containing the signal and the heat bath, resembling the role of Maxwell's demon. Analogous to the thermodynamics concepts, the energy of the signal can be split into two components, one is in analogy to "work", which corresponds to the part which can be observed by the observation; the rest can be regarded as "heat". Being able to partially observe the signal, the filter can extract energy from the signal and return it to the heat bath without increasing the entropy like a perfect Maxwell's demon [59]. In the stationary state, in the presence of observation, the overall rate of entropy change of the "universe" remains the same as the original one, but there is some energy circulating between

the heat bath, the conditional signal $\hat{X}$ and the filter, which shows similarity to a type of non-equilibrium statistical mechanics.

- When taking the signal and the conditional signal to make a joint process, in the linear Gaussian case, such as the Kalman-Bucy filter [80], it can be shown that the process is "physically reversible" in the sense that the interactive entropy flow of the time-reversed joint process is the same as that of the forward-time process in the stationary state. However, in nonlinear cases, this joint process is in general irreversible [71].

- In the case of a linear, partially observed, Gaussian system, since the signal, observation, and the conditional distribution of the signal are all Gaussian, the filter propagates the mean and covariance matrix of the conditional distribution. Through the physical analogy, the authors pointed out that "Bayesian filters achieves the maximum possible reduction in entropy from a given supply of observations, and stores no more information than is strictly necessary to do this" [80].

- The authors point out that the observations allow entropy reduction which providing a quantitative example for Landauer's Principle [57].

2.2.4 Time Reversibility in Visual Tracking System

In this dissertation, we are designing visual tracking systems that accept video data. The data is the record of the physical world. Although the dynamics of the objects moving in physical world could be complex and hard to model, once

the motion is recorded, it becomes a deterministic process, in the sense that the position of object is fixed in every time frame. If we play the video backward, it will exhibit the exact motion reversal property.

Since the purpose of visual tracking algorithm is to obtain the position of an object at every frame, a perfect tracker should give correct position of the object at every frame. If we view the output of the tracker, the estimated position, as a virtual object, in ideal situations when the tracker is perfect and the video is played back, this virtual object should also return to the starting point. This is referred to as time-reversibility of the tracking algorithm.

Most of the existing tracking algorithms are forward-only tracking. The identity of the object is usually loosely defined by its initial appearance in most tracking algorithms. As we analyzed before, the appearance of the object can change dramatically due to many possible factors during the tracking procedure, therefore the tracker can hardly know if it is tracking the correct object or not because using appearance information to verify the identity of the object is unreliable, especially when an updating strategy for the appearance model is adopted in the tracking algorithm. We notice that in practice neither the deterministic nor the stochastic algorithms can show time-reversibility property in the presence of tracking errors. Suppose a tracker starts from time 0 with a good initialization of the object and tracks forward to time t, the tracking procedure can be viewed as a process transporting information about the object contained at time 0 to time t. We claim that if the tracker tracks backward to time 0 and can not return to the initialization

point, then there is information loss during this round of the tracking procedure and its performance at time t is unlikely to be good.

Stochastic visual tracking algorithms usually assume that the object dynamics forms a temporal Markov chain. Brownian motion is a widely used motion model for generic object tracking using optimal filtering algorithms. We have already discussed their time-reversible properties. Time-reversibility in stochastic processes has statistical meanings which differ from those in deterministic processes. For a specific object tracking using optimal filtering algorithms, only one realization of the stochastic tracking process is performed; therefore, their statistical time-reversal properties discussed earlier are different from the one we define a perfect tracker, where it requires exact trajectory reversal. We will further discuss these points in detail in the next two chapters.

Chapter 3

Online Performance Evaluation of Visual Tracking Algorithms

3.1 Visual Tracking and Performance Evaluation

Visual tracking forms one of the most important components in a wide range of application domains. Robust tracking of features forms the primary input to classical vision problems such as structure from motion and registration. In addition, tracking finds use in diverse application areas such as surveillance, markerless motion capture and medical imaging. The need for robust tracking algorithms that work over a broad spectrum of application domains cannot be understated. However, practical realities and the diverse nature of data dictate that even the most sophisticated algorithm will have failure modes where the tracking performance is poor and the algorithm loses track. *In this chapter, we address the problem of automatic evaluation of tracking algorithms with the goal of detecting track failures and evaluation of tracking performance without the need for ground truth.*

There are multiple reasons why a self-evaluation framework is needed. Its most straightforward use is in online characterization of tracking performance to enable a system to recover from track failures. Further, in the context of dis-

tributed sensor network, evaluation of the performance of the tracking algorithm (associated with each modality) can be used to characterize its reliability for the tasks of multi-modal fusion.

Self-evaluation can also be used to rank different tracking algorithms based on their performance. In this sense, self-evaluation can be used to choose a tracking algorithm with better performance at run time. It also potentially allows for tracking algorithms to tune their parameters to the specifics of an individual video (as opposed to a training set, which may or may not capture the nuances of a single instance). While ground truth allows the same, it is not self-contained to the tracking algorithm and is not easily extensible.

There exist many evaluation schemes [8] [73] [82] that use ground-truth information to evaluate tracking algorithms, and more importantly rank-order them in terms of performance. The PETS[1] and CLEAR[2] workshops, along with the ETISEO [72] effort focused mainly towards characterizing algorithms in terms of performance *in the presence of ground truth*. The CAVAIR[3] and the VACE[4] efforts were geared towards evaluation of object detection and tracking [54, 67]. In addition to this, there has been some research on defining distance metrics for matching the ground truth information to the tracker outputs, and in tuning the parameters of the tracking algorithm [33]. However, collecting ground truth is time consuming, and has its own variabilities [61]. Further, performance evalua-

[1] http://petsmetrics.net
[2] http://isl.ira.uka.de/clear07
[3] http://homepages.inf.ed.ac.uk/rbf/CAVIAR/
[4] http://www.ic-arda.org/InfoExploit/vace/

tion using ground truth is not possible for real time field testing or on sequences which are unlabeled. This motivates the need for online performance characterization in the absence of ground truth.

Evaluation of tracking performance and detection of track failure is similar to the problem of model validation, especially when the underlying formulation is in terms of dynamical systems. Tracking performance is bound to deteriorate when the data violates the modeling assumptions significantly. There exist many ways to detect the incompatibility between the models and the observed data. For stochastic nonlinear systems, measurements based on the innovation error forms a common choice as an evaluation metric. The statistics of the innovation error can be cross-checked with those of the model (such as white Gaussian noise), and a hypothesis test can be performed to determine model validity. Similar metrics such as the tracking error (TE) and observation likelihood (OL), and their corresponding cumulative summations in time (CUSUM) have been used for change detection and model validation [92]. TE and OL detect only sharp changes which results in loss of track, and do not register slow changes. A statistic for detection of slow changes called the negative expected log likelihood of state (ELL) and its generalization, gELL are proposed in [92]. ELL is defined as a measure of inaccuracy between the posterior at time t and the t-step ahead prediction of the prior state distribution. Interestingly, as we point out later, the evaluation methodology proposed in this chapter mirrors the ELL method in spirit.

In [2] [34] [94], under the hypothesis that the model is correct, a random pro-

cess in the scalar observation space is shown to be a realization of independent and identically distributed variables uniformly distributed on interval $[0, 1]$. This result holds for any time series and may be used in statistical tests to determine the adequacy of the model. An extension to vector-valued measurements is presented in [23], where a χ^2-test for multi-dimensional uniform distribution is used to determine if the system behaves consistently. However, when it comes to visual tracking, as the observation could be in a very high-dimensional image space, the computation of the test statistics is infeasible. In [63], an entropy based criterion is used to evaluate the statistical characteristics of the tracked density function. The definition of good performance for tracking a single object is that the posterior distribution is unimodal and of low variance. In contrast, a multi-modal and a high variance distribution implies poor or lost tracking. In practice, tracking in the presence of multiple objects and clutter does lead to the presence of multi-modality in the object's posterior density. This, however, does not necessarily imply poor tracking.

While model validation and change detection literature offers formal and rigorous approaches to formulate the problem, in many cases, the underlying models for tracking are unable to handle wide variations that occur in visual tracking problems. Further, given the complexity of the visual information, it is virtually impossible to accurately model all the information in all its variabilities. Towards this end, there has been a body of research that exploits the inherent characteristics of tracking output to automatically characterize the performance.

In [28], Erdem et al. address an on-line performance evaluation method for contour tracking. Metrics based on color and motion differences along the boundary of the estimated object are used to localize regions where segmentation results are of similar quality, and combined to provide a quantitative evaluation of boundary segmentation and tracking. As an extension, [29] uses a feedback loop to adjust the weights assigned to the features used for tracking and segmentation. This method of evaluation is specific to contour-based tracking systems. We also presented a method for self-evaluation in [101] before. This empirical method evaluates the trajectory complexity, motion smoothness, scale constancy, shape and appearance similarity, combining each evaluation result to form a total score of the tracking quality. However, this heuristic method can only be applied to a static camera system.

In this chapter, we propose an online evaluation methodology that can be applied to many tracking algorithms to detect tracking failures and to evaluate tracking performance [100]. The intuition behind our algorithm lies in the reversibility of the physical motion exhibited by an object, and relating it to the time-reversibility of the tracking algorithm. When this tracking problem is defined in terms of dynamical systems exhibiting Markovian properties, we construct a time-reversed Markov chain for the sole purpose of evaluation. The posterior probability density of the time-reversed chain is propagated all the way back to the initial time instant when the tracking algorithm is initialized. The prior used to initialize the tracker is now compared to the posterior of the time-reversed

chain to form the evaluation statistic. For a well behaved system, the two probability distributions are expected to show proximity in some statistical sense, with significant discrepancies between them in the presence of tracking error. The proposed approach finds applicability in a host of tracking algorithms that use a dynamical system formulation. In this regard, the use of particle filtering for estimating inferences is very common given the non-linearity of most models and the non-Gaussian noise distribution. The proposed evaluation method involves filtering back to the initial time instant, and gets slower with increasing time. Hence, we also propose an approximation by tracking back and comparing the performance against a point in time where by prior verification we are confident that the performance is good. We analyze the performance of the evaluation methodology by extensive experimentation using a wide variety of videos. It is shown that when ground truth is available, the track failures detected by our approach correlate significantly with those validated by the ground truth. We also show the applicability of the core ideas for tracking algorithms which are not modeled as dynamical systems. Examples of such algorithms include the KLT feature tracker [64] [89] [85] and the mean-shift tracking algorithm [16] [17]. Finally, we show that the proposed evaluation algorithm can be used for ranking different tracking algorithms based on their performance.

3.2 Particle Filtering in Visual Tracking

In Chapter 2, we have summarized the necessary background of Bayesian filtering methods used in dynamical systems, in particular, the particle filtering method which is used widely in visual tracking systems [43] [44]. In this section, we briefly discuss some of the common state space approaches to visual tracking focusing in particular on the kind of motion and appearance models commonly used.

3.2.1 State Models

For rigid objects, most tracking algorithms formulate tracking over a state space that typically comprises of locations on the image plane, the scale and orientation of the object all of which can be re-parametrized as affine deformations of some basic shape. For non-rigid objects, the affine deformation state could be extended to include contour deformation parameters (usually encoded with splines or level sets). Finally, the state space may include components that relate to the appearance of the object, so as to characterize and track the changes in object's appearance with changing pose and illumination.

The state transition model for the dynamical system is usually the motion model describing the kinematics of the object. Depending on the requirements of the application, these could vary from a simple Brownian motion model or a constant velocity model, to activity specific motion models [93] when tracking complicated behaviors that have been learned a priori.

3.2.2 Observation Models

Finally, probably the most important component is the observation model, typically a characterization of the object's appearance encoded either as a gray-scale or color template, or a histogram of such features. The key property of the observation model is that it provides discriminability of object-specific features over background and other scene constructs. Further, the models are expected to be fairly robust to outliers. Finally, there is the need for robustness to changes in object pose and scene illumination. This can be achieved by explicitly modeling such pose and illumination parameters in the state space of the system, or by having observation models that are invariant (partially or otherwise) to such changes.

In this context, it is important to discuss the role of online appearance models for visual tracking. In many practical systems (especially in surveillance), most objects are opportunistic, with the tracking algorithm having no significant prior characterization of their appearance. In such a scenario, the only identification of object appearance is in the initial frame provided to the tracker, typically in the form of the prior density of the object. As the object moves in the scene, online appearance models (OAMs) try to adapt to the changing appearance of the object. However, the OAM needs to be updated in order to incorporate new features exhibited by the object without introducing undesirable background artifacts that could potentially cause the tracking algorithm to diverge. This results in two contradicting requirements for the adaptation rules used to update the OAM. The need for updating the appearance models to account for the changing appearance

of the object is balanced by the possibility that undesirable background artifacts might be introduced. Invariably, a strategy is chosen that balances these two effects. This leads to scenarios in visual tracking, where the appearance models may no longer represent the same object that was used in initialization. Hence, *this leads to a case where the tracking performance is poor not because of incompatibility between the models and the data (the premise of model validation) or because of lack of smoothness and continuity of tracks (the premise of heuristic works), but because the model characterizing the dynamical system are fundamentally flawed due to undesirable updates.*

In the next section, we outline our approach for performance evaluation, including detection of error such as the one described above. The key point that we like to retain from the discussions given above is the overwhelming role of the prior density in defining the object identity.

3.3 Online Performance Evaluation Using Time-Reversed Chains

It is insightful to understand the challenges in visual tracking, and where some of the existing tracking algorithms and evaluation schemes fail. We begin with a discussion of failure modalities of tracking algorithms and the challenges in designing a self-evaluation scheme.

3.3.1 Failure modes of Tracking Algorithms

Visual tracking needs to be robust against a wide variety of operating conditions, dealing with poor video resolution, occlusion, changes in pose and illumination, camera motion and clutter. Under such diverse operating conditions, descriptions of objects, such as appearance, color, shape and texture almost always change unpredictably. At the same time, motion consistency is a feature that most algorithms use to reduce the search space, and it is one feature that is frequently violated when the camera itself is moving.

The range of failures is even more enhanced when the tracking algorithm uses an adaptive and online observation (appearance, shape) model. Adaptive appearance models are crucial for achieving robustness to changing pose and illumination. However, there is almost always the posibility of incorporating undesirable features into the model, such as features from the background. However, in spite of the large variations in operating conditions, the identity encoded by the appearance and shape information at the initializing frame provides a reference for validation. This forms the basis for the intuition behind the algorithm proposed in this dissertation.

3.3.2 Intuition

Our goal is to provide a general, online evaluation method for visual tracking systems based on dynamical systems. We will refer the Markov chain associated with the tracker algorithm as the "forward" chain. The prior used to initialize the for-

ward chain is the reference distribution which we use to evaluate the performance of the tracking algorithm. In order to evaluate the tracking performance at a time instant (say $t = t_0$) we first need to account for the difference in time instant between the prior ($t = 0$) and the output of the tracker. To achieve this, we construct a time-reversed Markov chain with models that are similar to the forward chain. The key idea is to compute the posterior distribution of this time-reversed Markov chain at the initialization time ($t = 0$) and compare it to the prior of the forward chain. For algorithms employing OAMs, the *identity* of the object is defined in the initializing frame and the prior used to initialize the system. This prior information encodes all the knowledge given to the tracking algorithm, and arguably is most critical in determining the performance of the algorithm. In this sense, the tracking performance can be determined by verifying the output of the tracker at any particular time instant (say $t = t_0$) against the prior with suitable handling.

From the point of view of information captured in the tracking algorithms, the underlying intuition is that if, at time t, the tracker contains enough information about the object, then the ability to track well until time t along the forward Markov chain implies that it is very likely to be able to track back to the end along a time-reversed Markov chain equally well.

To get an intuitive understanding of the proposed algorithm, consider a video sequence in which the first frame and the last frame are identical (in camera placement as well as the location of every scene and object point). Good tracking

performance would require a tracking algorithm to localize the object in the last frame at the same location as the prior given in the first frame. Such an idea is exploited for detecting drift in feature point tracking in [108]. Our algorithm can be viewed as an extension of that idea for performance characterization.

3.3.3 Formalizing the Concept

The forward Markov chain describing the tracking algorithm is defined using the prior density $p(x_0)$, the state model $p(x_t|x_{t-1})$ and the observation models $p(y_t|x_t)$. At time T, given an observation sequence $\mathcal{Y}_T = \{y_1, \ldots, y_T\}$, the posterior is $\pi_T = p(x_T|\mathcal{Y}_T)$. To evaluate the performance of the system, we propose a backward time tracker that uses π_T as its prior and the observation sequence $\mathcal{Y}_T$ in the time-reversed order. Using the notation $q(\cdot)$ for probability density functions associated with the time-reversed system, the reverse tracker is formulated as follows. For evaluation at time T, the system is initialized at time $T+1$ and filtered through the observations $\mathcal{Y}_T$.

- **Prior at time $T+1$:**

$$\begin{aligned} q(x_{T+1}) &= p(x_{T+1}|\mathcal{Y}_T) \\ &= \int p(x_{T+1}|x_T)p(x_T|\mathcal{Y}_T)dx_T \end{aligned} \tag{3.1}$$

- **State Transition Model:** For $t \in (0, T)$,

$$q(x_t|x_{t+1}) = \frac{p(x_{t+1}|x_t)p(x_t)}{p(x_{t+1})} \tag{3.2}$$

  This can be directly computed from the models for most systems used to define the tracking problem.

- **Observation Model:** We retain the same observation model used in the forward model.

$$\forall t, q(y_t|x_t) = p(y_t|x_t) \tag{3.3}$$

With this characterization of the system, we can now filter the observation sequence $\mathcal{Y}_T^b = \{y_T, \ldots, y_1\}$ in reverse time. The posterior density function of this filter is of great interest to us. At time t, the posterior density $\pi_t^b = q(x_t|\mathcal{Y}_t^b) = q(x_t|y_T, y_{T-1}, \ldots, y_t)$.

We can now estimate the posterior density at time $t = 0$, π_0^b by recursion. From intuition, we expect this density to be close in some statistical sense to the prior density $p(x_0)$. the following property.

Suppose we initialize the time-reversed Markov chain using the density $p(x_{T+1})$ as opposed to $p(x_{T+1}|\mathcal{Y}_T)$. It is easy to verify that the final posterior distribution in the time-reversed process is equal to the smoothing result [55] at the beginning of the forward process using all the observations till time T, i.e, $\pi_0^b = p(x_0|y_{1,\ldots,T})$.

Now, π_0^b and the $p(x_0)$ are close in the sense that

$$\int x_0 p(x_0) dx_0 = \int_{\mathcal{Y}_t} \int_{x_0} x_0 \pi_0^b d\mathcal{Y}_t dx_0 \tag{3.4}$$

Suppose we compare $E(x_0)$ and $E_{\mathcal{Y}_t}(x_0)$, then on an average (over the ensemble set of possible observations) the two means will be the same.

It should be noted that the above result is true only when the time-reversed system is initialized with the prior $p(x_{T+1})$. However, for most tracking models, it is the observation model with its characterization of object appearance and/or shape that allows for discrimination of the object from the background. In this

sense, the observation model allows for accurate localization (or equivalently, low variance estimation) of the object with the state model used mainly to regularize and smoothen the result. Further, under the assumption that the data $\mathcal{Y}_T$ fits the underlying models, the density $p(x_{T+1}|\mathcal{Y}_T)$ is expected to localize the object better, in the sense of the sharpness of the density around its expected value. Hence, the system defined with prior $p(x_{T+1}|\mathcal{Y}_T)$ is *over-trained* and provides a model that fits the data better.

3.3.4 Evaluation Statistic

There exist distance metrics and measures for comparing density functions such as the Kullback-Leibler (KL) divergence and the Bhattacharya distance [26]. However, in our case, the distributions are represented by particles or samples from the density function. In general, given the differences in the individual proposal densities and random number generators, the exact locations at which the densities are sampled will be different. Computing the KL divergence or the Bhattacharya distance for such non-overlapping sample sets would require interpolation (using Parzen windows [26]) or the use of approximations such as the Unscented Transformation [35]. We circumvent this problem with the use of the Mahalanobis distance that depends only on the moments of the distributions.

Denoting p as the prior distribution $p(x_0)$ and π as the posterior of the time-reversed chain $q(x_0|\mathcal{Y}_T)$, the distance $d(p, \pi)$ between the two distributions can be

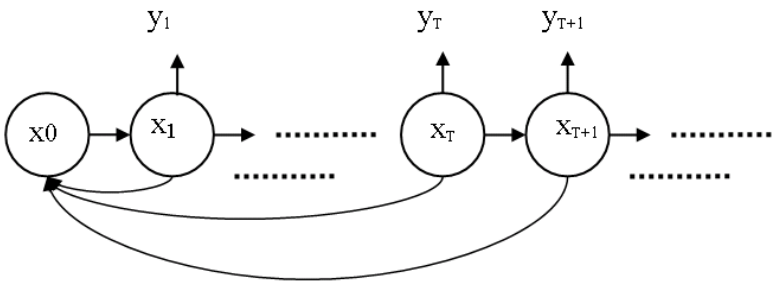

Figure 3.1: Schematic of the reference point used in the proposed algorithm. Evaluation of the performance of the tracker requires validation with the prior density using a time-reversed chain for suitable time normalization.

computed as:

$$\begin{aligned} d(p,\pi) &= (\mu_p - \mu_\pi)^T \Sigma_p^{-1} (\mu_p - \mu_\pi) + \\ & \quad (\mu_p - \mu_\pi)^T \Sigma_\pi^{-1} (\mu_p - \mu_\pi) \end{aligned} \tag{3.5}$$

where μ_p and Σ_p are the mean and the covariance matrix of the distribution p and μ_π and Σ_π are those of the distribution π, all of which can be easily computed or estimated from the particles or in some cases, analytically. An outline of the proposed evaluation framework is in Table 3.1.

3.3.5 Fast Approximation

The proposed evaluation framework poses a requirement to process (or track) across all the frames seen by the tracking algorithm. For such an algorithm, the computational requirements increase linearly with the number of frames (see Figure 3.1). This makes it increasingly harder for the evaluation algorithm to satisfy real time constraints.

However, a set of sufficient (though not necessary) conditions can be designed

Table 3.1: Outline of the Proposed Evaluation Algorithm.

To evaluate the performance of the tracker at time T, the density π_T is represented by the samples $\{x_t^{(i)}\}_{i=1}^N$,

1. Propagate the particles using $p(x_{T+1}|x_T)$ to get samples from $p(x_{T+1}|\mathcal{Y}_T)$,

$$\tilde{x}_{T+1}^{(i)} \sim p(x_{T+1}|x_T^{(i)}), i = 1, \ldots, N \tag{3.6}$$

2. Using the prior represented by the particle set $\{\tilde{x}_{T+1}^{(i)}\}_{i=1}^N$, iterate the steps 3, 4 and 5 for $t \in \{T, T-1, \ldots, 1\}$,

3. Proposition: At time t, propose a new particle set $\{\tilde{x}_t^{(i)}\}_{i=1}^N$ using the state transition model,

$$\tilde{x}_t^{(i)} \sim q(x_t|\tilde{x}_{t+1}^{(i)}), i = 1, \ldots, N \tag{3.7}$$

4. Weight Computation: Compute the weight $w_t^{(i)}$ associated with the particle $\tilde{x}_t^{(i)}$,

$$w_t^{(i)} = q(y_t|x_t^{(i)}) \tag{3.8}$$

5. Normalize the weights and resample them to obtain an unweighted particle set.

6. Using the particle set $\tilde{x}_0^{(i)} \sim q(\tilde{x}_0|\mathcal{Y}_T)$, compute mean $\hat{\mu}_\pi$ and covariance matrix $\hat{\Sigma}_\pi$ using sample statistics.

7. The evaluation statistic is computed using (3.5).

to alleviate this problem. We argue that if the performance at time T is good, then not only does the final posterior match well with the prior density, but that the posterior densities of the forward and backward tracker should match at all intermediate time instants. A fast approximation is now proposed using this observation. Suppose at time t_0, the performance of the system is evaluated to be good, then for an evaluation at a future time instant $t' > t_0$, the time t_0 can be used as a reference point in the place of $t = 0$ (see Figure 3.2). Extending this concept, we can recursively shift the reference point to keep a constant upper bound on the computational time for the evaluation. Let Δt be the time interval between successive reference points, i.e, the time instants $t_0 = 0, \Delta t, 2\Delta t, 3\Delta, \ldots,$ are used as the reference points. For a time instant t', the reference point chosen is $\Delta t \lfloor t'/\Delta t \rfloor$. However, the suitability of the approximation depends on the length Δt. The trade-off here is between the computation time, which is proportional to Δt and the ability to detect slow changes that are of the order Δt. A clever choice of Δt can go a long way in reducing the computational requirements of the proposed algorithm.

Finally, even with the approximation scheme described above, it might be difficult to achieve real-time processing for the evaluation at every time instant. However, online evaluation in real time is possible if we do not perform evaluation at every frame. For most practical systems, evaluation needs to be performed at regular time intervals. Choosing a fast approximation scheme with Δt as the time difference between reference points as well as the time instants when

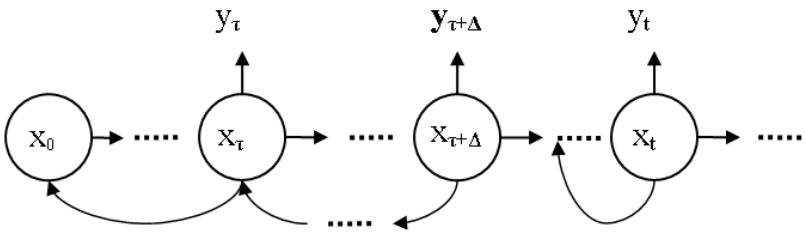

Figure 3.2: Schematic of the reference point used in the faster approximation to the proposed algorithm. As opposed to the implementation described in Figure 3.1, the approximation shifts the reference point from $t = 0$ to create multiple reference points separated by time interval of $\Delta t = \Delta$. This keeps the overall computational requirements for the evaluation scheme bounded.

evaluation is performed can go a long way in reducing the computing requirements for evaluation.

3.4 Extensions beyond particle filtering

3.4.1 Evaluations of the Kanade-Lucas-Tomasi tracker

The basic idea of the proposed evaluation methodology can be used for tracking algorithms that do not use particle filtering. Here, we show how to apply the evaluation method for feature point tracking using the KLT algorithm [64] [89] [85]. KLT is among the most widely used feature point trackers for many applications and we use it for showcasing our evaluation algorithm.

The original KLT algorithm works under the assumption of brightness constancy and small motion (typically, translation), that is, $I(t, \mathbf{p}) = I(t+1, \mathbf{p} + \mathbf{\Delta p})$ where $I(t, \mathbf{p})$ is the intensity at pixel coordinate $\mathbf{p} = (\mathbf{x}, \mathbf{y})$ in the frame at time t.

Under this assumption a linear system in $\mathbf{\Delta p}$ is solved to obtain the translation. In practice, the assumptions of brightness constancy and small motion used in the derivation of the solution are almost always violated eventually, leading to drifts in the tracking of feature points. In vision problems, especially those pertaining to geometry (such as structure from motion and estimation of epipolar geometry, homography), the presence of drift contributes to measurement errors which could subsequently be exaggerated by the following estimation algorithms.

The proposed evaluation methodology provides an elegant way to evaluate the tracking performance. As in the case of the particle filter, we formulate a KLT tracker for tracking back from the current frame to the initial frame. On the one hand, if the assumptions of brightness constancy and small motion are indeed satisfied and that the tracking remains stable and free of drift, the KLT tracker is expected to work well both forward and backward directions. Brightness consistency as a constraint is inherently time-reversible and with sufficient smoothness on the function $(I, \mathbf{x})$ and its derivatives, it can be shown that the forward and time-reversed systems behave similarly under the small motion assumption.

On the other hand, in the presence of drift due to model failure, when we do the backward tracking, the tracker does not go back to the initialization point due to the unmodeled errors that affect tracking. Therefore, the strategy used earlier for evaluation of particle filtering-based trackers along with the fast approximation techniques is also applicable to the KLT tracker. The interested reader is referred to a feature point algorithm described in [98] that uses this concept of time-reversal

for achieving robust tracking.

Finally, KLT as a tracking algorithm is a point tracker and does not estimate uncertainty in any specific form (such as density or covariances). We base our evaluation statistic on just the Euclidean distance between the initial point provided to the tracker and the result of the time-reversed KLT tracker. The Euclidean distance between the two points replaces the Mahalanobis distance used for the particle filtering scenario.

3.4.2 Evaluations of the Mean-Shift tracker

The proposed evaluation algorithm can also be extended to the mean-shift tracker [16] [17]. The mean-shift tracker contains two major components: object representation and localization. Histogram-based appearance representation is adopted for the object. Object localization is achieved through a mean-shift optimization process. The mean-shift tracker is popular due to its computational efficiency and ease of implementation. However, there are two major limitations that usually cause the traditional mean-shift tracker to fail [105]. The first limitation is that the basic mean-shift procedure assumes that the scale of the object remains unchanged during tracking, which may not be true in many practical situations. To handle scale change, it will bring uncertainty to the convergence of the tracker. The second limitation is that the traditional mean-shift tracker uses radially symmetric kernels which cannot adequately represent various object shapes. Therefore, like other tracking algorithms, the traditional mean-shift tracker may also often encounter difficulties during tracking, which makes it necessary to evaluate

its performance in real-time.

Based on the same idea we used for particle filter based trackers and the KLT tracker, we evaluate the mean-shift tracker using the distance between the forward and backward tracker. Here, the status of the tracking object can be characterized by the location (assuming the scale remains constant). Hence, simple Euclidean distance between the forward and backward kernel modes found by the mean-shift method is used for evaluation.

3.5 Discussion

In this section, we discuss some of the properties of the evaluation algorithm. In particular we highlight potential similarities between our algorithms and tools in existing literature. For example, ideas similar to time-reversal have been applied to the image registration problem where it is desirable for the forward and the backward maps to be inverses of each other [104] [14].

3.5.1 Similarity to the ELL

The proposed evaluation methodology is similar to the ELL statistic [92] in spirit, both involving posterior of the tracking algorithm and the prior at time $t = 0$. ELL propagates the prior density to time t and computes the inaccuracy between the t-step predicted prior and the posterior π_t. In contrast, the proposed methodology time reverses the posterior π_t back to the initial time using a time-reversed system and compares it against the prior at time $t = 0$. The main difference in our

formulation is that the t-step reverse prediction is *conditioned* on the observed data, while the t-step prediction in ELL is unconditional.

3.5.2 Smoothing filter

In Bayesian smoothing algorithms, the quantity of interest is $p(x_t|y_{1:T})$, the posterior of the state conditioned on all observations $y_{1:T}$, including those in the future. Computation of these smoothing posteriors involves running a forward PF and a backward PF and fusing their respective posteriors systematically [55]. However, in the smoothing algorithm, there are no new constraints that are used, in the sense, that the dynamical system model (prior + state transition + observation models) is still the same. However, the proposed evaluation method depends on this concept of time-reversibility of the physical models, which is a property that is extraneous to the basic definition of the dynamical systems. In this regard, the concept of smoothing filters and the evaluation methodology are two different concepts; it is possible to apply the evaluation methodology to the smoothing filter.

3.5.3 Failure Modes of the Proposed Algorithm

While the proposed evaluation algorithm works well across a wide range of tracking algorithms (see Section 3.6), there are some cases when it fails. Such failures vary with the selected tracking model and the specifics of data. In particular, we discuss two cases where the evaluation algorithm can potentially fail.

The first scenario deals with tracking algorithms that lock onto the initial posi-

tion, thereby losing track of a moving object. However, the time-reversed tracker used for evaluation will also remain locked at the initial position (of the forward

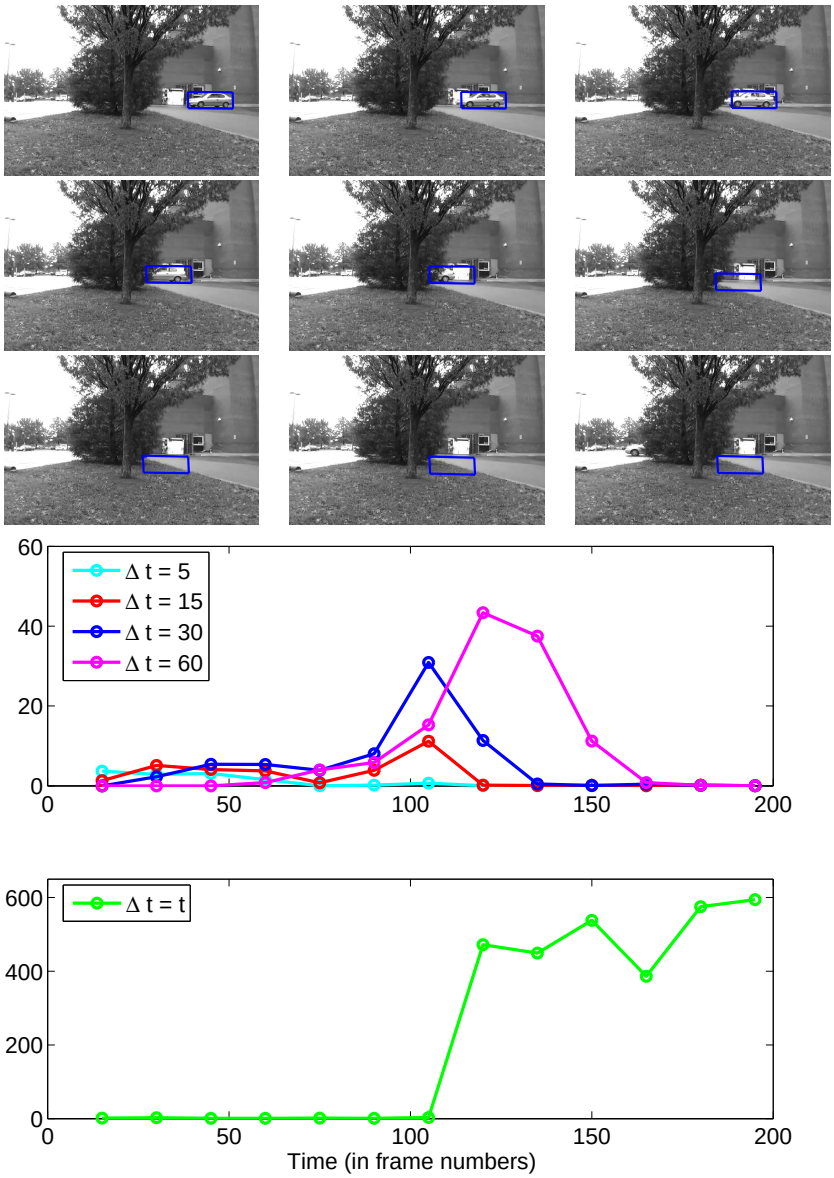

Figure 3.3: Performance evaluation over occlusion. The object is completely occluded by frame number 100. (Top left to bottom right) Tracking results at frame numbers 1, 20, 40, 60, 80, 100, 120, 135 and 150 (Bottom row) Evaluation results using the proposed algorithm ($\Delta t = t$) and its fast approximations ($\Delta t = 5, 15, 30, 60$).

tracker), and give low evaluation scores, indicating a good tracking performance. This is clearly a failure mode of the evaluation methodology, although for an unreasonable tracking algorithm. However, it highlights a potential scenario where the evaluation methodology might fail.

A second instance of failure involve trackers that are completely guided by their motion models. This could possibly happen due to the observations being rejected as outliers by the observation model, or in cases where a data association step associates a missing data state with the tracker. In such a case, the time-reversibility of the motion model (most commonly used motion models are time-reversible in the sense that the same model with different parameters can explain the time-reversed motion) would naturally guide the tracker back to its initial value.

A more realistic situation involves a combination of the two above-mentioned scenarios. Consider an example, where a tracking algorithm loses an object in the initial few frames of a video. For the remaining frames of the video, the output of the tracking algorithm is unpredictable. However, without sufficient observations to guide the estimate, the state transition model becomes the pre-dominant model in governing the evolution of the posterior density. For tracking algorithms that use a Brownian motion model on the state transition, the mean of the posterior does not change (and hence, remains close to the prior $p(x_0)$). The evaluation score in this case can possibly be of low value.

In short, the proposed method is very useful for many types of tracking prob-

lems; with certain potential failure modes that can be detected using simple heuristics. It is also noteworthy that the proposed evaluation might fail for a particular instance of data-algorithm pair, it does not have a consistent failure mode (say such as occlusion or illumination).

3.6 Experimental Results

In this section, we present experimental results of the proposed performance evaluation method with particle filtering-based visual trackers, the KLT and the mean-shift tracker. We first show that the proposed evaluation algorithm can detect various common failure modes in visual tracking systems using particle filters. We use the algorithm proposed in [110] as the representative tracking algorithm for this set of experiments. This algorithm uses a six-dimensional state space for capturing affine deformations, with a Brownian motion model for the state dynamics. The observation model is a template based OAM, which is a specific mixture of Gaussian model proposed in [46].

3.6.1 Evaluation under common Tracking scenarios

Figure 3.3 shows results for a video where the object is completely occluded. We used our evaluation algorithm once every 15 frames. The object undergoes occlusion around 100th frame. The proposed statistic and its fast approximations register peaks or sharp rises in value around this frame. It is noteworthy that evaluation using $\Delta t = 5$ does not seem large enough to capture the tracking

failure. However, a higher value of Δt registers the loss of track. Finally, as expected, inference using fast approximations is not useful after a track failure is registered. This is because that reference point against which the algorithm is being compared is corrupted.

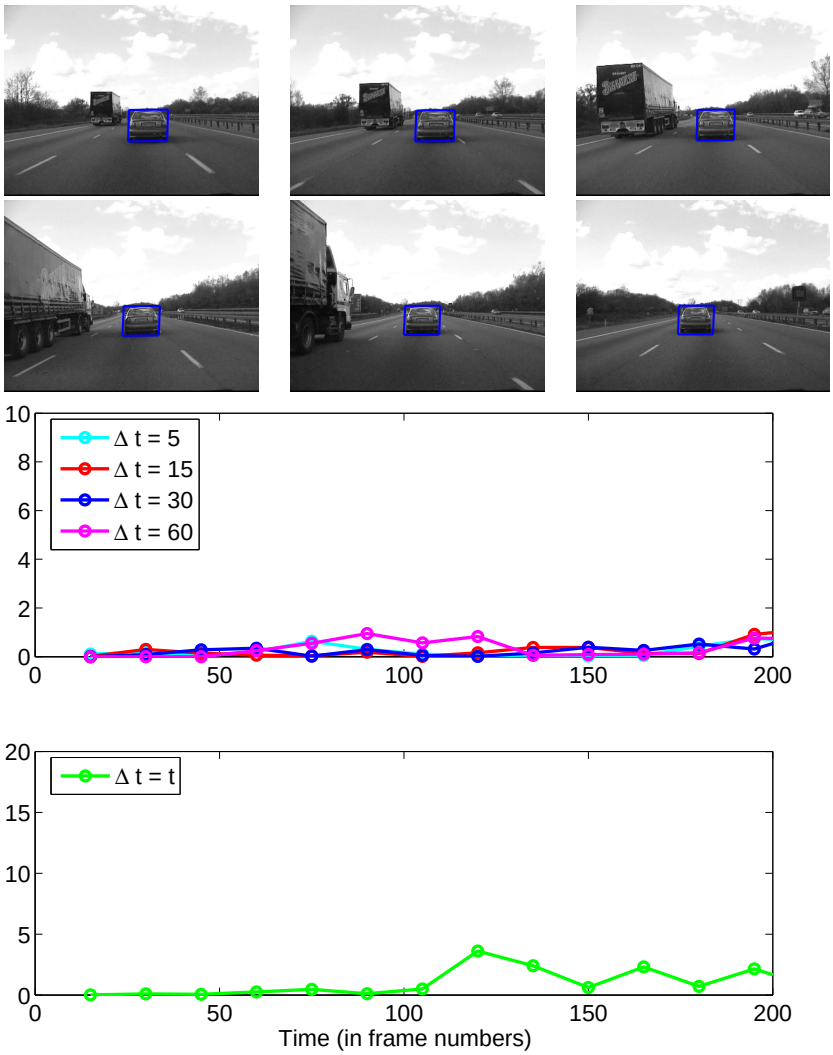

Figure 3.4: Performance evaluation over slow pose change. (Top three rows) Tracking results at frame numbers 1, 40, 80, 120, 160 and 200 (Bottom rows) Evaluation results using the proposed algorithm ($\Delta t = t$) and its fast approximations ($\Delta t = 5, 15, 30, 60$).

Figure 3.4 shows the results of evaluation for a sequence in which the object exhibits a small change in pose, easily tracked by the tracker. As expected, the proposed evaluation methodology generates a test statistic which takes low values indicating a good tracking performance. Figure 3.5 shows evaluation results on an aerial sequence in which the tracker loses track of the object due to the jerky motion of the camera. The test statistics registers sharp peaks around the point where the loss of track happens.

The proposed algorithm was tested on sequences in the PETS-2001 data set and the evaluation is compared with the ground truth. The comparison with the ground truth is done by computing the distance between the center of the object as hypothesized by the tracker to the ground truth. Figures 3.6 and 3.7 show the results on two sequences from the dataset. In Figure 3.6, the tracker tracks the object fairly well. Both the proposed statistic and the comparison against the ground truth take a low value. Figure 3.7 shows the evaluation results for a scenario involving tracking failure. While all statistics register the failure of track, the proposed statistic registers the track failure before the ground truth. This is because of the specific evaluation criterion used with the ground truth, which involves comparing only the centers of the object, while the bounding box is inaccurate before the loss of track (frame 60).

3.6.2 Receiver Operating Characteristic

To further give a statistical evaluation of the proposed evaluation method, we created a data set containing 40 sequences obtained from various scenarios, like out-

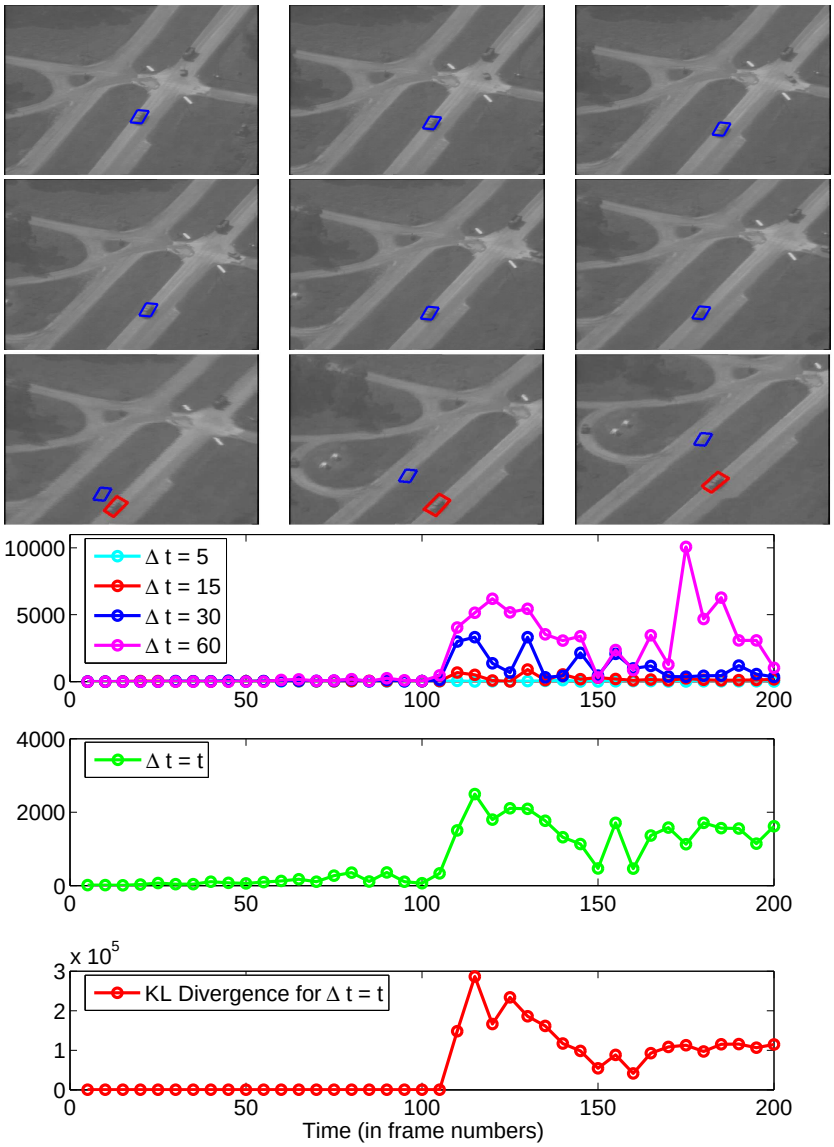

Figure 3.5: Performance evaluation in an aerial sequence. The tracker loses track of the object around frame 110 due to jerky camera motion. (Top three rows) Tracking results at frame numbers 1, 20, 40, 60, 80, 100, 120, 140 and 160. The true object location is marked in red after the algorithm loses track. (Bottom row) Evaluation results using the proposed algorithm ($\Delta t = t$), its fast approximations and the KL divergence between prior density and posterior of time-reversed chain.

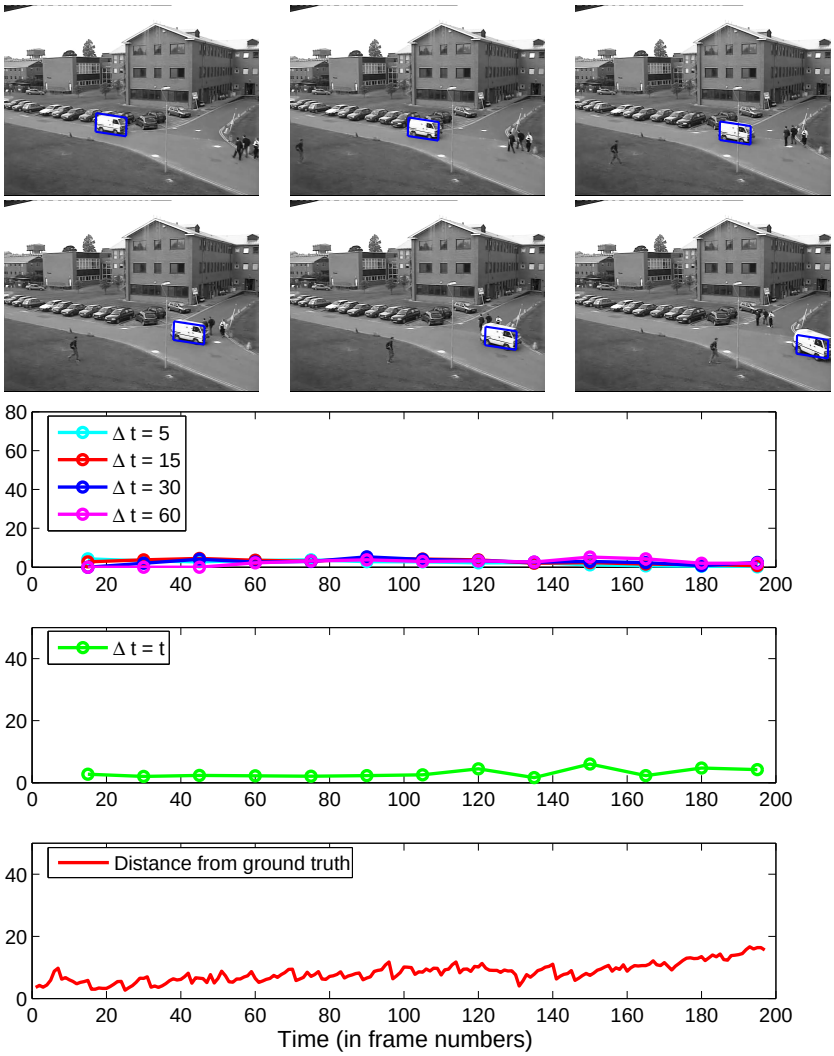

Figure 3.6: Performance evaluation on a PETS sequence including ground truth. (Top three rows) Tracking results at frame numbers 1, 30, 60, 90, 120 and 160. (bottom three rows) Evaluation results using proposed statistics and its fast approximations and the ground truth. Tracking performance remains fairly constant as shown by both the ground truth and the proposed evaluation strategy.

door/indoor, vehicle/human, optical/infrared, static/moving camera, ground/airborne, etc. These video sequences were each obtained from standard video datasets such as the PETS 2001, 2002 dataset, the aerial sequences from the VIVID dataset,

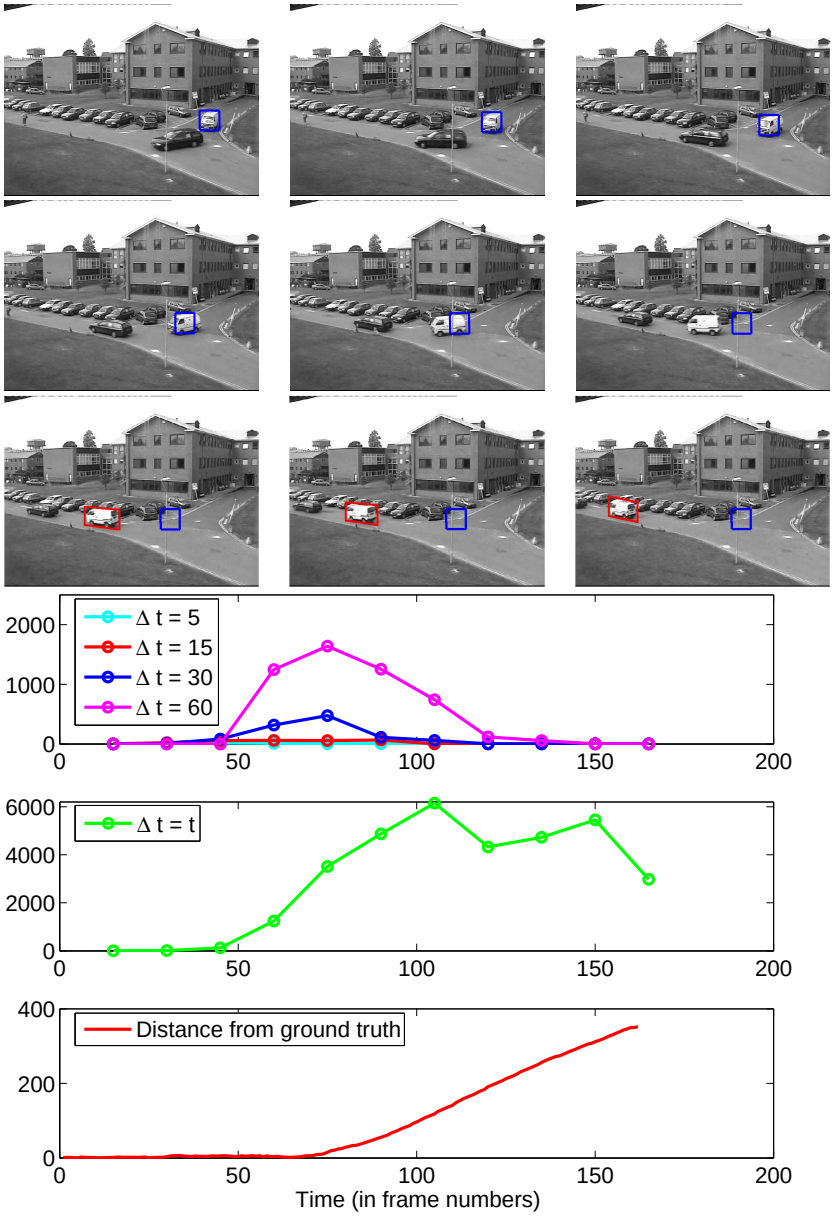

Figure 3.7: Performance evaluation for a PETS sequence including ground truth. (Top three rows) Tracking results at frame numbers 1, 20, 40, 60, 80, 100, 120, 140 and 160. The true object location is marked in red after the algorithm loses track. (bottom three rows) Evaluation results using proposed statistics and its fast approximations and the ground truth.

Figure 3.8: The collected data set for obtaining ROC curve of the proposed evaluation method.

the TSA dataset and other videos collected at the University of Maryland. Each sequence composes of 200 frames. The first frames of each sequence are shown in Figure 3.8.

Ground truth for each video was obtained manually, and comprises of a tight bounding box (parallelogram) around the object at frames $1, 20, 40, \ldots, 200$. A detection event corresponds to detecting the failure of the tracking algorithm. The true state of nature is obtained by using the spatial overlap between the ground truth and the region assigned as the object by the MAP estimate of the tracking algorithm. A low overlap between the two confirms that the tracking performance is poor, and is denoted as a detection of failure.

After obtaining the evaluation statistic values, we vary a threshold to get different detection and false alarm rates and plot the ROC curve. We plot operating

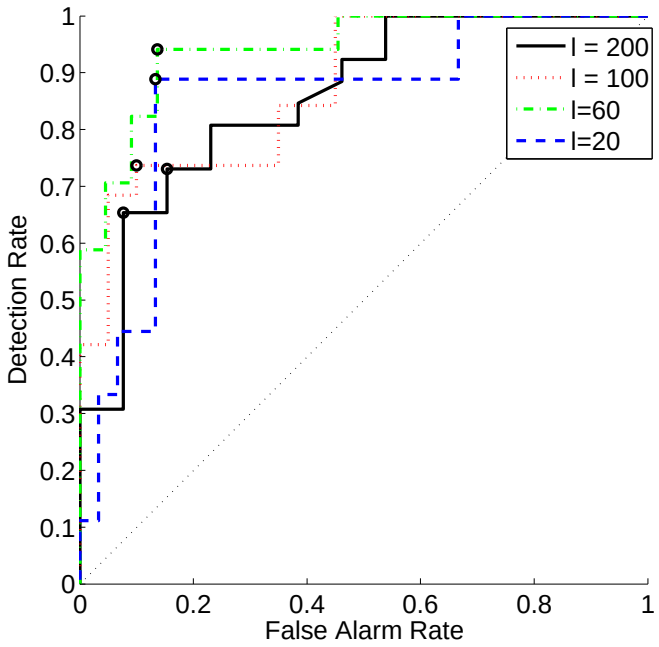

Figure 3.9: We characterize the performance of evaluation as the length of the video (number of frames) changes. The encircled points are the (Bayesian operating points) for equi-prior, and $0-1$ cost structure.

curves under various scenarios.

3.6.2.1 Length of the Video

We performed experiments characterizing the performance of the evaluation algorithm as the length of the video increases. This is to quantify the possible small degradation of performance as the length of the video increases. Figure 3.9 shows the ROC curves for videos of length $l = (20, 60, 100, 200)$ frames. Also marked are the Bayes' operating point for equi-prior and $0-1$ cost structure. This allows us to get a quantitative assessment of the Bayes risk and its degradation as the length of the video increases. This allows us to interpret the ROC curves better.

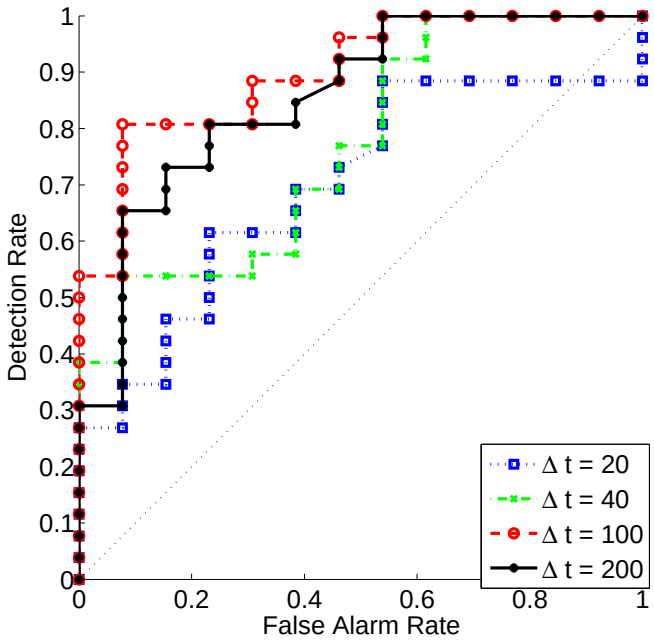

Figure 3.10: The ROC curves of the proposed evaluation method for OAM-based particle filtering. The evaluation was performed at the final frame of a 200 frame long video. Each line corresponds to a fast approximation scheme with different approximation length. Note that performance does not degrade much between the basic evaluation strategy $\Delta t = 200$ and an approximation $\Delta t = 100$.

For example, at $l = 60$ the detection probability is $P_D = 0.94$ at a false alarm probability $P_F = 0.13$, which falls to $P_D = 0.73$ when $l = 200$ (same the same false alarm rate P_F).

Length of the Video	20	60	100	200
Bayes Risk	0.12	0.1	0.18	0.21

Table 3.2: The Bayes risk of the evaluation algorithm.

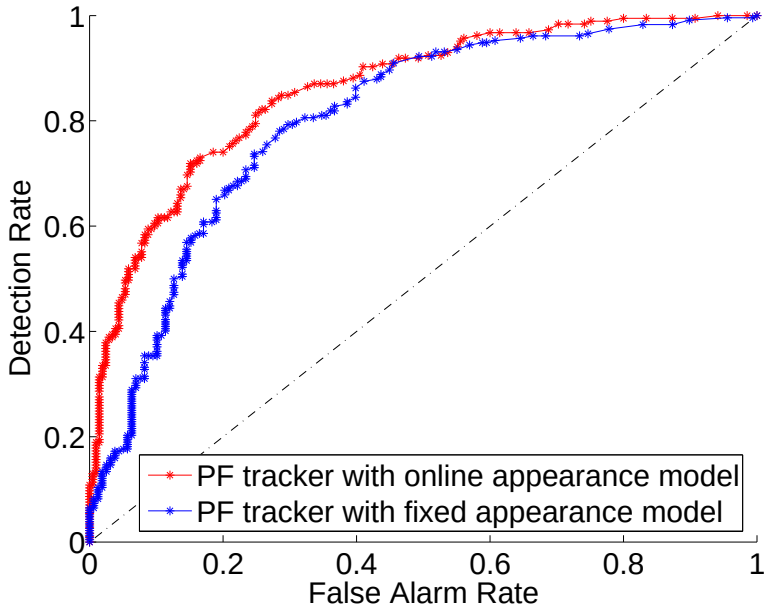

Figure 3.11: The performance comparisons of the proposed evaluation method for OAM-PF and FAM-PF trackers. The evaluation performance remains fairly same over two different tracking algorithms hinting at the robustness of the evaluation strategy over different tracking algorithms.

3.6.2.2 Fast Approximation

We next show the differences in performance between the basic evaluation method and its fast approximations at various Δt. The curves in Figure 3.10 show the ROC for evaluation at the last frame of the video (at $t = 200$) using the basic algorithm ($\Delta t = 200$) and fast approximations at $\Delta t = 20, 40, 100$. Note that in the fast approximation method, if an intermediate point is declared as a track failure, then all subsequent points are also declared as track failures. This contributes to the poor performance at $\Delta t = 20$. It is seen from the figure that with appropriate intervals, like 100 frames, the performance of the fast approximation strategy is comparable to the basic framework, while keeping the computation time constant.

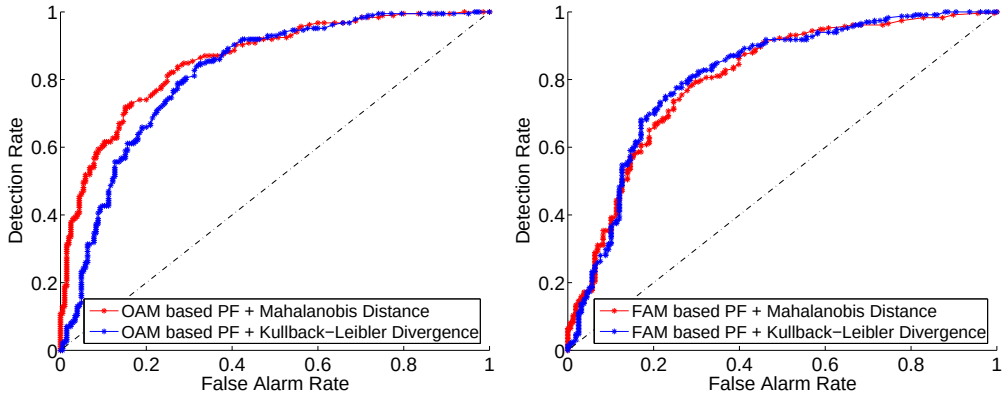

Figure 3.12: Performance comparisons of the proposed evaluation method by using Mahalanobis distance and KL divergence based evaluation statistics respectively. Evaluation performance seems fairly similar under either metric. This could possibly hint at the unimodality of the densities around the prior time instant $t = 0$. The similarity in evaluation justifies the use of the faster Mahalanobis distance in the place of KL divergence which is expensive to compute for point clouds.

3.6.2.3 Tracking Algorithm

We ran the evaluation method for particle filter-based tracking algorithms based on the OAM and a fixed appearance model (FAM). We show both the ROC curves in Figure 3.11. The test data set is the same for both trackers, while the evaluation performance is different by a small margin. Further, a comparison with the ROC curves in the evaluations of the KLT and mean-shift trackers (shown in Figures 3.16 and 3.14) suggests that the performance of the evaluation method may reveal some characteristics of the underlying tracking algorithm. We plan to explore this as a part of future work.

3.6.2.4 Choice of Evaluation Metric

In computing the evaluation statistic, we proposed to use Mahalanobis distance in place of distances such as the KL divergence which directly compares two densities. To test the effectiveness of this Mahalanbois metric, we also computed the KL divergence-based distance when using the basic framework where the computation is feasible given the Gaussian prior distribution. The comparisons in Figure 3.12 show that there is no significant difference between using the Mahalanobis distance and the KL divergence in our experiments. This could possibly be due to the unimodality of the densities around the prior time instant $t = 0$. The similarity in evaluation justifies the use of the faster Mahalanobis distance in place of the KL divergence which is expensive to compute for point clouds.

3.6.2.5 Evaluation of Mean Shift Tracker

Using the same data set as used for the particle filtering-based tracker evaluation, we tested the evaluation algorithm on the traditional mean-shift tracker. By excluding some sequences where the traditional mean-shift tracker completely fails from the very beginning, which makes the evaluation completely unreliable as we discussed earlier, the final test set contains 26 sequences and 260 evaluation points in total. Figure 3.13 shows the evaluation results for a sequence with slow tracking drift. The corresponding evaluation score for this sequence increases indicating the increasing drift in tracking. We use the same ground truth as in the particle filtering case. The true state of the track (failure or not) was determined

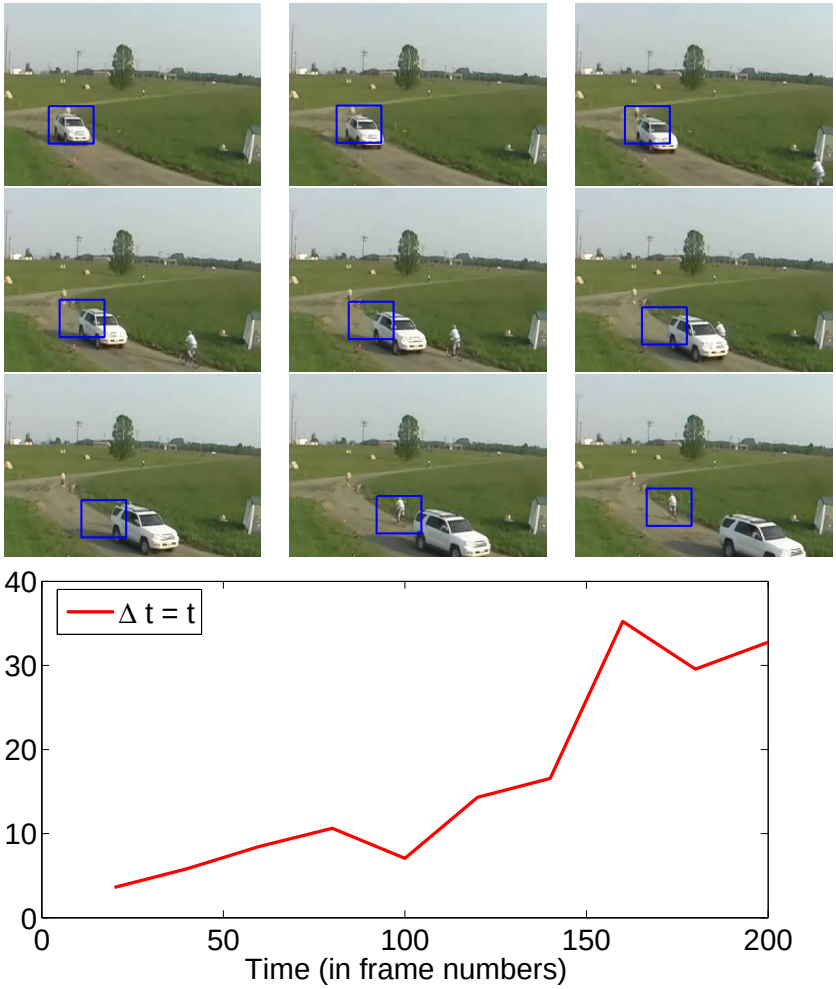

Figure 3.13: Performance evaluation for a sequence tracked by the mean shift tracker with slow tracking drift. (Top three rows) Tracking results at frame numbers 20, 40, 60, 100, 120, 140, 160, 180 and 200. (bottom) Evaluation results using the proposed statistics under the basic mode.

by comparing the hypothesized region to the ground truth. Lack of sufficient overlap between the two was labeled as a failed tracker. The evaluation metric was designed based on the distance between the output of the time-reversed mean-shift and the initial guess. The Euclidean distance between the two was used (as the scale of the tracker remains fixed, which makes the Euclidean distance almost

equivalent to spatial overlap). As before, we computed the ROC curve using the dataset of 26 sequences (see Figure 3.14) show-casing the performance of the evaluation method for the mean-shift tracker. We designed this work based on the code from http://www.cs.bilkent.edu.tr/ ismaila/MUSCLE/MSTracker.htm.

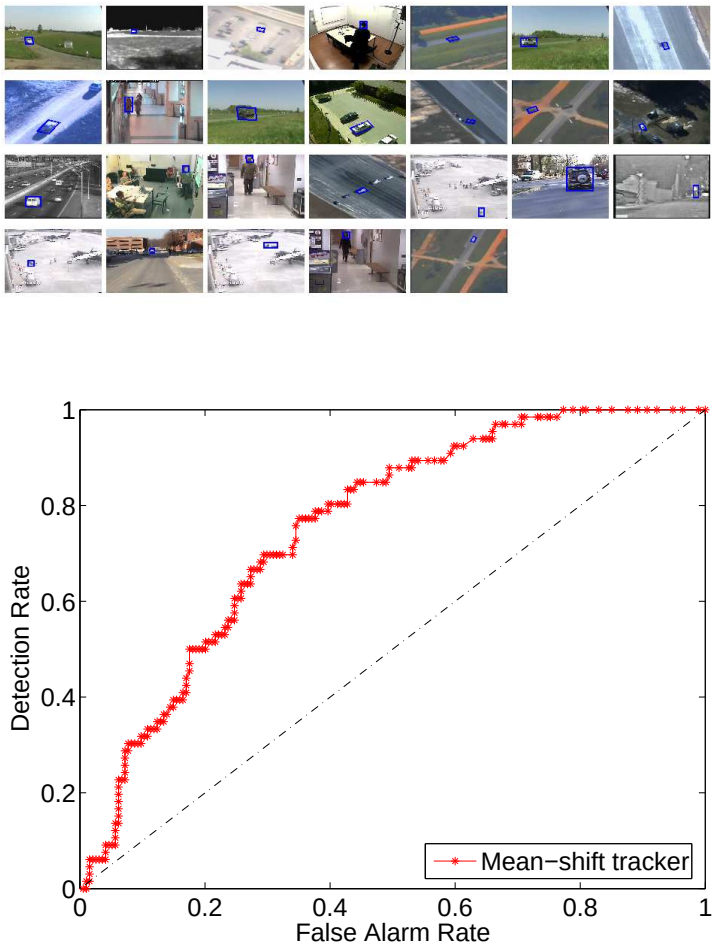

Figure 3.14: Evaluation results for the mean shift tracking algorithm over a dataset of 26 videos, snapshots from which are shown at the top. The ROC curve of the evaluation algorithm for the tracker using the basic mode is shown at the bottom. From 26 videos of 200 frames each, we obtained 260 evaluation points which were used to generate the ROC curves.

3.6.2.6 Evaluation of KLT Feature Tracker

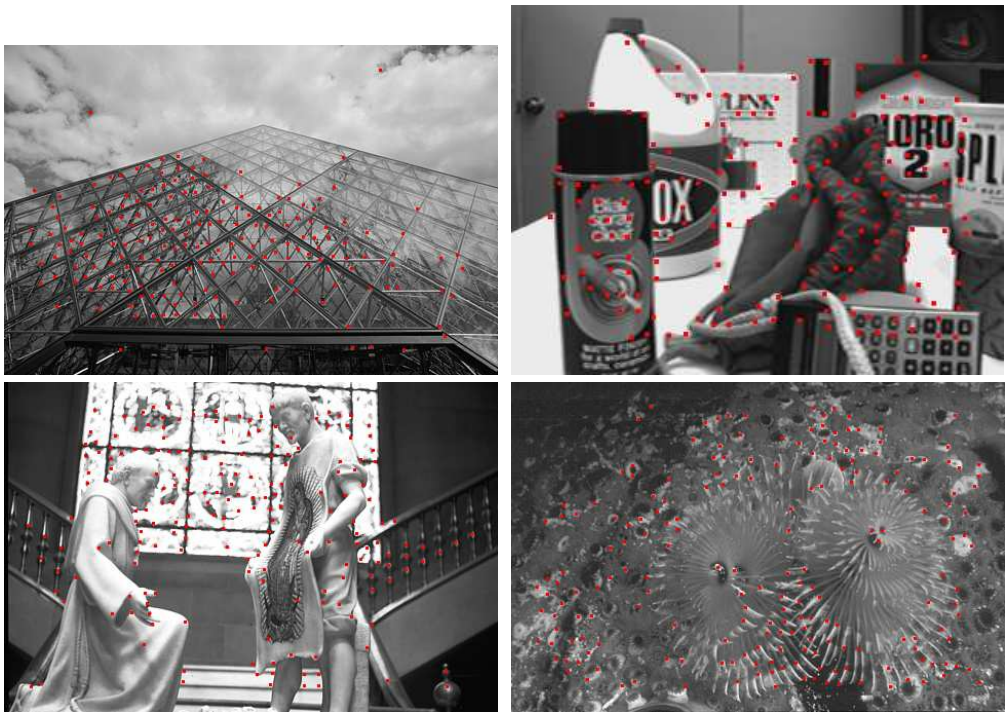

Figure 3.15: Test images used for the KLT tracking algorithm overlaid with the selected feature points. Each image was translated to create a synthetic video providing ground truth for evaluation.

We also tried to use the proposed method to detect tracking failures when the KLT feature tracker is used. The KLT is a feature point tracking algorithm and hence, we can generate multiple test cases from a single image. For our experiments, we used four images, and selected 200 features per image using the KL feature selection criterion (see Figure 3.15). Selecting 200 features per image gives us a mix of good and bad feature points (in terms of their tractability). We create a synthetic sequence by translating the images. This gives us the ground truth for the sequence. A feature point is considered to have drifted if it diverges

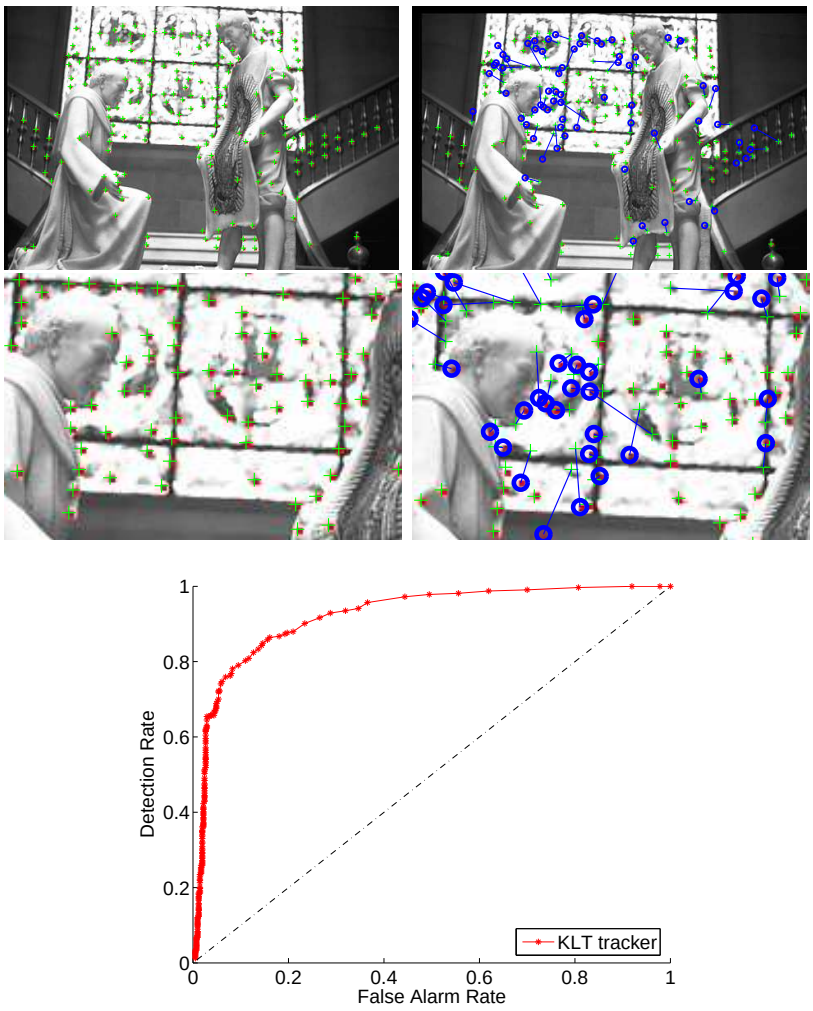

Figure 3.16: Performance of the evaluation method for the KLT feature point tracking algorithm. (Top left column) The initial frame and the enlarged details for KLT tracking. The red dot shows the initialization of the KLT tracker and the green plus sign shows the ground truth. (Top right column) The final frame and its enlarged details for KLT tracking. As we can see, many tracking feature points (red dot) have drifted away from their ground truth locations (green plus sign) and been detected by the evaluation algorithm (the blue circles indicate the drifted points which are connected with their ground truth locations by blue line). (Bottom) The ROC Curve of the evaluation method for KLT tracker using 4 images and 800 feature points.

by more than 2 pixels from the ground truth. As before, we use ROC curves to characterize the detection of drift using our evaluation methodology. The ROC curve in Figure 3.16 indicates that the evaluation method works very well for the KLT tracker. We also show some detection results in Figure 3.16.

3.6.3 Ranking the Performance of Trackers

We have showed above that the proposed online evaluation algorithm can detect tracking failures in the absence of ground truth data. In addition to this, the proposed algorithm can also be used to compare the performance of different trackers. We compared the performances of the three trackers used in this chapter: the particle filter-based tracker with OAM, the particle filter-based tracker with FAM and the mean-shift tracker. Since the KLT tracker is a feature tracking algorithm and requires a different test set, we did not include it in this ranking experiment.

The experiment was performed as follows. For each tracker, we count the number of tracking failures reported at different false alarm rates over the data set shown in Figure 3.14. For each tracker, we have 260 evaluation points (26 sequences, with 10 evaluation points each). We can see from the figure that at a false alarm rate of 0.6, the detection rates for all three trackers are very close, therefore we can compare the performance of each tracker in terms of the number of detected tracking failures at this point. Intuitively, a tracking algorithm with more detected track failure should correspond to a poorer tracking performance. From the figure, the ranking order for the three trackers we used here results in

(PF tracker with FAM) < (PF tracker with OAM) < (Mean-shift tracker), from left to right, worst to best performance. Notice that:

$$\begin{aligned} D_{\text{bad}} &= G_{\text{bad}} * \text{Detection Rate} + G_{\text{good}} * \text{False Alarm Rate} \\ G_{\text{bad}} + G_{\text{good}} &= \text{Total number of evaluation points} \end{aligned} \tag{3.9}$$

where D_{bad} is the detected number of failures, G_{bad} is the real number of failures (ground truth).

With the help of the ROC curves of the proposed evaluation algorithm together with the number of detected failures, we can recover the ground truth number of tracking failures. The results are: 152 (PF tracker with FAM), 90 (PF tracker with OAM) and 66 (Mean-shift tracker). As we can see, the ground truth ranking result of these three trackers gives the same ordering as the proposed evaluation algorithm.

It is noteworthy that the above comparison is valid only because that the detection rates for the tracking algorithms are similar at the false alarm rate of 0.6. At a different operating point where the detection rates are not similar (for the same false alarm) such a comparison becomes invalid as tracking with a higher detection rate tends to report larger number of detected failures.

3.6.4 Summary

To summarize, the following properties of the proposed evaluation scheme are highlighted. The proposed evaluation algorithm is shown to detect common failure modes in visual tracking and also compares favorably with ground truth based

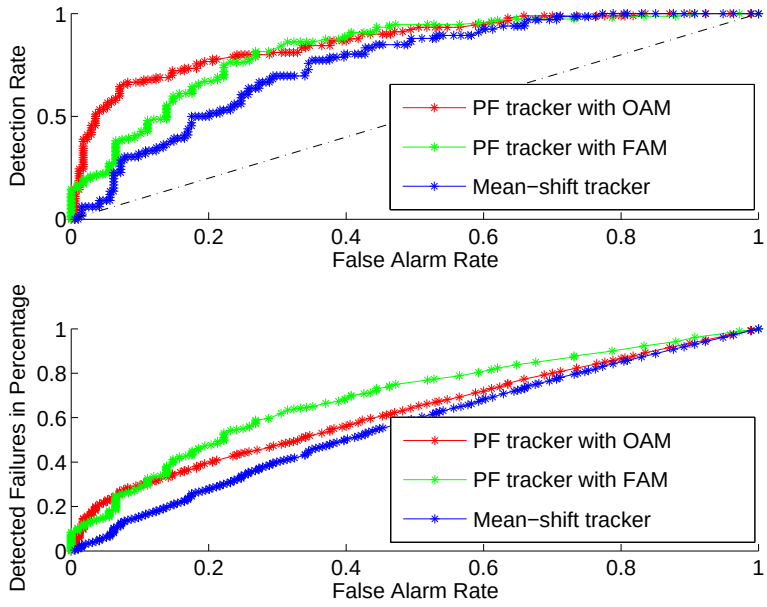

Figure 3.17: The ranking result of the three trackers: the particle filter-based tracker with OAM and FAM, the Mean-shift tracker. The above is the corresponding ROC curve of each tracker on the data set described in Figure 3.14. The bottom plot is the number of detected failures (the number is in percentage) using the proposed evaluation algorithm for each tracker at different false alarm rates.

evaluation. The value of Δt is shown to be critical in the efficiency of the fast approximations. A value of $\Delta t = 40$, or 60 seems reasonably large enough to register failures. It should be noted that fast approximations are meaningless after detection of failure, as the reference point against which they are compared does not correspond to good tracking. The choice of threshold to declare poor performance can be decided for a specific tracking system by inspection from the ROC curve. The choice is also influenced by the value of Δt. It can be seen that for all the experiments in this dissertation, the inference from the proposed evaluation agrees well with subjective evaluation of track failures. The supplemental material

includes videos showcasing the working of the evaluation algorithm.

3.7 Conclusion

In this chapter, we present a method to provide automatic and online evaluation of the tracking performance in visual systems without the knowledge of ground truth. The proposed evaluation algorithm works by verifying the prior at time $t = 0$ against the posterior of a time-reversed chain. The time-reversed chain is initialized using the posterior of the tracking algorithm. We characterize the performance of the algorithm using ROCs under various operating conditions. While the focus in the dissertation has been on systems using particle filtering, the evaluation method is fairly independent of the tracking algorithms used. In this regard, we show that the algorithm works well for other tracking approaches such as the KLT and the mean shift tracker. We also show that the evaluation methodology can also be used to rank different tracking algorithms according to their performance. We envision the use of the evaluation methodologies proposed in this dissertation for online verification and ranking of tracking performance. Future research efforts include tracking algorithms that optimize the evaluation metric so as to minimize the chances of track failure.

Chapter 4

Bi-directional Visual Tracking

4.1 Introduction

In the last chapter, we applied the time reversibility property of object motion to online performance evaluation of tracking systems. This online evaluation method can greatly improve the robustness of tracking systems and enable persistent tracking of objects. In this chapter, we continue to apply the time reversibility idea to improve the accuracy of visual trackers.

Given that the kinematic and dynamic information of the object can not be observed from the video directly, researchers try to infer the motion information through some observable properties of the object under motion, among which the appearance of the object is probably the most widely used. There are several tracking algorithms that have been developed based on the assumption that the appearance of the object is unchanged or the change can be explained using some object motion models. Most matching-based tracking algorithms fall in this category, along with the popular KLT [64] [89] [85] and the mean shift tracking algorithm [16] [17]. To improve the tracking performance, various 2D or 3D appearance models have been developed to handle the appearance change

during tracking. Researchers have also incorporated additional knowledge, like camera calibration, scene geometry, scene illumination, etc., for appearance-based tracking. Algorithms like CONDENSATION [44] and particle filtering based algorithms [25] directly incorporate the dynamic model of the object motion into tracking methods. These algorithms generally involve an object state transition model and an observation model, which combines both motion and appearance models of the object. The dynamic models used in these algorithms are generally loosely defined, which are good for general object motion, not just for a specific motion model, like constant speed, constant acceleration or a random walk plus some noise.

Researchers have taken various approaches to exploit and incorporate more information about the true object motion, like adaptive speed or trajectory prediction models [106]; however, to the best of our knowledge, a fundamental characteristic called time-reversibility of object motion has been ignored in most of the past and current works. Simply speaking, since all the targets to be tracked are macroscopic solid objects in the physical world and the physical laws of classical mechanics are time-symmetric, all the motion process of the targets should be time-reversible, which means that the time-reversed process satisfies the same dynamical equations as the original process. This motion reversibility directly implies the reversibility of tracking algorithms if they can perfectly follow the moving of objects. Therefore, a good tracking algorithm should perform equally well during backward tracking. However, most of the existing tracking algorithms

are designed by looking forward only in the time domain instead of looking bidirectionally during tracking.

If we look at the tracking problem as a black box shown in Figure 4.1, the inputs of the black box are the observations and object state at $t-1$, the output is the object state at the current frame t. The process of how the object evolves from the previous state to the current state is not totally obvious to the observers due to the limited information contained in the video data. We claim that irrespective of the specific trajectory of object motion, if we switch the time axis in the physical world, we can expect that the object will go back to the exactly same state at time $t-1$. Hence, if the tracking strategy does capture the true object motion during this time interval, which implies that the object state is well estimated at time t, then using the same tracking strategy, the backward estimated object state at time $t-1$ should be identical to the forward state at time $t-1$. On the contrary, if the tracking algorithms do not preserve the time-reversibility characteristic of the motion process, it is very possible that it has failed to track the object or may fail soon. In practice, it is very difficult for the forward tracking algorithms to maintain the time-reversibility property with time.

In this chapter, we present a new bi-directional KLT feature tracker. Instead of just looking forward in the time domain, we simultaneously perform both the forward and backward tracking using the time-reversibility constraint. The new algorithm reduces the possibility of the tracker getting stuck in local minima and significantly improves the tracking robustness and accuracy. The experimental

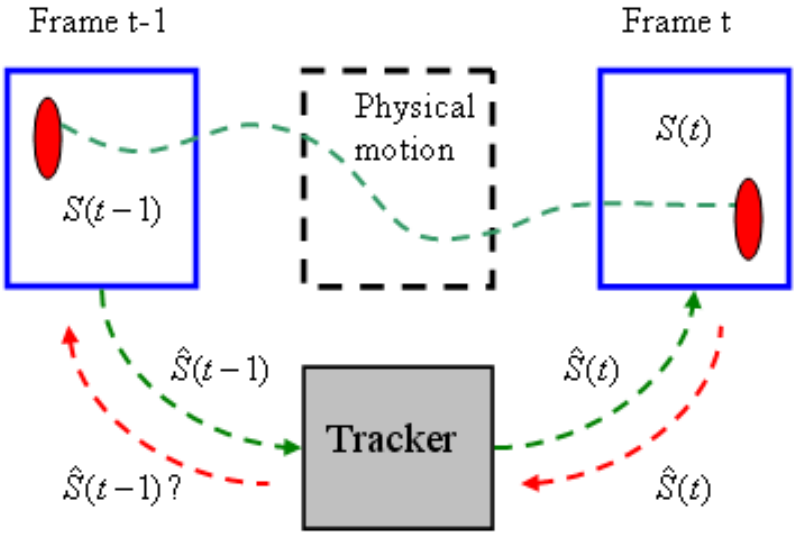

Figure 4.1: An illustration of visual tracking

results show that the improved KLT tracker significantly outperforms the original KLT tracker without demanding additional computational resources. The proposed time-reversibility constraint is general enough to be incorporated into many existing tracking algorithms, as well as for computing optical flow.

Before proceeding further, we clarify the notion of time-reversibility and distinguish it from backward processing or smoothing. Some algorithms also consider backward tracking [87] [10]; however, they perform forward and backward tracking separately and then merge the results in order to get better performance. There are some other works using backward tracking in video coding area [83] [84]; however, these strategies do not exploit the notion of time-reversibility in tracking. The consistent image registration method proposed by Christensen et al [14] [13] is probably the closest to exploiting time-reversibility in spirit.

CONDENSATION and particle smoothing methods [55] [45] [25] consider both the past and future observations to get a smoothed estimate, which also involves backward processing. There is no contradiction between the time-reversibility constraint and the Bayesian smoothing strategy. In Bayesian smoothing, the performance improvement is due to the backward-flow of information from future data while time-reversibility means that the entropy involved in the motion process is non-increasing. In practice, time-reversibility is only approximately satisfied due to noise or partial observations. Bayesian filtering or smoothing is trying to minimize the information decrease during tracking [80], which in effect is similar to using the time-reversibility constraint.

Many of the current tracking algorithms maximize a likelihood function or minimize a cost function. Since the likelihood functions usually have multiple local minima, especially for high dimensional data like images, the tracker may get stuck in some local minima in practice. In the approach using the time-reversibility constraint, the chance to find the global minima is higher because more constraints are incorporated into the optimization process. In the next section, we apply this bidirectional tracking strategy to the widely used KLT feature tracker.

4.2 The New KLT with the Time-Reversibility Constraint

In Chapter 2, we have introduced the KLT feature tracker. We notice that the final solution in (2.11) seems to smooth the forward and backward tracking results

by simply averaging them. In general the forward KLT results will differ from the backward KLT results due to asymmetry in image information. The symmetric expression in (2.8) may be confused with the time-reversibility constraint proposed in this dissertation. However, we can see from (2.8) that after switching to symmetric expression, the objective function tries to minimize the difference between $I(\mathbf{p}-\frac{\mathbf{d}}{2})$ and $J(\mathbf{p}+\frac{\mathbf{d}}{2})$ while the KLT method attempts to find the best match for the feature centering at $I(\mathbf{p})$. We will further explain this point in the next.

4.2.1 The New KLT using the Time-Reversibility Constraint

Following the original definitions in KLT, we propose a new objective function for KLT using the time-reversibility constraint.

$$\begin{aligned}(\hat{\mathbf{d}},\hat{\mathbf{d}}^{\mathbf{b}}) &= \arg\min_{\mathbf{d},\mathbf{d}^{\mathbf{b}}} \int\int_W [J(\mathbf{p}+\mathbf{d})-I(\mathbf{p})]^2 w(\mathbf{p}) d\mathbf{p} \\ &\quad + \int\int_W [I(\mathbf{p}+\mathbf{d}+\mathbf{d}^{\mathbf{b}})-J(\mathbf{p}+\mathbf{d})]^2 w(\mathbf{p}) d\mathbf{p} \\ &\quad + \lambda(\mathbf{d}+\mathbf{d}^{\mathbf{b}})^T(\mathbf{d}+\mathbf{d}^{\mathbf{b}}) \end{aligned} \tag{4.1}$$

where $\mathbf{d}^{\mathbf{b}}$ is the backward displacement vector when tracking from t to $t-1$, λ is a regularization parameter. Assuming $\mathbf{d}$ and $\mathbf{d}^{\mathbf{b}}$ are small, we can perform the following approximations using the first order Taylor expansion:

$$\begin{aligned} I(\mathbf{p}+\mathbf{d}+\mathbf{d}^{\mathbf{b}}) &\approx I(\mathbf{p}) + \nabla I(\mathbf{d}+\mathbf{d}^{\mathbf{b}}) \\ J(\mathbf{p}+\mathbf{d}) &\approx J(\mathbf{p}) + \nabla J\mathbf{d} \end{aligned} \tag{4.2}$$

By setting the derivatives with respect to $\mathbf{d}$ and $\mathbf{d}^{\mathbf{b}}$ to zero respectively, we

have the following constraints which are given in their discrete forms:

$$\begin{aligned} 0 &= \sum\sum_W [H(\nabla H - \nabla J)^T + (\nabla H^T \nabla I)\mathbf{d^b}] \\ &\quad + \sum\sum_W [(\nabla J^T \nabla J + \nabla H^T \nabla H)\mathbf{d}] + \lambda(\mathbf{d} + \mathbf{d^b}) \\ 0 &= \sum\sum_W [H\nabla I^T + (\nabla I^T \nabla H)\mathbf{d}] \\ &\quad + \sum\sum_W (\nabla I^T \nabla I)\mathbf{d^b} + \lambda(\mathbf{d} + \mathbf{d^b}) \end{aligned} \tag{4.3}$$

where $H = I - J$. Solving the above equation group, we finally get:

$$\begin{aligned} \mathbf{Ud} &= \varepsilon \\ \mathbf{U} &= AD^{-1}C + \lambda D^{-1}C - \frac{1}{2}B \\ \varepsilon &= (A + \lambda I)D^{-1}(V - W) + \frac{1}{2}(S - R) \end{aligned} \tag{4.4}$$

where the definitions of A, B, C, D, V, W, S, R are given below:

$$\begin{aligned} A &= \sum\sum_W (\nabla I)^T \nabla I; B = \sum\sum_W (\nabla I)^T \nabla J; \\ C &= \sum\sum_W (\nabla J)^T \nabla J; D = \sum\sum_W (\nabla J)^T \nabla I; \\ R &= \sum\sum_W I(\nabla I)^T; S = \sum\sum_W J(\nabla I)^T; \\ V &= \sum\sum_W I(\nabla J)^T; W = \sum\sum_W J(\nabla J)^T; \end{aligned} \tag{4.5}$$

If we re-write the original KLT equation using the above defined variables, we get:

$$\begin{aligned} \mathbf{Z} &= \frac{1}{2}(A + B + C + D); \\ \mathbf{e} &= \frac{1}{2}(R - S + V - W); \end{aligned} \tag{4.6}$$

4.2.2 Comparison to the Original KLT

By comparing the new KLT and the original KLT, we find that although all the variables appear in both the original and the new KLT equations, the interactions between these variables are different. The original KLT just linearly combines the forward-only and backward-only mappings while in the new KLT process, the forward and backward are performed simultaneously, actually improving the performances of each other.

Also, we notice that the new KLT has almost the same computational cost as the original one. Actually in practice, the new KLT even requires less computations than the original one because the required number of iterations is lower to achieve the same performance.

An interesting observation is that even when $\lambda = 0$, the new KLT still has a completely different expression from the original KLT. When $\lambda = 0$, $\mathbf{U}$ and ε are:

$$\begin{aligned} \mathbf{U} &= AD^{-1}C - \frac{1}{2}B \\ \varepsilon &= AD^{-1}(V - W) + \frac{1}{2}(S - R) \end{aligned} \tag{4.7}$$

This is due to the second term in (4.1). In our experiments, we found that the new KLT outperforms the original KLT even when λ equals to 0. The reason is that the first two terms in (4.1) still involve the interaction between forward and backward processing, implicitly enforcing the time-reversibility constraint. In fact, the second and third terms in (4.1) both improve the tracking performance. We will further study their contributions in section 4.3.

4.2.3 Good Features to Track

Shi and Tomasi [85] presented a method to select good features to track. The method is based on the rationale that if both the eigenvalues of the matrix R as defined above are larger than some threshold, which implies that the KLT equation can be robustly solved, then the feature in general will exhibit complex textures and be good for tracking.

Since the new KLT has the same form as the original one, we can also judge if a feature is good for tracking or not. Actually in both symmetric KLT and the proposed new KLT, the corresponding matrix contains information from two images, so the explicit physical interpretation of the eigenvalues is hard to see. Both the original KLT matrix and the new KLT matrix show good properties in terms of their stability in solving the equations. We studied the condition number of the both matrices, which is in general an indication of good stability if the value is close to 1. The experimental result shows that the condition number of the new KLT matrix is on the average more closer to 1 than the original KLT. This provides an explanation for the improvement due to the new constraint.

As different tracking algorithms lead to different matrices, evaluating if a feature is good or not for tracking is not completely determined by the characteristic of the feature itself. A bad feature in one algorithm can be good in the other. In the experiments, we find that the proposed new KLT can track some features well when the original KLT fails to track them.

4.3 Experimental Results and Discussions

We implemented the new KLT algorithm based on the latest C code of the original KLT algorithm which is widely used and can be downloaded from the website: http://www.ces.clemson.edu/ stb/klt/.

4.3.1 Performance Evaluation on Clean Sequences

4.3.1.1 On Real Sequence With No Ground Truth

First, we compare the results on the sequence contained in the KLT code package, which we call 'the Table sequence'. In the Table sequence the camera exhibits a rotation from left to right. To fairly evaluate the performance, we used the same set of parameters for both algorithms. We also disabled some thresholds for removing bad features during the tracking procedure unless they are out of the image region.

We selected 200 feature points for both algorithms using the same method contained in the package which is based on [85], so the starting feature points are the same for both algorithms. The iteration number is set to 10 which is sufficient for both algorithms to converge. Figure 4.2 shows the results for both algorithms at the starting and ending frames together with some enlarged details. The feature points are colored as red dots. We circled the points where the two algorithms differ significantly, say, by more than 3 pixels. By visual inspection, we found that the new KLT kept tracking well on those points while the original KLT lost track of them.

Figure 4.2: Tracking results of the original KLT (left column) and the proposed KLT (right column) using the time-reversibility constraint at the starting and ending frames. The green circled points on the left side differ significantly with the yellow circled points on the right side. It is easy to see that the new KLT keeps tracking those points well while the original KLT fails to track them.

error	$\lambda =$	0	0.05	0.2	2	20
n		1427	1427	1427	1427	1426
mean	original KLT	0.2351	0.2351	0.2351	0.2351	0.2353
	new KLT	0.1429	0.1424	0.1435	0.1487	0.1429
variance	original KLT	1.7419	1.7419	1.7419	1.7419	1.7431
	new KLT	0.7935	0.7936	0.7928	0.7983	0.7898

Table 4.1: The generated random translation is uniform between 0 and 12 pixels in both directions, where we can see the best performance is achieved when $\lambda = 0.05$. The value of λ is normalized by the size of the feature window. n is the number of points involved in the experiment. The value of error is in pixel unit.

4.3.1.2 On Synthesized Sequence With Generated Ground Truth

To further quantitatively evaluate the performance, we use the first frame of the Table sequence to generate a new sequence with random translations in both x and y directions. Therefore all the features points have the same motion as the generated random translations. Two-level image pyramids are used in both algorithms. The results of the algorithms are then compared with the ground truth. The effect of different λ has been studied on two synthetic sequences with different motions. We summarize the quantitative results in Table 4.1 and 4.2. For the sequence with translation distributed between 0 to 12, the mean error curves with respect to λ are plotted in Figure 4.3 with red and blue lines for the original and new KLT respectively. As we can see, the new KLT using the time-reversibility constraint consistently outperforms the original KLT in different λ values. The average improvement is more than 35%.

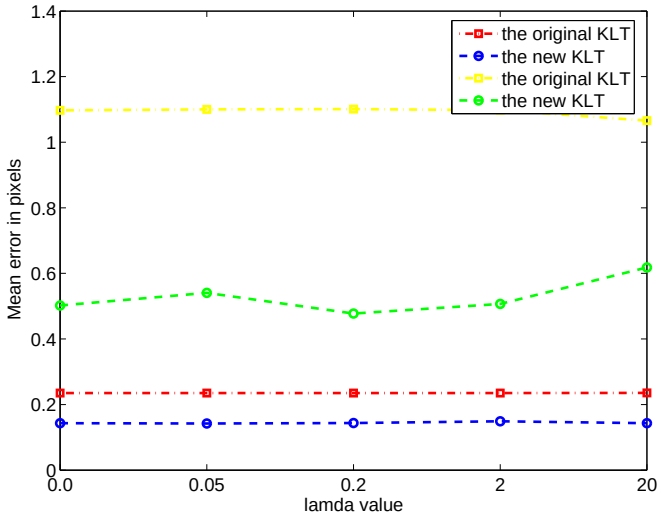

Figure 4.3: The mean errors of the original KLT and the new KLT. The red and blue lines are the results for the sequence with translation uniformly distributed from 0 to 12; the yellow and green lines are the results for the sequence with translation uniformly distributed from 0 to 20.

In this sequence, the change due to λ is quite small between $\lambda = [0, 20]$. Comparing the error at $\lambda = 0$ and others, there is not a significant change while the performance is still much better than the original KLT. This result shows that the second term in (4.1) does contribute to improving the results. Both the average errors of the original and new KLT methods on this sequence are quite small, primarily due to the motion being small translation. From these results, we probably can say that the improvement on good sequences mainly comes from the second term.

We also generated another sequence with increased translation magnitude. We can see the improvement due to the third term in Table 4.2. We find the value of λ to achieve the best performance increased in the new sequence with larger

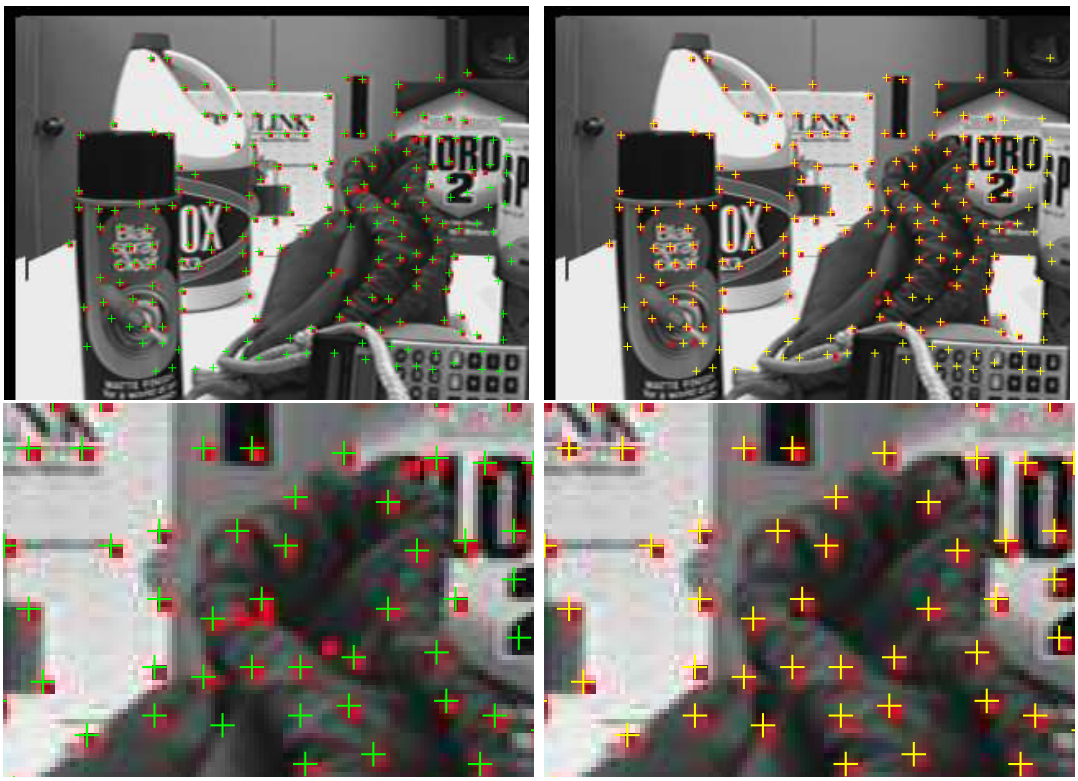

Figure 4.4: Tracking results of the original KLT (left column) and the proposed improved KLT (right column) at the ending frame on the generated sequence with known ground truth. The green and yellow cross signs provide the ground truth positions of the feature points. The up-left corner of the small red block should overlay on the center of the cross if tracking were perfect.

motion. The mean errors of both algorithms are shown in yellow and green curves respectively in Figure 4.3. This result tells us that the third term helps more when the original KLT has more difficulties in tracking than in situations where it is easier to track. Figure 4.4 shows the tracking results of both algorithms at the ending frame, where λ used in the new KLT equals to 0.2.

error	$\lambda =$	0	0.05	0.2	2	20
n		1235	1232	1231	1233	1237
mean	original KLT	1.0973	1.1001	1.1009	1.0974	1.0659
	new KLT	0.5019	0.5405	0.4777	0.5066	0.6181
variance	original KLT	40.2333	40.3281	40.3601	40.2967	39.2340
	new KLT	9.9528	13.7966	9.2691	9.4448	11.3291

Table 4.2: The generated random translation is uniform between 0 and 20 pixels in both directions, where we can see the best performance is achieved at $\lambda = 0.2$. The value of λ is normalized by the size of the feature window. n is the number of points involved in the experiment.The value of error is in pixel unit.

4.3.2 Performance Comparison on Noisy Sequences

We add Gaussian noise to the sequence with random translations uniformly distributed between 0 and 12. The variance of the zero-mean noise is 0.005 which is normalized by the range of the image intensity. Table 4.3 shows the results of both algorithms for different λ; and the mean errors are plotted in Figure 4.5. It is seen that the best performance is achieved at $\lambda = 40$, which confirms the above conclusion that larger value of λ should be used for more difficult sequences. This is because the values of the first two intensity evaluation terms increase for more difficult sequences; thus to make the third term comparable to the first two terms, a larger λ is needed. Figure 4.6 provides the tracking results for both algorithms, where λ equals to 40 in the proposed new KLT.

error	$\lambda =$	0	2	20	40	60
n		1311	1310	1282	1269	1264
mean	original KLT	1.3192	1.3158	1.3395	1.3331	1.3210
	new KLT	0.9666	0.9278	0.9180	0.9165	0.9306
variance	original KLT	11.9968	12.0254	12.3895	12.1590	11.7519
	new KLT	2.4290	2.3080	3.3098	3.4253	3.5468

Table 4.3: Results on the noisy sequence with random translation uniformly distributed from 0 and 12 pixels in both directions, where we see the best performance is achieved at $\lambda = 40$. The value of λ is normalized by the size of the feature window. n is the number of points involved in the experiment. The value of error is in pixel unit.

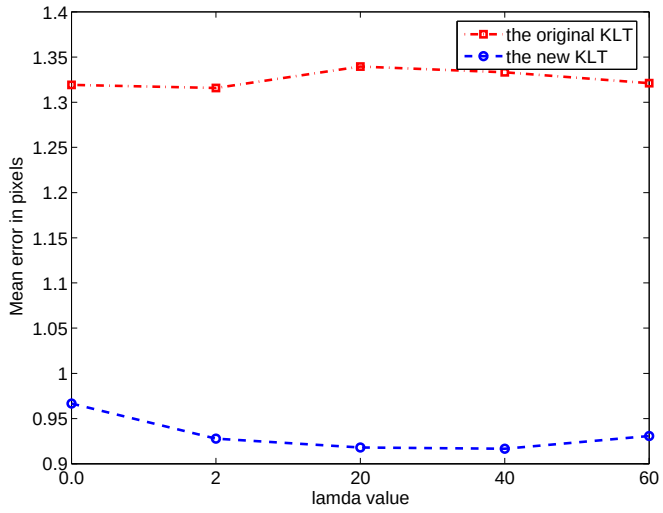

Figure 4.5: The mean error of the original KLT and the new KLT. Gaussian noise is added in the sequence with translations uniformly distributed from 0 to 12.

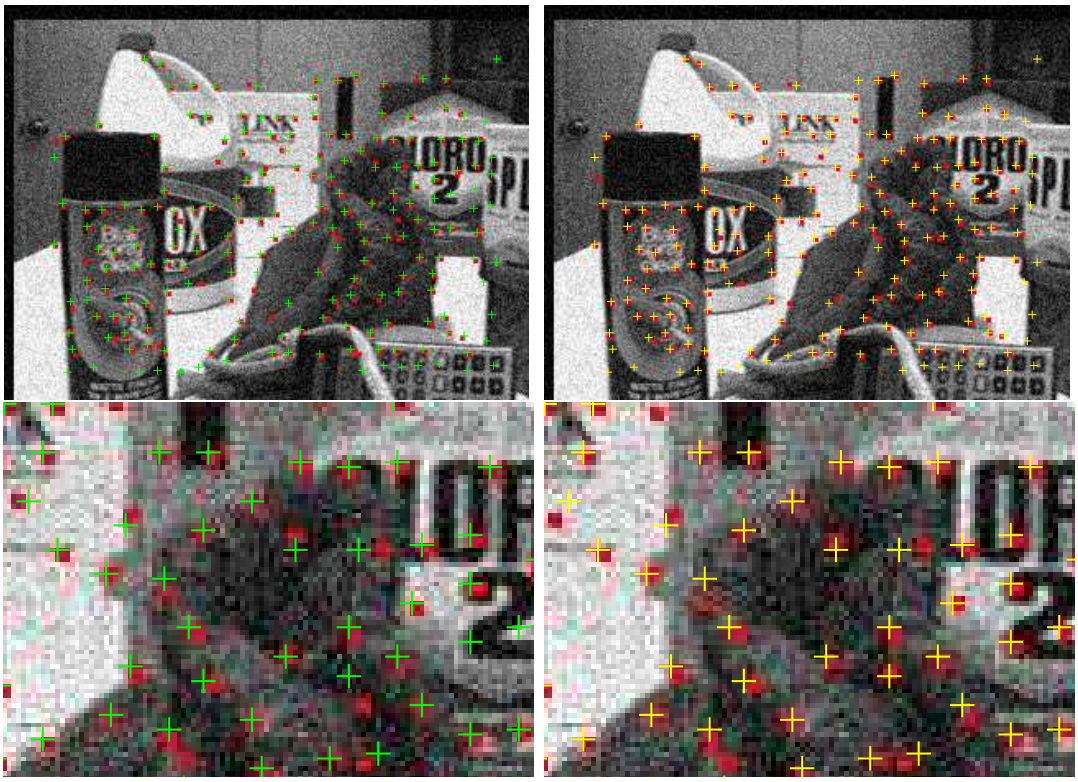

Figure 4.6: The tracking results of the original KLT (left column) and the improved KLT (right column) on a noisy sequence at the ending frame. The green and yellow cross points provide the ground truth positions of the feature points. The up-left corner of the small red block should overlay on the center of the cross if tracking were perfect.

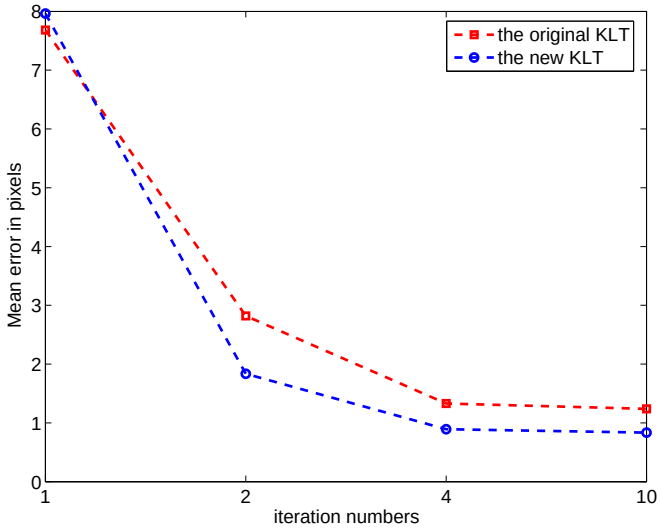

Figure 4.7: The mean error of the original KLT and the new KLT for different iteration numbers from 1 to 10.

4.3.3 Speed Comparison

The new KLT using the time-reversibility constraint is a real-time algorithm, which does not add noticeable increase in computational cost, compared to the original KLT. This can be expected from the similar linear equations solved in both the original and new KLT. We plot the mean error of both algorithms under different iteration numbers in Figure 4.7. It can be seen that the new KLT even converges a little bit faster than the original KLT.

4.3.4 Good Features to Track

We studied the condition number in the original KLT matrix $\mathbf{Z}$ and the new KLT matrix $\mathbf{U}$. The condition number here is defined as the absolute ratio of the maximal eigenvalue to the minimal eigenvalue of a matrix. When solving a linear

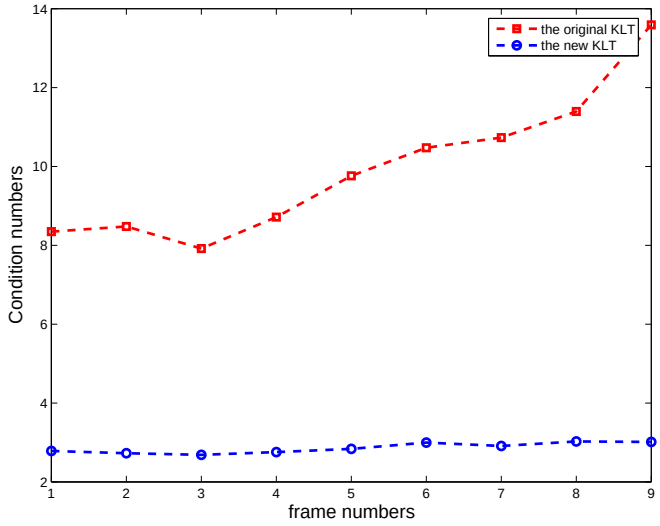

Figure 4.8: The average condition number of the matrices for 200 points at each frame during tracking.

equation like $\mathbf{M}x = \mu$, if the condition number of $\mathbf{M}$ is large, even a small error in μ may cause a large error in x. In the original Table sequence, the average condition number of the original KLT matrix is about 8.9171 while the condition number of the new KLT matrix is about 2.6621. This is based on the evaluations of 200 points across 10 frames. And from Figure 4.8, we also find that the condition number increases in the original KLT while it remains nearly constant in the new KLT based on the time-reversibility constraint. We believe that this explains the performance improvement due to the new constraint, from the view of numerical computation stability.

4.3.5 Additional Results on Large Object Tracking

To test the improvement of the new tracking strategy using the time-reversibility constraint on tracking large objects, we performed an exhaustive search based tracking on a generated very noisy sequence. The objective functions used are the same as the original KLT and the new KLT. The difference is in that in KLT, a gradient decent like search method is used while an exhaustive search method is used here. The results are shown in Figure 4.9. We can see that using the time-reversibility constraint, the block can be tracked well while it can not be tracked without such a constraint. Thus we believe that by combining the time-reversibility constraint with some large-object tracking algorithms could improve the tracking performance too.

4.4 Conclusions

In this chapter, we exploited the time-reversibility constraint in designing visual tracking algorithms, which has not been studied before. We applied this idea to the popular KLT feature tracking algorithm and developed a new KLT algorithm using the time-reversibility constraint. Extensive experimental results were presented comparing the performance between the original KLT and the new KLT. The results show that the performance of the new KLT algorithm has been significantly improved. A simple experiment on tracking large object is also given, which shows that the proposed strategy is very promising for tracking large objects. The work on improving other tracking algorithms, such as mean shift tracking, and optical

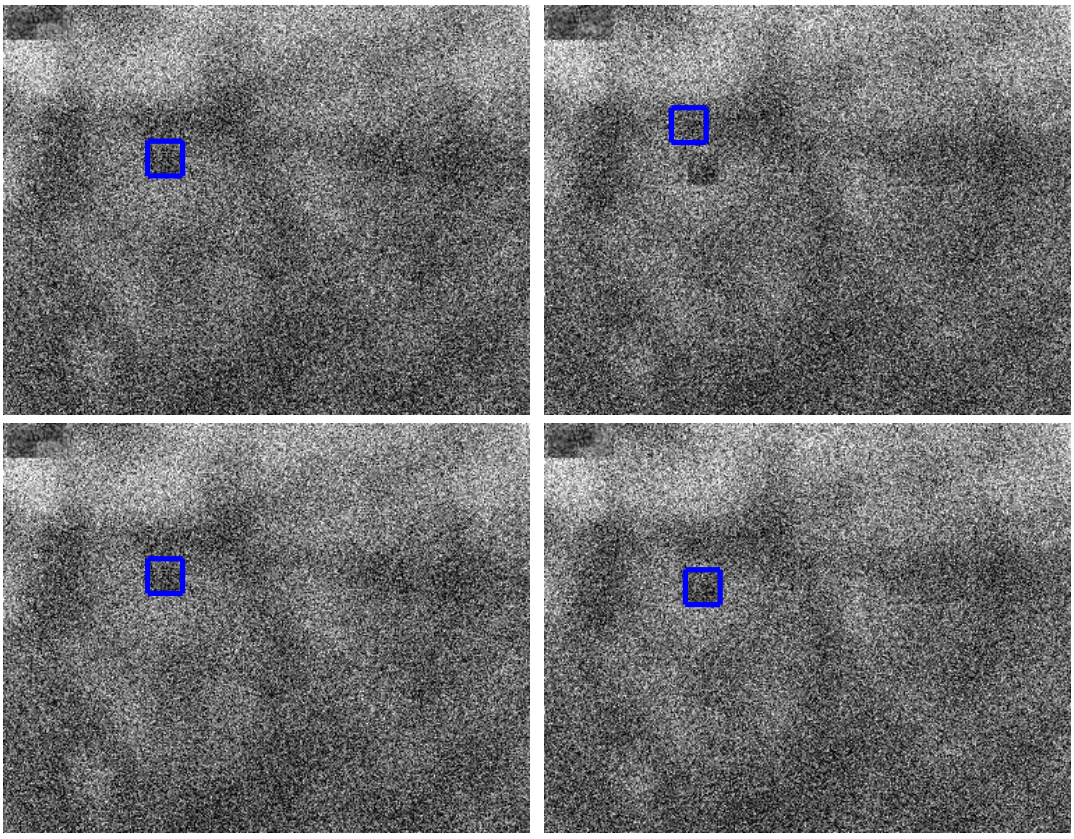

Figure 4.9: Tracking the black block by searching in a neighborhood around the object using a fixed appearance model. The top row shows the result without the time-reversibility constraint at the starting and ending frames, which fails to track the block. The bottom row shows the result using the time-reversibility constraint at the corresponding frames, which obtained good tracking of the block.

flow methods will be studied in the future.

Chapter 5

Boosted Ranking Model for Model Alignment

Model alignment has been an active research topic in computer vision for over two decades. The task of model alignment is to adjust a shape model or template to be in accordance with the shape of an object in images. Model alignment has broad interactions with many other vision tasks, such as, object detection/tracking/recognition, activity analysis, human expression recognition, etc. The performance of model alignment is very important for subsequent applications; however, the accuracy and robustness of present algorithms are still not sufficient for practical applications on large data sets.

5.1 Model Alignment Problem

Model alignment can be viewed as a special registration problem. In general, from the perspective of data to be processed, the registration problem can be divided into three categories: image to image, graphics to graphics and graphics to image. Usually the registration procedures are converted to minimize a distance metric or maximize a similarity measure which is defined to indicate how good the current registration is.

Model alignment task belongs to the third category, which tries to overlay a

deformable shape template on an image to indicate the locations of object components or features. Unlike the other two categories where the distance metric or similarity measure can be straightforwardly defined, in the model alignment problem, it is rather tricky when one has to evaluate the distance or similarity between two different types of data: images and graphical shape models. This intrinsic difficulty makes model alignment and other similar registration problems very challenging yet attractive to many researchers. Therefore, how to avoid or solve this problem actually characterizes various algorithms developed. Intuitively, there are two ways to handle this problem: either convert images to graphical features or build image instance from the shape model. The first method extracts graphical features from the image, like edge, corners, structures, etc., and measures the distance between graphical features and the shape model; however, due to the difficulty in extracting enough clean and meaningful graphical features from images, this method has not been successful in practice. The second method compares the object image and some "standard" image instance built from the training data to measure the distance in image domain. Algorithms based on this strategy have dominated the deformable model based alignment/registration area. The well-known ASM [19], AAM [5] [18] and the most recently developed BAM [62] algorithms belong to this category. In the following, we will introduce these algorithms for solving the face alignment problem, which is useful for developing recognition algorithms for non-frontal face images.

5.2 Prior Work

Cootes et al. proposed successful ASM and AAM algorithms in 1992 [19] and 2001 [18] respectively. The majority of the prior work in face alignment is based on ASM, AAM or their variations [20] [21] [22] [24] [52] [81] [96].

5.2.1 Active Shape Model and its extensions

We first look into the ASM algorithm proposed by Cootes et al [19]. In ASM, an object shape is represented by a set of landmark points. The coordinates of the 2D shape landmarks, $(x_i, y_i), i \in [1, v]$, are concatenated in some predefined order to form a vector denoted by $\mathbf{s}$, where $\mathbf{s} = (x_1, y_1, x_2, y_2, ..., x_v, y_v)^T$. Then the statistical shape model, which is also called the point distribution model (PDM), is built using principal component analysis (PCA):

$$\mathbf{s} = \mathbf{s}_0 + \Sigma_{i=1}^{n} p_i \mathbf{s}_i \tag{5.1}$$

where $\mathbf{s}_0$ is called the mean shape and $\mathbf{s}_i$ is the i^{th} shape basis, the coefficient p_i is the shape parameter computed by projecting $\mathbf{s}$ onto the ith shape basis.

Differing from non-statistical active models like Kass's Active Contour Models (ACM or 'snakes') [53] which only aim at imposing smoothness on the curves, statistical shape models learnt from a training set using PCA analysis represent the shape model in a parametric way. It gives a compact representation of allowable variations while prohibiting arbitrary deviations. The active shape model uses a local appearance model. By sampling along the profile normal to the boundary in the training set, it builds a statistical model of grey-level structure for each

landmark. When aligning the shape model to a new face image, the ASM fitting procedure alternates between searching local landmark, and the projections of the statistical shape model.

The ASM-based algorithms often suffer from local minima problem while searching for local landmarks because the 1-D profile appearance model is insufficient to distinguish feature points from their neighbors. Many algorithms have been presented to improve the performance, including more sophisticated representative and discriminative local appearance models. some examples are Gabor wavelet presented by Jiao et al. [47], nonlinear kNN-classifier by Ginneken, et al. [91], and boosted local appearance classifiers like Harr wavelet [30], FisherBoost [90], and real-adaboost [107].

Except for the convergence problem caused by local appearance models, inadequate interaction between search steps over shape parameters and local landmarks also makes the two alternative fitting procedures less efficient and prone to converge to a wrong solution in some circumstances.

5.2.2 Active Appearance Model and its extensions

In the original AAM algorithm proposed by Cootes et al. [18], both the statistic shape and texture models are built using a manually labeled training set and the principal component analysis.

At the same time, they also try to learn how to adjust the model parameters for fitting an image after both the texture and shape models have been built. The shape model is the same as in the ASM algorithm. In a similar way, the statistical

texture model can be constructed from the shape normalized training images:

$$\mathbf{g} = \mathbf{g}_0 + \Sigma_{i=1}^{m} r_i \mathbf{g}_i \tag{5.2}$$

The purpose of AAM is to find the optimal shape and texture parameters, $\mathbf{p}$ and $\mathbf{r}$, by minimizing the residual error between the synthesized model instance and the given image I:

$$E = \|\mathbf{g}_0 + \Sigma_{i=1}^{m} r_i \mathbf{g}_i - I(W(\mathbf{x}; \mathbf{p}))\|^2 \tag{5.3}$$

where $\mathbf{x}$ is the pixel coordinates inside the image patch defined by the current shape model parameters $\mathbf{p}$, and $W(\cdot)$ warps this image patch to the normalized shape.

This is a difficult high-dimensional optimization problem as there are usually dozens of shape and texture parameters to be optimized. The direct gradient descent algorithms could be very slow and often get stuck in local minima. Observing that the residual image contains useful information to adjust the model parameters towards a better fit, the authors used linear regression to establish the relationship between the residual image and the model parameters:

$$\begin{aligned} \Delta\mathbf{p} &= \mathbf{A}_s \Delta\mathbf{I}; \\ \Delta\mathbf{r} &= \mathbf{A}_g \Delta\mathbf{I}; \end{aligned} \tag{5.4}$$

where

$$\Delta\mathbf{I} = \mathbf{g}_0 + \Sigma_{i=1}^{m} r_i^0 \mathbf{g}_i - I(W(\mathbf{x}; \mathbf{p}^0)) \tag{5.5}$$

and

$$\begin{aligned} \Delta\mathbf{p} &= \mathbf{p}^{opt} - \mathbf{p}^0 \\ \Delta\mathbf{r} &= \mathbf{r}^{opt} - \mathbf{r}^0 \end{aligned} \tag{5.6}$$

$\mathbf{r}^0, \mathbf{p}^0$ are initial model parameters.

Notice that $\mathbf{A}_s, \mathbf{A}_g$ are constants. However this constant linear relationship is incorrect in general as pointed out by other researchers [5]. The authors have also noted that this linear relationship only exists in a small neighborhood around the true solutions [18].

Besides the statistical shape model, AAM algorithms also construct a generative global statistical texture model to generate a synthetic image that can be directly compared to the given face image. AAM learns a locally linear relationship between model parameter displacements and the residual errors between the synthetic image and the image from the training data. The associated search algorithm then uses this linear model to predict changes to the current parameters to get a better fit. AAM algorithms are expected to achieve more efficient fitting and better convergence than ASM, as more information is used in training and fitting processes.

The AAM-based approaches perform well on small data sets. However, learning the linear prediction models based on texture difference vectors demands significant resources when the training data set gets larger. The demand on memory makes AAM training very difficult even with a moderate number of images.

Furthermore, the constant locally linear relationship between the error image

and the model parameters is in general incorrect, which affects both the convergence rate and fitting accuracy. Hence, Baker and Matthews presented the Inverse Compositional(IC) and Simultaneously Inverse Compositional(SIC) algorithms for AAM fitting to avoid inaccurate modeling of this relationship [65]. The authors use this efficient analytical gradient descent algorithm to minimize the squared error between the given image and the synthetic image. However, like other gradient descent algorithms, IC/SIC needs extra measures to avoid the local minima problem. The authors also point out in [37] that if AAM fails to fit, the main reason is due to the difficulty of the fitting process rather than the inability of the AAM to model the image.

It has been reported that the AAM alignment performance degrades quickly when processing an image that is not in the training set [62]. Also, according to Gross, et al. [37], person specific AAMs are easier to build than generic AAMs. The global eigenspace-based appearance model is asserted to be the main reason to cause these problems.

Most of the AAM body of work is based on the generative model of [5], which greatly improves the efficiency of the AAM-based face alignment. Some AAM variations include discriminative fitting methods [24] [81]. Other representative works include [60] [111]. There are also many other improvements to AAM, for handling occlusion, illumination change and other problems in alignment [38] [36] [50] [51], but the generalization and convergence problems are seldom addressed.

5.2.3 Bossted Appearance Model

Lately, motivated by some learning based-tracking and detection algorithms [4] [41] [97], Liu presented [62] a boosted appearance model-based face alignment approach to alleviate the generalization problem of conventional AAM algorithms. Instead of training a generative eigenspace-based appearance model, BAM tries to train a two-class classifier which is able to distinguish correct and incorrect alignments using the GentleBoost algorithm and Haar wavelet features. Fitting a shape model into an image is an optimization process of maximizing the classification score.

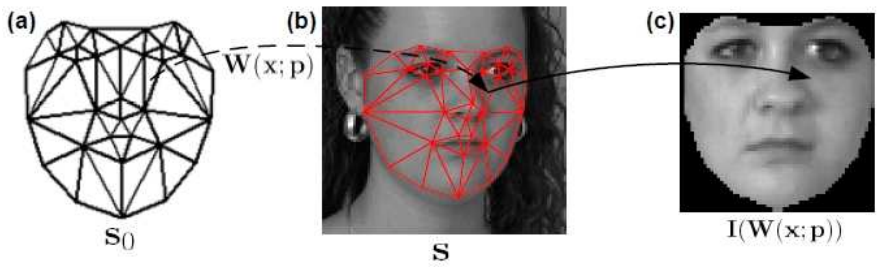

Figure 5.1: Shape Model and Warping Function. (a) Representation of the mean shape. (b) The face image with a superimposed shape. (c) The face image warped to the mean shape domain.

The appearance model of BAM is simply a collection of m features $\{\varphi_i\}_{i=1,\ldots,m}$, computed over the shape-normalized face image $I(W(x;p))$ as shown in Figure 5.1. The popular rectangular Haar-like features [74] [95] are chosen here, mainly because of their computational efficiency, which exploits the integral image representation [95], and because of their success in face-related applications [62] [95].

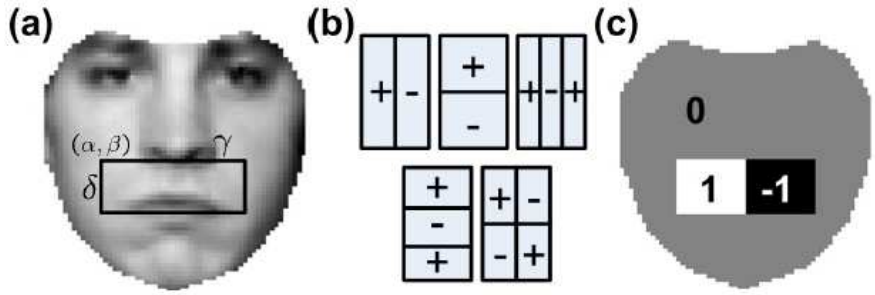

Figure 5.2: Appearance Features. (a) Warped face image with feature parametrization. (b) Representation of the five feature types used by the appearance model. (c) Notional template A.

As shown in Figure 5.2, a rectangular feature can be parameterized as follows

$$\varphi \doteq A^T I(W(x;p)) \tag{5.7}$$

which is intended as the inner product between the vectorized version of an image template A, and the vectorized version of the warped face image. The inner product between the template and the warped image is equivalent to computing the rectangular feature using the integral image. The image template A can in turn be parameterized by $(\alpha, \beta, \gamma, \delta, \tau)$, as shown in Figure 5.2(a), where (α, β) is the top-left corner, γ and δ are the width and height, and τ is the feature type. Figure 5.2(b) shows the feature types used in this model.

By using discriminative texture features, the BAM algorithms are more robust to texture variations than AAM-based algorithms. And the boosting method is capable of learning from a large dataset and generalizing well to new data.

While the idea of using trained classifiers to find the best matches is also used in some ASM-based algorithms to search for the best local landmarks [30] [90] [107], BAM differs from these algorithms in its fitting procedure. Owing

to the continuous output of the GentleBoost classifier, BAM uses an analytical gradient accent algorithm, which is very similar to IC/SIC methods, except BAM maximizes the classification score instead of the residual error.

In essence, BAM only learns how to distinguish good and bad alignments. Without learning the relationship between the texture change and the shape parameters change, the BAM fitting procedure is also prone to get stuck in local maxima. Another potential problem of BAM is that the classification performance may drop as the size of the dataset increases, which will result in poor alignment accuracy.

5.3 Boosted Ranking Model(BRM)

From the analysis given above, we observe that most alignment algorithms focus on how to build more sophisticated texture/shape models and the optimization criterion without considering the fitting procedure at the same time. However, the fitting procedure usually involves the optimization process in very high dimensional image and shape spaces. Insufficient or incorrect information(like the constant linear relationship) of the fitting process will lead to inefficient or bad alignment.

In this section, we propose a novel and efficient strategy to learn how to improve the score function to simplify the fitting procedure. On the one hand, our method is different from the rigid constant linear models used to establish the relationship between an image and its synthesized model instance, which has been

proved to be incorrect in general and an obstacle to improving the alignment performance; on the other hand, unlike IC/SIC and BAM algorithms which do not exploit global knowledge about the optimization space, our method provides sufficient guidance in modifying the shape parameters all the way toward optimal solutions. This is achieved by learning through a continues boosted ranking algorithm on the deformed face manifold built from a training set. The proposed method enables the optimization process carried on a local-extrema-free surface from a random start location to the optimal solution. This greatly improves the fitting performance and accuracy.

5.3.1 The Intuition Behind BRM

In fact, from the above AAM formulation, it is easy to see that given the initial model parameters, the synthesized model instance $\mathbf{g}_0 + \Sigma_{i=1}^{m} r_i^0 \mathbf{g}_i$ will be fixed for different images, then according to 5.4, the optimal solution is actually determined by $I(W(\mathbf{x}; \mathbf{p}^0))$. This means that from a partially overlayed image patch, one can infer the whole image information, which is obviously infeasible.

Matthews and Baker [5] proposed IC/SIC algorithms to efficiently minimize the residual error directly. They use gradient information to guide the fitting direction; the Gauss-Newton gradient decent algorithms are then alternatively applied to modify the shape and texture model parameters; however, without a global view of the optimization space, using only local information leads to converge to local minima easily, especially in such high dimensional image/shape joint spaces.

How to correctly utilize the information of an object which is partially contained in the initialization and intermediate image patches? To record all the (initialization, optimal solution) pairs is impossible and inefficient. Even if we can do that, there is still ambiguity when fitting a new image because of the one to many mapping between the initial image patch and optimal solutions. Relying only on the local image gradient is unreliable in some circumstances and insufficient in general.

In this chapter, we propose a novel strategy to learn how to utilize the image information to guide the fitting process. To illustrate this idea, let us first take a look at the data manifold consisting of image patches defined by the correct and incorrect shape models. Given an image I and the hand labeled shape model $\mathbf{s}^0$, by perturbing $\mathbf{s}^0$ to get distorted shape models denoted by $\mathbf{s}^j$, we can get as many image patches as possible which are defined by the correct and perturbed shape models. Now we can warp all these image patches into the normalized shape and denote the normalized patches as $I(W(\mathbf{x}, \mathbf{s}^j))$. These patches lay on a manifold in a high-dimensional image space. Using classic nonlinear dimension reduction methods like Isomap [88] or LLE [79], etc, we can unfold this manifold and compute their geodesic distances to the patch warped from $\mathbf{s}^0$. This geodesic distance is a global indication of how far it is to a specific reference point. In our case, the dimension of this manifold is determined by the number of shape eigen-bases. For example, if we use $\mathbf{s}^j = \mathbf{s}_0 + \Sigma_{i=1}^n p_i^j \mathbf{s}_i$, then the vector $(p_1^j, p_2^j, ..., p_n^j)$ identifies each patch in this manifold and hence $\|\mathbf{p}^0 - \mathbf{p}^j\|^2$ is a good approximation of geodesic

distance. If we treat $\mathbf{p}^j$ as variables to minimize this distance using gradient descent like methods, the optimal direction can always be given by $\mathbf{p}^0 - \mathbf{p}^j$ and this process contains no local minima. Therefore, instead of minimizing the residual error of texture images in (5.3), we aim at directly minimizing the shape distance:

$$D = \|\mathbf{p}^0 - \mathbf{p}\|^2 \tag{5.8}$$

However, when fitting a new image without knowing $\mathbf{p}^0$, we can not minimize (5.8) directly. Also, like in minimizing (5.3), we have analyzed that prediction of $\mathbf{p}^0$ using imperfect $\mathbf{p}$, if it is possible, is unreliable and likely to converge to local minima.

5.3.2 Learn a Score Function without Local Extrema

Noticing that the useful information about how to get closer to the true solution is actually contained in the image patch defined by the shape model, we can define a function F on $I(W(\mathbf{x}, \mathbf{p}))$ to extract the useful image information and have the following property:

$$\|F(I(W(\mathbf{x}, \mathbf{p}^0)) - F(I(W(\mathbf{x}, \mathbf{p}))\|^2 \propto \|\mathbf{p}^0 - \mathbf{p}\|^2 \tag{5.9}$$

then we can minimize the following distance instead of (5.8):

$$D_F = \|F(I(W(\mathbf{x}, \mathbf{p}^0)) - F(I(W(\mathbf{x}, \mathbf{p}))\|^2 \tag{5.10}$$

In the original AAM and IC/SIC algorithms, they actually use $F(I) = I$. Here we propose to use machine learning methods to learn how to build the function F.

Different from boosted regression methods used in [21] and BAM which is a two class regression method, usually requiring $F(I) = \theta(\mathbf{p})$, where $\theta(\cdot)$ has countable discrete scalar outputs, we emphasize that it is not proper to force a rigid equality sign, especially when F is a continues function which is desired in many cases, based on the following considerations:

- First, for different images, it is unreasonable to require $F(I) = F(G)$ when $\mathbf{p}_I = \mathbf{p}_G$ or $\theta(\mathbf{p}_I) = \theta(\mathbf{p}_G)$.
- Second, even for the same images, it is still unreasonable to require:

$$F(I(W(\mathbf{x}, \mathbf{p}_1))) = F(I(W(\mathbf{x}, \mathbf{p}_2))) \quad (5.11)$$
$$when \quad \theta(\mathbf{p}_1) = \theta(\mathbf{p}_2)$$

In our method, as shown in Figure 5.3, we only expect the function F to preserve the ranking order along the steepest descending direction between any given start point $\mathbf{p}$ and $\mathbf{p}^0$, namely:

$$F(I(W(\mathbf{x}, \alpha\Delta\mathbf{p}))) > or < F(I(W(\mathbf{x}, \beta\Delta\mathbf{p}))) \quad (5.12)$$
$$where \Delta\mathbf{p} = \mathbf{p} - \mathbf{p}^0, and, 0 < \alpha < \beta < 1$$

By doing this, on the one hand, if F can perfectly satisfy the above requirement, then there will be no local extrema during fitting process, otherwise it will contradict (5.12); on the other hand, comparing to the regression requirement, (5.12) is more reasonable and flexible for learning F. To learn such a function, we need a rank learning algorithm instead of regression or classification. We also claim here

that this ranking idea fits the essence of fitting process most. More details of the proposed training and fitting algorithms are given in the next section.

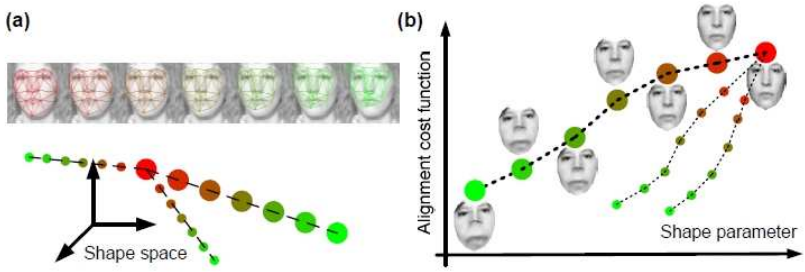

Figure 5.3: Performance evaluation on a PETS sequence including ground truth. (Top three rows) Tracking results at frame numbers 1, 30, 60, 90, 120 and 160. (bottom three rows) Evaluation results using proposed statistics and its fast approximations and the ground truth. Tracking performance remains fairly constant as shown by both the ground truth and the proposed evaluation strategy.

5.4 The Detailed BRM Algorithm

From the above analysis, we have concluded that we want to learn a function F that can preserve the ranking order along the vector $\mathbf{p} - \mathbf{p}^0$ for any given start point $\mathbf{p}$. In practice, we should keep in mind that if $\|\mathbf{p} - \mathbf{p}^0\|$ is too large, say the overlapping between $I(\mathbf{p})$ and $I(\mathbf{p}^0)$ is too small or even no overlapping, there is no way that we can learn this ranking information to help fitting since there is insufficient or no information about the fitting direction contained in the initialization. Hence, the perturbation of the shape parameters should remain in a reasonable range.

5.4.1 Balanced Positive and Negative Sample Sets

Given a training set with hand labeled shape models $\{(I^1, \mathbf{s}^1), (I^2, \mathbf{s}^2), ..., (I^N, \mathbf{s}^N)\}$, the statistical shape model$\mathbf{s} = \mathbf{s}_0 + \Sigma_{i=1}^{n} p_i \mathbf{s}_i$ can be trained in the same way as in ASM/AAM. Now we start to learn function F as required by (5.12). However, it is infeasible to exploit all the possible initialization $\mathbf{p}$ and α, β. We solve this problem by sampling on the data manifold and learning a continues function F which behaves like (5.12) in the neighborhood around each sample; therefore, as long as the sampling is dense enough, we can still get an approximate but good enough solution as required.

Considering a face dataset consisting N labeled face images, by perturbing the shape parameters $\mathbf{p}$ in different directions, we can get a series of perturbed shape models for each training sample, then we sample along $\mathbf{p} - \mathbf{p}^0$ to get a vector of discrete samples in ranking order. Suppose for each training samples, we randomly perturb U times to get K starting points, and for each starting point we take V samples along the steepest descent direction. U, V can be different for different images. The perturbed shape parameters can be denoted by:

$$\{p_i + v\Delta p_u\}_{i=1,...,N;u=1,...,U;v=1,...,V} \tag{5.13}$$

Using pertubed shape parameters, we can generate the perturbed training images $\{I_i^{(u,v)}\}$, such that

$$I_i^{(u,v)} \doteq I_i(W(x; p_i + v\Delta p_u)) \tag{5.14}$$

Figure 5.4 shows some examples of the generated training images. Since our

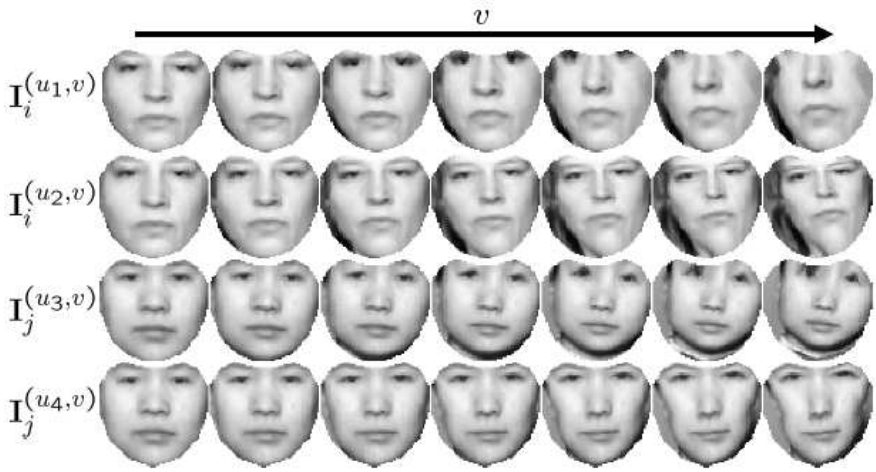

Figure 5.4: Training Samples. Top two rows and bottom two rows are training samples generated from the same face image (I_i and I_j respectively). Samples from each row have been generated from the original shape-normalized face image on the left, and with shape-perturbation parameters u1, u2, u3, u4. From left to right the parameter v increases, and the shape parameter is varying according to Equation (5.13)

goal is to learn a function that can maintain the order of the generated traning samples, the training task can be handled by ranking algorithms. Before learning the rank function, we need to break the long ordering sequence into pairwise data, in which a positive sample can be defined as the consecutively ordered pair $x_{+uv} = (I_i^{(u,v+1)}, I_i^{(u,v)})$, and will be labeled with $y_{+uv} = +1$ in our training algorithm. Similarly, a negative training sample is defined as the inverse ordered pair $x_{-uv} = (I_i^{(u,v)}, I_i^{(u,v+1)})$, and will be labeled with $y_{-uv} = -1$. Therefore, the training sets of positive and negative samples in our BRM algorithm are given by:

$$\begin{aligned} \text{Positive Set} &= \{x_{+iuv}\}_{i=1,\ldots,N;u=1,\ldots,U;v=1,\ldots,V-1} \\ \text{Negative Set} &= \{x_{-iuv}\}_{i=1,\ldots,N;u=1,\ldots,U;v=1,\ldots,V-1} \end{aligned} \tag{5.15}$$

It is noticable that the positive and negative sets have the same cardinality,

which means the they are totally balanced. Therefore, our training algorithm will not suffer from unbalanced training sets as in the BAM algorithm. To learn a function $F(I)$ which can preserve the ranking order for each training sample, rank learning methods are needed.

5.4.2 Rank Learning Algorithm:

Rank learning, together with ordinal regression and preference learning, refer to inductive learning methods from ordered data in the area of supervised learning. Tailored to solve problems between classification and metric regression, they have many applications in information retrieval, econometric models, classical statistics and other social research areas. In contrast to using statistical models, machine learning-based rank learning/ordinal regression methods have drawn a lot of attentions in recent years. Herbrich et al. [40] proposed a support vector learning method for ordinal regression which models the ranks as intervals on a real line and optimizes the loss function based on the true ranks and features. This method was followed and extended by many other researchers [56] [77] [102] [1]. However, the absolute numerical ranks used in ordinal regression formulation restricts its applications where the relative ranking preferences are available between pairs of training data. Later, by converting paired data ranking into binary classification problems, Joachims proposed a ranking SVM method which has been successfully used to improve search engines [48]; Freund et al. designed the RankBoost algorithm in the setting of collaborative filtering [31]. These algorithms and their derivatives have found abundant applications mostly in the area of information

retrieval.

In the Computer Vision community, [3] [32] have utilized ranking algorithms for shape and image retrieval. Some researchers introduced the RankBoost [66] algorithm into shape localization and detection. Yan et al. [103] proposed the Constrained RankBoost approach to model the likelihood of local features associated with the landmarks of an object in their RPBF shape localization framework. In RPBF, the shape inferring process is conducted through maximizing the posterior probability composed of likelihood of local features and prior probability of shape; however, the assumption that the local features around different key points are independent is obviously inaccurate, which reduces the chance that the true shape can be found from solving the posterior probability even using sophisticated optimization techniques. Zheng et al. [109] also used RankBoost in their example based shape detection work. By creating a sorted image list using predefined warping templates for each image in the training set, the authors use RankBoost to learn a relative similarity function. When detecting the shape in a new image, an exhaustive testing with all the warping templates applied to this image will be conducted and the top k candidates with the largest responses will be selected to build the detected shape using the kernel-weighted average. This method lacks sound shape reconstruction theory from a set of un-orthogonal base shapes.

In order to compare with BAM method, we developed a GentleBoosted ranking algorithm to learn the rank function in our experiments. We also tested the RankBoost algorithm and observed that the performances of both algorithms are

To learn a strong rank function $F(I)$ from paired training samples

1. Build training samples according to 5.15: $(I^i_{k,l}, I^i_{k,l+1})$, $i \in [1, N], k \in [i, K], l \in [1, L]$. Initialize weights $w_m = 1/M$, $M = N * K * L$.

2. for $t = 1, 2, ..., T$ **do**

(a) Find the optimal weak rank function $h_t(I)$ to minimize the weighted least-squares mis-ranking error:

$$h_t(I) = \underset{h \in H}{\operatorname{argmin}} \sum_{m=1}^{M} w_m [1 - (h(I_m) - h(I_{m+1}))]^2 \tag{5.16}$$

(b) Update $F(I) = F(I) + h_t(I)$.

(c) Update the weights by $w_m = w_m e^{-h_t(I_m)}$ and normalize the weights such that $\sum_{m=1}^{M} w_m = 1$

3. Output the rank function $F(I) = \sum_{t=1}^{T} h_t(I)$.

Table 5.1: Outline of the proposed training algorithm.

similar.

We generated the positive and negative samples as described above. In our settings, each positive sample pair has its corresponding negative sample pair and their contributions to the ranking cost function are exactly the same; therefore, in our training algorithm, we actually only consider the positive sample pairs as the input. As a side benefit, our algorithm will not suffer from the unbalanced positive/negative samples as classification based methods. The training algorithm is summarized as Algorithm 5.1.

Figure 5.5 shows the selected Harr features by our training algorithm. We

see that most features are well aligned with the boundaries of the natural facial features, hence containing abundant image information.

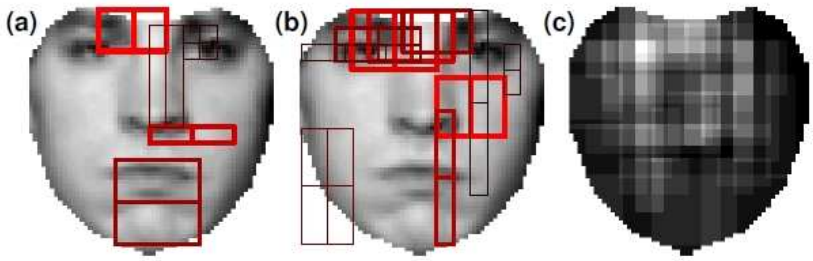

Figure 5.5: Selected Appearance Features. (a) Representation of the top 5 Haar features selected by Algorithm 1. (b) Representation of the top 6-15 Haar features. (c) Spatial density map of the top 50 Haar features. Most features are well aligned with the boundaries of natural facial features.

5.4.3 Fitting On a Ranked Surface

The weak function used in our experiments is the monotonic arctan function, which can be substituted by other continuous sigmod functions. Using the Harr wavelet features, the analytical expression of the weak function is constructed as:

$$h_t(\mathbf{p}) = \frac{2}{\pi} \arctan(g_t \mathbf{A}_t I(\mathbf{W}(\mathbf{x}; \mathbf{p}) - v_t) \tag{5.17}$$

Therefore, the learned rank function can be written as:

$$F(\mathbf{p}) = \sum_{t=1}^{T} \frac{2}{\pi} \arctan(g_t \mathbf{A}_t I(\mathbf{W}(\mathbf{x}; \mathbf{p}) - v_t) \tag{5.18}$$

As the objective of our training algorithm, this function is designed to preserve the ranking order along the steepest descending direction from any given start point $\mathbf{p}$ to the true solution $\mathbf{p}^0$. This guarantees that the optimal solution can be

found by even using the basic gradient ascent method. The derivative of function F with respect to $\mathbf{p}$ is:

$$\frac{dF}{d\mathbf{p}} = \frac{2}{\pi}\sum_{t=1}^{T}\frac{g_m[\nabla I\frac{\partial \mathbf{W}}{\partial \mathbf{p}}]^T\mathbf{A}_t}{1+[g_t\mathbf{A}_t I(\mathbf{W}(\mathbf{x};\mathbf{p})-v_t]^2} \tag{5.19}$$

Like IC/SIC and BAM algorithms, by triangulating the landmark sets and defining a piecewise affine warping function, we can pre-compute the *Jacobian* $\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$. More details can be found from IC/SIC and BAM algorithms [5] [62]. Since the rank function has exactly the same form as BAM's classification function, our fitting procedure is straight forward.

Figure 5.6 shows a few face images with green initial shape masks are well aligned by the red shape masks resulting from our fitting algorithm.

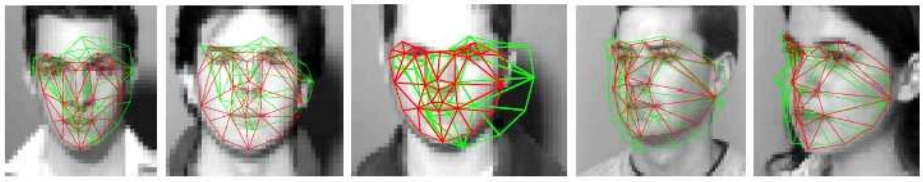

Figure 5.6: Alignment Examples. Face images with superimposed initial face model (green), and aligned face model (red).

5.4.4 Comparison with Other Models

The BRM belongs to the same class of models as the BAM model [62]. Therefore, compared to AAM [18] [65], it enjoys the same benefits, such as robustness to partial occlusions, improved alignment speed, and ability to incorporate knowledge about both good and bad alignments, while being substantially more parsimonious. In comparison with BAM, our model is the outcome of a very different

problem formulation. More precisely, the BAM is produced by learning a strong classifier that is able to distinguish between correct and incorrect alignment, and the results in [62] empirically show that face alignment can be achieved via gradient ascent on the corresponding classifier score function. However, there is no guarantee that the gradient will be aimed at improving the alignment (because the strong classifier can distinguish only between right or wrong fittings). On the other hand, we consider this fundamental issue at the outset, and propose to solve the alignment problem by looking for a score function that is concave, hence optimizable via gradient ascent. This leads to learning a strong classifier that is able to say whether by switching from one alignment to another one we are actually making an improvement, as opposed to saying whether or not the alignment is correct. Another advantage (as opposed to BAM), is also the fact that positive and negative training sets naturally have the same cardinality, which makes the training problem balanced. These advantages lead to a superior alignment performance of the BRM over the BAM, as we will show in the experiment section.

5.5 Experimental Results

5.5.1 Face Dataset

We start by describing the dataset used for training and testing our approach. It is composed of a total of 964 images constituted from the aggregation of three publicly available datasets: ND1 [12] (534 images of 200 subjects appearing in frontal

Figure 5.7: Face Dataset Samples. ND1 database [12] (left), FERET database [75] (center), and BioID database [86] (right).

view), FERET [75] (200 images of 200 subjects appearing in different pose), and BioID [86] (230 images of 23 subjects appearing under different background and lighting conditions). Figure 5.7 shows some typical face images from the datasets. Each image has 33 manually labeled landmarks. To speed up the training process, we down-sample each image so that the face width is roughly 40 pixels. We divide the 964 images in three parts, namely Set 1, Set 2, and Set 3. Set 1 contains the 200 images from FERET, and 200 images from ND1 (one image per subject). Set 2 contains the remaining 334 images from ND1. Set 3 is the BioID dataset. Set 1 is used as training dataset. All the three sets are used in the alignment tests. In particular, Set 2 allows for testing the performance over unseen data of seen subjects (because different images of them have been used for training), whereas Set 3 allows for testing the performance over unseen data of unseen subjects (never used for training). Note that Set 3 is particularly challenging because the subjects are captured under different cluttered background, and illumination.

5.5.2 Training

Throughout the section we compare three models: the proposed BRM, BAM, and an adaptation of Rank- Boost [31] that uses the same weak rankings, and training pairs of the BRM. We do not compare our model against AAM-based methods [5] [18], as it has been shown in [62] that the BAM outperforms them. We train the three models with Set 1, which originates the training samples $\{I_i^(u, v)\}$, where $i = 1, \cdots, 400, u = 1, \cdots, 10$ and $v = 0, \cdots, 6$, corresponding to 24000 positive (and also negative) training pairs. In contrast, the BAM uses 400 positive and 4000 negative samples, since each image generates 10 negative samples. The resulting appearance models are such that the BRM and RankBoost have 50 weak rankings, whereas the BAM has 50 weak classifiers. The shape model has 33 shape bases and it is the same for all the models.

5.5.3 Convergence Properties

Figure 5.8 plots the false alarm rate (FAR) of the strong feedback functions of both BRM and RankBoost, as a function of the number of weak rankings, when the miss-detection rate on the training set is set to 0%. This shows that the BRM converges faster than RankBoost. In particular, for 50 weak rankings the FARs of BRM and RankBoost are 1.44%, and 6.58%, respectively.

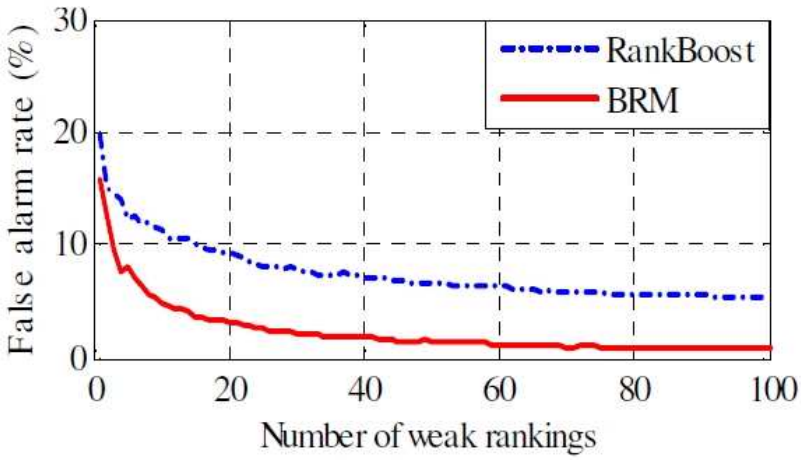

Figure 5.8: Feedback Function Performance. False alarm rate of the strong feedback function when the miss-detection rate on the training set is set to 0%.

5.5.4 Score Function Concavity

Figure 5.9(a) presents the learned score (ranking) function F for 3 images, perturbed along 10 different shape-perturbation parameters. Figure 5.9(b) presents the score function for 100 images of Set 1, each of which is perturbed along one shape-perturbation parameter. Both cases highlight the concavity properties of F.

Another way to show the score function is by using grayscale values, as in Figure 5.10, where each column represents F computed for one image, and each image has been produced by varying the intensity of only two shape bases. The range of perturbation is ± 1.6 times the eigenvalue of the corresponding bases. For both seen data in Figure 5.10(a), and unseen data in Figure 5.10(b), F shows concave properties, as required by construction, with the brightest pixel in the center, and intensity fading towards the borders.

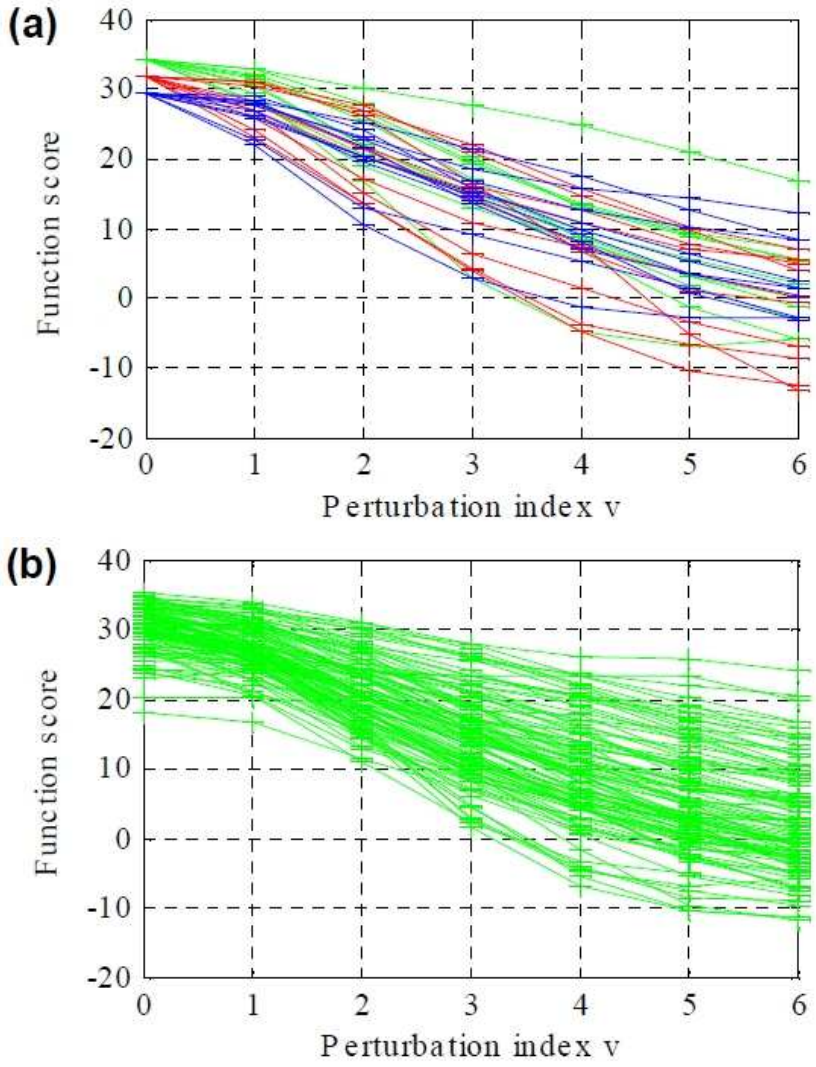

Figure 5.9: Alignment Score Function Profile. (a) Score functions of 3 images, corresponding to 10 shape-perturbation parameters. (b) Score functions of 100 training images, each of which corresponding to one shape-perturbation parameter.

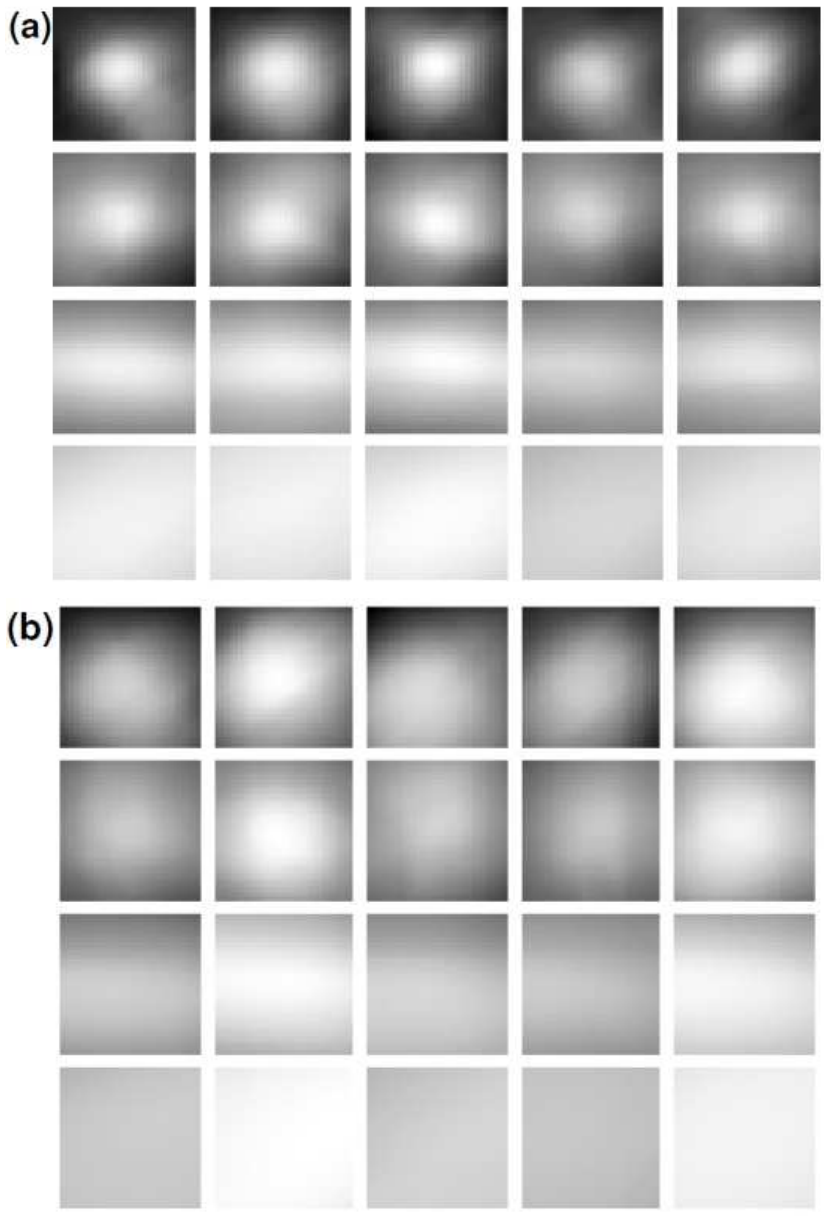

Figure 5.10: Alignment Score Function Surface. Score function F of 5 images randomly selected from Set 1 (a), and 5 images from Set 2 (b), one per column. Each image is produced by varying the shape parameter corresponding to two shape bases at a time. From the top to the bottom rows we vary: (p1; p2), (p3; p4), (p5; p6), and (p7; p8). F is concave in both seen and unseen data, and this ensures high frequency of convergence of the alignment.

5.5.5 Ranking Performance

Using the same methodology for building the training sets of pairs, we build two testing sets of pairs, one from Set 1, and one from Set 3, and test the ranking performances of BRM, BAM, and RankBoost. The correct ranking rates are reported in Figure 5.11(a), which shows the superiority of the BRM versus the BAM, especially for Set 3, highlighting the stronger generalization capabilities of the BRM to unseen data. Also, the BRM performs slightly better than RankBoost on both sets, and therefore it is expected to achieve better alignment performance as well. Figure 5.11(b) shows that BRM outperforms the BAMalso in a much harder scenario, where testing pairs are built from Set 3, but with half, and one quarter of the perturbation used to produce Figure 5.11(a). The reader may notice the slight drop in ranking performance of both methods as the perturbation becomes smaller, because it makes the ranking task more difficult.

5.5.6 Alignment Performance

In order to evaluate the alignment quality of the modeling framework, we randomly perturb the ground truth landmarks of a face image, and use them as initial conditions to align the model. The procedure is repeated multiple times on each image of the testing set in order to perform a statistical evaluation of the result. The initial positions of the landmarks are generated by perturbing the components $\{p_i\}$ of the shape parameter with independent Gaussian noise with variances that are multiples of the eigenvalues of the corresponding shape

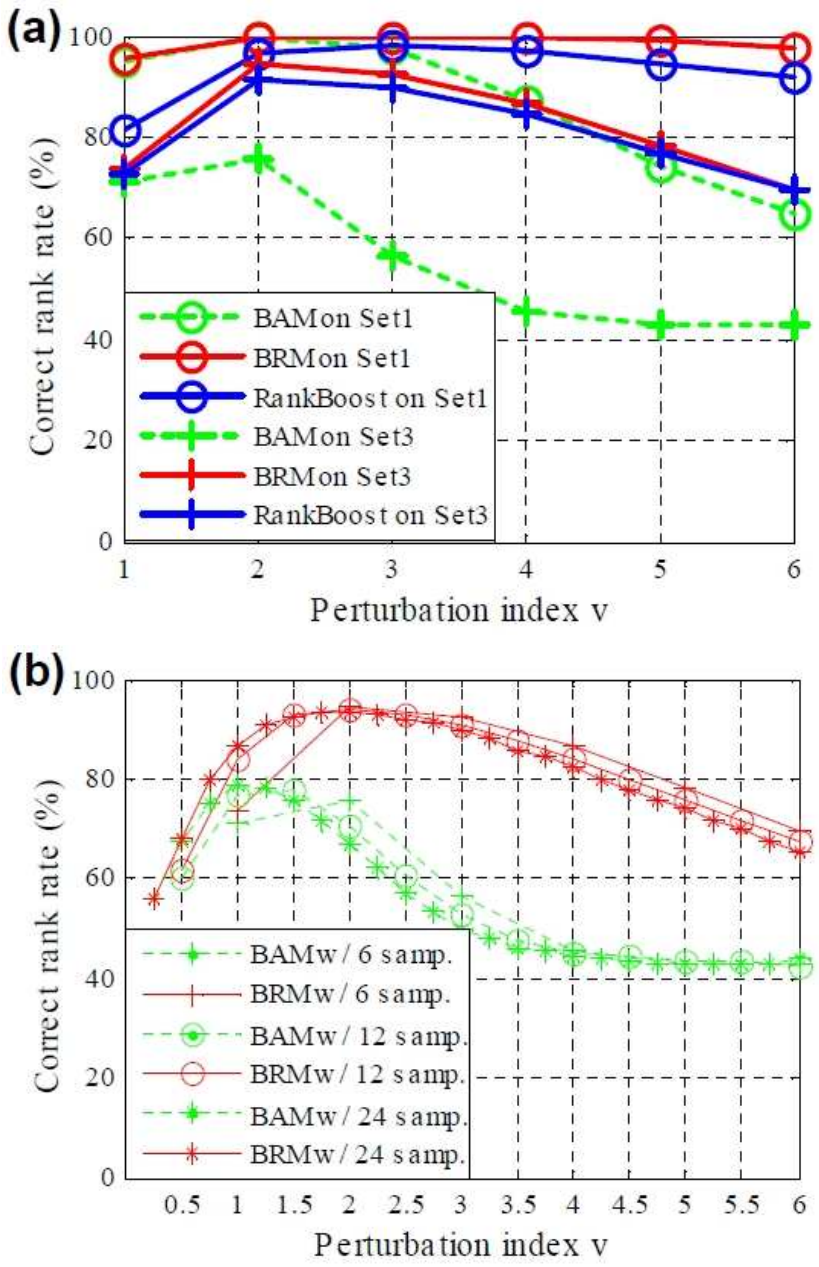

Figure 5.11: Ranking Performance. (a) Correct ranking rates of the BRM, BAM, and RankBoost on test pairs from Set 1 and Set 3. (b) Correct ranking rates of the BRM and BAM on test pairs sampled from Set 3, but with half (12 samples) and one quarter (24 samples) of the perturbation used in (a).

bases. An alignment is said to have converged if the Root Mean Square Error (RMSE) between the aligned landmarks and the ground truth is less than one pixel. Finally, we assess the alignment robustness and accuracy by computing: (a) the Average Frequency of Convergence (AFC), given by the number of trials where the alignment converges divided by the total number of trials; and (b) the histogram of the RMSE (HRMSE) of the converged trials, which measures how close the aligned landmarks are to the ground truth.

We test BRM and BAM under the same conditions. For example, both algorithms are initialized with the same set of randomly perturbed landmarks. Both algorithms have the same constant v in equation (5.13), and also the same termination condition. That is, if the number of iterations is larger than 55 or the RMSE between consecutive iterations is less than 0.025 pixels. Figures 5.12(a)(c)(e) plot the AFC of the BRM and BAM against the amount of the initial landmarks perturbation, computed over Set 1, Set 2, and Set 3, respectively. In particular, for each perturbation value, each image of each set is randomly perturbed 5, 6, or 9 times depending on whether it belongs to Set 1, Set 2, or Set 3, respectively.

The AFC plots in Figure 5.12 show that BRM-based alignment is substantially more robust than BAM-based alignment for both seen and unseen data. In contrast, the accuracy improvement of the BRM over the BAM, demonstrated by HRMSE, is not as large as the AFC melioration. For example, on Set 3 the average ($\pm$ the standard deviation) BRM-RMSE is 0.5745 ± 0.1725, whereas the average BAM-RMSE is 0.6533 ± 0.1594. This means that, when approaching convergence,

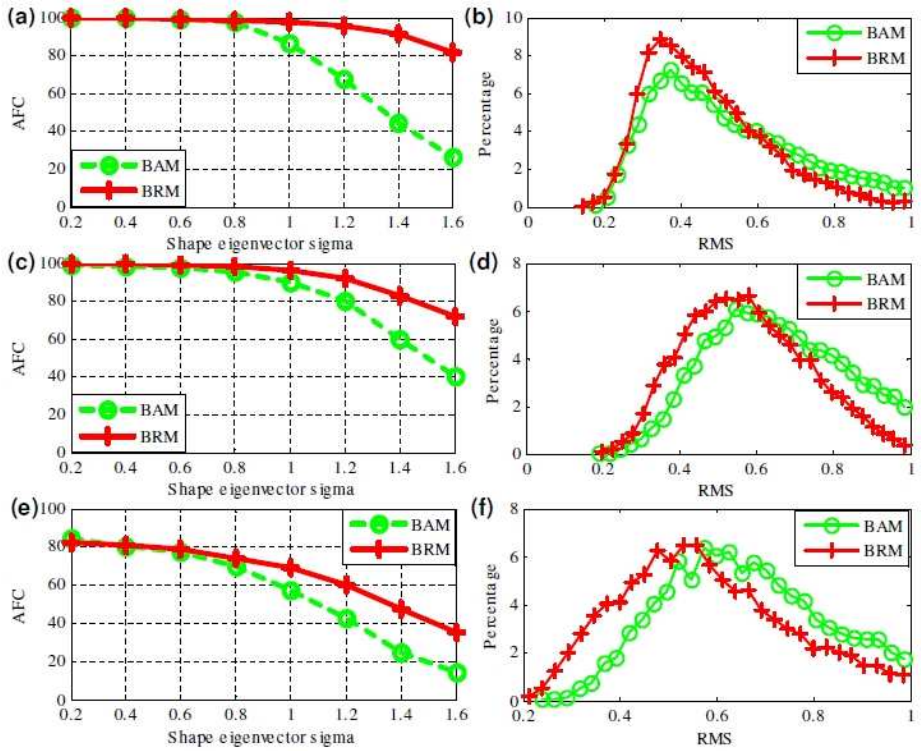

Figure 5.12: Alignment Performance. From top to bottom, AFC and HRMSE of both BAM and BRM computed on Set 1, Set 2, and Set 3, respectively. The HRMSE is computed on the trials where both algorithms converge.

BAM and BRM have comparable ability to rank pairs. This aspect is confirmed also by the left-most plot of Figure 5.11(b). Speed. When computing Figure 5.12(c) on a low-end PC, we recorded the time and number of iterations taken by our MatlabTM implementation of the BAM, and of the BRM, to converge. When both algorithms converge, the BAM takes an average of 8.06 iterations, and 0.122 seconds, whereas the BRM takes an average of 7.4 iterations, and 0.112 seconds. We attribute this improvement to the superior property of the ranking function of the BRM, compared to the classifier function of the BAM.

5.6 Conclusion

We have introduced the Boosted Ranking Model (BRM), a new discriminative face model suitable to perform face alignment. The BRM is associated to a score function learned from data, which is meant to be local extrema free to ensure that fitting can be achieved via gradient ascent. Learning a BRM corresponds to training a boosted classifier with a particular structure, that makes it equivalent to learning a boosted ranking function. This is done by extending GentleBoost to rank-learning, which we found to work better than other methods. The BRM outperforms the BAM for both seen and unseen subjects, especially in terms of alignment robustness (due to the concave properties of the score function), while slightly improving the accuracy and computational speed. This is a parsimonious model (especially if compared with the AAM), with enhanced generalization properties, that holds the promise of fitting multiple face models to new subjects in real-time. Our approach is not bounded to work with faces, and it could be extended to work with other objects of interest. Moreover, the idea of building a local extrema free function through rank-learning could be applied to other vision problems, such as discriminative object tracking, which could greatly benefit from a smooth and local extrema free tracking score function.

Chapter 6

Future Research Directions

In this dissertation, we have exploited the notion of time-reversibility in visual tracking. We developed an online performance evaluation method for visual tracking systems and presented a bi-directional visual tracking framework. In addition, we proposed a boosted ranking algorithm to reduce the local extrema in the score function learned from training samples. In this chapter, we discuss some future directions and summarize the dissertation.

6.1 Future Directions

We applied the time-reversibility constraint to the well-known KLT tracker and developed a new KLT algorithm. Extensive experimental results were given comparing the performances between the traditional KLT and the new KLT. The results show the new KLT using the time-reversibility constraint significantly outperforms the traditional one. We are working toward applying this strategy to other tracking algorithms, like mean-shift tracker and CONDENSATION/particle filtering-based trackers. Also, the time-reversibility constraint can be combined with optical flow to get more accurate motion estimates.

As we know, the model alignment problem can be viewed as a more accurate

tracking problem. In fact the face alignment algorithm we developed can be directly used as a face tracking algorithm. This motivates us to extend this work to develop a visual tracking algorithm with similar motivation, that is, to avoid being stuck in local extrema during tracking. However, general object tracking presents its own difficulties: In face alignment, the human face has a relatively stable structure and the same topology, so we can learn a statistical shape and appearance models to help alignment. In the general object tracking problem, different objects have different appearances and shapes, so we cannot use the same strategy to learn the shape and appearance model. This is the major obstacle for developing similar tracking algorithms. We will further explore the possibility developing a novel rank learning algorithm for visual tracking.

6.2 Closing Summary

In this dissertation, we have looked into a seldom noticed but very intrinsic property of object motion, time-reversibility, in visual tracking systems. We also exploited a new learning algorithm for optimization problems. We sucessfully applied these new algorithms to solve visual tracking and model alignment problems. In the future, we will continue to explore the applications of these ideas to different problems.

Bibliography

[1] A. Agarwal, "Learning to rank networked entities," *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge*, pp. 14–23, August 2006, Philadelphia, USA.

[2] C. Andrieu, A. Doucet, S. Singh, and V. Tadic, "Particle methods for change detection, system identification, and control," *Proceedings of the IEEE*, vol. 92, pp. 423–438, March 2004.

[3] V. Athitsos, J. Alon, S. Sclaroff, and G. Kollios, "Boostmap: A method for efficient approximate similarity rankings," *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, vol. 2, pp. 268–275, June 2004, Washington DC, USA.

[4] S. Avidan, "Support vector tracking," *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, vol. 1, pp. 184–191, December 2001, Kauai, Hawaii, USA.

[5] S. Baker and I. Matthews, "Lucas-Kanade 20 years on: A unifying framework," *Int. J. Computer Vision*, vol. 56, pp. 221–255, March 2004.

[6] Y. Bar-Shalom and T. E. Fortmann, "Tracking and data association," *San Diego,California: Academic Press, Inc.*, 1988.

[7] S. Birchfield, "Derivation of kanade-lucas-tomasi tracking equation," *Unpublished*, January 1997.

[8] J. Black, T. Ellis, and P. Rosin, "A novel method for video tracking performance evaluation," *Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance(VS-PETS)*, pp. 125–132, 2003.

[9] P. Bremaud, *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*. Springer, 2nd edition, 2001.

[10] T. Caljon, V. Enescu, P. Schelkens, and H. Sahli, "An offline bidirectional tracking scheme," *Advanced Concepts for Intelligent Vision Systems*, pp. 587–594, September 2005, Antwerpen.

[11] K. Cannons, "A review of visual tracking," *Technical Report*, September 2008.

[12] K. Chang, K. Bowyer, and P. Flynn, "Face recognition using 2D and 3D facial data," *In Proceedings of ACM Workshop on Multimodal User Authentication*, pp. 25–32, 2003.

[13] G. Christensen, "Consistent linear-elastic transformations for image matching," *Information Processing in Medical Imaging*, vol. 1613, pp. 224–237, June 1999.

[14] G. Christensen and H. Johnson, "Consistent image registration," *IEEE Transactions on Medical Imaging*, vol. 20, pp. 568–582, July 2001.

[15] K. L. Chung and J. B. Walsh, *Markov Processes, Brownian Motion, and Time Symmetry*. Springer, Second Edition, 2005.

[16] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 142–149, June 2000, Hilton Head Island, SC, USA.

[17] ——, "Kernel-based object tracking," *IEEE Transaction on Pattern Analysis Machine Intellegence*, vol. 25, pp. 564–575, 2003.

[18] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 681–685, June 2001.

[19] T. Cootes, C. Taylor, D. Cooper, and J. Graham, "Training models of shape from sets of examples," *Int'l Conf. on British Machine Vision Conference*, pp. 9–18, 1992, Leeds, UK.

[20] D. Cristinacce and T. F. Cootes, "Facial feature detection and tracking with automatic template selection," *7th International Conference on Automatic Face and Gesture Recognition(FGR)*, pp. 429–434, April 2006, Southampton.

[21] ——, "Boosted regression active shape models," *Int'l Conf. on British Machine Vision Conference*, vol. 2, pp. 880–889, September 2007, Warwich, UK.

[22] G. Dedeoglu, T. Kanade, and S. Baker, "The asymmetry of image registration and its application to face tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 807–823, May 2007.

[23] F. der Heijen, "Consistency checks for particle filters," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, pp. 140–145, January 2006.

[24] R. Donner, M. Reiter, G. Langs, P. Peloschek, and H. Bischof, "Fast active appearance model search using canonical correlation analysis," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 1690–1694, October 2006.

[25] A. Doucet, N. D. Freitas, and N. Gordon, *Sequential Monto Carlo Methods in Practice.* New York, NY: Springer-Verlag, 2001.

[26] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition).* Wiley-Interscience, 2000.

[27] A. Einstein, *Investigations on the Theory of Brownian Movement.* New York: Dover, 1956.

[28] C. Erdem, A. Murat Tekalp, and B. Sankur, "Metrics for performance evaluation of video object segmentation and tracking without ground-truth," *Proc. IEEE Int'l Conf. on Image Processing*, vol. 2, pp. 69–72, October 2001, Thessaloniki, Greece.

[29] C. Erdem, B. Sankur, and A. Tekalp, "Non-rigid object tracking using performance evaluation measures as feedback," *Proc. IEEE Int'l Conf. on Com-*

put. Vis. and Patt. Recog., vol. 2, pp. 323–330, December 2001, Hawaii, USA.

[30] F. Zuo, P. de With, "Fast facial feature extraction using a deformable shape model with Haar-wavelet based local texture attributes," *Proc. IEEE Int'l Conf. on Image Processing*, vol. 3, pp. 1425–1428, October 2004, Singapore.

[31] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," *Journal of Machine Learning Research*, vol. 4, pp. 933–969, December 2003.

[32] A. Frome, Y. Singer, F. Sha, and J. Malik., "Learning globally consistent local distance functions for shape-based image retrieval and classification," *Proc. IEEE Int'l Conf. on Computer Vision*, pp. 1–8, October 2007, Rio de Janeiro, Brazil.

[33] B. Georis, F. Bremond, M. Thonnat, and B. Macq, "Use of an evaluation and diagnosis method to improve tracking performances," *International Conference on Visualization, Imaging and Image Processing*, pp. 827–832, september 2003, Benalmadena, Spain.

[34] R. Gerlach, C. Carter, and R. Kohn, "Diagnostics for time series analysis," *Journal on Time Series Analysis*, vol. 20, pp. 309–330, 1999.

[35] J. Goldberger, S. Gordon, and H. Greenspan, "An efficient image similarity measure based on approximations of KL-divergence between two Gaussian mixtures," *Proc. IEEE Int'l Conf. on Computer Vision*, vol. 1, pp. 487–493, October 2003, Nice, France.

[36] R. Gross, I. Matthews, and S. Baker, "Constructing and fitting active appearance models with occlusion," *IEEE Intl. Conf. on Computer Vision and Pattern Recognition Workshop (CVPRW'04)*, vol. 5, pp. 72–80, June 2004, Washington, DC, USA.

[37] ——, "Generic vs. person specific active appearance models," *Image and Vision Computing*, vol. 23, pp. 1080–1093, November 2005.

[38] ——, "Active appearance models with occlusion," *Image and Vision Computing*, vol. 24, pp. 593–604, June 2006.

[39] M. Haw, "Einstein's random walk," *Physics World*, vol. 18, no. 1, 2005.

[40] R. Herbrich, T. Graepel, and K. Obermayer, "Support vector learning for ordinal regression," *In Proceedings of the Ninth International Conference on Articial Neural Networks*, pp. 97–102, 1999.

[41] A. Hidaka, K. Nishida, and T. Kurita, "Face tracking by maximizing classification score of face detector based on rectangle features," *ICVS '06: Proceedings of the Fourth IEEE International Conference on Computer Vision Systems*, p. 48, 2006, Washington, DC, USA.

[42] H. Iro, *A Modern Approach to Classical Mechanics*. World Scientific (Singapore, River Edge, NJ), 2002.

[43] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," *European Conference on Computer Vision*, vol. 1, pp. 343–356, April 1996, Cambridge, UK.

[44] ——, "CONDENSATION – conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, pp. 5–28, 1998.

[45] ——, "A smoothing filter for condensation," *Proceedings of the 5th European Conference on Computer Vision*, vol. 1, pp. 767–781, June 1998, Freiburg, Germany.

[46] A. Jepson, D. Fleet, and T. El-Maraghi, "Robust online appearance models for visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1296–1311, October 2003.

[47] F. Jiao, S. Li, H.-Y. Shum, and D. Schuurmans, "Face alignment using statistical models and wavelet features," *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, vol. 1, pp. 321–327, June 2003, Nice, france.

[48] T. Joachims, "Optimizing search engines using clickthrough data," *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 133–142, June 2002, Edmonton, Alberta, Canada.

[49] S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte, "A new approach for filtering nonlinear systems," *In American Control Conference*, pp. 1628–1632, June 1995, Seattle, WA, USA.

[50] F. Kahraman, M. Gokmen, S. Darkner, and R. Larsen, "An active illumination and appearance (AIA) model for face alignment," *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, pp. 1–7, June 2007, New York, NY, USA.

[51] F. Kahraman and M. Gokmen, "Illumination invariant face alignment using multi-band active appearance model," *Pattern Recognition and Machine Intelligence*, vol. 3776, pp. 118–127, December 2005.

[52] A. Kanaujia and D. N. Metaxas, "Large scale learning of active shape models," *Proc. IEEE Int'l Conf. on Image Processing*, vol. 1, pp. 265–268, September 2007, San Antonio, TX, USA.

[53] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Computer Vision*, vol. 1, pp. 321–331, January 1988.

[54] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, M. Boonstra, and V. Korzhova, "Performance evaluation protocol for face, person and vehicle detection & tracking in video analysis and centent extraction," *(VACE-II) CLEAR - Classification of Events, Activities and Relationships, Tampa, January*, 2006.

[55] M. Klass, M. Briers, N. D. Freitas, A. Doucet, S. Maskell, and D. Lang, "Fast particle smoothing: If I had a million particles," *Proc. IEEE Int. Conf. on Machine Learning*, pp. 481–488, September 2006, Pittsburgh, Pennsylvania, USA.

[56] S. Kramer, G. Widmer, B. Pfahringer, and M. De Groeve, "Prediction of ordinal classes using regression trees," *Intelligent Systems*, vol. 47, pp. 1–13, September 2001.

[57] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM J. Res.*, 1961.

[58] S. L. Lauritzen, *Thiele: Pioneer in Statistics*. Oxford University Press, 2002.

[59] H. Leff(Editor) and A. F.Rex(Editor), *Maxwell's Demon 2: Entropy, Classical and Quantum Information, Computing*. Institute of Physics Publishing, 2003.

[60] L. Liang, F. Wen, Y. Q. Xu, X. Tang, and H. Y. Shum, "Accurate face alignment using shape constrained markov network," *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, vol. 1, pp. 1313–1319, June 2006, New York, NY, USA.

[61] T. List, J. Bins, J. Vazquez, and R. Fisher, "Performance evaluating the evaluator," *Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 129–136, October 2005, Nice, France.

[62] X. Liu, "Generic face alignment using boosted appearance model," *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, vol. 2, pp. 1079–1088, June 2007, Minneapolis, Minnesota, USA.

[63] L. Lu, X. Dai, and G. Hager, "A particle filter without dynamics for robust 3D face tracking," *IEEE Proc. Computer Vision and Pattern Recognition Workshops (CVPRW)*, vol. 5, pp. 70–70, June 2004, Washington DC, USA.

[64] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," *International Joint Conference on Artificial Intelligence*, pp. 674–679, August 1981, Vancouver, BC, Canada.

[65] I. Matthews and S. Baker, "Active appearance models revisited," *Int. J. Computer Vision*, vol. 60, pp. 135–164, November 2004.

[66] R. Meir and G. Ratsch, "An introduction to boosting and leveraging," *In Advanced Lectures on Machine Learning: Machine Learning Summer School 2002*, pp. 118–183, 2004.

[67] J. Nascimento and J. Marques, "Performance evaluation of object detection algorithms for video surveillance," *IEEE Transactions on Multimedia*, vol. 8, pp. 761–774, August 2006.

[68] E. Nelson, *Dynamical Theories of Brownian Motion.* Princeton University Press, 1967.

[69] N. J. Newton, "Dual Kalman-Bucy filters and interactive entropy production," *SIAM J. Control Optim.*, vol. 45, pp. 998–1016, 2006.

[70] ——, "Dual nonlinear filters and entropy production," *SIAM J. Control Optim.*, vol. 46, pp. 1637–1663, 2007.

[71] ——, "Interactive statistical mechanics and nonlinear filtering," *Journal of Statistical Physics*, vol. 133, pp. 711–737, November 2008.

[72] A. T. Nghiem, F. Bremond, M. Thonnat, and V. Valentin, "ETISEO, performance evaluation for video surveillance systems," *IEEE Conference on Advanced Video and Signal Based Surveillance*, pp. 476–481, September 2007.

[73] A. Nghiem, F. Bremond, M. Thonnat, and R. Ma, "A new evaluation approach for video processing algorithms," *IEEE Workshop on Motion and Video Computing*, pp. 15–15, February 2007.

[74] C. P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," *Proc. IEEE Int'l Conf. on Computer Vision*, pp. 555–562, January 1998, Bombay, India.

[75] P. J. Phillips, M. Hyeonjoon, S. A. Rizvi, and P. J. Rauss, "The FERET evaluation methodology for face-recognition algorithms," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1090–1104, October 2000.

[76] M. B. Propp, *The Thermodynamic Properties of Markov Processes.* Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA,USA, 1985.

[77] M. Richardson, "Beyond pagerank: Machine learning for static ranking," *In WWW 06: Proceedings of the 15th international conference on World Wide Web*, pp. 707–715, 2006, Edinburgh, Scotland.

[78] S. M. Ross, *Introduction to probability models.* Academic Press; 9th edition, 2007.

[79] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, December 2000.

[80] S. K. Mitter and N. J. Newton, "Information and entropy flow in the Kalman-Bucy filter," *Journal of Statistical Physics*, vol. 118, pp. 145–176, January 2005.

[81] J. Saragih and R. Goecke, "A nonlinear discriminative approach to aam fitting," *Proc. IEEE Int'l Conf. on Computer Vision*, pp. 1–8, October 2007, Rio de Janeiro, Brazil.

[82] T. Schlogl, C. Beleznai, M. Winter, and H. Bischof, "Performance evaluation metrics for motion detection and tracking," *International Conference on Pattern Recognition*, vol. 4, pp. 519–522, August 2004, Cambridge, UK.

[83] T. Shanableh and M. Ghanbari, "Backward tracking of B-pictures bidirectional motion for interframe concealment of anchor pictures," *Proc. IEEE Int'l Conf. on Image Processing*, vol. 3, pp. 396–399, 2000, Vancouver, BC, Canada.

[84] ——, "The importance of the bi-directionally predicted pictures in video streaming," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, pp. 402–414, March 2001.

[85] J. Shi and C. Tomasi, "Good features to track," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 593–600, June 1994, Seattle, Washington, USA.

[86] M. B. Stegmann, B. K. Ersboll, and R. Larsen, "Fame-a flexible appearance modeling environment," *IEEE Trans. on Medical Image*, vol. 22, pp. 1319–1331, October 2003.

[87] J. Sun, W. Zhang, X. Tang, and H.-Y. Shum, "Bidirectional tracking using trajectory segment analysis," *Proc. IEEE Int'l Conf. on Computer Vision*, vol. 1, pp. 717–724, October 2005, Beijing, China.

[88] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, pp. 2319 – 2323, 2000.

[89] C. Tomasi and T. Kanade, "Detection and tracking of point features," *Carnegie Mellon University Technical Report CMU-CS-91-132*, April 1991.

[90] J. Tu, Z. Zhang, Z. Zeng, and T. Huang, "Face localization via hierarchical condensation with fisher boosting feature selection," *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, pp. 719–724, June 2004, Washington DC, USA.

[91] B. Van Ginneken, A. F. Frangi, J. J. Staal, B. M. t. H. Romeny, and M. A. Viergever, "A non-linear gray-level appearance model improves active shape model segmentation," *MMBIA '01: Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, p. 25, 2001, Washington, DC, USA.

[92] N. Vaswani, "Additive change detection in nonlinear systems with unknown change parameters," *IEEE Trans. Signal Processing*, vol. 55, pp. 859–872, March 2006.

[93] A. Veeraraghavan, R. Chellappa, and M. Srinivasan, "Shape-and-behavior encoded tracking of bee dances," *IEEE Transaction on Pattern Analysis Machine Intellegence*, vol. 30, pp. 463–476, March 2008.

[94] J. Vermaak, C. Andrieu, A. Doucet, and S. Godsill, "Particle methods for Bayesian modelling and enhancement of speech signals," *IEEE Trans. Speech and Audio Processing*, vol. 10, pp. 173–185, March 2002.

[95] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Computer Vision*, vol. 57, pp. 137–154, 2004.

[96] C. Vogler, Z. Li, A. Kanaujia, S. Goldenstein, and D. Metaxas, "The best of both worlds: Combining 3D deformable models with active shape models," *Proc. IEEE Int'l Conf. on Computer Vision*, pp. 1–7, October 2007, Rio de Janeiro, Brazil.

[97] O. Williams, A. Blake, and R. Cipolla, "Sparse Bayesian learning for efficient visual tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 1292–1304, August 2005.

[98] H. Wu, R. Chellappa, A. S. Sankaranarayanan, and S. K. Zhou, "Robust visual tracking using the time-reversibility constraint," *Proc. IEEE Int'l Conf. on Computer Vision*, pp. 1–8, October 2007, Rio de Janeiro, Brazil.

[99] H. Wu, A. Sankaranarayanan, and R. Chellappa, "In situ evaluation of tracking algorithms using time reversed chains," *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, pp. 1–8, June 2007, Minneapolis, MN, USA.

[100] H. Wu, A. C. Sankaranarayanan, and R. Chellappa, "Online empirical evaluation of tracking algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 99, June 2009.

[101] H. Wu and Q. Zheng, "Performance self-evaluation of visual tracking systems," *Army Science Conference*, 2004, Washington DC, USA.

[102] J. Xu and Y. Cao, "Ranking definitions with supervised learning methods," *In Proceedings of the 14th International World Wide Web Conference*, pp. 811–819, February 2005, Chiba, Japan.

[103] S. Yan, M. Li, H. Zhang, and Q. Cheng, "Ranking prior likelihood distributions for Bayesian shape localization framework," *Proc. IEEE Int'l Conf. on Computer Vision*, vol. 1, pp. 51–58, 2003, Nice, France.

[104] S. Yeung and P. Shi, "Stochastic inverse consistency in medical image registration," *Medical Image Computing and Computer-Assisted Intervention*, pp. 188–196, 2005.

[105] A. Yilmaz, "Object tracking by asymmetric kernel mean shift with automatic scale and orientation selection," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–6, June 2007, Minneapolis, MN, USA.

[106] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, December 2006.

[107] L. Zhang, H. Ai, S. Xin, C. Huang, S. Tsukiji, and S. Lao, "Robust face alignment based on local texture classifiers," *Proc. IEEE Int'l Conf. on Image Processing*, vol. 2, pp. 354–357, September 2005, Genova.

[108] Q. Zheng and R. Chellappa, "Automatic feature point extraction and tracking in image sequences for arbitrary camera motion," *International Journal of Computer Vision*, vol. 15, pp. 31–76, June 1995.

[109] Y. Zheng, X. S. Zhou, B. Georgescu, S. K. Zhou, and D. Comaniciu, "Example based non-rigid shape detection," *Proc. European Conf. on Computer Vision*, pp. 423–436, May 2006, Graz, Austria.

[110] S. K. Zhou, R. Chellappa, and B. Moghaddam, "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Trans. Image Processing*, vol. 13, pp. 1491–1506, November 2004.

[111] Y. Zhou, L. Gu, and H. J. Zhang, "Bayesian tangent shape model: Estimating shape and pose parameters via Bayesian inference," *Proc. IEEE Int'l Conf. on Comput. Vis. and Patt. Recog.*, vol. 1, pp. 109–116, June 2003, Madison, Wisconsin, USA.