VISUAL TRACKING WITH MOTION ESTIMATION AND ADAPTIVE TARGET APPEARANCE MODEL EMBEDDED IN PARTICLE FILTERING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ERKAN BAŞER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2008

Approval of the thesis:

VISUAL TRACKING WITH MOTION ESTIMATION AND ADAPTIVE TARGET APPEARANCE MODEL EMBEDDED IN PARTICLE FILTERING

submitted by **ERKAN BAŞER** in partial fulfilment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by**,**

Prof. Dr. Canan, Özgen _____
Dean, Graduate School **of Natural and Applied Sciences**

Prof. Dr. İsmet Erkmen _____
Head of Department, **Electrical and Electronics Engineering**

Prof. Dr. Mübeccel Demirekler _____
Supervisor, **Electrical and Electronics Eng. Dept., METU**

Examining Committee Members:

Prof. Dr. Aydan Erkmen _____
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Mübeccel Demirekler _____
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Gözde Bozdağı Akar _____
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. Aydın Alatan _____
Electrical and Electronics Engineering Dept., METU

Dr. Tayfun Çimen _____
Senior System and Control Engineer, Roketsan

Date: 24.09.2008

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Erkan Başer

Signature:

ABSTRACT

VISUAL TRACKING WITH MOTION ESTIMATION AND ADAPTIVE TARGET APPEARANCE MODEL EMBEDDED IN PARTICLE FILTERING

Başer, Erkan

M.Sc., Department of Electrical and Electronics Engineering

Supervisor: Prof Dr. Mübeccel Demirekler

September 2008, 103 Pages

In this thesis we study Particle Filter for visual tracking applications. The sequential Monte Carlo methods or Particle Filter provides approximate solutions when the tracking problem involves non-linear and/or non-Gaussian state space models. Also in this study, in order to make the visual tracker robust against change in target appearance and unexpected target motion, an adaptive target appearance model and a first order motion estimator are embedded in particle filtering. Additionally, since pixels that don't belong to target makes the motion estimation biased, the algorithm includes robust estimators to make the tracker reliable.

Within the scope of this thesis the visual tracker proposed in [5] is implemented and the same problem is solved by proposing a Rao-Blackwellized Particle Filter. To deal with problems encountered during the implementation phase of the algorithm some improvements are made such as utilizing learning rate for the computation of adaptive velocity estimation. Moreover, some precautions are taken such as checking the velocity estimations to validate them.

Finally, we have done several experiments both in indoor and outdoor environments to demonstrate the effectiveness and robustness of the implemented algorithm.

Experimental results show that most of the time the visual tracker performs well. On the other hand the drawbacks of the implemented tracker are indicated and we explain how to eliminate them.

Keywords: Visual Tracking, Particle Filtering, Adaptive Target Appearance Model, Motion Estimation, Rao-Blackwellized Particle Filtering

ÖZ

GÖRSEL HEDEFLERİN PARÇACIK FİLTRELEMESİ İÇİNE GÖMÜLMÜŞ HIZ TAHMİNİ VE ADAPTİF HEDEF GÖRÜNTÜSÜ MODELİ İLE İZLENMESİ

Başer, Erkan

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Prof Dr. Mübeccel Demirekler

Eylül 2008, 103 Sayfa

Bu tezde görsel hedef takip uygulamaları için parçacık filtrelemesine çalışıldı. Sıralı Monte Carlo metotları veya parçacık filtrelemesi hedef izleme problemi doğrusal olmayan ve/veya Gaussian olmayan bir durum uzayına sahip olduğunda problemin çözümünü yaklaşık olarak yapan yöntemlerdir. Ayrıca bu çalışmada görsel hedef izleyicinin hedefin görüntüsünde meydana gelen değişimlere ve beklenmedik hedef hareketlerine karşı daha dirençli olması amacıyla adaptif hedef görüntüsü ve birinci dereceden hız tahmini yapan kestirici de parçacık filtrelemesi içine gömülmüştür. Bundan başka hız tahmini sırasında hedefe ait olmayan piksellerin tahmin doğruluğunu en az derecede etkilemesi için bu piksellere karşı duyarlı olmayan kestiriciler kullanmıştır.

Bu tez kapsamında [5]'te önerilen görsel hedef takip algoritması yazılıp uygulanmış ayrıca aynı problem Rao-Blackwellized parçacık filtrelemesi ile çözülmeye çalışılmıştır. Algoritmayı geliştirme esnasında karşılaşılan problemlerin çözümü için örneğin, hız tahmininde öğrenme katsayısının kullanımı ve yapılan tahminlerin önceden belirlenen değerler ile geçerleşmesi gibi bağzı iyileştirmeler yapılmıştır.

Yazılan algoritmanın etkinliğini ve sağlamlığını göstermek amacıyla hem iç ortamda hem de dış ortamda bir çok deney yapılmıştır. Deneysel sonuçlar göstermiştir ki yazdığımız hedef izleyici çoğu zaman başarı ile seçilen hedefi izlemiştir. Diğer taraftan bu hedef izleyicinin eksikleri gösterilip bunların nasıl ortadan kaldırılacağı da açıklanmıştır.

Anahtar Kelimeler: Görsel Hedef İzleme, Parçacık Filtrelemesi, Adaptif Hedef Görüntüsü Modeli, Hız Tahmini, Rao-Blackwellized Parçacık Filtrelemesi

ACKNOWLEDGEMENTS

I am greatly thankful to my supervisor Prof. Dr. Mübeccel Demirekler for her remarkable affect on every step of this thesis. Throughout this period, by her appreciated advices, thought systems and cheerful personality, had I the chance of improvement.

The author acknowledges all those who contributed to the completion of this work in this or that way.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION AND LITERATURE SURVEY

1.1 Literature Survey and Motivation

In recent years many visual tracking algorithms use features extracted from the target region together with a motion model to be more robust and efficient than conventional ones. Extracted features can be local or template correspondences. Local correspondences are local features of the target region derived from image intensities such as edges, lines, corners etc. On the other hand template correspondences use characteristics of the target image region such as color histogram of the image. In other words they make direct and complete use of all available image intensity within the target region [1]. The benefit of the local correspondences is that tracking algorithm becomes robust to partial occlusion of the target. Also they are invariant to geometric transformations such as translations, rotations and changes in scale [2]. Alternatively template correspondences make possible to track the objects, templates of which is too complex to be modeled by local features [3]. However, template correspondences can cause to lose the target if appropriate appearance of the target can't be captured during the tracking. Main factors causing this problem are deformation of the target appearance due to motion of the target, illumination changes and outliers. Therefore, a reliable and online target appearance model is needed to adapt the target appearance without losing its stable structure. In this way the algorithm can be respectively robust and insensitive to the geometric transformations and outliers as in the case of using local correspondences in the tracking algorithm. In [4], Jepson et. al. present an adaptive appearance model that consists of three components. In this model the first component captures the stable and slowly varying properties of the target appearance. This component for each target pixel is modeled as a Gaussian

distribution with small variance, whereas the second component determines outlier pixels when target is occluded and thus it is accepted as a uniform distribution over the observation domain (gray scale images). Finally, the last component of the model learns the change in target appearance so that it enables to use the proposed appearance model in the visual tracking algorithms. In other words until capturing the stable structures of the target appearance tracking algorithm makes use of this component of the model. Furthermore, after rapid and unexpected changes in target appearance it is more reliable to track the target according to this component of the model than stable one. Consequently, the components of this online adaptive appearance model are combined by means of Bayesian approach to estimate the expected appearance of the target in future frames. In [5] Zhou et. al. improve Jepson's study by embedding this online adaptive appearance model into a visual tracker with particle filtering. Moreover, they preferred to use robust statistics to deal with outliers and a fixed target template that is the most expected one instead of the second component used in Jepson's online adaptive appearance model. In both of these studies the parameters of the model components are updated by means of Expectation Maximization (EM) algorithm.

Besides target features modern tracking algorithms need a motion model characterizing the kinematic behavior of the target. In general they are time invariant models with unchanging noise variances. In other words system models, consisting of state transition matrix, input gain (if any) and noise covariance are all predetermined and fixed. However, in many situations they must be adaptive to changes in kinematic behavior of the target. In [5] this problem is solved via an adaptive motion model. In this article, the parametric motion model is selected as an affine transformation and the change in its six parameters, which are called adaptive velocity, are computed. Another approach is to look for the optimum values of the motion parameters by locating local minimum of a cost function via one of the two search methods, which are called deterministic gradient descent and stochastic diffusion [6]. In this approach if target moves smoothly and slowly, the new position and pose of the target is determined by searching the minimum of the performance surface constructed by the cost function through deterministic gradient

descent method, which is initiated by the previous values of the particles. However, the deterministic gradient descent can't be an appropriate approach if target moves rapidly or deforms quickly. In that case second approach is used and particles are propagated by a stochastic motion model. In [7] authors illustrate that the motion information can be derived using stochastic background model developed by detection algorithm. The proposed algorithm incorporates this model into the appearance based tracker with particle filtering and so the pixels of incoming image are labeled as background or foreground pixels as a result of a hypothesis test. Afterwards, the appearance based tracker estimates the most probable target region making use of the cue provided by the detection algorithm.

In target tracking algorithms the aim is to estimate the values of the state space variables that indicate the location and pose of the object at every time step. In modern target tracking applications these variables are estimated in the Bayesian framework. [5], [6], [8], [9] and [10] illustrate that PF (Particle Filter) can be used to obtain a solution to these recursive Bayesian Filtering problems. In these articles state space is composed of translation and deformation parameters of target features and observation model relates these state space variables to the target template. Using these models or probability distribution functions characterizing them values of state space variables are estimated through approximated posterior distribution of these variables.

All appearance based trackers mentioned up to now try to locate the target by utilizing target motion model in recursive Bayesian particle filtering. To do so they use a target template and match it to an image region indicated by particles. It is a kind of image registration process which is performed by minimizing the difference between target template and image region cropped out of the incoming frame. Difference is measured by a cost function and it creates a performance surface where optimum values of the state variables are searched. In literature Sum of Square Difference (SSD,) correlation and L2 norm are popular cost functions used for this purpose. Usually Gauss-Newton method is utilized to search for the global minimum of the performance surface where optimum state variables lie [5], [6]. It is a numerical way to search the best match and uses first order Taylor series

expansion to obtain quadratic performance surface of the state variables [11]. Although the optimization with first order Taylor series expansion enables the problem to be solved easily, it causes local minima/maxima problem [10]. In [6] many different hypotheses about the target's position and pose are formed by particle filtering and so local minima maxima problem can be solved.

Since particle filtering is an interference technique to represent the posterior distribution by a set of particles and their associated importance weights, it is crucial how to distribute particles over the state space [12]. Generating particles via Monte-Carlo Methods make particles possible to come from regions of the state space where the most probable values of the state space variables exist. In [12], [13] and [14] special Monte-Carlo Methods called SIS (Sequential Importance Sampling) and Sampling Importance Resampling (SIR) are used to distribute particles over the state space effectively. In [5] Zhou et al used motion information obtained by Hyperplane Approximation (HA) to predict the change in state space variables between two successive frames; thus particles are distributed over the state space close to predictions depending on process noise that indicates how well the prediction is done. Also in [1] another approach for template matching called Jacobian Approximation (JA) is used to obtain the motion information (change in state variables). Both of these methods assume that the target's movement and its deformation evolve in time according to a parametric motion model and target's illumination doesn't change during overall tracking. On the other hand in [3] the authors indicated that HA is better than JA to predict the target's motion. The reason is that HA uses available data directly to predict the parameters of the motion model, whereas JA uses the data to obtain the Jacobian matrix that relates the change in pixel intensities w.r.t the change in state variables at first and then predicts the parameters of the motion model using the data and the Jacobian matrix indicating the gradient of gray scale intensities.

Nowadays particle filtering has become a popular stochastic method used in visual trackers. Most of the aforementioned visual trackers mentioned here also use particle filter (PF) to perform the tracking process. The reason why particle filtering became so popular is because it doesn't require that the system and observation

models to be linear and driven by Gaussian noises, whereas they are prerequisites for linear optimal filters like Kalman [15]. The dimension of the state space is extremely important for visual trackers employing PF so that the increase in the dimension of the state space requires large number of particles should be employed to make robust and accurate estimates. However, in this case, the computational performance of the algorithm decreases due to increased computational load. To deal with these problems Rao-Blackwellized particle filtering is developed. In this filter if it is possible, the state space is partitioned into two parts and then one of them is estimated using PF while other part is tractable analytically via Kalman filter (KF) [16]. In [17] the authors present a visual tracker where the selected reference points within the object region are tracked via Rao-Blackwellized particle filtering. In this article state space consists of translational motion variables and variables related to the geometric structure of the reference points. Translational motion variables evolve according to a nonlinear system model subject to Gaussian noise, whereas other state variables, conditioned on the nonlinear ones, evolve according to a linear system model driven by Gaussian noise. In the algorithm proposed by [15] the constraints imposed by the camera-scene configuration is utilized to partition the state space into a linear and nonlinear subspaces and so Rao-Blackwellized Particle Filtering (RBPF) becomes applicable. In their work, targets are modeled using ellipses with eight parameters describing their dynamics, where four of them are related to the motion of the ellipse (positions and velocities on 2-D planar surface) and estimated by PF while other four parameters estimated by KF indicate the size of the ellipse and rates of scale changes. Also, authors of this article proved that RBPF performs better than classical PF for this filtering problem. Finally, Schinder and Frank [18] presented a visual tracker employing RBPF. In their work the tracked objects (insects) are modeled as constellation of flexible parts where the configuration of each part are described by pose and shape variables. The pose variables (2D position and rotation) are estimated via traditional PF and shape variables are analytically tracked using image registration technique based on Lucas-Kanade algorithm. Furthermore, they used an appearance model learned by means of Expectation Maximization (EM) for each part of the object also; they utilized these models in computation of likelihoods and image registration process.

Besides HA and JA approaches for template matching processes, another solution to the template matching problem is to use Gaussian Pyramid approach [19]. In this approach resolution of the incoming frame is reduced and a pyramid with n levels is constructed using the obtained images from coarsest resolution to the original one. Starting from top of the pyramid and using the template image whose resolution is reduced at the same level, change in target motion parameters are predicted via Lukas-Kanade Algorithm and then it becomes input to the next level of the pyramid; thus the computational saving is achieved using low resolution images and estimation is improved as approaching to the bottom of the pyramid [11]. Consequently, the estimation obtained from original image and target template becomes final result.

Since these motion estimation procedures depend on pixel intensities, they are sensitive to illumination changes. Therefore, in [1] illumination of the target is assumed to be constant to make reliable motion estimations. However, in real environment illumination of the target changes due to automatic exposure adjustment, change in light source irradiance and motion of the target [20]. To deal with this problem Zhou at el [5] utilized adaptive appearance model that absorbs the change in target illumination. Also, in [21] two methods are proposed to either compensate the illumination changes or reduce the sensitivity of template matching algorithms to the change in illumination. In the first method incoming frames are preprocessed by sobel filter; thus edge images, which are less sensitive to the illumination changes, are obtained. However, while this method decreases the sensitivity to the change in illumination, it causes other problems like decrease in computational performance of the tracking algorithm and increase in image noise. Second method compensates illumination changes by two different ways: (i) using statistics of image region and (ii) using a linear model for illumination compensation parameters. Since the distribution of pixel intensities are less affected by illumination changes first approach in the second method is proposed to be used in compensating the illumination changes.

Another important problem for visual trackers is the occlusion of the target by other objects. In this case motion estimations depending on pixel intensities deviate from

the true values and so tracker loses the target in long time period. To deal with this problem in [1], [5] and [9] the influence of the pixels classified as outliers is decreased on the motion estimation by windsorizing function. Also, the importance weights of the particles are computed by using this function in [5] and [9] to obtain robust estimations of the target state. In [22] the author indicates that with this optimization by down-weighting doubtful observations sharp decisions such as keeping or rejecting an observation is eliminated and so better estimations are obtained.

1.2. Scope

Within the scope of this thesis, we have implemeted the visual tracker that Zhou et. al. in [5] proposed. However, we didn't deal with occlusion handling and adaptive number of particles that based on asympotic relative efficieny. Also, since we don't have still images of the selected targets like in [23], and no information about the most observed template of the targets, F-component of the appearance model (fixed template of the target object) proposed in [5] isn't utilized in our target appearance model.

In this study, the implemented algorithm consists of adaptive target appearance model and a gradient based motion estimator. In this way, both state transition and observation models become adaptive [5]. Here, our state variables are parameters of the similarity transformation to crop out the region of interest. Also, in order to increase the effectiveness of the algorithm and handle nonlinearities we utilized Particle Filtering (PF). Moreover, to deal with some problems encountered during the implementation phase of the algorithm some improvements are added and some precautions are taken such as using learning rate and motion history in the computation of adaptive velocity. Finally, in appendix-A we propose Rao-Blackwellized Particle Filtering to solve the same problem in a more efficient way. We know that Rao-Blackwellized Particle Filter achieves variance reduction with an optimal filter as Kalman Filter and so using Rao-Blackwellized Particle Filter we the accuracy of the algorithm is increased.

1.3. Outline

The remaining part of the thesis is organized as follows:

In Chapter-2 Particle Filtering is introduced and also Sequential Importance Sampling (SIS) method which is the fundamental and basic solution to most of the particle filtering problems will be explained. Also, we will give a brief overview about some variations of particle filters.

Chapter-3 presents similarity transformation which is an image warping function and the popular interpolation techniques are reviewed. The process of image cropping is explained in this chapter.

In Chapter-4 an adaptive appearance model (Expectation Maximization (EM) algorithm) for the tracked object, is introduced. Also, here we will examine how to update parameters of the model components in detail.

Chapter-5 covers the motion estimation process and discusses its limitations. Finally, an overview is given about the robust estimators at the end of this chapter.

In Chapter-6 the main steps of the proposed algorithm by Zhou and et.al. [5] is introduced. Also some fundamental functions of this algorithm are explained clearly.

Chapter-7 provides experimental results on the tracked objects and conclusions about this thesis are presented in Chapter-8.

Finally, some preliminary work about using Rao-Blackwellization techniques is introduced in Appendix-A

.

CHAPTER 2

SEQUENTIAL MONTE CARLO ESTIMATION

Let's consider a system that is described in the state space where the state vector of the system consists of the state variables containing all relevant information about the system and also the observation gives partial information about the state of the system. The fundamental solution to obtain the state of a system is provided by Bayesian Filtering. Bayesian Filtering is a statistical inference technique that approximates the posterior distribution of the state variables if the system and observation models are available; thus all statistical information about the system becomes available and the estimate of the state variables can be achieved [24].

Kalman filter is a recursive linear method in Bayesian framework to make optimal estimate of the state variables. It assumes that the posterior distribution of the state is Gaussian and characterized by two statistical parameters of the state vector, mean and covariance matrix [14]. Kalman filter makes minimized mean square error estimation if the system and the observation models are accurately determined [25].

In reality many system and/or observation models are nonlinear so the posterior distribution becomes non-Gaussian and KF can't be applied to obtain an optimal solution. In this case some suboptimal filters are used such as Extended Kalman Filter (EKF) Unscented Kalman Filter (UKF) and Particle Filter (PF) to achieve approximate solutions to the posterior distribution.

In this chapter a brief review of EKF and UKF will be given and then sequential Monte Carlo based nonlinear filters (PF) will be described in detail.

2.1. Suboptimal Nonlinear Filters

EKF is based on linearization of the nonlinear system dynamics and/or nonlinear observation function around the latest estimation or prediction of the state variables. Since the accuracy of the linearization depends on how accurate the estimates are, the EKF diverges quickly when estimates become inaccurate. Unlike the EKF, the UKF doesn't linearize the nonlinear functions [12]. It is known as a deterministic sampling technique where the posterior distribution represented using a minimal set of carefully chosen sample points. In UKF fixed number of deterministically selected particles is used where these particles completely capture the true mean and covariance of the approximated posterior density. When they are propagated through nonlinear state space models, the predicted state and predicted measurement are obtained and then the weighted average of the sample mean and covariance are calculated. In this way, the parameters of Gaussian approximation for the posterior distribution are reestimated [14]

Monte Carlo methods are also suboptimal solutions under nonlinear and/or non-Gaussian filtering problems. Since the posterior distribution can't be computed analytically by KF or its other versions like EKF or UKF, these methods use statistical samples (random samples) generated by Monte Carlo simulations and estimate the unknown state in Bayesian framework by utilizing the efficient approximation of the posterior distribution [13]. In the following parts of this chapter we will focus on a popular Monte Carlo based recursive Bayesian estimator known as Particle Filter (PF) and some fundamental Monte Carlo sampling methods used in particle filtering will be explained.

2.2 Particle Filter

The PF is a recursively implemented Monte Carlo based statistical signal processing technique used to make estimations in Bayesian framework [26]. The aim of the PF is to approximate the posterior distribution of the state vector in a numerical way [27]. Here, the particles represent the probable state of the system and the associated weights indicate the likelihood of the state represented by the particles. In other words the particles and their associated weights reflect the value of the

posterior distribution in a specific location of the state space [28]. Accurate approximation of the posterior density via these particles and their associated weights is crucial since the posterior distribution of the state vector contains all relevant statistical information about the dynamics of the system [29] thus; particles are distributed or guided according to a particular sampling method to provide accurate representation of the posterior distribution. Finally, using the approximation to the posterior density the estimate of the state $\hat{x}_k$ can be made either in a minimum mean square error (MMSE) or a maximum a posteriori (MAP) sense [5]:

$$\begin{aligned} \hat{x}_k &= \hat{x}_k^{MMSE} \approx \sum_{i=1}^{J} w_k^i x_k^i \\ \hat{x}_k &= \hat{x}_k^{MAP} \approx \arg\max_{x_k} p(x_k \mid z_{1:k}) = \arg\max_{x_k} w_k^j \end{aligned} \tag{2.1}$$

Where $\hat{x}_k$ denotes the estimate of the state vector, J is the number of particles and $z_{1:k}$ denotes all measurements up to time index k. In the MMSE sense particles whose weights are high dominate the state estimate of the system, whereas in the MAP sense the state estimate of the system is determined by the one particle which has the highest weight in the set of particles. Since accuracy of these estimates depends on how well the posterior distribution is approximated, the number of particles should also be large enough to make accurate approximations [29].

To deal with the sampling problem of particles some Monte Carlo methods are proposed. If the posterior density is available at each time step, the particles are generated by sampling this density in the region of importance to achieve efficiency [13]. However, the posterior density can be unknown or it can be difficult to make use of the posterior density to draw the samples, but it becomes efficient to sample from an easy-to-implement distribution covering the posterior density in the range of the values of the state variables [14]. This distribution is called importance density and denoted as $q(x_k \mid z_{1:k})$. There are two fundamental Monte Carlo sampling methods in Particle Filtering and their use in PF will be explained clearly in the subsequent parts of this chapter.

2.3 Particle Filtering Methods

2.3.1 Sequential Importance Sampling Algorithm (SIS)

SIS algorithm is the mostly used particle filtering algorithm [12]. It is a Monte Carlo method to implement the recursive Bayesian estimation based on choosing an importance density to guide the particles in the state space. In order to do so, it must generate new particles according to the importance density and calculate their associated weights to approximate the posterior density function by using the existing particle set $\left\{x_{0:k-1}^{i}\right\}\Big|_{i=1}^{J}$ representing previous posterior density, $p(x_{0:k-1}|z_{1:k-1})$ when new observation becomes available [30].

To develop the idea behind SIS let's assume that $\left\{x_k^i, w_k^i\right\}\Big|_{i=1}^{J}$ denotes a set of particles with associated weights and approximates numerically the posterior filtering density of the state vector at discrete time index k. To ensure this set of particles represent a proper distribution the weights are normalized such that $\sum_{i=1}^{J} w_k^i = 1$; then the discrete weighted approximation of the posterior filtering density is given by

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{N} w_k^i \delta(x_k - x_k^i) \tag{2.2}$$

However, because of the reasons mentioned before, the sampling from true posterior distribution can't be always possible. Therefore, the samples must be drawn from a proper importance density, $q\left(x_{0:k}|z_{1:k}\right)$ satisfying the following property [12]:

$$p(x_{0:k}|z_{1:k}) > 0 \Rightarrow q(x_{0:k}|z_{1:k}) > 0 \quad \text{for all x} \in \mathbb{R}^{n_x} \tag{2.3}$$

The above property imposes the assumption that the support of the importance density includes the support of the posterior distribution, $p(x_{0:k}|z_{1:k})$ [30].

By using the old particles $\left\{x_{0:k-1}^{i}\right\}\Big|_{i=1}^{J}$ with new ones $\left\{x_{k}^{i}\right\}\Big|_{i=1}^{J}$ the weighted approximation of importance density is obtained like in (2.2), thus the estimate of the state vector (conditional mean of the state vector) is given by

$$\begin{aligned} E\left[x_k \left| z_{1:k}\right.\right] &= \int x_k \frac{p(x_{0:k}\left| z_{1:k}\right.)}{q(x_{0:k}\left| z_{1:k}\right.)} q(x_{0:k}\left| z_{1:k}\right.) dx_k \\ &= \int x_k w(x_k) q(x_{0:k}\left| z_{1:k}\right.) dx_k \\ &\triangleq \sum_{i=1}^{J} w(x_k^i)\delta(x_{0:k} - x_{0:k}^i) \end{aligned} \tag{2.4}$$

Where $w(x_k^i)$ denotes the associated weight of ith the particle and it is defined to be

$$w(x_k^i) = \frac{p(x_{0:k}^i \left| z_{1:k}\right.)}{q(x_{0:k}^i \left| z_{1:k}\right.)} \tag{2.5}$$

Since SIS is a recursive Monte Carlo method, to evaluate the associated weights with new measurements in a recursive manner importance distribution must be in the form of

$$\begin{aligned} \mathrm{q}(\mathrm{x}_{0:k}\left| z_{1:k}\right.) &= q(x_k \left| x_{0:k-1}, z_{1:k}\right.) q(x_{0:k-1}\left| z_{1:k}\right.) \\ &= q(x_k \left| x_{0:k-1}, z_{1:k}\right.) q(x_{0:k-1}\left| z_{k,} z_{1:k-1}\right.) \\ &= q(x_k \left| x_{0:k-1}, z_{1:k}\right.) q(x_{0:k-1}\left| z_{1:k-1}\right.) \end{aligned} \tag{2.6}$$

The density $q(x_k \left| x_{0:k-1}, z_{1:k}\right.)$ in the above equation indicates that new set of particles are distributed according to the existing ones representing $\mathrm{q}(x_{0:k-1}\left| z_{k,} z_{1:k-1}\right.)$.

In order to derive the update equation of importance weights the posterior distribution of the state vector is factorized as follows:

$$\begin{aligned} \mathrm{p}(\mathrm{x}_{0:k}\left| z_{1:k}\right.) &= \frac{p(z_k \left| x_{0:k}, z_{1:k-1}\right.) p(x_{0:k}\left| z_{1:k-1}\right.)}{p(z_k \left| z_{1:k-1}\right.)} \\ &\propto p(z_k \left| x_k\right.) p(x_k, x_{0:k-1}\left| z_{1:k-1}\right.) \\ &= p(z_k \left| x_k\right.) p(x_k \left| x_{k-1}\right.) p(x_{0:k-1}\left| z_{1:k-1}\right.) \end{aligned} \tag{2.7}$$

Substituting (2.7) and (2.6) into (2.5), the equation that updates the importance weights is obtained as

$$w_k^i \propto \frac{p(z_k \left| x_k^i \right.) p(x_k^i \left| x_{0:k-1}^i \right.) p(x_{0:k-1}^i \left| z_{1:k-1} \right.)}{q(x_k^i \left| x_{0:k-1}^i, z_{1:k} \right.) q(x_{0:k-1}^i \left| z_{1:k-1} \right.)}$$
$$= w_{k-1}^i \frac{p(z_k \left| x_k^i \right.) p(x_k^i \left| x_{0:k-1}^i \right.)}{q(x_k^i \left| x_{0:k-1}^i, z_{1:k} \right.)} \quad (2.8)$$

It is more common to estimate the posterior filtering density instead of the posterior distribution of the entire history. Therefore, the particle trajectory $\left\{ x_{0:k-1}^i \right\} \Big|_{i=1}^J$ and observation history $z_{1:k-1}$ are discarded and so importance distribution becomes $q(x_k \left| x_{k-1}, z_k \right.)$ [12]. New form of the weight update equation and the posterior filtering density are then respectively given by

$$w_k^i \propto w_{k-1}^i \frac{p(z_k \left| x_k^i \right.) p(x_k^i \left| x_{k-1}^i \right.)}{q(x_k^i \left| x_{k-1}^i, z_k \right.)} \quad (2.9)$$

$$p(x_k \left| z_{1:k} \right.) \approx \sum_{i=1}^N w_k^i \delta(x_k - x_k^i) \quad (2.10)$$

According to new form of importance distribution and (2.9) SIS algorithm updates particles and associated weights recursively as new measurement arrives. The sequential importance sampling PF is summarized in Algorithm 2.1 [14].

Algorithm 2.1: SIS Particle Filter

$$\left[\left\{ x_k^i, w_k^i \right\} \Big|_{i=1}^J \right] = SIS\left[\left\{ \left\{ x_{k-1}^i, w_{k-1}^i \right\} \Big|_{i=1}^J \right\}, z_k \right]$$

- FOR i=1:J

  -Draw $x_k^i \sim q(x_k \left| x_{k-1}^i, z_k \right.)$

  -Assign the particle weight, w_k^i according to (2.9)

- End FOR

In this algorithm as time proceeds weight of the most of the particles will approach small values so; only few particles will have significant contribution to the approximation of the posterior distribution [12]. In other words, the variance of the

weights will increase over time and hence propagation of the particles whose contribution to the posterior density is negligible causes both the waste of the computational sources and divergence of the estimate [30]. This phenomenon is called weight degeneracy problem. To solve this problem two methods are proposed [14]:

i. To choose a good proposal distribution that matches to the true posterior distribution as far as possible.

ii. Resample the particles by discarding the ones with small weights and drawing new ones using the particles which have significant contribution to the filtering density (Resampling Method).

We will discuss the choice of proposal distribution and resampling method and explain their pros and cons in the subsequent subparts.

i. Choice of Proposal Distribution

From the point of view of providing both computational efficiency and accuracy of SIS PF the choice of the importance density is a critical design issue [12]. The optimal proposal distribution that minimizes the variance of the weights has been introduced in [12] and [14] as follows

$$q(x_k \mid x_{k-1}, z_k)_{opt} = p(x_k \mid x_{k-1}, z_k) \tag{2.11}$$

Using Bayes' Theorem (2.11) can be rewritten as

$$q(x_k \mid x_{k-1}, z_k)_{opt} = \frac{p(z_k \mid x_k)\, p(x_k \mid x_{k-1})}{p(z_k \mid x_{k-1})} \tag{2.12}$$

By substituting this optimal importance density into (2.9) the weight of each particle is computed as

$$w_k^i \propto w_{k-1}^i p\left(z_k \mid x_{k-1}^i\right) \tag{2.13}$$

The above equation indicates that the importance weights at time k can be computed before the particles are distributed. As a consequence of using optimal

importance density a large computational effort devoted to propagate the particles which makes little contribution to the $p(x_k|z_{1:k})$ can be eliminated and so computational efficiency is maintained [12]. However, the optimal importance density has two difficulties: (i) sampling from $p(x_k|x_{k-1},z_k)$ and (ii) evaluating the following integral over the new state of the system [14]:

$$p\left(z_k|x_{k-1}^i\right)=\int p\left(z_k|x_k\right)p\left(x_k|x_{k-1}^i\right)dx_k \tag{2.14}$$

Because of these reasons, the optimal importance density can't be used as a solution to the weight degeneracy in all particle filtering problems. Therefore, resampling is suggested to deal with the weight degeneracy problem in a more easy-to-implemented way.

ii. Resampling

It is a simple method to cope with the weight degeneracy problem in Particle Filtering algorithms. The main idea behind the resampling is to discard the particles whose contribution to the estimation of $p(x_k|z_{1:k})$ is low and generate new ones from more promising ones [13]. As a result of accurate reflection of the density by particle cloud, the computational efficiency and accuracy of the estimator is maintained via resampling. Resampling can be used in each iteration of the SIS algorithm after calculating estimates or only at specific time steps when the weight degeneracy is severe. Hence the algorithm needs a function that measures the severity of degeneracy. A good measure of the weight degeneracy is effective sample size, N_{eff} and it is shown in [12], [13] and [14] to be

$$N_{eff}=\frac{1}{1+\mathrm{var}\left(w_k^{*i}\right)} \tag{2.15}$$

Where w_k^{*i} denotes true weight for ith particle.

Since $w_k^{*i}=p\left(x_k^i|z_{1:k}\right)\Big/q\left(x_k^i|x_{k-1}^i,z_{1:k}\right)$, it is not always possible to calculate true importance weights exactly [14], N_{eff} is estimated as follows

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N} \left(w_k^i \right)^2} \tag{2.16}$$

Here w_k^i is the normalized weight of ith particle at time k and computed using (2.9)

There are two extreme cases where estimated effective sample size gets boundary values: (i) if all weights are uniform then; $\hat{N}_{eff} = N_p$ where N_p is the total number of particles and (ii) if all weights except one is zero then; $\hat{N}_{eff} = 1$. Since only one particle makes contribution to reflect the posterior density, the second case indicates the severe degeneracy. Hence small N_{eff} is accepted as a sign of severe degeneracy [12].

When $\hat{N}_{eff} < N_{th}$ where $N_{th} = N_p/2$ or $N_p/3$, resampling is carried out. As a result of resampling all particles have uniform weights, $N_{eff} = N_p$ [13]. A preferred and simple resampling scheme is Systematic Resampling and its execution is described in [12].

Using resampling method, the main steps of generic particle filter is defined in Algorithm 2.2 [12]

Algorithm 2.2: Generic Particle Filter

$$\left[\left\{ x_k^i, w_k^i \right\} \Big|_{i=1}^{J} \right] = PF\left[\left\{ \left\{ x_{k-1}^i, w_{k-1}^i \right\} \Big|_{i=1}^{J} \right\}, z_k \right]$$

- Filtering via SIS (Algorithm 2.1)

  $$\left[\left\{ x_k^i, w_k^i \right\} \Big|_{i=1}^{J} \right] = SIS\left[\left\{ \left\{ x_{k-1}^i, w_{k-1}^i \right\} \Big|_{i=1}^{J} \right\}, z_k \right]$$

- Calculate $\hat{N}_{eff}$ using (2.16)
- IF $\hat{N}_{eff} < N_{th}$

  -Execute resampling
- End IF

Execution of the of a single cycle generic particle filter [13] is depicted in the following figure.

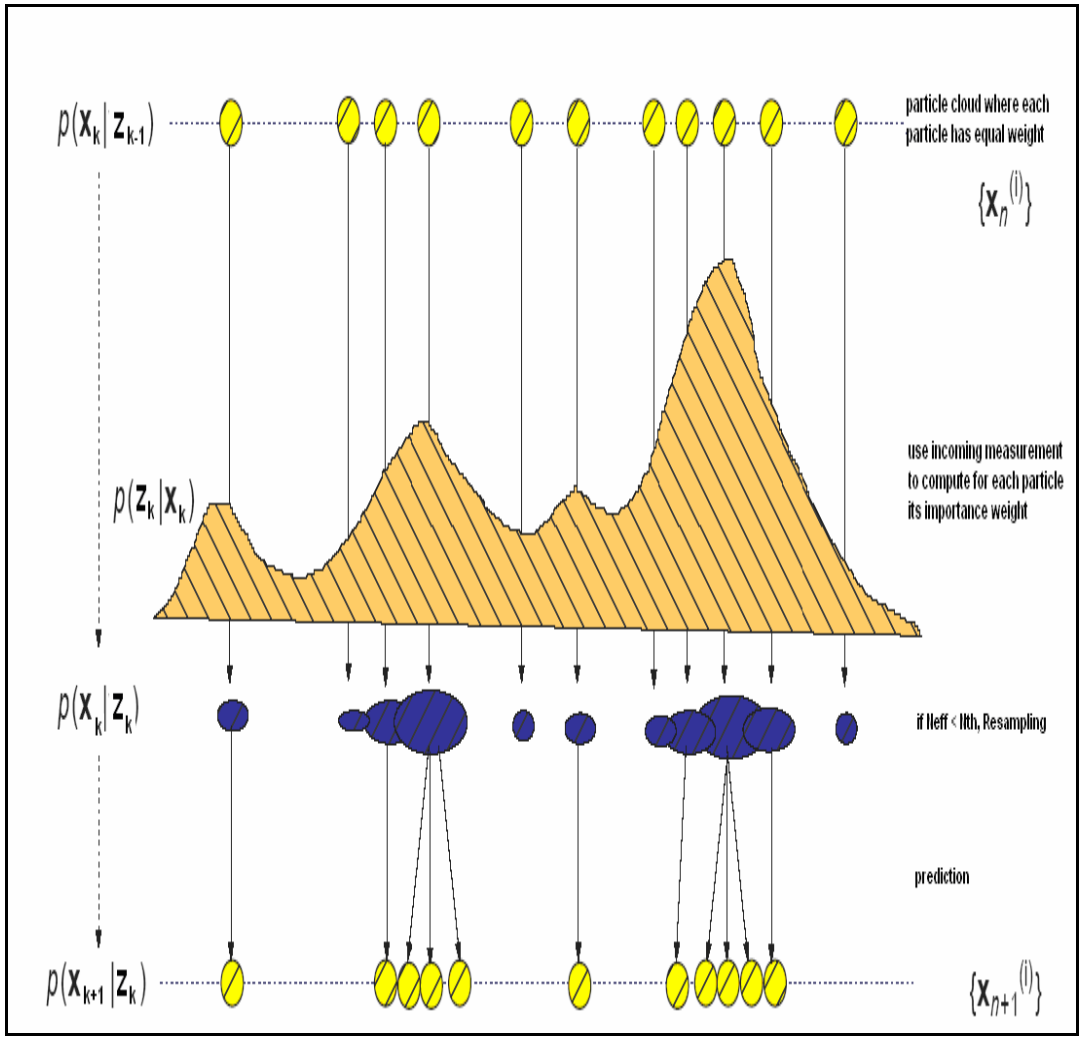

Figure-2.1 Execution of a Single Cycle Generic Particle Filter

2.3.2 Variations of Particle Filters: Examples

The sequential importance sampling algorithm is the basis of the most of the particle filtering algorithms [12]. Here only three of them will be introduced briefly. All of these particle filters differs from SIS algorithm by the choice of importance density and/or handling of weight degeneracy problem [14].

i. Sampling Importance Resampling (SIR): In this version of particle filtering, importance density is chosen to be prior density, $x_k^i \sim p\left(x_k \left| x_{k-1}^i \right.\right)$ and resampling is executed at every time step. Because of selected importance density, importance weights of the particles are computed without knowledge of the last measurement which may cause quick weight degeneracy [30]. To deal with this problem Auxiliary Sampling Importance Algorithm is developed.

ii. Auxiliary Sampling Importance Resampling Algorithm (ASIR): It is a variant of SIR algorithm [12] to deal with the problem mentioned above by choosing an importance density that includes information from the measurement. At the first stage of this filter new particles are predicted from existing ones without process noise and then the particles in the regions of high likelihood are determined. Finally, poor particles are discarded and good ones are augmented with different process noise realizations

iii. Regularized Particle Filter (RPF): It is a solution to the problem called sample impoverishment which arises due to resampling. To deal with this problem it executes regularization step in resampling: it calculates empirical statistic of the particle set (empirical covariance of particle set) before conducting resampling and samples are generated from resampled ones by using this statistic and a kernel density (usually Epanechnikov or Gaussian).

In this work SIR PF is used with the motion estimation that predicts the change in state vector. After making the motion estimation the state of the target is predicted and then particles are sampled using the system model where the system noise variance is a function of the estimation quality. Since particles are drawn according to the prior density function that characterize the state transition model, importance weight of new particles are given by observation likelihoods, $w_k^i \propto p\left(z_k \left| x_{k-1}^i \right.\right)$.

CHAPTER 3

GEOMETRIC SPATIAL TRANSFORMATIONS

In visual object tracking the image of the target deforms geometrically and shifts in the image plane via the translational motion of the target. The implemented algorithm is a correlation based tracker where correlation is translational invariant but requires that template and target image has same scale and orientation. Therefore, the tracker needs a system model that includes the parameters related to the geometric distortion of the target image as well as the kinematic parameters of the target. In this thesis both the translational motion of the target and geometric distortions are modeled by parametric similarity transformation. It is a geometric transformation function that changes the existing spatial relationship between the pixels [31].

This chapter is organized as follows: First, the similarity transformation will be overviewed and then two resampling techniques to perform the geometric transformations between the input and output images will be discussed. Also, some basic geometric transformations and popular interpolation techniques will be introduced. Finally we conclude this chapter by explaining the procedure to crop out the image regions.

3.1. Forward and Inverse Mapping

Similarity transformation includes the following basic operations: uniform scaling of the target region, rotational and translational motion of the target pixels on the image plane. The change in the spatial relationship between pixels with this transformation is established as

$$f(\mathrm{x}; s, \phi, \mathbf{d}) = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \begin{bmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix} \mathbf{x} + \begin{bmatrix} d_x \\ d_y \end{bmatrix} \tag{3.1}$$

Where **x** denotes a pixel coordinate, s is the uniform scale parameter, ϕ is the rotation parameter and $\{d_x, d_y\}$ are the parameters of the translational motion.

Since the pixel coordinates of the output image is given as a function of the input pixel coordinates, the transformation is called forward mapping [33]. Figure-3.1 illustrates the forward mapping where each pixel ***x*** in input image ***f*(*x*)** is sent to its corresponding locations ***x'*** in output image ***g*(*x'*)** via given mapping function ***h*(*x*)** [32].

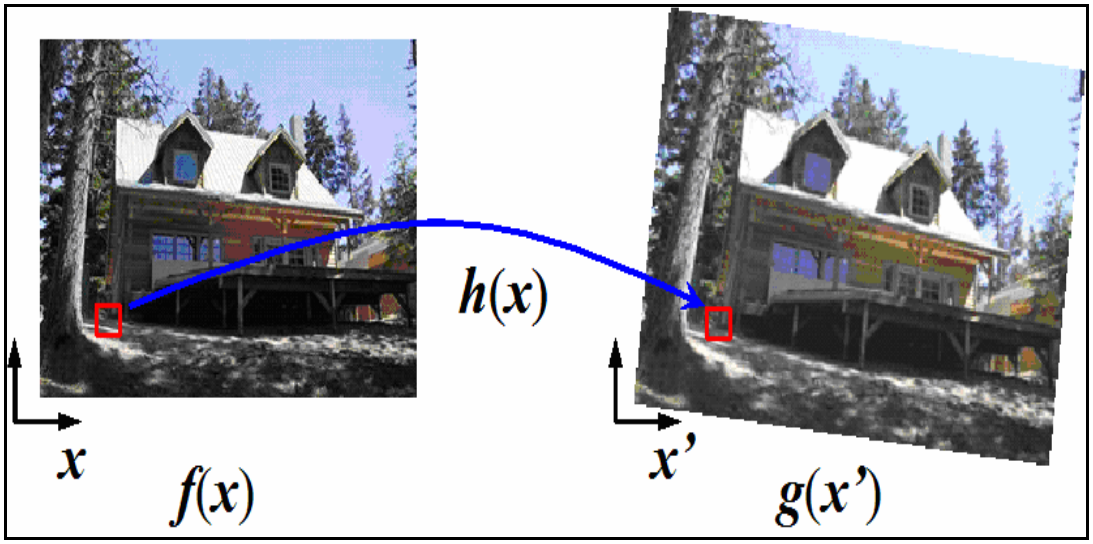

Figure-3.1 Forward Mapping

In geometric image transformations two basic operations are performed: (1) coordinate transformation between input and output images and (2) intensity interpolation. Transformation with forward mapping can't always perform one to one coordinate transformation between input and output images so input pixel intensities can't be assigned directly to the locations in the output image pointed out by the transformation function. Moreover, as a result of forward mapping it is possible that output image contains some pixel locations where a value is never assigned or a value assigned more than once [33]. Another way of performing

spatial transformation which is easier and more efficient than forward mapping inverse mapping where output pixels are mapped onto the input image [31]. However, some output pixel coordinates may be transformed to non-integer values as a result of inverse mapping. To deal with this problem interpolation is utilized to assign the intensity values to these output pixels. The Figure-3.2 illustrates the inverse mapping where each pixel in output image $g(x')$ is obtained from its corresponding location x in input image $f(x)$ via the given inverse mapping function $h^{-1}(x')$ [32].

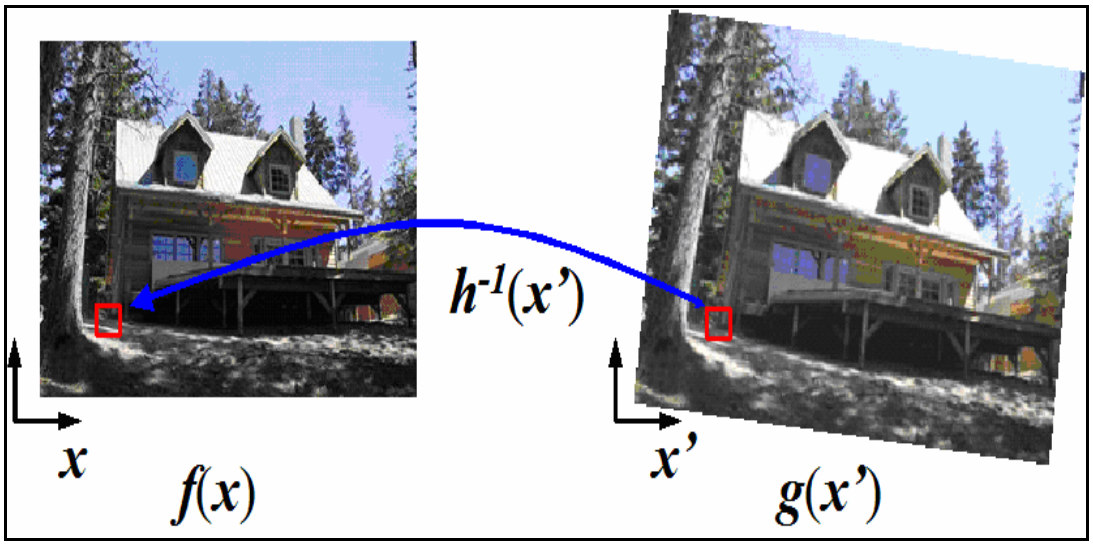

Figure-3.2 Inverse Mapping

Other 2D geometric transformations are called as translation, euclidean, affine and projective. Table-3.1 [32] shows transformation matrix, number of degrees of freedom and representative symbols of these geometric transformations.

Table-3.1 Type of 2D Geometric Transformations

Name	Matrix	# D.O.F	Symbol
Translation	$[I\|d]_{2x3}$	2	
Euclidean	$[R\|d]_{2x3}$	3	
Similarity	$[sR\|d]_{2x3}$	4	
Affine	$[A\|d]$	6	
Projective	$[H]_{3x3}$	8	

The matrix notation used in this table can be written for each geometric transformation explicitly as follows [32]:

- Translation: x’ = x + d
- Rotation: x’ = R x + d
- Similarity: x’ = s R x + d
- Affine: x’ = A x + d
- Perspective: x’ ≈H x

Besides the translation motion, pixel coordinates in all other transformations are given in homogenous coordinates as $x = [\chi \ y \ 1]^T$. In homogenous coordinates 2D vector denoted as $[\chi \ y]^T$ is represented by $[\chi \ y]^T$ where a/w=x and b/w=y.

The relationship between these transformations is shown in Figure-3.3 [34]. As can be seen from the figure some transformations are special cases of more general ones. Here the symbol I denotes the identity transformation that leaves every pixel coordinates unchanged.

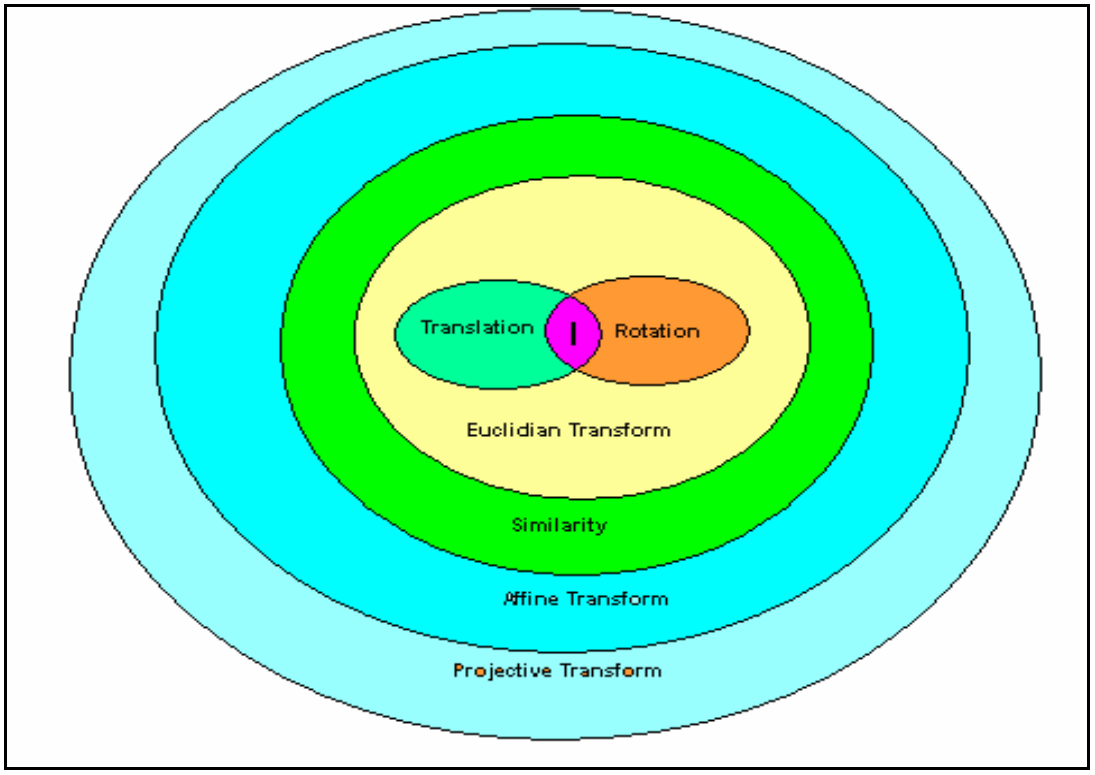

Figure-3.3 Nested Set of Geometric Transformations

3.2. Elementary Geometric Transformations

In this section we will introduce elementary geometric transformations. All of them are represented in homogeneous coordinates. The power of this notation is that a sequence of rotation, scaling and translation can be performed by simply a single matrix such that it is a product of such elementary transformations in a given sequence.

1. Rotation

The pixels are rotated about some fixed point and fixed coordinate system. Usually positive rotations are assumed to be in the counter clockwise direction. The Figure-3.4 shows the rotation through an angle Θ about the z-axis.

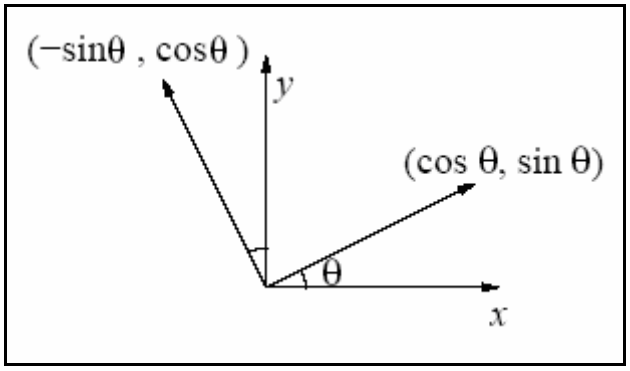

Figure-3.4 Rotation

In homogenous coordinates the rotation is expressed as follows

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\Theta) & -\sin(\Theta) & 0 \\ \sin(\Theta) & \cos(\Theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.2}$$

According to the rotation matrix given in (3.2) the rotation has the following properties:

- It consists of orthonormal vectors describing base vectors in the new coordinate system.
- Since the norm of the column vectors is equal to one, rotation doesn't change scale of the image. It only rotates the image about some fixed point.
- Since the columns of rotation matrix are orthonormal to each other, it is a unitary matrix, that is;

$$R \times R^T = I \text{ and so } R^{-1} = R^T \tag{3.3}$$

2. Translation

It moves pixels to new locations by adding translation parameters $\{d_x\, d_y\}$ to their 2D coordinate values. This operation is expressed by (3x3) homogenous matrix as follows:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.4}$$

Translation is the most basic geometric transformation and has the following properties [34]:

- There exist a unique inverse of the transformation matrix such that it undoes the translation
- It is closed under a series of translational geometric transformation.
- The composition has associative property under this elementary transformation.

3. Scaling

Scaling an image means resizing it by means of multiplying its dimensions by a scalar. In similarity transformation uniform scaling expands or compresses the image using same scalar for each of the axes. In homogenous coordinates this elementary transformation is given by

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.5}$$

However in affine transformation the image is resized by nonuniform scaling such that different scalars are used per dimensions.

3.3. Interpolation Techniques

Here, continuous image signal is approximated from discrete one and then sampling is performed at points where geometric transformation indicates. Three practical approaches are common in usage: (i) nearest neighbor interpolation (ii) bilinear interpolation and (iii) bicubic interpolation. These interpolation techniques are introduced briefly as follows:

Nearest Neighbor Interpolation: In this method calculated coordinates are rounded to nearest integer values and intensity value at this location is assigned to the location in the transformed image [31]. This operation can be performed by

convolving input image with continuous function of one pixel width square pulse and then taking the result of the convolution at the spatial location indicated by transformation. The expression for the function used in nearest neighbor interpolation in 1-D is given by

$$h(x)=\begin{cases}1 & \text{if } |x|\le 0.5\\ 0 & \text{otherwise}\end{cases} \tag{3.6}$$

Since ,as the above equation indicates that, only one pixel (sample) from input image contributes to the determination of the intensity value of a pixel on the transformed image, the quality of the nearest neighbor interpolation is poor but its implementation is simple [31].

Bilinear Interpolation: This interpolation method explores 4 neighbor pixels to estimate the intensity value at given location by the geometric transformation. It calculates a weighted average of the pixel intensities where weights are proportional to the distance between pixel locations and the given location by the geometric transformation. In 1D the interpolation function (interpolant) is given by

$$h(x)=\begin{cases}1-|x| & \text{if } |x|\le 1\\ 0 & \text{otherwise}\end{cases} \tag{3.7}$$

It requires two samples to produce interpolated value. However, in 2D images bilinear interpolation is performed by convolving image with continuous triangle function in (3.7) row by row and column by column [32]. This means that 4 pixels are used in estimation of the pixel intensity value on transformed image.

Using (3.7) and four neighbor pixels bilinear interpolation can be written as follows [32]:

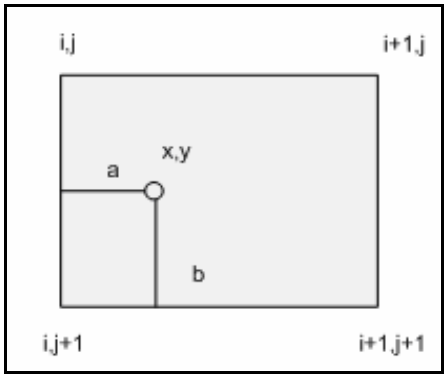

Figure-3.5 Bilinear Interpolation

$$I(x,y)=(1-a)\times b\times I(i,j)+a\times bI(i+1,j)+ (1-a)\times(1-b)\times I(i,j+1)+a\times(1-b)\times I(i+1,j+1) \quad (3.8)$$

Bicubic Interpolation: This is the most complex interpolation in all of three interpolation techniques, but it gives better results by preserving the details than bilinear interpolation. It explores the sixteen nearest neighbor pixels of a given location by geometric transformation. The intensity value at a given point is obtained using the following interpolation kernel (Mexican Hat) [35].

$$h(x)=\begin{cases}\frac{2}{3}-0.5*|x|^2(2-x) & 0\le|x|\le 1\\ \frac{1}{6}(2-|x|)^3 & 1\le|x|\le 2\\ 0 & 2\le|x|\end{cases} \quad (3.9)$$

3.4 Image Warping

In the preceding sections the parameters of the geometric transformation are all unknown and thus these parameters must be estimated in order to crop out the image regions. However, only available information is that output and input images to estimate these parameters.

In this thesis before guiding the particles in the state space it is needed to predict the target motion. It is equivalent to estimate the transformation parameters that maps the region cropped out of the current frame to the target region in the previous frame. The details of this estimation process will be given in Chapter 5. However, this estimation is an area based or correlation-like feature matching process where input image is registered against reference image by means of estimated parameters [36]. To make this estimation robust to noise and illumination changes the image regions are cropped as follows [5]:

i First of all, using the translational motion parameters the center of the image region is determined and then using other parameters of the similarity transformation spatial relationship between the pixels within the image region are redefined.

ii Then, zero-mean unit variance normalization is performed by means of statistics extracted from the pixels within the target region in order to decrease the effect of the photometric variations (illumination changes) to the result of the estimation [21].

$$I_{norm}\left(f(\mathrm{x};s,\phi,\mathbf{d}),k\right)=\frac{I\left(f(\mathrm{x};s,\phi,\mathbf{d}),k\right)-m}{\sigma} \tag{3.10}$$

Where m and σ are respectively the mean and standard deviation values of the pixels and given by

$$m=\frac{1}{d}\sum_{x\varepsilon r} I\left(f(\mathrm{x};s,\phi,\mathbf{d}),k\right) \qquad \sigma^2=\frac{1}{d}\sum_{x\varepsilon r}\left(I\left(f(\mathrm{x};s,\phi,\mathbf{d}),k\right)-m\right)^2 \tag{3.11}$$

Here d denotes the number of pixels within the image region.

CHAPTER 4

ADAPTIVE APPEARANCE MODEL WITH EXPECTATION MAXIMIZATION (EM) ALGORITHM

Visual trackers usually include an appearance model of the target if it utilizes the motion estimation to track the target. There are two common approaches to estimate the motion. First approach uses a fixed template of the target whereas the second one uses a recently observed target image in the motion estimation procedure. When they are compared in terms of performance, the first approach can be reliable over short durations, but its performance degrades under appearance changes [4]. However, the second one has advantages of adapting to rapid change in the target appearance, but the adaptation can cause the models to drift away from the target characteristics because of clutter or image noise [4], [37]. By combining these two approaches in an appearance model, the model will adapts the target template to the changes in appearance and meanwhile it becomes robust to the occlusion and outliers. In this way a reliable visual tracker utilizing an adaptive appearance model can be achieved.

In this chapter an adaptive appearance model that is a mixture of two Gaussian distributions will be described. Also, Expectation Maximization (EM) algorithm that updates the parameters of this appearance model will be discussed in detail.

4.1 Adaptive Appearance Model

The appearance model used in this thesis is composed of two components: (i) stable component to capture the stable and slowly varying features of the target (ii) transient component to capture the rapid changes in target appearance. These two components explain the observed pixel intensity values in terms of stable and transient templates of the appearance model.

Since a pixel intensity value at a time is explained by these two components, which are assumed to be Gaussian distributions, the parameters characterizing them must be computed in real time. The appearance model at discrete time k is represented as $A_k = \{S_k, W_k\}$ where S_k and W_k denote the stable and the transient components respectively. Their distributions for ith pixel are given by

$$S_k(i) : N(I_k(i); \mu_{s,k}(i), \sigma^2_{s,k}(i)) \tag{4.1}$$

$$W_k(i) : N(I_k(i); I_{k-1}(i), \sigma^2_w) \tag{4.2}$$

Here $I_k(i)$ denotes the intensity value of the ith pixel according to the components of the target appearance model at time k and $\mu_{s,k}(i)$ and $\sigma^2_{s,k}(i)$ are slowly time varying functions specifying the mean and the variance of the ith pixel. The parameters that characterize the transient distribution are previous intensity value of the ith pixel $I_{k-1}(i)$ and a constant variance σ^2_w.

Until a significant time precedes the appearance model can't capture the stable or slowly varying features of the target. During this period stable component becomes unreliable and transient component makes possible the visual tracker to track a given image region [37]. Also, it is more appropriate to track the target according to transient component of the model after rapid changes in target appearance due to target motion [37].

Finally, these two distributions are used to compute the likelihood of the hypothesized target region specified by the state vector θ_k as follows [5]

$$p(I_k | a_k) = \prod_{i=1}^{d} \sum_{j=s,w} m_{j,k}(i) N(I_k(i); \mu_{j,k}(i), \sigma^2_{j,k}(i)) \tag{4.3}$$

Where $p(I_k | a_k)$ is observation likelihood function, which says how well the target image I_k specified by θ_k matches to the target appearance model, d is the number of pixels within the target region, $m_{j,k}$ is mixing probability, $\mu_{j,k}$ and $\sigma^2_{j,k}$ are respectively the mean and the variance of distributions at discrete time k. All of these appearance model parameters are denoted as a_k in (4.3). After the optimum

target region is determined, current target appearance model is updated using the pixel intensity values within that region.

4.2 Model Update via EM and Initialization

To update the current appearance model we have to estimate the parameters, namely, the mean and the variance of the model components $\{\mu_{j,k}, \sigma^2_{j,k}\}$ and mixture probabilities $m_{j,k}$. This estimation process adapts the parameters of the distributions in a recursive manner. Furthermore, it is assumed that the past observations affect the new values of the parameters under an exponential envelope [5]. Thus, the contributions of observations from the present to the past are downweighted exponentially. In [5] exponential envelope which is the envelope of the log-likelihood with forgetting factor α is given as

$$S_k(n) = \alpha e^{-\frac{(k-n)}{\tau}} \quad \text{for } n \le k \tag{4.4}$$

Where $\tau = n_s / \log(2)$ and n_s is half-life of the envelope in frames which is the time required for the decaying quantity to fall to one half of its initial value. The forgetting factor α enables the sum of all exponential weights to be equal to 1 and its calculation is given by

$$\begin{aligned} \sum_{n=-\infty}^{k} S_k(n) &= \sum_{n=-\infty}^{k} \alpha e^{-\frac{(k-n)}{\tau}} = 1 \\ &= \alpha(e^0 + e^{-1/\tau} + e^{-2/\tau} + \ldots + 0) = 1 \\ &= \alpha e^{-1/\tau}(e^{1/\tau} + e^0 + e^{-1/\tau} + e^{-2/\tau} + \ldots + 0) = 1 \\ &= \alpha e^{-1/\tau}(e^{1/\tau} + \frac{1}{\alpha}) = 1 \end{aligned} \tag{4.5}$$

Thus the forgetting factor becomes equivalent to

$$\alpha + e^{-1/\tau} = 1 \Leftrightarrow \alpha = 1 - e^{-1/\tau} \tag{4.6}$$

To calculate the mixing probabilities and parameters defining the Gaussian distributions (model components), steps of the expectation maximization (EM) algorithm is applied. First of all, ownership probabilities of the distributions for each pixel have to be calculated as (E-step) [4]:

$$q_{j,k}\left(I_n(i)\right)\Big|_{j=s,w} = \frac{m_{j,k}\left(i\right)N(I_n(i);\mu_{j,k}(i),\sigma^2_{j,k}(i))}{\sum_{l=s,w} m_{l,k}\left(i\right)N(I_n(i);\mu_{l,k}(i),\sigma^2_{l,k}(i))} \qquad \begin{matrix} i=1,...,d \\ n\le k \end{matrix} \tag{4.7}$$

The above equation indicates the ownership probability of the jth distribution. In other words it is the probability that the datum $I_n(i)$ is owned by the jth model component at discrete time k. Here, EM algorithm makes soft assignments by computing the ownership probabilities such that they become lower bounds to the true posterior assignment probabilities [38].

Next, M-step is performed by taking the partial derivatives of the log-likelihood function w.r.t the unknown model parameters denoted as $a_{j,k} = \left\{m_{j,k}, \mu_{j,k}, \sigma^2_{j,k}\right\}\Big|_{j=S,W}$ and so the unknown parameters are updated. Another point of view to understand the meaning of the steps in EM algorithm is that Once the current target region becomes available, in E-step the algorithm determines the lower bound to the posterior assignment probabilities of the pixels to the model components and then in M-step it maximizes this lower bounds by computing maximum likelihood values of the model parameters [38]. The log-likelihood with exponential envelope is given by

$$L(a_k) = \sum_{n=-\infty}^{k} S_k(n)\log(p(I_n|a_k)) \tag{4.8}$$

To compute the optimum mixing probability of each distribution, the constraint $\sum_{j=s,w} m_{j,k} = 1$ is included into the L function with Lagrange multiplier λ. Note that this is due to the fact that $\sum_{j=s,w} m_{j,k} = \sum_{j=s,w} q_{j,k}\left(I_n(i)\right) = 1$

$$L(a_k) = \sum_{n=-\infty}^{k} S_k(n)\log(p(I_n|a_k)) + (1-\sum_{j=1}^{2} m_{j,k})\lambda \tag{4.9}$$

Take the derivative of the above equation w.r.t $m_{j,k}$ and then set it to zero

$$\frac{\partial L(a_k)}{\partial \mathrm{m}_{\mathrm{j,k}}(i)} = \sum_{n=-\infty}^{k} \frac{\partial S_k(n)\log(p(I_n|a_k))}{\partial \mathrm{m}_{\mathrm{j,k}}(i)} + \frac{\partial}{\partial \mathrm{m}_{\mathrm{j,k}}(i)}(1-\sum_{j=1}^{2}\mathrm{m}_{\mathrm{j,k}}(i))\lambda = 0$$
$$= \sum_{n=-\infty}^{k} S_k(n)\frac{N(I_n(i);\mu_{j,k},\sigma_{j,k}^2)}{\sum_{j=s,w} m_{j,k}N(I_n(i);\mu_{j,k},\sigma_{j,k}^2)} - \lambda = 0 \quad (4.10)$$

By multiply the numerator and denominator by $m_{j,k}$ (4.10) can be rewritten as follows

$$\frac{1}{m_{j,k}(i)}\sum_{n=-\infty}^{k} S_k(n)\frac{m_{j,k}(i)N(I_n(i);\mu_{j,k}(i),\sigma_{j,k}^2(i))}{\sum_{j=s,w} m_{j,k}(i)N(I_n(i);\mu_{j,k}(i),\sigma_{j,k}^2(i))} - \lambda = 0 \quad (4.11)$$

Substituting (4.7) into (4.11) the mixing probability $m_{j,k}$ is computed as

$$m_{j,k}(i) = \frac{1}{\lambda}\sum_{n=-\infty}^{k} S_k(n)q_{j,k}(I_n(i)) \quad (4.12)$$

We know that the sum of mixing probabilities of model components for each pixel is equal to 1 then, λ becomes the normalization constant.

In a similar way optimum mean and variance of the stable model for each datum can be computed by taking the derivative of the log-likelihood function w.r.t these parameters.

The equation that gives the optimum mean is derived as follows:

$$\frac{\partial L(a_k)}{\partial \mu_{s,k}(i)} = \sum_{n=-\infty}^{k} S_k(n)\frac{\partial \log(p(I_n|a_k))}{\partial \mu_{s,k}(i)}$$
$$= \sum_{n=-\infty}^{k} S_k(n)\frac{\frac{\partial p(I_n(i)|a_k)}{\partial \mu_{s,k}(i)}}{p(I_n(i)|a_k)} \quad (4.13)$$

Where $p(I_n(i)|a_k)$ is the likelihood of observing intensity of ith pixel from the mixture of two components and expressed as

$$p(I_n(i)|a_k) = \sum_{j=s,w} m_{j,k}(i)N(I_n(i);\mu_{j,k}(i),\sigma_{j,k}^2(i)) \quad (4.14)$$

Making the use of (4.14), (4.13) is rewritten as

$$\sum_{n=-\infty}^{k} S_k(n) \frac{m_{s,k}(i) \frac{\partial N(I_n(i); \mu_{s,k}(i), \sigma_{s,k}^2(i))}{\partial \mu_{s,k}(i)}}{\sum_{j=s,w} m_{j,k}(i) N(I_n(i); \mu_{j,k}(i), \sigma_{j,k}^2(i))} = 0 \tag{4.15}$$

Here, the derivative of the Gaussian distribution w.r.t $\mu_{s,k}(i)$ is expressed as

$$\frac{\partial N(I_n(i); \mu_{s,k}(i), \sigma_{s,k}^2(i))}{\partial \mu_{s,k}(i)} = \frac{(I_n(i) - \mu_{s,t}(i)) N(I_n(i); \mu_{s,k}(i), \sigma_{s,k}^2(i))}{\sigma_{s,k}^2(i)} \tag{4.16}$$

Substituting (4.16) into (4.15), the equation that gives the optimum mean value of the stable distribution for ith pixel becomes

$$\frac{1}{\sigma_{s,k}(i)} \sum_{n=-\infty}^{k} S_k(n) \frac{m_{s,k}(i)(I_n(i) - \mu_{s,k}(i)) N(I_n(i); \mu_{s,k}(i), \sigma_{s,k}^2(i))}{\sum_{j=s,w} m_{j,k}(i) N(I_n(i); \mu_{j,k}(i), \sigma_{j,k}^2(i))} = 0$$

$$\mu_{s,k}(i) = \frac{\sum_{n=-\infty}^{k} S_k(n) \frac{m_{s,k}(i) I_k(i) N(I_n(i); \mu_{s,k}(i), \sigma_{s,k}^2(i))}{\sum_{j=s,w} m_{j,k}(i) N(I_n(i); \mu_{j,k}(i), \sigma_{j,k}^2(i))}}{\sum_{n=-\infty}^{k} S_k(n) \frac{m_{s,k}(i) N(I_n(i); \mu_{s,k}(i), \sigma_{s,k}^2(i))}{\sum_{j=s,w} m_{j,n}(i) N(I_n(i); \mu_{j,k}(i), \sigma_{j,k}^2(i))}} \tag{4.17}$$

$$= \frac{M_{s,k}^1(i)}{M_{s,k}^0(i)}$$

Here $M_{s,k}^1$ denotes the 1st order moment of the stable distribution and $M_{s,k}^0$ corresponds to mixing probability of the stable model given by (4.12) [5].

Finally, the optimum variance of the stable distribution for each pixel is derived as follows:

First of all using the optimum mixing probability and mean values the derivative of the log-likelihood w.r.t $\sigma_{s,k}^2(i)$ is taken as.

$$\frac{\partial L(a_k)}{\partial \sigma_{s,k}^2(i)} = \sum_{n=-\infty}^{k} S_k(n) \frac{\partial \log(p(I_n(i)|a_k))}{\partial \sigma_{s,k}^2(i)} = 0 \tag{4.18}$$

Where the derivative of the log likelihood function for ith pixel is given by

$$\frac{\partial N(I_n(i);\mu_{s,k}(i),\sigma^2_{s,k}(i))}{\partial \sigma^2_{s,k}(i)} = \frac{m_{s,k}(i)\left[(I_n(i)-\mu_{s,k}(i))^2-\sigma^2_{s,k}(i)\right]N(I_n(i);\mu_{s,k}(i),\sigma^2_{s,k}(i))}{\sum_{j=s,w} m_{j,k}(i)N(I_n(i);\mu_{j,k}(i),\sigma^2_{j,k}(i))} \tag{4.19}$$

Substituting (4.19) into (4.18) the following equation is obtained

$$\frac{\partial L(a_k)}{\partial \sigma^2_{s,k}(i)} = \sum_{n=-\infty}^{k} S_k(n) \frac{m_{s,k}(i)\left[(I_n(i)-\mu_{s,k}(i))^2-\sigma^2_{s,k}(i)\right]N(I_n(i);\mu_{s,k}(i),\sigma^2_{s,k}(i))}{\sum_{j=s,w} m_{j,k}(i)N(I_n(i);\mu_{j,k}(i),\sigma^2_{j,k}(i))} \tag{4.20}$$

Then using (4.17) the above equation is rewritten as

$$\frac{\partial L(a_k)}{\partial \sigma^2_{s,k}(i)} = \sum_{n=-\infty}^{k} S_k(n) \frac{m_{s,k}(i)\left[I_n(i)^2-\sigma^2_{s,k}(i)\right]N(I_n(i);\mu_{s,k}(i),\sigma^2_{s,k}(i))}{\sum_{j=s,w} m_{j,k}(i)N(I_n(i);\mu_{j,k}(i),\sigma^2_{j,k}(i))} - {\mu_{s,k}}^2(i)M^0_{s,k}(i) \tag{4.21}$$

Thus, the equation giving optimum variance is expressed as

$$\sigma^2_{s,k}(i)M^0_{s,k}(i) = \sum_{n=-\infty}^{k} S_k(n) \frac{m_{s,k}(i)I_n(i)^2 N(I_n(i);\mu_{s,k}(i),\sigma^2_{s,k}(i))}{\sum_{j=s,w} m_{j,k}(i)N(I_k(i);\mu_{j,k}(i),\sigma^2_{j,k}(i))} - {\mu_{s,k}}^2(i)M^0_{s,k}(i)$$

and (4.22)

$$\sigma^2_{s,k}(i) = \frac{M^2_{s,k}(i)}{M^0_{s,k}(i)} - {\mu_{s,k}}^2(i)$$

Here, $M^2_{s,k}$ denotes the second order moment of the stable distribution.

Note that the optimum mean and variance of the stable component depends on the first two moments of the data. Using (4.17) and (4.22) the equation that computes the mth order data moments can be given by

$$M^m_{n,k}(i) = \sum_{n=-\infty}^{k} S_k(n) I_n(i)^m q_{s,k}(I_n(i)) \tag{4.23}$$

Here, the mth order data moment is a function of the ownership probability, which must be calculated for all data observations up to current time. It is a time consuming operation and makes the algorithm to be impractical for real time applications [4]. Therefore, the computation of the data moment is approximated as follows [4], [5]

$$M^m_{s,k}(i) = \alpha I_k(i)^m q_{k,s}(I_k(i)) + (1-\alpha)\sum_{n=-\infty}^{k-1} S_{k-1}(n) I_n(i)^m q_{s,k}(I_n(i)) \tag{4.24}$$

In the above equation instead of recalculating the ownership probabilities of all observations $q_{s,k}\left(I_n(i)\right)$,old ones $q_{s,n}\left(I_n(i)\right)$ are used. Similarly, the mixing probability of the distributions given by (4.12) is approximated in a same manner as in [4], [5]

$$m_{j,k}(i) = \alpha q_{j,k}\left(I_k(i)\right) + (1-\alpha) m_{j,k-1}(i) \tag{4.25}$$

Since the ownership probabilities of the past observations aren't recalculated, the accuracy of the algorithm becomes poor when the target is under rapid movement [4]. The indicators of this case are low mixing probability and high variance for the stable distribution. However, the accuracy of the algorithm is quite well when the motion of the target is slow and so the parameters of the distributions don't change rapidly [4]

EM algorithm start with initial guess for all parameters, mixing probabilities, mean and variances that characterize the distributions [38]. In this thesis, these parameters are initialized as follows

$$m_{s,0} = 0.15 \; and \; m_{w,0} = 0.85 \tag{4.26}$$

Here, the mixing probability of the stable component is set to a smaller value than the transient one. The reason is that the stable component is unreliable until a significant time passes to capture the stable or slowly varying features of the target [4]. Also, the constraint for mixing probabilities at each time index k is given by

$$\sum_{j=s,w} m_{j,k} = 1.0 \tag{4.27}$$

Furthermore, the mean of the each distribution initially holds the normalized pixel intensity value and first two moments of the stable component is set as [5]

$$M_{s,0}^{1}\left(i\right) = m_{s,0} I_0(i) \tag{4.28}$$

$$M_{s,0}^{2}\left(i\right) = m_{s,0}\sigma_{s,0}^{2} + I_0\left(i\right)^2 \tag{4.29}$$

Where $\mathrm{I}_0\left(\mathrm{i}\right)$ denotes a normalized pixel intensity within the target region at discrete time index k= 0 and the variance of these components are initially assigned to the following values

$$\sigma_w = 0.75 \; and \; \sigma_{s,0} = \sigma_w / 5 \tag{4.30}$$

At each iteration previous parameter estimations become an initial guess to the recursive parameter estimation procedure. To avoid singular case, where one model becomes sufficient to represent the distribution of the observation, some constraints are taken against computed values of these parameters via EM algorithm. In this sense the following limits are defined.

Lower bound of any mixing probability is limited as

$$\min m_{j,k} \Big|_{j=s,w} = 0.1 \tag{4.31}$$

Also, the variance of the stable component is never allowed to be greater than the variance of the transient one [4]. Thus, instead of using a fixed variance for the transient component of the model, it is computed as a function of the variance of the stable component as

$$\sigma_{w,k}^2 = 5\sigma_{s,k}^2 \tag{4.32}$$

An adaptive online appearance model consisting of these two components gives the algorithm a large benefit. Because pose and illumination changes affect the target appearance and so motion estimation that does not utilize an adaptive target appearance model can deviate from the optimum estimate and it makes the performance of the algorithm poor. However, adaptive online appearance model allows coping with these difficulties. Thus, accurate positioning of the target is achieved [37].

To conclude this chapter a flowchart of adaptive appearance model is given in Figure-4.1.

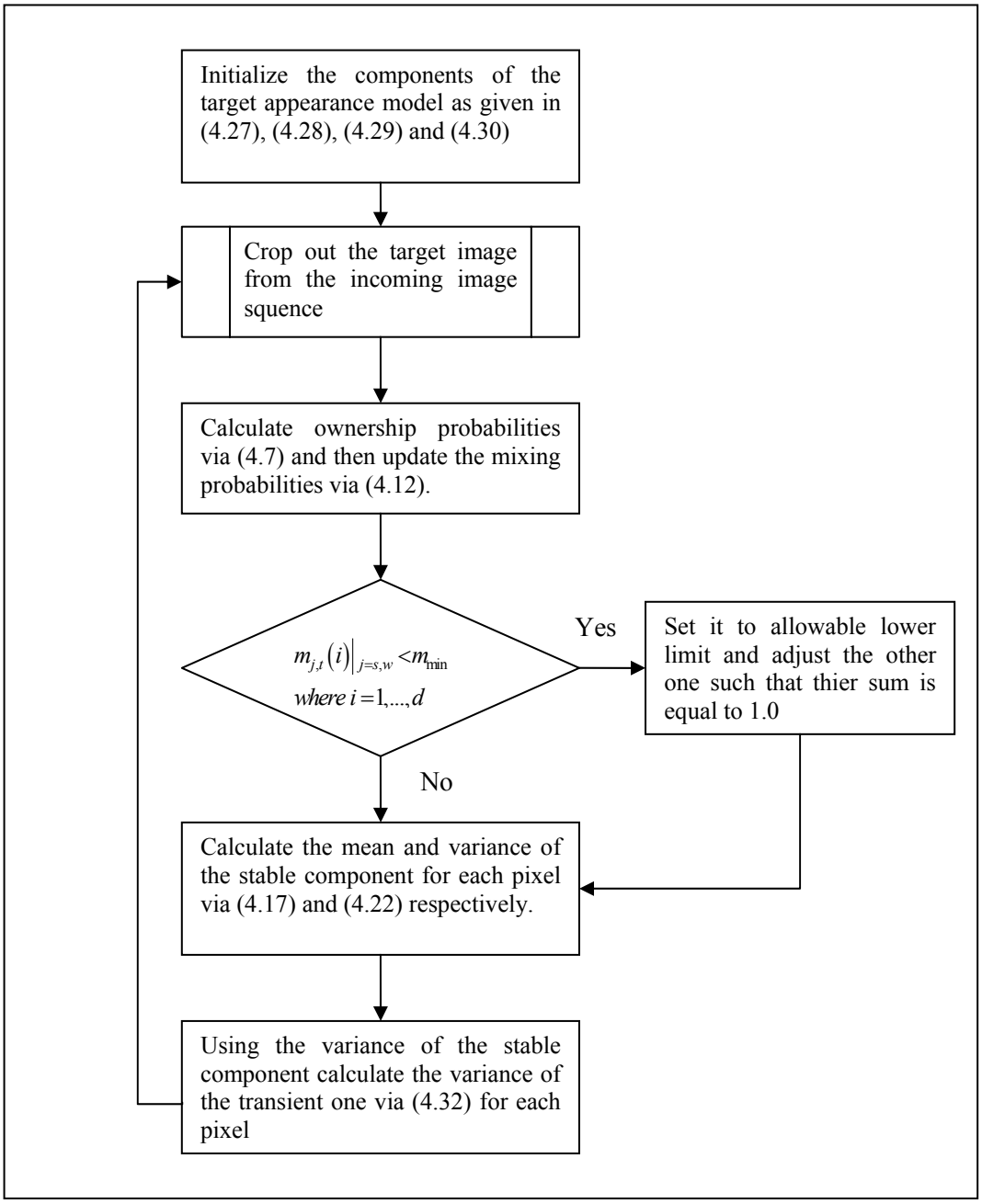

Figure-4.1 Flowchart of Adaptive Appearance Model via EM Algorithm

CHAPTER 5

MOTION ESTIMATION

In this thesis to increase the accuracy of the visual tracker state transition model is made adaptive to the changes in kinematic behaviors of the target. Adaptation of the state transition model is provided by means of adaptive velocity estimation. It is based on image registration process where the reference template is matched to the most probable region in the incoming frame by making predictions about the state variables.

In this chapter the calculation of the adaptive velocity will be expressed in detail and then the limitations will be discussed. Furthermore, a method to measure the quality of the estimation given in [5] will be introduced and finally the last section makes a brief review of robust estimators.

5.1 Iterative SSD Based Motion Estimation

The components of the motion estimation are: (i) initial guess to the motion parameters, (ii) geometric transformation (image warping) and (iii) refinement of the estimated motion parameters in an iterative manner.

Since in the template based visual tracking the continuity of the motion is assumed, initial guess is simply equal to the estimated motion parameters from the previous frame [4]. In motion estimation the aim is to update these parameters by minimizing the distinction between the target region and a reference target template thus; the movement of the target patch from one frame to next is obtained by means of updated motion parameters [19].

The motion estimators which calculate the updates to the predicted motion parameters vary according to the motion model. In all cases, however the estimation

involves a cost function like Sum of Square Difference (SSD) [22]. SSD compares a feature template with the image regions cropped out of the incoming frame and so it registers the template to the most reasonable image region [22]. SSD is a correlation method and expressed as follows:

$$SSD = \sum_{n=1}^{N} \left[I(\chi_n, k_o) - I(f(\chi_n; \Theta_k), k) \right]^2 \quad (5.1)$$

Where $\chi_n = \begin{bmatrix} x_n & y_n \end{bmatrix}^T \; n = 1,...,N$ are the pixel coordinates within the target region, I(.) gives brightness values of these pixels and f(.) is a warping function that transforms the pixel locations within the target region to the corresponding locations in the reference template according to the motion parameters denoted as Θ_k .

SSD gives the quality of the parameter estimation. Therefore, the smaller SSD means the better estimation of the motion parameters. To minimize SSD an iterative optimization technique is needed by searching on SSD surface thus; the improved motion parameter estimation can be achieved [39]

A standard numerical approach to solve such a problem is to apply Newton's method to find the update to motion parameters Θ_k . Since the current observation $I(f(\chi; \Theta_k), k)$ is a nonlinear function of the motion parameters, the performance surface (SSD) isn't a quadratic function of Θ_k . Therefore, Gauss-Newton method is utilized and so the observation function is linearized around the previously estimated motion parameters and a quadratic performance surface of Θ_k is obtained.

For later developments, it is convenient to write (5.1) in matrix notation. For this reason a vector composed of the target pixel intensities is obtained via scanning the target region row by row after it is aligned with reference target template by means of warping function. Then, the image of the rectified target region is expressed in this form as follows:

$$I\left(f(\chi,\Theta_k),k\right)=\begin{bmatrix} I\left(f(\chi_1,\Theta_k),k\right) \\ I\left(f(\chi_2,\Theta_k),k\right) \\ \dots \\ I\left(f(\chi_N,\Theta_k),k\right) \end{bmatrix} \tag{5.2}$$

Where $\chi_n \quad n=1,...,N$ denotes the locations within the target region. Using this vector notation (5.1) can be rewritten as

$$SSD=\left\|I\left(f(\chi,\Theta_k),k\right)-I\left(\chi,k_0\right)\right\|^2 \tag{5.3}$$

The derivation of the adaptive motion vector (v_k) using the above equation is then given as follows:

If the magnitudes of the components of v_k are small, it is possible to linearize current observation $I\left(f(\chi,\Theta_t),k\right)$ around the previously estimated motion parameters $\hat{\Theta}_{k-1}$ via the first order Taylor series expansion in order to obtain a quadratic cost function of Θ_k.

$$I\left(f(\chi,\Theta_k),k\right)\approx I\left(f(\chi,\hat{\Theta}_{k-1}),k\right)+\nabla_{\Theta_k} I\left(f(\chi,\Theta_k),k\right)\Big|_{\Theta_k=\hat{\Theta}_{k-1}}\left[\Theta_k-\hat{\Theta}_{k-1}\right] \tag{5.4}$$

Here, the term $\Theta_k-\hat{\Theta}_{k-1}$ is called adaptive motion vector. By substituting (5.4) into (5.3) the quadratic cost function of the unknown quantity v_k is achieved as

$$\begin{aligned} SSD &\approx \left\| I\left(f(\chi,\hat{\Theta}_{k-1}),k\right)+\nabla_{\Theta_k} I\left(f(\chi,\Theta_k),k\right)\Big|_{\Theta_k=\hat{\Theta}_{k-1}} v_k - I\left(\chi,k_0\right)\right\|^2 \\ &\approx \left\| Av_k-b\right\|^2 \end{aligned} \tag{5.5}$$

Where A denotes the Jacobian matrix $\nabla_{\Theta_k} I\left(f(\chi,\Theta_k),k\right)$ evaluated at the previous estimate of the motion parameters and b denotes the difference between image patches.

To compute the update for the initial estimates of the motion parameters the derivative of SSD w.r.t v_k is taken and then assigned to zero [19].

$$\frac{\partial SSD}{\partial v_k}=0 \tag{5.6}$$

Where the derivative can be written as

$$\frac{\partial SSD}{\partial v_k} = \left(\nabla_{\Theta_k} I\left(f(\chi,\Theta_k),k\right)\Big|_{\Theta_k=\hat{\Theta}_{k-1}}\right)^T \left(\left(I\left(f(\chi,\hat{\Theta}_{k-1}),k\right)+\nabla_{\Theta_k} I\left(f(\chi,\Theta_k),k\right)\Big|_{\Theta_k=\hat{\Theta}_{k-1}} v_k\right) - I(\chi,k_0)\right) \qquad (5.7)$$
$$= A^T\left(Av_k - b\right)$$

In (5.7) the Jacobian matrix $\nabla_{\Theta_k} I\left(f(\chi,\Theta_k),k\right)\Big|_{\Theta_k=\hat{\Theta}_{k-1}}$ can be expressed for one pixel location $\chi_n = \begin{bmatrix} x_n & y_n \end{bmatrix}^T$ using the chain rule as follows [1]:

$$\nabla_{\Theta_k} I\left\{f(\chi_n,\Theta_k),k\right\} = \nabla_f I\left\{f(\chi_n,\Theta_k),k\right\}^T f_{\Theta_k}\left(\chi_n,\Theta_k\right) \qquad (5.8)$$

Where the first term on the left hand side of the above equation is approximated as

$$\nabla_f I\left\{f(\chi_n,\Theta_k),k\right\} \equiv \left[\frac{\partial}{\partial x} I\left\{f(\chi_n,\Theta_k),k\right\} \quad \frac{\partial}{\partial y} I\left\{f(\chi_n,\Theta_k),k\right\}\right]^T \qquad (5.9)$$

The above equation indicates that the partial derivatives of the observation w.r.t warping function corresponds to a vector that consists of spatial partial derivatives of the current observation at $f(\chi_n,\Theta_k)$ and also the second term $f_{\Theta_k}\left(\chi_n,\Theta_k\right)$ in (5.8) is the (2x6) jacobian matrix of the warping function w.r.t the motion parameters at $\Theta_k = \hat{\Theta}_{k-1}$. In order to derive this jacobian matrix the equation describing the motion model is required. The motion parameters (parameters of the similarity transformation) are expressed as follows:

$$\Theta_k = \begin{bmatrix} a_{11} & a_{12} & a_{21} & a_{22} & d_x & d_y \end{bmatrix}^T \qquad (5.10)$$

By using these parameters the following equation describing this geometric transformation is established:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_{11}x + a_{12}y + d_x \\ a_{21}x + a_{22}y + d_y \end{bmatrix} \qquad (5.11)$$

The above equation indicates that location of the ith pixel denoted as (x,y) is transformed to the location (x′,y′) using warping function with motion parameters.

Taking the partial derivatives of (5.11) w.r.t motion parameters, the jacobian of the transformation matrix can be expressed as

$$f_{\Theta_k}\left(\chi,\Theta_k\right) = \begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{bmatrix} \qquad (5.12)$$

By making the use of (5.12) and spatial derivatives of the observation the Jacobian matrix in (5.8) can be obtained as

$$\begin{aligned}\nabla_{\Theta_k} I\{f(\chi_n,\Theta_k),k\} &= \nabla_f I\{f(\chi_n,\Theta_k),k\}^T f_{\Theta_k}(\chi_n,\Theta_k) \\ &= \left[\frac{\partial I\{f(\chi_n,\Theta_k),k\}}{\partial x} \;\; \frac{\partial I\{f(\chi_n,\Theta_k),k\}}{\partial y}\right]\begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{bmatrix}\end{aligned} \tag{5.13}$$

Writing the above equation for all target pixels the jacobian of the current observation w.r.t the motion parameters is derived as follows [1]:

$$\nabla_{\Theta_k} I\left(f(\chi,\Theta_k),k\right) = \begin{bmatrix} \nabla_f I\{f(\chi_1,\Theta_k),k\}^T f_{\Theta_k}(\chi_1,\Theta_k) \\ \nabla_f I\{f(\chi_2,\Theta_k),k\}^T f_{\Theta_k}(\chi_2,\Theta_k) \\ \vdots \\ \nabla_f I\{f(\chi_N,\Theta_k),k\}^T f_{\Theta_k}(\chi_N,\Theta_k) \end{bmatrix} \tag{5.14}$$

However, in this thesis instead of calculating the Jacobian matrix as given in (5.14) we have used the method mentioned in the motion estimation process in [5]. The details of this method are given in Chapter-6. Also, we will utilize (5.13) and (5.14) to drive our adaptive velocity estimation in Appendix-A.

The meaning of the Jacobian matrix can be expressed by taking the partial derivative of the equation (5.4) w.r.t time under assumption that rectified image is a continuous time varying function

$$\frac{\partial I\left(f(\chi,\Theta_k),k\right)}{\partial k} \approx \frac{\partial I\left(f(\chi,\hat{\Theta}_{k-1}),k\right)}{\partial k} + \nabla_{\Theta_k} I\left(f(\chi,\Theta_k),k\right)\Big|_{\Theta_k=\hat{\Theta}_{k-1}} \frac{\partial}{\partial k}\left[\Theta_k - \hat{\Theta}_{k-1}\right] \tag{5.15}$$

Here, the jacobian of the current observation relates the change in motion parameters to the change in brightness values in the target region [1]. The solution of (5.4) w.r.t v_k benefits from the similar relationship as well but it uses the changes in brightness values to infer the motion parameters of the target [1]. Also, the multiplication in each row in (5.14) can be interpreted as projecting the change in brightness values onto the changes in transformation parameters for each pixel [1].

5.2 The Limitations and Estimation Quality

i. In iterative refinement process rectification of the current observation with new motion parameters is needed. But rectification is a cumbersome process and also it introduces error into estimation process due to interpolation [40].

Every pixel provides one constraint for the parameters of the motion model. Therefore, the number of target pixels must be equal to the number of the parameters at least [40].

ii. The accuracy of the estimate depends on how well the linearization of the current observation is done. Linearization via Taylor series expansion is good when the change in parameters is small. Therefore, several iterations are needed if initial estimate is far away from the true one [41].

iii. All of the above is valid if only change in subsequent images of the target is completely described by f(.), i.e., there are no changes in the illumination of the target [5]. However, the adaptive appearance model overcomes this problem and so the similarity transformation becomes suitable motion model. Also, to avoid contrast variations zero-mean unit variance normalization is utilized [1].

In [5] the quality of the motion estimation is measured as

$$\varepsilon_k = \frac{1}{d}\sum_{i=1}^{d}\left\{\sum_{j=s,w} m_{j,k}\frac{\left(I\left(f(\chi_i,\Theta_k),k\right)-\mu_{j,k}(i)\right)^2}{2\sigma_{j,k}^2(i)}\right\} \tag{5.16}$$

Where ε_k denotes the distance (error) between current observation $I\left(f(\chi,\Theta_k),k\right)$ and the reference target template also Θ_k denotes the iteratively updated motion vector by v_k. Initially it is assumed that $\Theta_k = \hat{\Theta}_{k-1}$ which is the optimum estimate of the motion parameters that minimizes ε_{k-1}. In iterative estimation procedure the result of each iteration is used as input to the next iteration such that $\Theta_k^n = \Theta_k^{n-1} + v_k^{n-1}$ [5] where n denotes the iteration number. After the first iteration the computed v_k might not be accurate, but indicates a good minimization direction for ε_k. Therefore

several iterations are performed to compute the more accurate estimations of motion parameters.

However, a threshold is needed to determine the adequate score for the quality. If this threshold is satisfied, iterative motion estimation is terminated. Otherwise, the search continues. A constant threshold can't be appropriate to determine the adequate estimation quality, since the appearance model of the target changes due to motion of the target, lighting conditions and sensor noise [6]. Therefore, it is more appropriate to calculate the threshold via Kalman Filter so; it should be set to a value by considering how challenging it is to track the given target. In this sense a constant velocity model can be used and the score of the motion estimation associated to the particle which has the highest importance weight is accepted as the measurement [6].

This tracker runs efficiently if the stable component of the target model becomes more dominant. In other words, the mixing probability of the stable component should be larger than mixing probability of the transient component. The quality of the estimation given in (5.16) uses statistics from target appearance model. Considering these facts (5.16) measures the estimation quality by depending on the pixels whose mixing probability for the stable component is large (the variance of the stable component for these pixels are low). That is, these pixels are more influential to determine the quality of the motion estimation.

5.3 Robust Estimators

Besides the limitations given in section 5.2 the least mean square estimators can't tolerate the bias introduced by the outliers of the data. In other words these estimators are sensitive to outliers and so the result easily deviates from the optimum estimate. To analyze robustness of an estimator three measures are used [42]:

1. Efficiency: Ability to reach to the optimal estimates when there is noisy data and outliers.

2. Breakdown point: Defined as percentage of outliers that an estimator can tolerate. For example for a robust estimator the breakdown point should be at most equal to $1/(p+1)$ percent of all data where p denotes the number of states in the system model.

3. Computational Complexity

Here, we will present two well known M-estimators: (1) Huber estimator and (2) Tukey's biweight estimator. These estimators down-weight the dubious observations rather than rejecting them, thus the estimate is less influenced by these observations [22]. In [42] M-estimators are established as

$$\arg\min_{\Theta} \sum_{i} \rho\left(y_i - M\left(x_i, \Theta\right)\right) \tag{5.17}$$

Here y_i and x_i respectively denote the reference data and observations. Also, M is the parameterized transformation function whose parameters are represented by θ. ρ is the function which generates a cost for a given error. In general function ρ corresponds to

$$\rho\left(x\right) = -\log f\left(x\right) \tag{5.18}$$

Where f is the likelihood of observations. Since these estimators maximize the likelihood function of the observations while minimizing their cost functions, they are called maximum likelihood estimators (MLEs) [22].

In [22] Huber estimator is given by

$$p\left(x\right) = \begin{cases} x^2 & \text{if } |x| \le c \\ c\left(2|x| - c\right) & \text{otherwise} \end{cases} \tag{5.19}$$

Where c denotes the outlier threshold. In [5] it is assigned to 1.435 and in [42] it is accepted as 1.345 to obtain %95 efficiency. Also, in [42] it is adjusted by setting it to large values at the beginning of the estimation process then it is lowered as estimation accuracy improves.

For Tukey's biweight estimator function ρ is defined in [42] as follow

$$p(x)=\begin{cases}\dfrac{c^2}{6}\left[1-\left[1-\left(\dfrac{x}{c}\right)^2\right]^3\right] & \text{if } |x|\le c \\ \dfrac{c^2}{6} & \text{otherwise}\end{cases} \tag{5.20}$$

To analyze the robustness of these estimators the influence function is introduced in [22], [42] and [43]. The influence function characterizes the sensitivity to the bias that an outlier or noisy data induce on the estimate [42]. Given the estimator it is defined by

$$\varphi(x)=\frac{d\rho(x)}{dx} \tag{5.21}$$

These estimators and their influence functions are plotted in Figure 5.1 and Figure-5.2 where the influence functions are given by

$$\varphi_{Huber}(x)=\begin{cases}-c & \text{if } x<-c \\ 2x & \text{if } |x|<c \\ c & \text{if } x>c\end{cases} \tag{5.22}$$

$$\varphi_{Tukey's}(x)=\begin{cases}x\left(c^2-x^2\right)^2 & \text{if } |x|\le c \\ 0 & \text{otherwise}\end{cases} \tag{5.23}$$

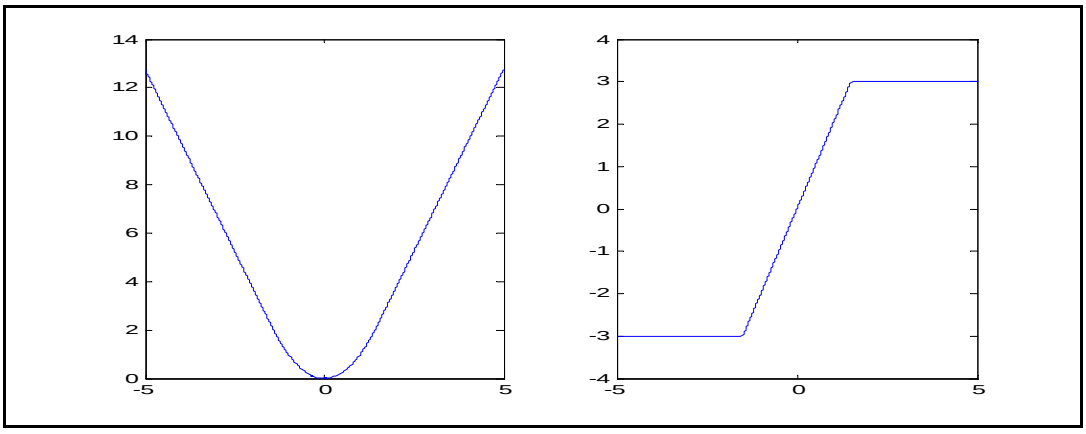

Figure 5.1 (a) Huber Estimator, (b) Influence Function φ (c=1.50)

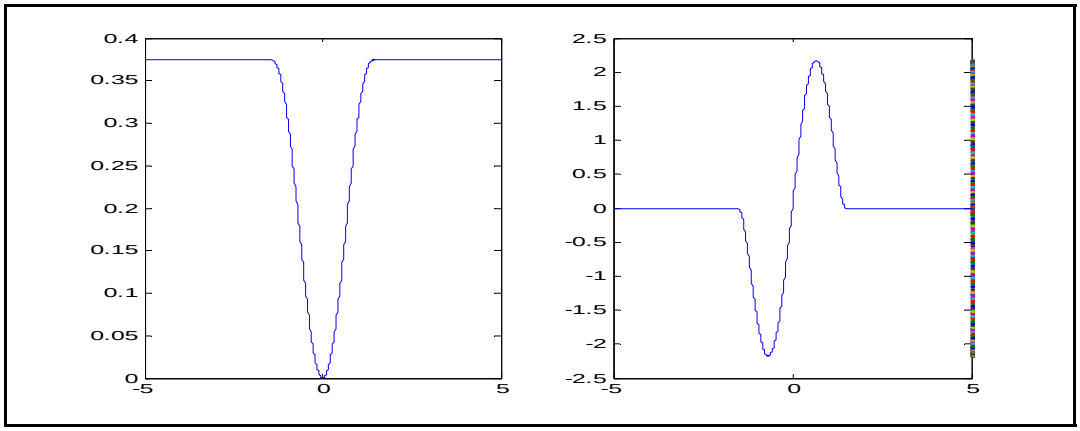

Figure 5.2 (a) Tukey's biweight Estimator, (b) Influence Function φ (c=1.50)

For least mean square estimators, the influence of outlier increases at an accelerating rate (2x), while for L1 estimators (|x|), it becomes a constant (sign(x)). According to Figure-5.1 and Figure 5-2 the influence functions are bounded as error increases over predetermined threshold. When Huber estimator (Windsorizing) is used, a mixture of least mean square and L1 estimators is seen: for small values of error, ρ increases at an accelerating rate, but once the predetermined threshold is reached (1.50 in this example), the influence of error becomes constant.

Tukey's biweight (also known as bisquare) function behaves in a similar way to the least mean square estimator at first, but for larger errors, the function tapers off and thus the influence of large errors on the result is reduced [42].

CHAPTER 6

EXECUTION OF THE PROPOSED VISUAL TRACKING ALGORITHM

In this chapter the main steps of the visual tracking algorithm proposed by Zhou will be introduced. Also, some fundamental functions of this algorithm such as adaptive velocity estimation, adaptive noise and observation likelihood calculations will be explained clearly. In this thesis we have implemented this algorithm to see its robustness and efficiency against target pose variations and also changes in target kinematic parameters during the overall tracking. Moreover, to increase its efficiency and robustness we have used learning rate and robust statistics in the adaptive velocity estimation and perform illumination compensation via pixel statistics.

6.1. Main Steps of the Visual Tracking Algorithm

1. After the target is marked out from the incoming frame by hand, the initialization phases of both particle filter and object appearance model are invoked by the main part of the program.

   a. The state vector (similarity parameters) describing the kinematics of the target is set as follows:

$$\left[a_{11}, a_{12,}\, a_{21,}\, a_{22}, d_x, d_y\right]^T = \left[1.0,\, 0.0,\, 0.0,\, 1.0,\, 0.0,\, 0.0\right]^T \qquad (6.1)$$

   Here the first four variables are used in geometrical transformation to scale and rotate the target region and the remaining ones are used in translational motion of the target. With the given initialization in (6.1) the initial target state indicates no geometric transformation and translational motion.

b. The particle set $\left\{\Theta_o^i, 1/J\right\}\Big|_{i=1}^{J}$ is initialized by selecting J=100 particles from a uniform distribution $p\left(\Theta_0\right)$

c. Adaptive appearance model $A_k = \left\{s_k, w_k\right\}$ which has two components denoted by s and w (stable and wandering components) is initialized: Gray level pixel intensities within the target region are used to set the mixture centers $\mu_{0,i}\Big|_{i=s,w}$ of the model components to their initial values. The initial variance of these components are assigned as;

$$\sigma_{w,0} = 0.75 \text{ and } \sigma_{s,0} = \frac{\sigma_{w,0}}{5.0} \tag{6.2}$$

Also, the mixing probabilities of the model components $m_{0,j}$ are set to 0.15 and 0.85 for i = s, w respectively. Since stable features of the target evolve after a period of time and the appearance of the target can change rapidly after the initialization, the mixing probability for the wandering component denoted as w is set to a higher value than the stable component denoted as s [4].

d. Finally, OCD flag is set to false to indicate no occlusion

2. The following steps are performed when a new image is captured during the tracking phase of the main program.

Before beginning to tell these steps of the implemented visual target tracker, in order to provide initial insight about the operations performed within the tracker its flowchart is given in Figure-6.1

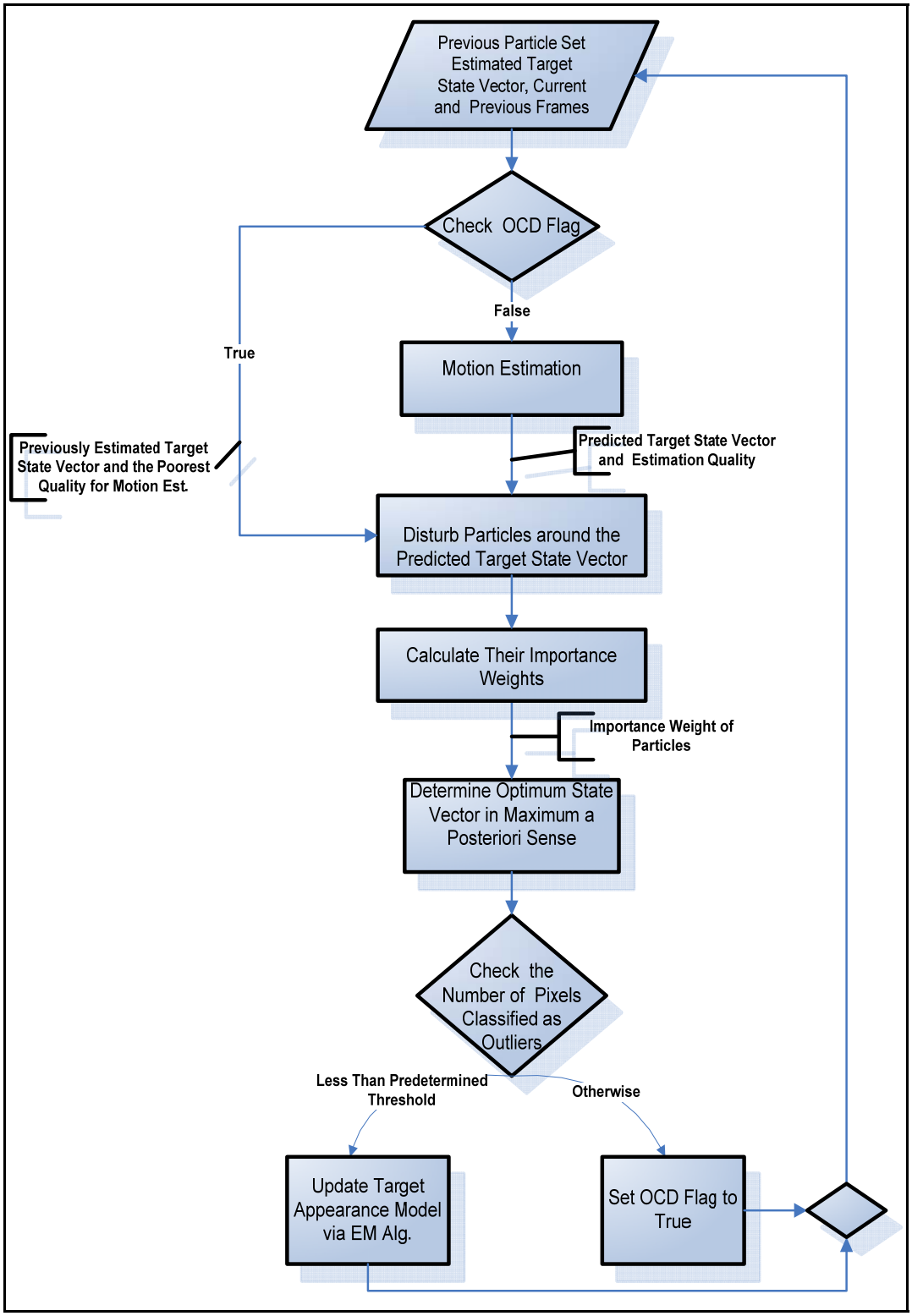

Figure-6.1 Flowchart of the Target Tracking Phase

a. If OCD flag is false, compute the adaptive velocity estimate, v_k by making several iterations until the estimate converges to the optimum value. Otherwise set it to 0. Algorithm 6.1 shows the steps of the adaptive velocity estimate.

b. Generate new particles:

- Sample the adaptive process noise from the density of

$$U_k^i = R_k * U_0 \tag{6.3}$$

Where R_k denotes a scalar that is a function of the estimation quality for the adaptive velocity. Otherwise, if OCD flag was true, it would be set to the highest predetermined value. Also, in (6.3) U_0 denotes a standardized process noise vector which is zero mean Gaussian and its standard deviation matrix is equal to:

$$U_0 = block - diag\left[10/180,\ 10/180,\ 10/180,\ 10/180,\ 10,\ 10\right] \tag{6.4}$$

See Algorithm 6.2 to get further insight into the computation of U_k.

- Using the system model ($\theta_k = \theta_{k-1} + v_k + U_k$) generate new particles as follows

$$\Theta_k^i = \hat{\Theta}_{k-1} + v_k + U_k^i \quad i = 1,...,J \tag{6.5}$$

Here $\hat{\Theta}_{k-1}$ denotes the predicted target state vector through the adaptive velocity estimation

c. Update the importance weights of each particle by calculating observation likelihoods, $p\left(Z_k \left| \Theta_k^i \right.\right)$. Algorithm 6.3 gives the details of this calculation.

d. Perform MAP estimate to find the optimum state and then update the target state vector.

e. Set OCD flag to true if the ratio of the number of pixels classified as outlier over the total number of pixels within the target region exceeds 0.20. The threshold is set to 0.20 since in [42] it is said that the breakdown point of a robust estimator is at most equal to $1/(p+1)$ where p is the number of states in the system model. In this thesis our state space is composed of 4 variables and thus the threshold is assigned to 0.20.

f. If OCD flag is false, update the target appearance model using pixels within the target region $\hat{Z}_k$ cropped out of the current frame by the updated target state vector.

The steps of the algorithm given here are detailed in the next sub-section by explaining the fundamental functions clearly.

6.2 Fundamental Components of the Algorithm

Algorithm 6.1: Adaptive Velocity Estimate

Using the previous particle set $\left\{\Theta_{k-1}^{i}, w_{k-1}^{i}\right\}\Big|_{i=1}^{J}$ and updated adaptive appearance model A_k through the previous target region $\hat{Z}_{k-1}$, new state of the target is predicted by using incoming image as follows:

- To develop the details of the algorithm assume that the current target state is represented by Θ_k at the discrete time index k so that the target region cropped out of the incoming frame is defined by

$$I\left(f(\chi,\Theta_k),k\right) \simeq \hat{Z}_{k-1} \tag{6.1.1}$$

Where, f is the image warping function and χ denotes the location of pixels within previous target region, $\hat{Z}_{k-1}$.

- $I\left(f(\chi,\Theta_k),k\right)$ is approximated by Taylor series expansion up to the first order terms around the previous estimate of the target state vector $\hat{\Theta}_{k-1}$ and this approximation yields to

$$\begin{aligned} I(f(\chi,\Theta_k),k) &\simeq I\left(f(\chi,\hat{\Theta}_{k-1}),k\right)+\nabla_{\Theta_k} I(f(\chi,\Theta_k),k)\Big|_{\Theta_k=\hat{\Theta}_{k-1}^{i}}\left[\Theta_k-\hat{\Theta}_{k-1}\right] \\ &= I\left(f(\chi,\hat{\Theta}_{k-1}),k\right)+C_k v_k \end{aligned} \tag{6.1.2}$$

Here the shift in the state variables or adaptive velocity $\Theta_k - \hat{\Theta}_{k-1}$ is denoted as v_k and C_k is the jacobian matrix of the image region w.r.t target state variables.

- By substituting (6.1.1) into (6.1.2) the equation that computes the adaptive velocity estimate is derived as

$$v_k \simeq -B_k\left[I\left(f(\chi,\hat{\Theta}_{k-1}),k\right)-\hat{Z}_{k-1}\right] \tag{6.1.3}$$

Where B_k denotes the inverse of the Jacobian matrix C_k.

The above equation is the same as the adaptive velocity estimation obtained from (5.7) except that in (6.1.3) we use target region cropped out of the previous frame as target reference template.

- To estimate the pseudo inverse of C_k, (6.1.3) is evaluated with previous particle set as follows

$$v_k^i \simeq -B_k\left[I\left(f(\chi,\Theta_{k-1}^i),k-1\right)-\hat{Z}_{k-1}\right] \tag{6.1.4}$$

Where the difference in motion vectors $v_k^i = \Theta_k - \Theta_{k-1}^i$ defines adaptive velocity w.r.t ith particle also, $I\left(f(\chi,\Theta_{k-1}^i),k-1\right)$ is the image region cropped out of the previous frame via the parameters represented by ith particle and denoted as Z_{k-1}^i.

- Differences in motion vectors $\Theta^i_{k-1} - \hat{\Theta}_{k-1}$ and image patches $I\left(f(\chi, \Theta^i_{k-1}), k\right) - \hat{Z}_{k-1} = Z^i_{k-1} - \hat{Z}_{k-1}$ are stacked into the following matrices:

$$\Theta^{\delta}_{k-1} = \left[\Theta^1_{k-1} - \Theta_{k-1}, ..., \Theta^J_{k-1} - \Theta_{k-1}\right], \; Z^{\delta}_{k-1} = \left[Z^1_{k-1} - \hat{Z}_{k-1}, ..., Z^N_{k-1} - \hat{Z}_{k-1}\right] \tag{6.1.5}$$

- Using the above matrices the LS solution for B_k is given by

$$B_k = \left(\Theta^{\delta}_{k-1} Z^{\delta\ T}_{k-1}\right)\left(Z^{\delta}_{k-1} Z^{\delta\ T}_{k-1}\right)^{-1} \tag{6.1.6}$$

- However, the matrix $\left(Z^{\delta}_{k-1} Z^{\delta\ T}_{k-1}\right)$ is usually singular then, it isn't possible to take the inverse of it. To cope with this problem Zhou et. al in [5] proposed to use singular value decomposition (SVD) and so using SVD the matrix Z^{δ}_{k-1} is decomposed as follows:

$$Z^{\delta}_{k-1} = USV^T \tag{6.1.7}$$

By substituting SVD of the Z^{δ}_{k-1} into the (6.1.6) B_k is computed as

$$B_k = \Theta^{\delta}_{k-1} VS^{-1} U^T \tag{6.1.8}$$

Actually the mathematical calculation given by (6.1.8) is obtained by taking the pseudo-inverse of the matrix composed of the differences between image patches, Z^{δ}_{k-1}. The best way to compute the pseudo-inverse of a rank deficient matrix A is to use SVD. Let's assume that SVD of matrix A is given by $A = USV^T$ then pseudo-inverse of A is shown as [44]

$$\begin{aligned} A^{+} &= \left(AA^T\right)^{-1} A^T \\ &= VS^{-1}U^T \end{aligned} \tag{6.1.9}$$

Where U and V are orthonormal matrices whose columns are called left and right singular vectors and S is the diagonal singular matrix.

Finally, substituting (6.1.8) into (6.1.3) the adaptive velocity estimate is computed. However, to make the estimation robust to bias induced by outliers the influence of each data are weighted by the means of a (dxd) diagonal matrix whose ith element is given by

$$L_k(i) = \frac{1}{x}\frac{dp(x)}{dx} = \begin{cases} 1 & \text{if } |x| < c \\ c/|x| & \text{otherwise} \end{cases} \qquad (6.1.10)$$

Where x is a pixel intensity in the difference image $I\left(f(\chi,\hat{\Theta}_{k-1}),k\right)-\mu_{s,k-1}$ normalized by the standard deviation of the transient (wandering) component. Here $\mu_{s,k-1}$ denotes the mean of the stable component for each pixel at discrete time index k-1. Also, c denotes outlier threshold value and ρ is Huber estimator which has been introduced in Chapter-5.

- Finally using (6.1.10) the adaptive and robust velocity estimation is established as [5]

$$v_k \simeq -B_k L_k\left[I\left(f(\chi,\hat{\Theta}_{k-1}),k\right)-\hat{Z}_{k-1}\right] \qquad (6.1.11)$$

In this thesis to estimate the adaptive velocity an iterative robust and reweighted least square estimation is performed. At each iteration previous estimates are used as input to the current one and so the estimation quality is improved until the estimate convergences to optimum value. Furthermore, the learning rate is utilized in iterative adaptive velocity estimation to control both the speed of convergence and accuracy of the algorithm. The learning rate is set to 0.5 in the first iteration of the adaptive velocity estimation and then it is set to 0.25 for next iterations. Finally to prevent inaccurate velocity estimations for small size targets, velocity estimates are checked with predetermined thresholds (2 sigma values of the similarity transformation parameters).

Algorithm 6.2: Adaptive Noise

Using the adaptive velocity estimate v_k and previously updated target appearance model A_k, the calculation of the adaptive noise U_k is defined as follows:

- New state of the target is predicted with adaptive velocity estimate, v_k as

$$\tilde{\Theta}_k^n = \tilde{\Theta}_k^{n-1} + v_k^{n-1} \tag{6.2.1}$$

Here, n denotes the iteration number in the iterative and robust reweighted least square (IRLS) and $\tilde{\Theta}_k^n$ denotes the prediction in the nth iteration such that this prediction will become pivotal point for the next iteration.

- Using the geometrical transformation function f with the predicted target state after the adaptive velocity estimation crop out the target region from the current image:

$$I\left(f(\chi, \tilde{\Theta}_k^n), k\right) \simeq \tilde{Z}_k \tag{6.2.2}$$

- The quality of the prediction, ε_t is measured as a distance between $\tilde{Z}_k$ and previously updated target appearance A_k :

$$\varepsilon_k = \frac{1}{d} \sum_{n=1}^{d} \left\{ \sum_{i=s,w} m_{i,k-1} \left(\frac{\tilde{Z}_k(n) - \mu_{i,k}(n)}{\sigma_{i,k}} \right)^2 \right\} \tag{6.2.3}$$

Here, if ε_k is small, it implies good prediction of the adaptive velocity then; particles are perturbed with the small process noise to absorb the residual motion. Otherwise, if ε_k is large, it implies poor prediction then; particles are perturbed with large process noise to cover the abrupt changes in the motion of the target [5]

- Using the prediction quality the scale of the standardized noise, R_t is computed as

$$R_k = \max(\min(R_o\sqrt{\varepsilon_k}, R_{\max}), R_{\min}) \quad (6.2.4)$$

Where R_{min} and R_{max} are respectively the lower and upper bound to cover the motion of the target [5] and R_0 is the nominal value for R_k. In this thesis R_{max}, R_{min} and R_0 are respectively assigned to 1.0, 0.50 and 0.25.

- Otherwise, if occlusion has been declared, the adaptive velocity estimation is skipped and the scale of the standardized noise is given by

$$R_k = R_{max} \quad (6.2.5)$$

Algorithm 6.3: Observation Likelihood Calculation

After making an accurate adaptive velocity estimation particles are generated by (6.5) then, to update the state of the target the algorithm needs the importance weights of the particles. In [5] the importance weights correspondence to the observation likelihoods and the calculation of this likelihood is given by the following steps:

- Obtain the image regions $I\left(f(\chi,\Theta_k^i),k\right)$ via the particle states denoted as Θ_k^i and then using gray level pixel intensities within these regions calculate the normalized innovations:

$$\nu_s = \frac{\left|\text{pixelInt} - \mu_{s,k}\right|}{\sigma_{s,k}} \tag{6.3.1}$$

Where pixelInt denotes the gray level pixel intensity in a particular location within the image region given by $I\left(f(\chi,\Theta_k^i),k\right)$ also $\mu_{s,k}$ and $\sigma_{s,k}$ are respectively the mean and standard deviation of the corresponding position w.r.t the stable component of the appearance model.

- If ν_s is less than predetermined outlier threshold denoted as c which is assigned to 1.435, the value added by this pixel to the stable component of the observation likelihood is given by

$$likelihood_s = \left(2\pi\sigma_{s,k}{}^2\right)^{-\frac{1}{2}} \exp\left(-0.5\nu_s{}^2\right) \tag{6.3.2}$$

- Otherwise, this pixel is classified as outlier and it makes the following contribution to the stable component of the observation likelihood as

$$likelihood_s = \left(2\pi\sigma_{s,k}{}^2\right)^{\frac{1}{2}} \exp\left(-c(\nu_s - 0.5c)\right) \tag{6.3.3}$$

Thus, the contribution of outlier pixels to the computation of the observation likelihood is decreased and the likelihood calculation becomes both reliable and robust.

- Repeat the same steps to calculate contribution of this pixel to the wandering component of the observation likelihood using statistics of the wandering component, $\mu_{w,k}$ and $\sigma_{w,k}$.

- Combine these contributions by using mixing probabilities of the model components and then take the logarithm of the result. Repeat these steps for all pixels within the image region and sum their contributions to obtain the observation likelihood as.

$$likelihood = \sum_{n=1}^{d} \ln\left(m_{stable,n} * likelihood_{s,n} + m_{wander,n} * likelihood_{wander,n}\right) \qquad (6.3.4)$$

Where $m_{stable,n}$ and $m_{wander,n}$ are respectively the mixing probabilities of the model components and d is the total number of the pixels within the hypothesized target region.

- Finally after adding the last contribution to the observation log-likelihood, the observation likelihood of the particle is given by

$$\exp(likelihood / number\ of\ pixels) \qquad (6.3.5)$$

CHAPTER 7

SIMULATION RESULTS

In this section we will demonstrate the tracking results obtained from the visual target tracker. For this purposes we have tested the algorithm in different scenarios, but here we only demonstrate the results obtained from a synthetic video sequence and two real video sequences. The algorithm is written in C++ environment and run on 2.0 GHz Pentium Core 2 Duo CPU with 1 GB of RAM.

Synthetic video images are used to evaluate the performance of the implemented algorithm statistically. In synthetic scenarios target makes both linear and nonlinear movements. It moves with a constant velocity along the vertical axis whereas its movement is sinusoidal along the horizontal axis. No occlusion and illumination changes occur but the target size changes from 7.5x7.5 to 30x30 pixels. The aim of selecting such a scenario is to observe the ability of the algorithm under a complex target motion where both 2D translational motion and scale change occur

In this experiment, the target of interest is a moving square. A template used for this experiment is obtained by marking out the target in the first image by hand. The results are obtained by the values of the design parameters given in Chapter 6. Sample images given in Figure-7.1 are outputs of the tracking algorithm where the outer rectangle bounding the tracked object shows the tracking result. Also, Figure-7.2 plots both the trajectory of the target centroid and its position estimates given by the tracking algorithm. Although the square has different kinematic behaviors along the two axes, the tracking algorithm is still successful.

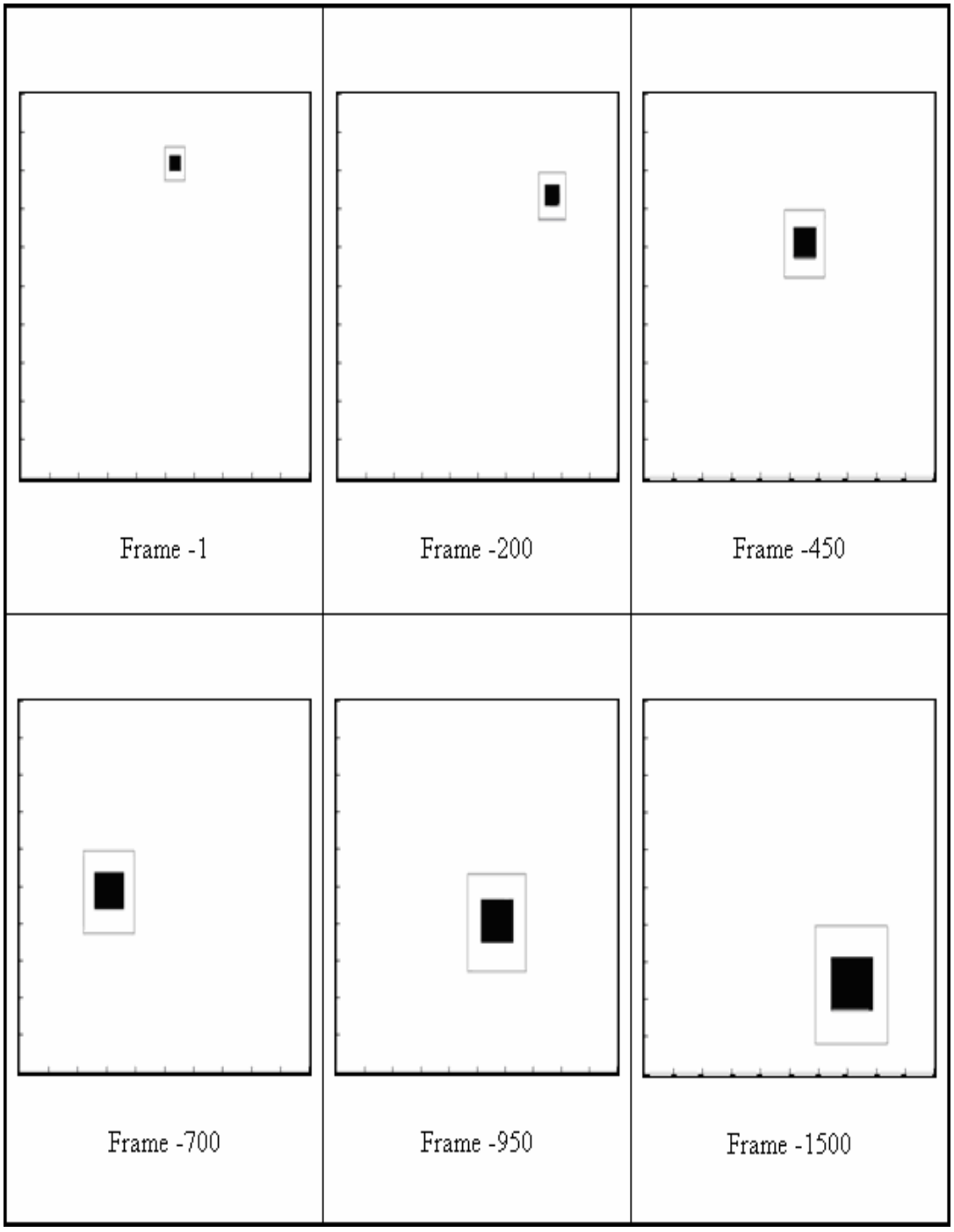

Figure-7.1 Tracking Results on Moving Square Sequence using 100 Particles

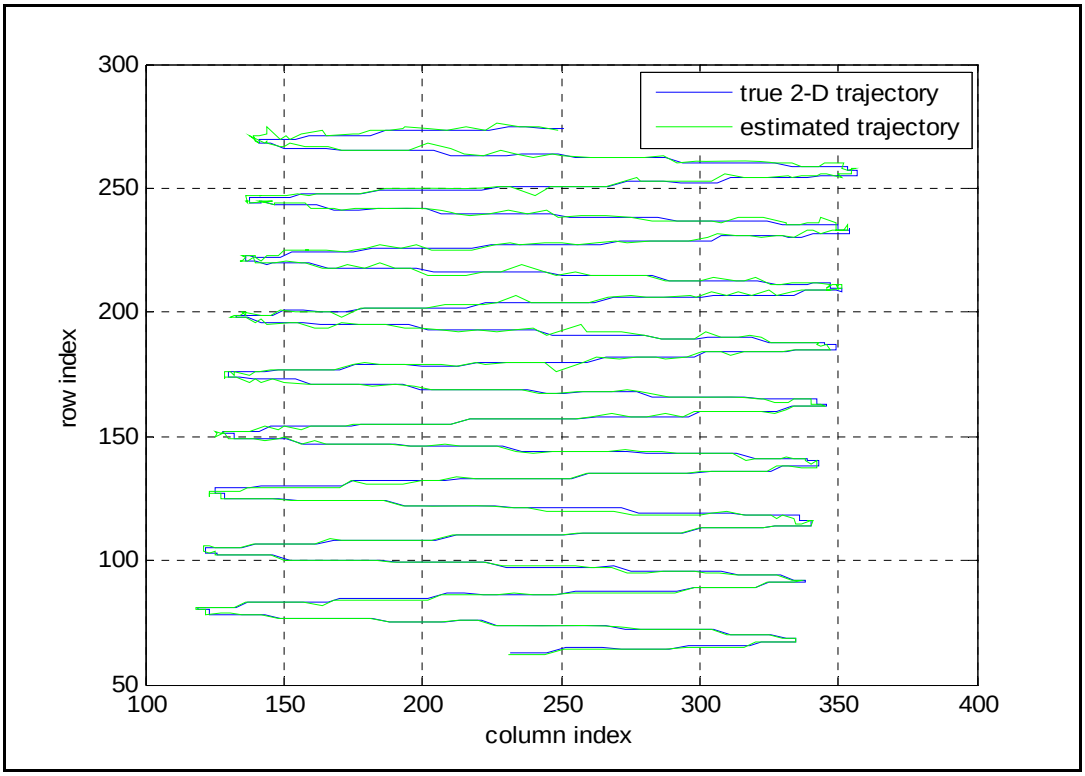

Figure-7.2 Trajectory of Moving Square and Its Estimation with 100 Particles

To measure the quality of the proposed algorithm we computed the mean square error (MSE) of the translational motion parameters where the computation of MSE is given by:

$$\mathrm{MSE}_{\mathrm{trans}}(t)=\left(x(t)-\hat{x}(t)\right)^2+\left(y(t)-\hat{y}(t)\right)^2 \qquad (7.1)$$

Here x and y represent the true position of the moving square along the horizontal and vertical axis respectively at time t while $\hat{x}$ and $\hat{y}$ denote the corresponding estimations given by the tracking algorithm.

Figure-7.3 plots MSE for the translational motion parameters. The average MSE for these parameters is computed as 3.3 pixels. In other words, it means that average standard deviation from the true position is approximately 1.8 pixels.

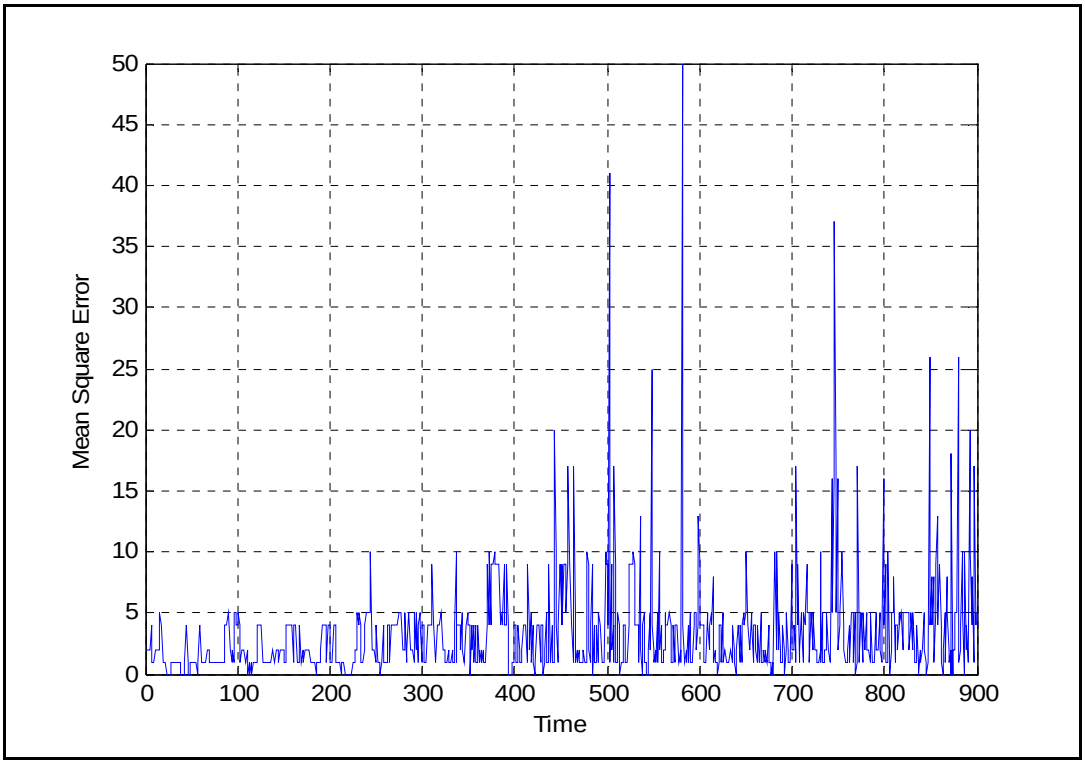

Figure-7.3 MSE for Translational Motion Parameters with 100 Particles

Figure-7.4 shows both true and estimated scales of the target vs. time during overall tracking process where using deformation parameters of the similarity transformation the scale of the target is calculated as $\sqrt{\left(a_1^2+a_2^2+a_3^2+a_4^2\right)/2}$ [5]. As can be seen from this figure the scale of the target increases as time proceeds but the tracking algorithm follows change in scale variable where the average deviation from the accurate value of the scale variable is about 0.07.

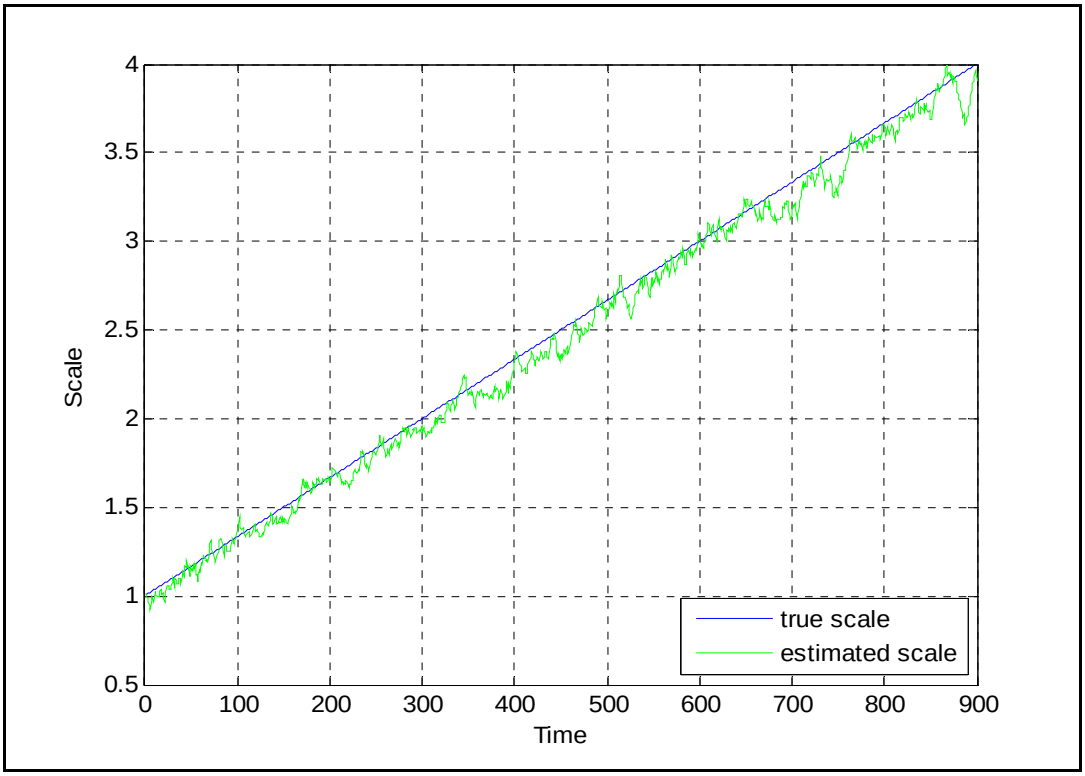

Figure-7.4 Scale Estimate for the Moving Square with 100 Particles

The computational performance of the algorithm for synthetic video sequence on the machine of which the configuration is given at the beginning of this chapter is shown in Figure-7.5. It shows the frame processing time of the algorithm during the overall tracking process. According to this figure the average processing time is about 8 frames per second with 100 particles.

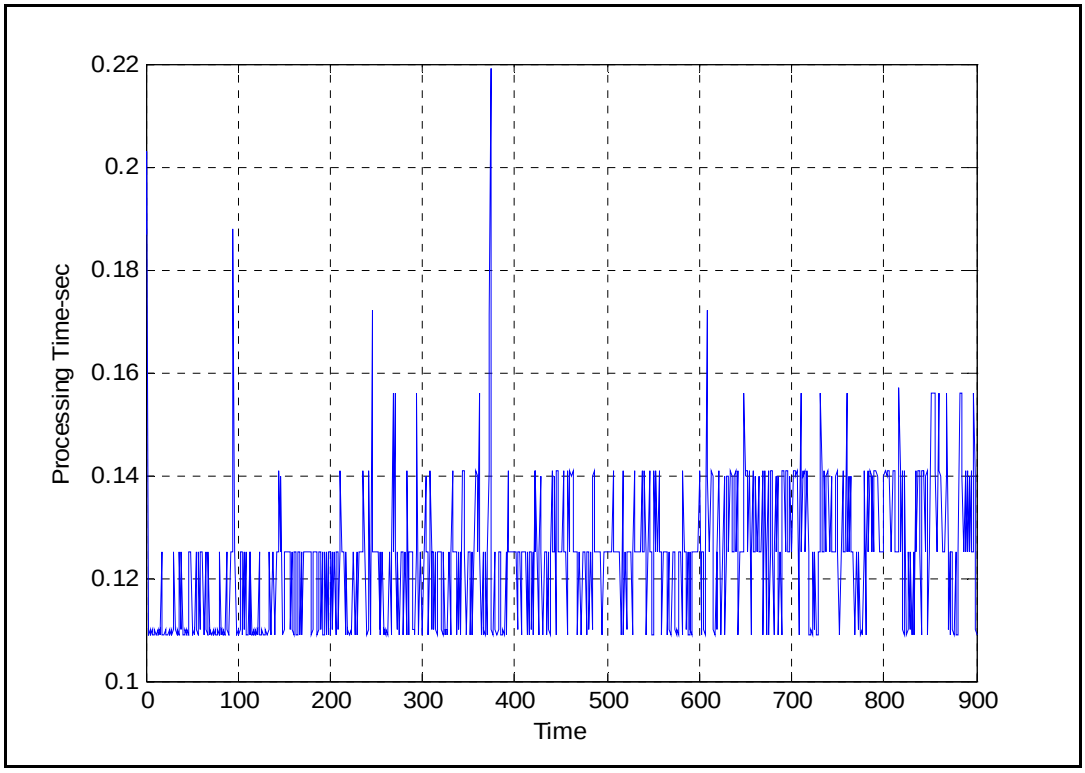

Figure-7.5 Performance of the Tracking Algorithm with 100 Particles

As we can see from Figure-7.5 the performance of the algorithm degrades slightly as the size of the object increases. The reason is that as the size of the object increases the more pixels are involved in the warping process of the image regions indicated by the particles.

To illustrate importance of particle number, we tested the algorithm on synthetic video sequence again after setting the number of particles to 50 and 200. The following figures show the accuracy and computational performance of the algorithm for these cases. As pointed out in MSE plot, the accuracy of the algorithm is better when the number of particles is increased. However, the computational performance of the algorithm degrades as the number of hypothesis indicated by the particles increases.

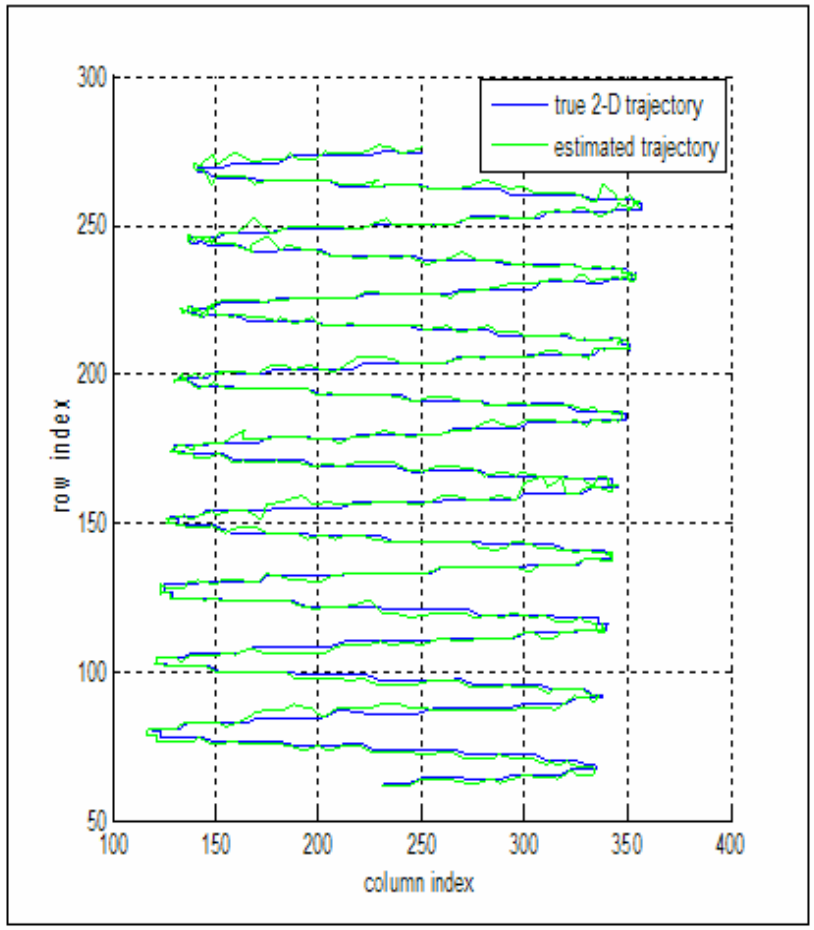

Figure-7.6 Trajectory of the Moving Square and Its Estimation with 50 Particles

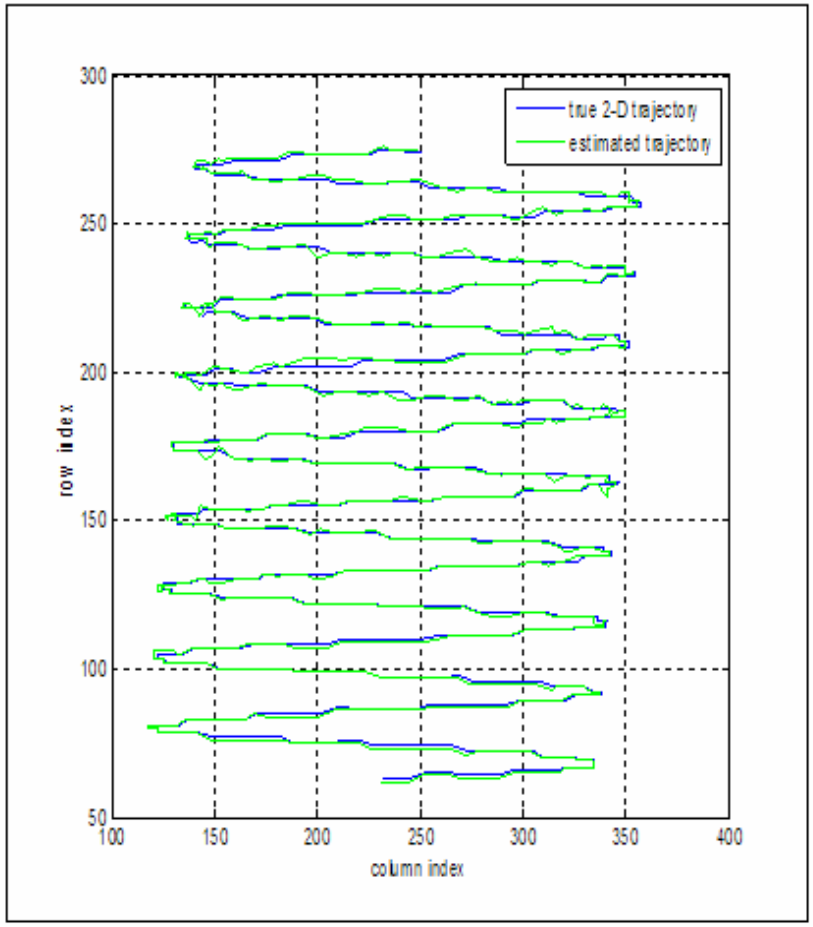

Figure-7.7 Trajectory of the Moving Square and Its Estimation with 200 Particles

To illustrate the importance of the particle number, MSE of translational motion parameters with 50 and 200 particles are computed and shown in figures 7.8 and 7.9.

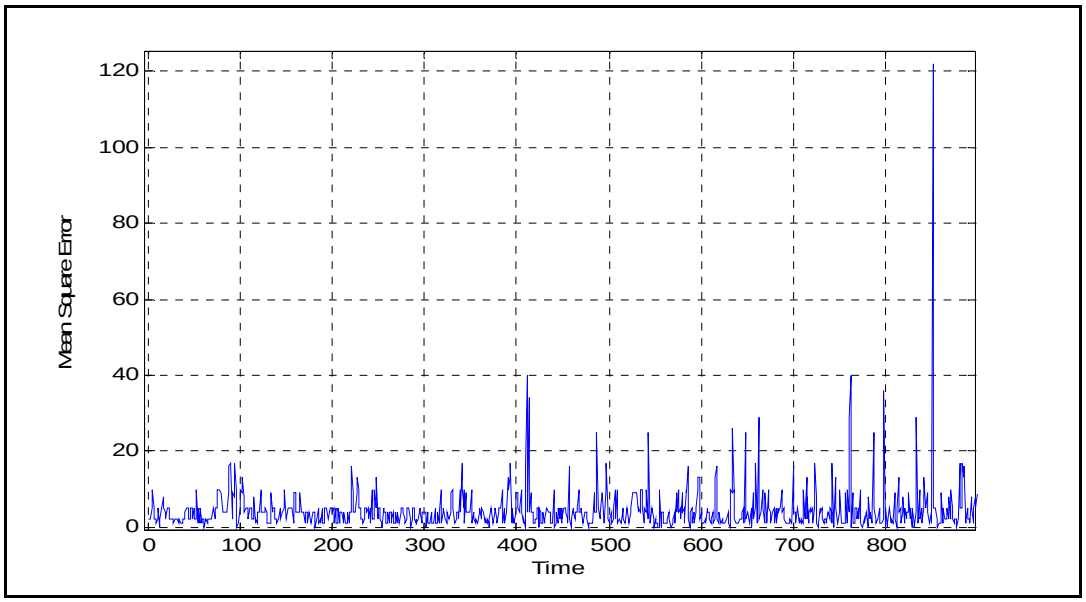

Figure-7.8 MSE for Translational Motion Parameters with 50 Particles

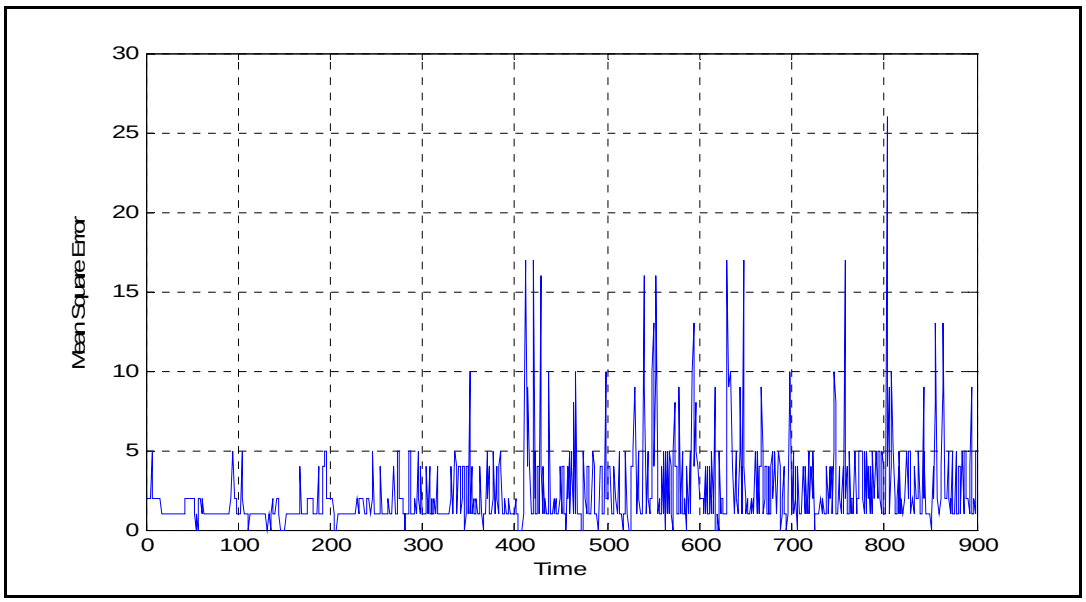

Figure-7.9 MSE for Translational Motion Parameters with 200 Particles

As can be seen from these two figures MSE is inversely proportional to the number of particles. The reason is that when the number of particles is increased, the number of hypothesis for target pose and position increases as well and consequently the algorithm makes better estimation for the posterior distribution of the state space variables.

From figures 7.8 and 7.9 the average MSE of displacement variables with 50 and 200 particles are calculated as 4.4 and 2.4 pixels respectively. It means that when the number of particles is doubled the average accuracy of the estimated translational motion parameters is improved about 38 percent. On the other hand the accuracy of these variables decreases about 33 percent when the number of particles is reduced by half.

Estimated scale for moving square is illustrated in figures 7.10 and 7.11 when the employed number of particles is 50 and 200 respectively.

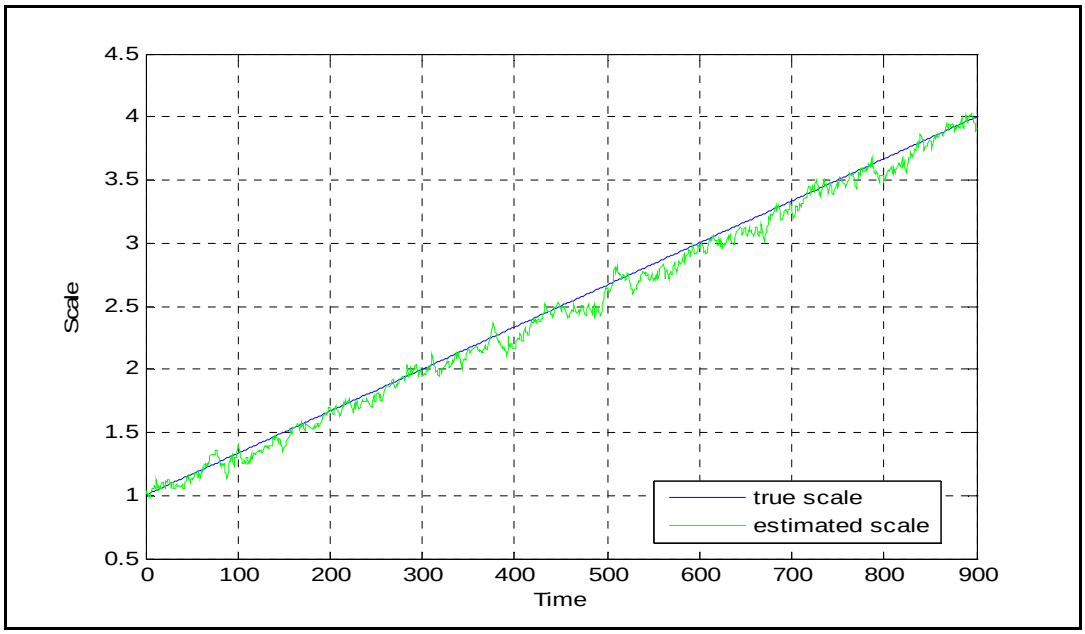

Figure-7.10 Scale Estimate for the Moving Square with 50 Particles

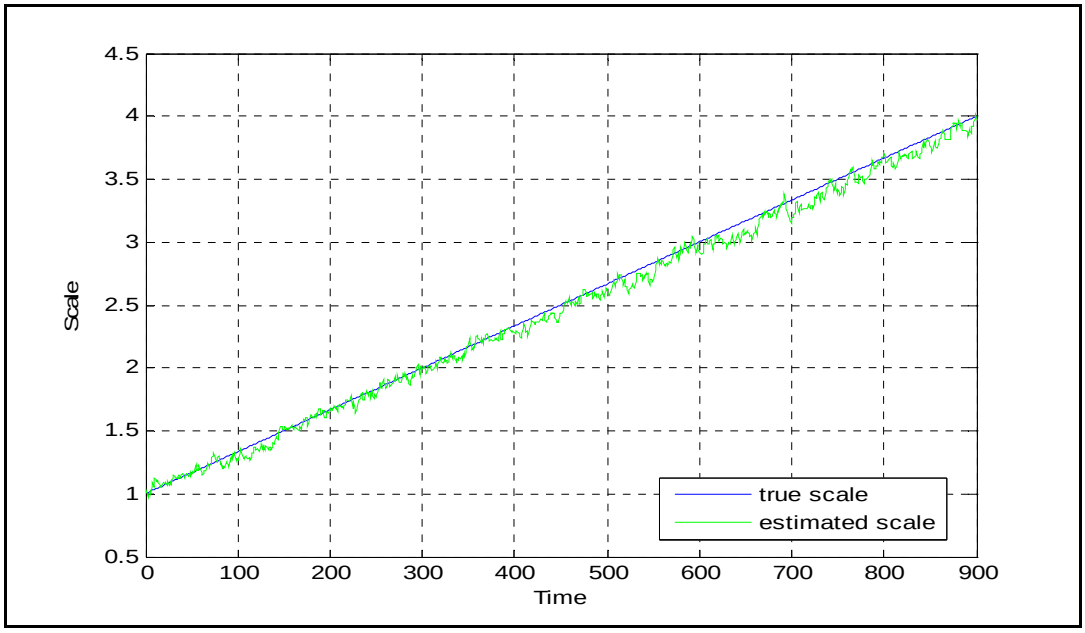

Figure-7.11 Scale Estimate for the Moving Square with 200 Particles

The average MSE of the estimated scale is computed as 0.005 and 0.004 for 50 and 200 particles. In other words, average deviation from the true scale is about 0.07 and 0.06 for the given number of particles. Thus, the improvement on scale estimation becomes about 15 percent when the number of particle is doubled. Nevertheless, the accuracy of the scale estimation doesn't change significantly. Since scale parameter takes value w.r.t a slowly time varying linear model and also noise entering this model is assumed to have small variance, scale estimates indicated by particles doesn't change significantly around the predicted target scale and importance weight of the particles are affected by estimates of the displacement parameters much more than the estimate of the scale parameter.

While the number of particle is increased, we expect that the accuracy of the algorithm will improve whereas the performance of the algorithm will decrease because of the increased number of hypothesis indicated by particles for the target pose and position in each frame. Figure-7.12 shows the frame processing time during the overall tracking process for both 50 and 200 particles. According to this figure the average processing time with 50 and 200 particles are about 20 and 3

frames per second respectively. When we compare the average processing times of the implemented tracking algorithm with 50, 100 and 200 particles, it is observed that the performance of the tracking algorithm with 200 particles is slower 3 times than it is with 100 particles, whereas the performance of the algorithm with 50 particles is faster 2.5 times than it is with 100 particles.

Also, in Figure-7.12 a decrease in the computational performance of the algorithm with 200 particles is obvious while the size of the tracked object increases. We know that the more pixels are involved in the warping process of the image regions indicated by the particles while the size of the object increases. But with 200 particles increase in processing time is multiplied by a higher constant (200) and so the change in computational time becomes more obvious.

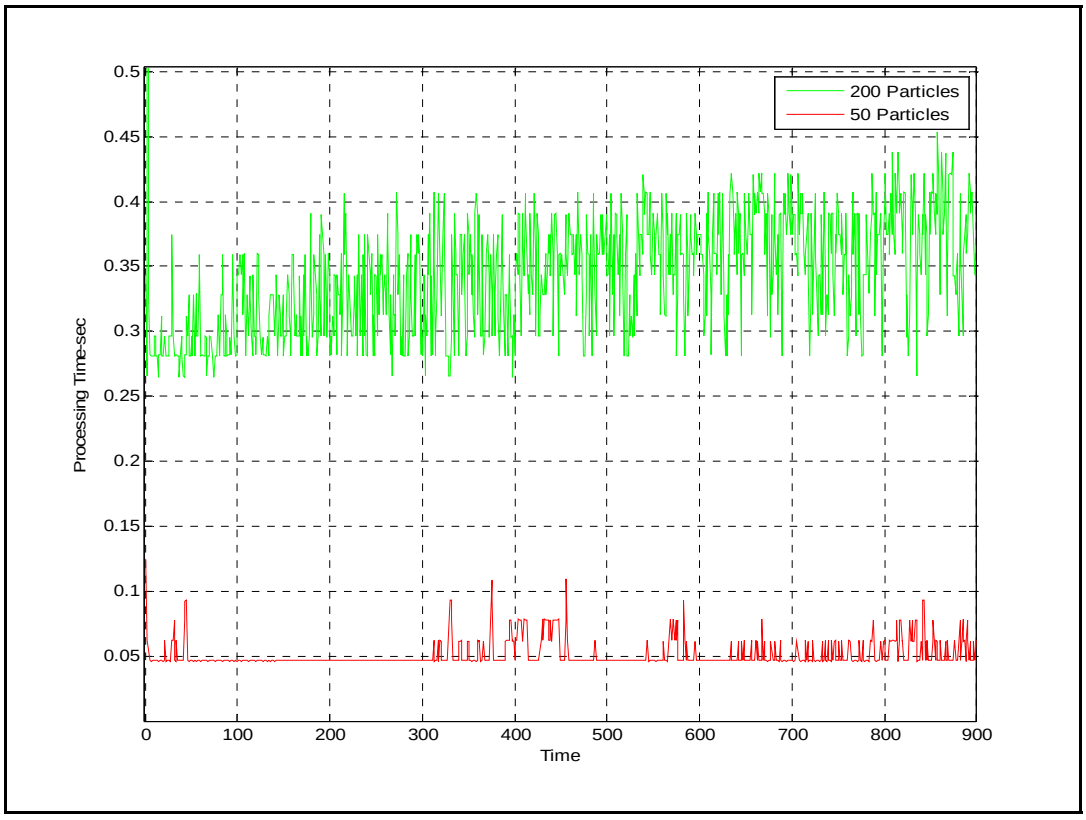

Figure-7.12 Performance of the Tracking Algorithm with 50 and 200 Particles

To demonstrate the effectiveness of Particle Filtering we compare the implemented algorithm with the algorithm that doesn't utilize Particle Filtering after the motion estimation. We know that using Particle Filtering after motion estimation increases the accuracy of the algorithm via hypothesized target states whose distribution in the state space depends on the quality of the motion estimation. In this sense we analyzed the performance of the algorithm and observed effect of Particle Filtering on synthetic video sequence.

On the other hand, in the implemented visual tracker the motion estimation process uses previous particle distribution and image patches cropped out of the previous frame. Therefore, after motion estimation which gives estimated target state vector, we have to perturb particles around the estimated state depending on the quality of the motion estimation. This distribution is only utilized for motion estimation in the next frame and not for the estimation of the target state vector precisely as in the implemented algorithm. Here, 100 particles are used to predict the target state vector in motion estimation procedure and other design parameters are left same as the first experiment. For this case Figure 7-13 illustrates the performance of the algorithm.

Figure-7.13 indicates that as time increases the performance of the algorithm degrades. The reason for this degradation is the increasing number of pixels classified as outliers. Since particle filtering isn't used to improve the prediction of the state vector after motion estimation, tracker slightly drifts away from the target region and so the number of outlier pixels increases.

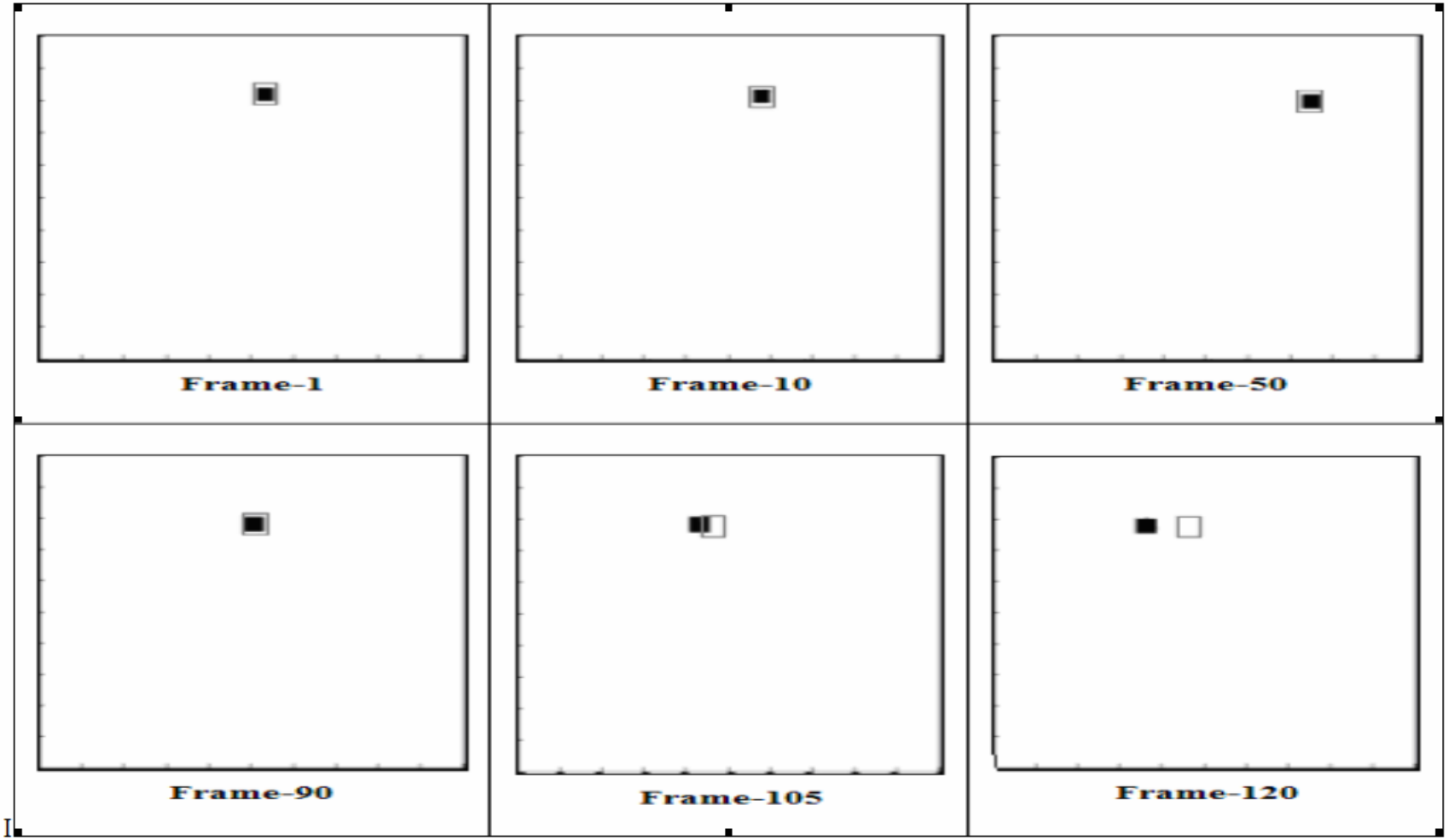

Figure-7.13 Tracking Results with 100 Particle used Only in Motion Estimation Process

Table-7.1 gives us the performance of the implemented algorithm on moving square under these test cases.

Table-7.1 Performance of the Algorithm under Different Test Cases

Test Case	With 50 Particles	With 100 Particles	With 200 Particles	With 100 Particles (Used in Motion Est.)
D-MSE	4.4	3.3	2.4	NA
S-MSE	0.005	0.005	0.004	NA
Average Proc. Time (frame/sec)	20	8	3	NA

Here, D-MSE, S-MSE denotes computed average MSE values for displacement variables and scale variable respectively. Note that when particle distribution is used only in motion estimation process, the implemented tracker fails to track the target after a period of time. Therefore NA (Not Applicable) is used for performance criteria.

In second test we track a human face using the same design parameters with 100 particles. In this video sequence both camera and object motion is present. Also, the scale of the human face changes and it rotates as well. Figure-7.10 shows the output images from the tracking algorithm where the white rectangle bounding the human face indicates the tracking result.

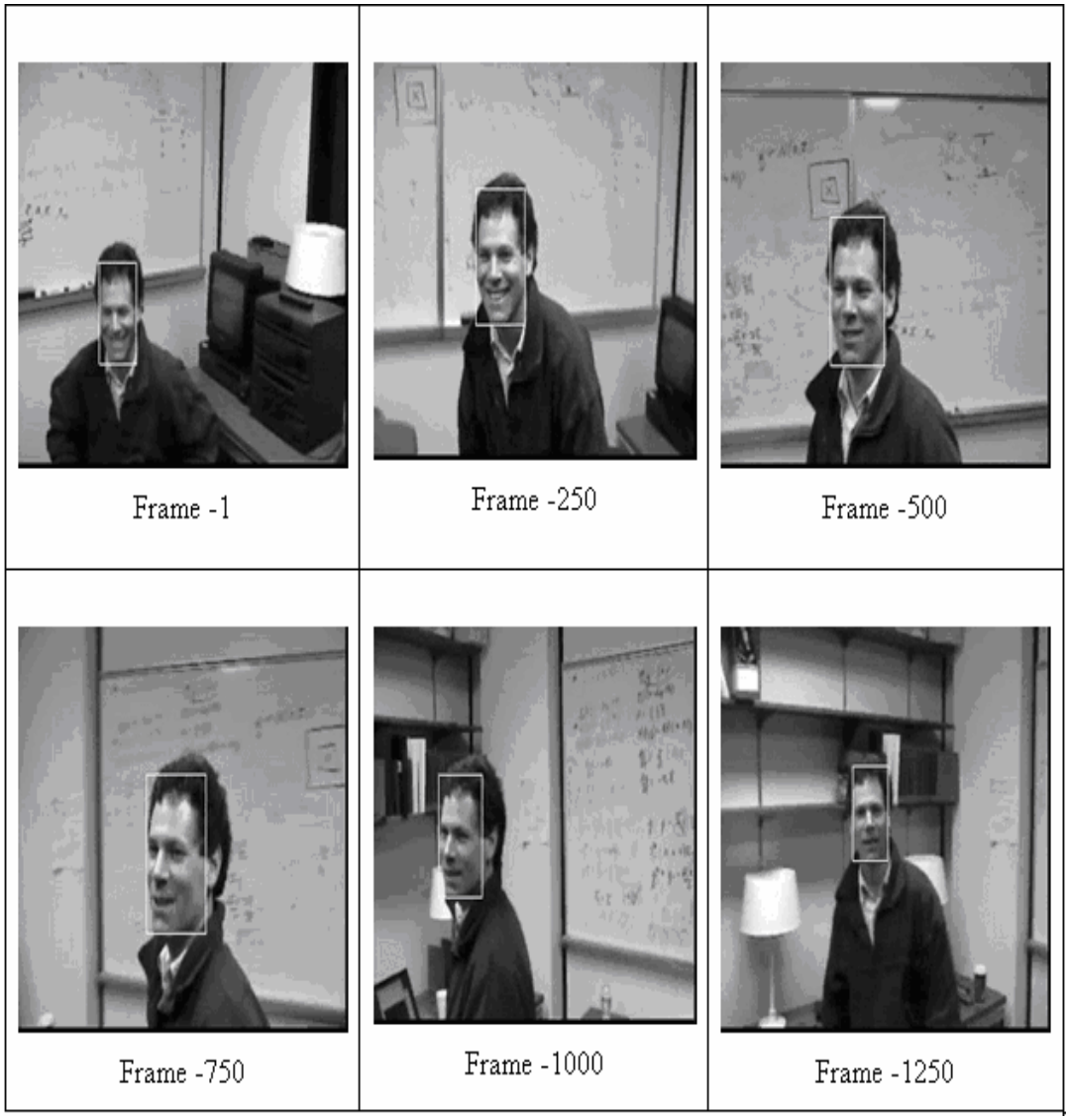

Figure-7.10 Tracking Results on Human Face with 100 Particles

From the results, the implemented visual tracking algorithm can correctly track the target in the video sequence even though appearance and size of the object changes during the overall tracking process. These changes make it difficult to lock on the human face, but with the adaptive appearance model, similarity transformation and particle filtering this algorithm resolve these difficulties.

Finally, we evaluate the proposed visual tracking algorithm in outdoor environment. A black moving car is selected as a target object where size and appearance of the

car changes during the tracking process. In Figure-7.11 some sample output images from the tracking algorithm is shown.

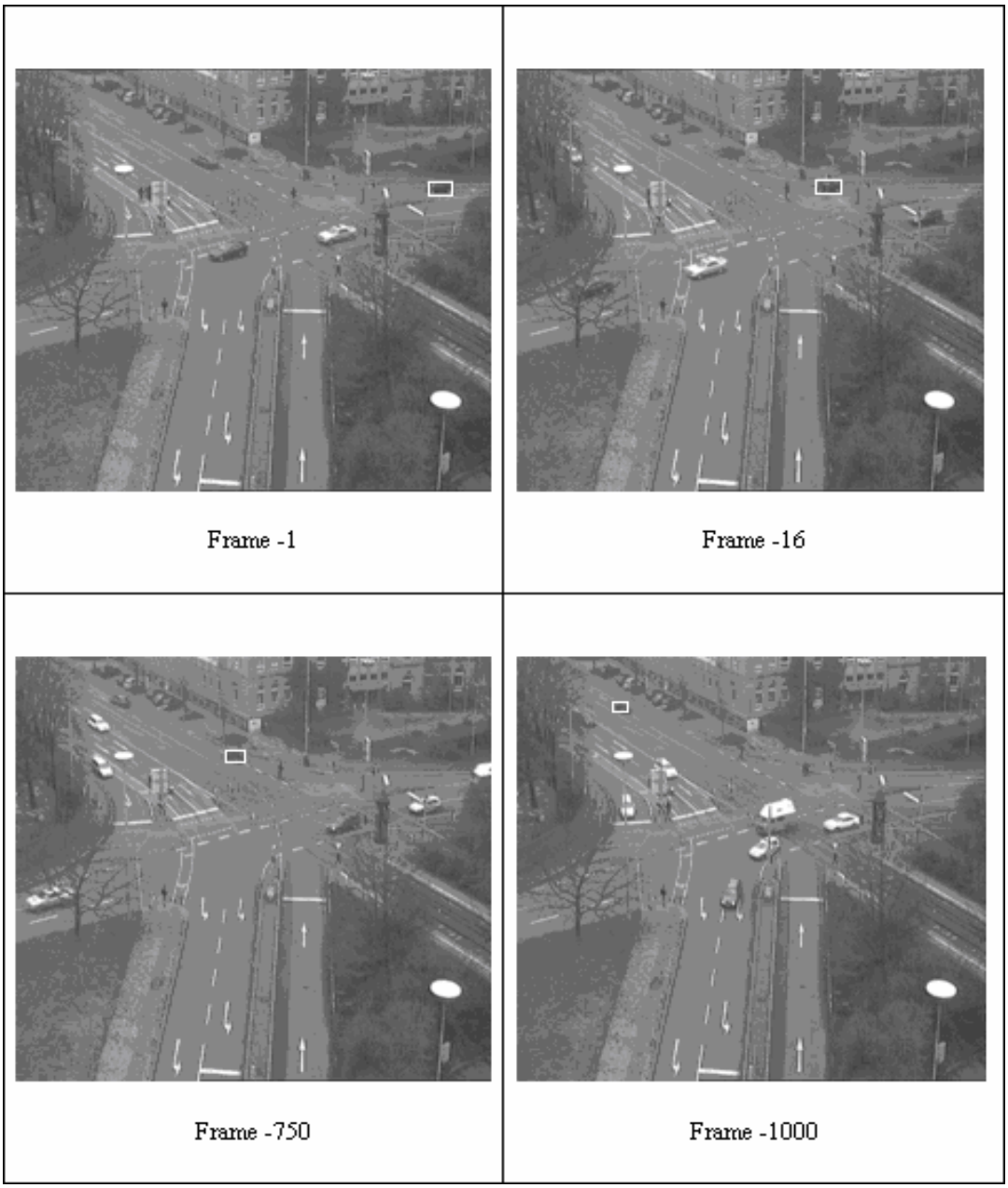

Figure-7.11 Tracking Results on Black Moving Car with 100 Particles

As seen from the above figure the proposed visual tracker has ability to track the selected object under pose and scale changes. However, we model the change in pose of the target as parametric similarity transformation thus; our model can't cover the different scale changes in different axis of the bounding box. Therefore the bounding box doesn't cover the whole of the target accurately when the scale isn't uniform for both of the axes.

CHAPTER 8

CONCLUSION

In this thesis we have studied the visual tracker proposed by Zhou et. al in [5]. However, we have used similarity transformation instead of affine transformation to make the algorithm simple. Also, our adaptive appearance model consists of two components rather than three ones. Moreover, the learning rate has been utilized to control the speed and accuracy of the motion estimation and some precautions have been taken to check that the motion estimation is acceptable or not.

Since our measurement model (cropped target region) is a nonlinear function of the state variables, the sequential Monte Carlo methods, or particle filtering has been used to solve the tracking problem. Also, the algorithm uses an adaptive appearance model that adapts to slowly varying features of the target while it maintains the stable ones of the observed image of the target during the tracking process. Furthermore, to cover the unexpected target motion the algorithm includes adaptive state transition model. By using the previous target appearance and particle set the motion estimation for the target pose and position is calculated then the particles are distributed around the expected target region by adaptive noise depending on the quality of the estimation thus, we obtain more accurate estimations for the target pose and position.

To examine the proposed tracker we have done several tests consisting of tracking visual objects in indoor and outdoor environments. In these tests we usually obtain good results. On the other hand our tracker can't track the object well when the target makes a complex motion composed of two or more basic geometric transformations: translation, rotation and change in scale. The reasons why performance of the tracker degrades in this situation are: First of all, much more

particles are needed to handle these kinds of complex target motions accurately. Although Zhou and et al in [5] have used adaptive number of particles we didn't benefit this approach to cope with this problem in order to run the algorithm in real-time as closely as possible. Secondly, the robust estimator prevent the estimate to be biased from the non-target pixels but in image warping process pixels don't belong to the target cause the motion estimation to deviate from the true value. Lastly, we have used similarity transformation that handles change in scale by a uniform scale parameter. However, when the target makes a motion and horizontal and vertical scales change in different magnitudes, the implemented tracker begins to cover non-target region and so it shift away from the tracked object. Since, the scale of the target depends on the configuration between the camera and the target, we think that it is better to calculate the scale of the target by considering target and camera positions thus our state space decreases and so the less number of particles are needed.

To cope with these problems under a single framework we proposed Rao-Blackwellized Particle Filter in Appendix-A. In this approach state space is partitioned into two parts. The target position is estimated in a numerical way via Kalman Filter while the non-uniform scale and rotation parameters are approximately inferred by Particle Filtering. By using Rao-Blackwellized Particle Filtering the algorithm allows more hypotheses for scale and rotation parameters by the same number of particles used in classical Particle Filter thus, the accuracy of the algorithm improves.

REFERENCES

[1] Gregory D. Hager and Peter N. Belhumeur, ,"Efficient Region Tracking With Parametric Models of Geometry and Illumination", IEEE Transactions on Pattern analysis and Machine Intelligence, Vol. 20, No. 10, October 1998

[2] Sebastian Nowozin, "Object Classification using Local Image Features", Master Thesis, Technische Universitat Berlin, May 8, 2006

[3] Jurie, F and Dhome, M., "Hyperplane Approximation for Template Matching", Pattern Analysis and Machine Intelligence, IEEE Transactions on Volume 24, Issue 7, Jul 2002 Page(s):996 – 1000

[4] Allan D. Jepson, David J. Fleety and Thomas F. El-Maraghi, "Robust Online Appearance Models for Visual Tracking", IEEE Conference on Computer Vision and Pattern Recognition, Kauai, 2001, Vol. I, pp. 415–422

[5] Shaohua Kevin Zhou, Rama Chellappa, and Baback Moghaddam, "Visual Tracking and Recognition Using Appearance-Adaptive Models in Particle Filters", Image Processing, IEEE Transactions on Volume 13, Issue 11, Nov. 2004 Page(s): 1491 – 1506

[6] Josephine Sullivan and Jens Rittscher, "Guiding Random Particles by Deterministic Search", Eighth International Conference on Computer Vision (ICCV'01) - Volume 1 p. 323

[7] Aswin C Sankaranarayanan, Rama Chellappa and Qinfen Zheng, "Tracking Objects in Video Using Motion and Appearance Models", Image Processing, 2005. ICIP 2005. IEEE International Conference on 11-14 Sept. 2005, Volume: 2, On page(s): II- 394-7

[8] Fleck, Sven; Hoffmann, McElory; Hunter, Karin; Schilling, Andreas, "PFAAM-An Active Appearance Model based Particle Filter for both Robust and Precise Tracking", Computer and Robot Vision, 2007. CRV apos;07. Fourth Canadian Conference, 28-30 May 2007, Page(s):339 - 346

[9] Sukwon Choi and Daijin Kim, "Robust Face Tracking Using Motion Prediction in Adaptive Particle Filters", Lecture Notes in Computer Science Publisher :Springer Berlin / Heidelberg ISSN0302-9743 (Print) 1611-3349 (Online) Volume 4633/2007 Image Analysis and Recognition

[10] Kawamoto, K., Kitakyushu, "Affine Modeling of Targets in Video Sequences by Particle Filters", World Automation Congress, WAC '06 on 24-26 July 2006, Budapest, On page(s): 1-6

[11] James R. Bergen, P. Anandan, Keith J. Hanna, and Rajesh Hingorani, “Hierarchical Model-Based Motion Estimation”, Proceedings of the Second European Conference on Computer Vision 1992, Pages: 237 - 252

[12] Branko Ristic, Sanjeev Arulampalam, Neil Gordon, “Beyond the Kalman Filter: Particle Filters for Tracking Applications” , Artech House Radar Library, 2004

[13] ZHE CHEN, “Bayesian Filtering: from Kalman Filters to Particle Filters, and Beyond”

[14] Arulampalam, M.S.; Maskell, S.; Gordon, N.; and Clapp, T., “A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking”, Signal Processing, IEEE Transactions on Volume 50, Issue 2, Feb 2002, Page(s):174–188

[15] Xinyu Xu and Baoxin Li “Rao-Blackwellised Particle Filter for Tracking with Application in Visual Surveillance”, Proceedings 2nd Joint IEEE International Workshop on VS-PETS, Beijing, October 15-16, 2005

[16] Gustaf Hendeby, Rickard Karlsson and Fredrik Gustafsson, “A New Formulation of the Rao-Blackwellized Particle Filter”, Statistical Signal Processing, 2007. SSP '07. IEEE/SP 14th Workshop on: 26-29 Aug. 2007, On page(s): 84-88

[17] Dong, J.F.; Wijesoma, W.S.; and Shacklock, A.P, “An Efficient Rao-Blackwellized Genetic Algorithmic Filter for SLAM”, Robotics and Automation, 2007 IEEE International Conference on, 10-14 April 2007, Page(s):2427 – 2432

[18] Grant Schindler and Frank Dellaert, ”A Rao-Blackwellized Parts-Constellation Tracker” , Springer Berlin, Book: Dynamical Vision, 2007, Pages:178-189

[19] Julian Eggert, Chen Zhang, and Edgar Körner, “Template Matching for Large Transformations”, Artificial Neural Networks, 17. International Conference (ICANN). Springer Verlag, September 14 2007, Volume 4669/2007, pages 169-179.

[20] Hailin Jin, Paolo Favaro, and Stefano Soatto, ”Real-Time Feature Tracking and Outlier Rejection with Changes in Illumination”, Eighth International Conference on Computer Vision (ICCV'01), July 2001, Volume 1 , Pages: 684-689

[21] Christoph Graβl, Timo Zinβer, and Heinrich Niemann “Illumination Insensitive Template Matching with Hyperplanes”, EURASIP Journal on Applied Signal Processing archive Volume 2005 , Issue 1 January 2005, Pages: 2375 - 2390

[22] B.D Ripley, “Robust Statistics ”, M.Sc. in Applied Statistics, 2004

[23] Rama Chellappa, and Shaohua Kevin Zhou, “Chapter 8. Face Tracking and Recognition from Video”, Handbook of Face Recognition, Springer

[24] Arthur Gelb, Joseph F. Kasper, Jr., Raymond A. Nash, Jr, Charles F. Price, Arthur A. Sutherland, Jr., “Applied Optimal Estimation”, THE M.I.T. PRESS Massachusetts Institute of Technology, Massachusetts, and London, England, 1974

[25] Samuel Blackman and Robert Popoli, “Design and Analysis of Modern Tracking Systems”, Artech House radar Library, 1999

[26] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P-J. Nordlund. “Particle filters for positioning, navigation and tracking”. IEEE Transactions on Signal Processing, 2002, Page(s): 425-437

[27] Schon, T.; Gustafsson, F.; and Nordlund, P.-J. “Marginalized Particle Filter for Mixed Linear/Nonlinear State Space Models”, Signal Processing, IEEE Transactions on Volume 53, Issue 7, July 2005, Page(s): 2279 - 2289

[28] Karlsson, R.; Schon, T.; Gustafsson, F. “Complexity Analysis of the Marginalized Particle Filter”, Signal Processing, IEEE Transactions on Volume 53, Issue 11, Nov. 2005, Page(s): 4408 - 4411

[29] Niklas Svenzen, “Real Time Implementation of Map Aided Positioning Using a Bayesian Approach”, Linköping Studies in Science and Technology, 2002

[30] Thomas Schön, “On Computational Methods for Nonlinear Estimation”, Linköping Studies in Science and Technology Thesis No. 1047

[31] Rafael C. Gonzales and Richard E. Woods, “Digital Image Processing”, Prentice Hall, 2008

[32] Yung-Yu Chuang, “Digital Visual Effects: Image Warping/Morphing”, Spring 2006

[33] Bernd Jahne, “Digital Image Processing”, Springer 2005.

[34] MIT Graphics Class Lecture-7, 6.837 Intro to Computer Graphics Fall 2001

[35] Philippe Thevenaz, Thierry Blu and Michael Unser, “Image Interpolation and Resampling”, in Handbook of Medical Image Processing, 2000

[36] Barbara Zitova, Jan Flusser, and Filip ˇ Sroubek, “Image Registration: A Survey and Recent Advances”, Institute of Information Theory and Automation Academy of Sciences of the Czech Republic, ICIP 2005

[37] Jaco Vermaak, Patrick Pérez, Michel Gangnet and Andrew Blake “Towards Improved Observation Models for Visual Tracking: Selective Adaptation”, Computer Vision ECCV 2002,Volume 2350/2002 Springer, Pages: 645-660

[38] The Expectation Maximization Algorithm, Frank Dellaert College of Computing, Georgia Institute of Technology Technical Report number GIT-GVU-02-20, February 2002

[39] Kevin Nickels and Seth Hutchinson, “Estimating Uncertainty in SSD-Based Feature Tracking”, Image and Vision Computing, 2002, volume 20 pages 47-58

[40] Richard Szeliski, Lecture Notes: “Motion estimation”, Introduction to Computer Vision CS223B, Winter 2005 Stanford University,

[41] Richard Szeliski, “Image Alignment and Stitching: A Tutorial”, Foundation and Trends in Computer Graphics and Vison, Volume 2, Issue 1, January 2006

[42] J-M Odobez and P Bouthemy,” "Robust Multiresolution Estimation of Parametric Motion models," J. Vis. Commun. and Image Repres., Vol. 6, No. 4, Pages. 348-365, 1995.

[43] A. D Jepson ,”2503:Robust Estimation”, Lecture Notes, Toronto University 2004

[44] Gilbert Strang, “Linear Algebra and Its Applications”, Third Edition, Thomson Learning, 1988

[45] Arnaud, E., and Memin, E., “An Efficient Rao-Blackwellized Particle Filter for Object Tracking”, mage Processing, 2005. ICIP 2005. IEEE International Conference on Volume 2, 11-14 Sept. 2005, Page(s): II - 426-9

[46] X. Xu, and B.Li, “Rao-Blackwellised Particle Filter for Tracking with Application in Visual Surveillance”, Second Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS), in conjunction with ICCV 2005, Oct 15-21, Beijing.

[47] Schon, T.B., Karlsson, R.,and Gustafsson, F., “The Marginalized Particle Filter in Practice”, Aerospace Conference, 2006 , 4-11 March 2006, Page(s): 11 pp. –

[48] Yaakov Bar-Shalom, X. Rong Li, and Thiagalingam Kirubarajan, “Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software “, Wiley, New York, 2001.

[49] Gustaf Hendeby, “Performance and Implementation Aspects of Nonlinear Filtering” Linköping studies in science and technology. Dissertations. No. 1161

[50] Thomas Schön, ”Introducing the Marginalized Particle Filter-Exploiting Structures in State-Space Models”, 5th Swedish-Russian Control Conference

APPENDIX A

FUTURE WORK

Rao-Blackwellized Particle Filtering

Sequential Monte Carlo methods (Particle Filters) aim to approximate posterior distribution. To do so, samples are drawn from state space according to an importance distribution. However, huge number of particles is required when the dimension of the state space is large. Otherwise, particle filtering becomes unsuccessful to appropriately represent the posterior distribution [45]. Nevertheless, the great number of particles makes particle filtering technique inappropriate to real time applications. To overcome this problem dynamic state space is partitioned into two subspaces if it is analytically possible. According to generalized state space model [27] the first part of the state space has variables that evolve according to a nonlinear system equation. These variables contain information about the second part of the state space and also they have nonlinear relationship with observations in measurement equation. Given these state variables the second part of the state space consists of the state variables which system and measurement models are both linear and subject to Gaussian noises. Since state space models are linear and subject to Gaussian noises for these state variables Kalman Filter gives optimal solution for the evaluation of their posterior distribution whereas particle filtering is appropriate to evaluate the posterior distribution of the first set of state variables having nonlinear state space models [46].

To get more insight about state space models consider a state vector Θ_t which is partitioned as

$$\Theta_t = \left[\Theta_t^p \;\; \Theta_t^l \right] \tag{A.1}$$

Where Θ_t^l denotes state vector in linear-Gaussian environment and Θ_t^p denotes nonlinear state vector represented by particles. The state space models for these variables are established as follows:

$$\Theta_{t+1}^{p} = f_t^p\left(\Theta_t^p\right) + A_t^p\left(\Theta_t^p\right)\Theta_t^l + G\left(\Theta_t^p\right)w_t^p \tag{A.2}$$

$$\Theta_{t+1}^{l} = f_k^l\left(\Theta_t^p\right) + A_t^l\left(\Theta_t^p\right)\Theta_t^l + G\left(\Theta_t^p\right)w_t^l \tag{A.3}$$

$$y_t = h\left(\Theta_t^p\right) + H\left(\Theta_t^p\right)\Theta_t^l + e_t \tag{A.4}$$

Where the system and measurement noises (w_t^p, w_t^l and e_t) are assumed to be independent zero-mean and Gaussian. As mentioned before it is clear that given nonlinear state Θ_t^p there is a linear-Gaussian sub-structure in (A.3) and (A.4).

The technique mentioned above is called Rao-Blackwellization and it improves the performance of the algorithm as variance reduction. Performance improvement (variance reduction) can be justified in two ways: (i) Intuitively, Kalman Filter is the optimum estimator in MMSE sense for states in linear and Gaussian environment and also the dimension of the posterior distribution where particles are sampled is decreased. Consequently; the smaller dimension of the state space is represented by the particles, so the posterior distribution is approximated more effectively. (ii) Theoretical justification of variance reduction thus the accuracy improvement in Rao-Blackwellized particle filter can be given as follows [13].

Let g(Θ_t^p, Θ_t^l) be a function of these two random variable sets. The expectation of this function w.r.t the posterior distribution is established as

$$E\left[g\left(\Theta_t^p, \Theta_t^l\right)\right] = \iint g\left(\Theta_t^p, \Theta_t^l\right) p\left(\Theta_t^p, \Theta_t^l \middle| \mathbf{Y_t}\right) d\Theta_t^l d\Theta_t^p \tag{A.5}$$

and this quantity is assumed to be an estimator of a function of state. Here, the linear state variables are marginalized out as given in [13]:

$$\lambda\left(\Theta_t^p\right) = \int g\left(\Theta_t^p, \Theta_t^l\right) p\left(\mathbf{Y_t} \middle| \Theta_t^p, \Theta_t^l\right) p\left(\Theta_t^l \middle| \Theta_t^p\right) d\Theta_t^l \tag{A.6}$$

Using the above definition, expectation of the estimator can be rewritten as follows

$$E\left[g\left(\Theta_t^p,\Theta_t^l\right)\right]=\frac{\int\lambda\left(\Theta_t^p\right)p\left(\Theta_t^p\right)d\Theta_t^p}{\int p\left(\mathbf{Y_t}\middle|\Theta_t^p\right)p\left(\Theta_t^p\right)d\Theta_t^p}$$
$$=\frac{\int\lambda\left(\Theta_t^p\right)p\left(\Theta_t^p\right)d\Theta_t^p}{p\left(\mathbf{Y_t}\right)} \tag{A.7}$$

By exploiting linear substructure conditioned on Θ_t^p the linear Gaussian state variables are marginalized out and so the estimation of the marginalized variables becomes analytically possible via an optimal filter (Kalman Filter) [27]. In other words, the density $p\left(\Theta_t^l\middle|\Theta_t^p\right)$ is analytically tractable whereas $p\left(\Theta_t^p\right)$ is approximately inferred by particle filtering [13] as

$$p\left(\Theta_t^p\right)=\sum_{i=1}^{N}\omega_t^{p,i}\delta\left(\Theta_t^{p,i}-\Theta_t^p\right) \tag{A.8}$$

Where, $\omega_t^{p,i}$ is the associated importance weight of the ith particle and N denotes the number of particles.

Thus, the expectation of the estimator conditioned on the nonlinear part is given by

$$E\left[g\left(\Theta_t^p,\Theta_t^l\right)\middle|\Theta_t^p\right]\approx\sum_{i=1}^{N}\omega_t^{p,i}\lambda\left(\Theta_t^{p,i}\right) \tag{A.9}$$

The variance reduction is the result of the information about the nonlinear state variables that ease to the process of the estimation. The following theorem sometimes referred to as Rao-Blackwellization enables to compare the variance of crude (standard PF) and marginalized particle Filters [47]

$$\mathrm{var}\left[g\left(\Theta_t^p,\Theta_t^l\right)\right]=\mathrm{var}\left[E\left\{g\left(\Theta_t^p,\Theta_t^l\right)\middle|\Theta_t^p\right\}\right]+E\left[\mathrm{var}\left(g\left(\Theta_t^p,\Theta_t^l\right)\middle|\Theta_t^p\right)\right] \tag{A.10}$$

Since the second term on the right hand side of the equation is nonnegative it yields the following conditional inequality

$$\mathrm{var}\left[E\left\{g\left(\Theta_t^p,\Theta_t^l\right)\middle|\Theta_t^p\right\}\right]\le\mathrm{var}\left[g\left(\Theta_t^p,\Theta_t^l\right)\right] \tag{A.11}$$

Thus, it has been shown that the variance of the conditional estimator is less than the variance of the crude one. See Appendix-B for the proof of (A.11)

Utilization of Kalman Filter for each particle improves the accuracy of the algorithm whereas the computational complexity depends on the state space models used in Rao-Blackwellization [12], [27].

In this thesis the full system state vector, Θ_t parameterize a motion model consisting of non-uniform scaling, rotation and translation. Parameters of this geometric transformation can be divided into two groups: (i) deformation parameters s_x , s_y and Φ, (ii) translation parameters d_x and d_y. First component of the full system state vector i.e., Θ_t^p includes the deformation parameters related to non-uniform scaling and rotation of the object whereas second one, denoted as Θ_t^l consists of the parameters related to displacement and their derivatives.

$$\Theta_t^p = \begin{bmatrix} s_t^x & s_t^y & \Phi_t \end{bmatrix} \tag{A.12}$$

$$\Theta_t^l = \begin{bmatrix} d_{x,t} & \dot{d}_{x,t} & d_{y,t} & \dot{d}_{y,t} \end{bmatrix} \tag{A.13}$$

The motion model that represents the evolution law of the nonlinear state vector can be written as

$$\Theta_{t+1}^p = \Theta_t^p + v_t^p + u_t^p \tag{A.14}$$

Where v_t^p is the part of the adaptive velocity estimate related to deformation parameters. The model is driven by process noise u_t^p which is assumed to be independent zero mean and Gaussian of covariance

$$C_t^p = \mathrm{diag}\begin{bmatrix} \mathrm{q}_{s_x} & \mathrm{q}_{s_y} & \mathrm{q}_{\Phi} \end{bmatrix} \tag{A.15}$$

Where q_{s_x} q_{s_y} and q_{Φ} denote variances for each deformation variable.

Adaptive velocity v_t is recursively estimated until a significant convergence of the estimate is achieved. The details of the computation are given in Chapter 5. The important feature of the adaptive velocity estimation is that it is a linear function of both parts of the state space variables. The following equation is used for estimation of the adaptive velocity:

$$v_t = g\left(\Theta_t^p, \Theta_t^l\right) = \Theta_t - \hat{\Theta}_{t-1} \tag{A.16}$$

The right hand side of the above equation is equal to a constant and its computation according to ith particle is obtained using (5.7) as

$$v_t^i = \left[\nabla_{\Theta_l}{}^T I\{f(\chi;\Theta_l),t\}\Big|_{\Theta_l=\Theta_{l-1}^i} \nabla_{\Theta_l} I\{f(\chi;\Theta_l),t\}\Big|_{\Theta_l=\Theta_{l-1}^i}\right]^{-1} \nabla_{\Theta_l}{}^T I\{f(\chi;\Theta_l),t\}\Big|_{\Theta_l=\Theta_{l-1}^i} \left[I\{f(\chi;\hat{\Theta}_{t-1}),t-1\} - I\{f(\chi;\Theta_{t-1}^i),t\}\right] \tag{A.17}$$

Where Θ_{t-1}^i and $\hat{\Theta}_{t-1}$ are respectively the system state vector represented by ith particle with predicted linear states and the estimated target state vector at time t-1. Also, f is the image warping function that transforms the pixel coordinates according to the given transformation parameters and this transformation is expressed as

$$f\left(\chi,\Theta_t\right) = \begin{bmatrix} s_x\cos(\Phi) & -s_x\sin(\Phi) \\ s_y\sin(\Phi) & s_y\cos(\Phi) \end{bmatrix}\chi + \begin{bmatrix} \mathrm{d_x} \\ d_y \end{bmatrix} \tag{A.18}$$

Here, the transformation is implemented by using the deformation parameters to distort the whole image geometrically and then, target region centered at $\chi_{\text{target}} = \{\mathrm{x}_{\text{target}}, \mathrm{y}_{\text{target}}\}$ is cropped with the same size as the target template [5]. In this procedure, the center of the target is calculated by the estimated variables via Kalman Filter.

In (A.17) the evaluation of image gradients w.r.t previous particle states increases the computational load. Therefore, we prefer calculating the template image gradient by assuming that the parameter set represented by the ith particle and associated linear states gives the exact target state. Using the brightness constraint and (5.13) it can be shown that the image gradient can be written in terms of template image gradient as [1]

$$\nabla_f I\{f(\chi;\Theta_t),t\} = \nabla_x I\left(\chi,\mathrm{t}_0\right)^T f_x(\chi;\Theta_t)^{-1} \tag{A.19}$$

Thus the jacobian matrix in (A.17) can be redefined in accordance with our state variables as

$$\nabla_{\Theta_t} I\{f(\chi,\Theta_t),t\} = \begin{bmatrix} \left[\frac{\partial}{\partial x} I(\chi,t_0)\ \frac{\partial}{\partial y} I(\chi,t_0)\right]_{\chi=\chi_1} f_\chi\left(\chi,\Theta_{t-1}^i\right)^{-1}\Big|_{\chi=\chi_1} f_{\Theta_t}\left(\chi_1,\Theta_t\right)\Big|_{\Theta_t=\Theta_{t-1}^i} \\ \left[\frac{\partial}{\partial x} I(\chi,t_0)\ \frac{\partial}{\partial y} I(\chi,t_0)\right]_{\chi=\chi_2} f_\chi\left(\chi,\Theta_{t-1}^i\right)^{-1}\Big|_{\chi=\chi_2} f_{\Theta_t}\left(\chi_2,\Theta_t\right)\Big|_{\Theta_t=\Theta_{t-1}^i} \\ \cdots \\ \left[\frac{\partial}{\partial x} I(\chi,t_0)\ \frac{\partial}{\partial y} I(\chi,t_0)\right]_{\chi=\chi_N} f_\chi\left(\chi,\Theta_{t-1}^i\right)^{-1}\Big|_{\chi=\chi_N} f_{\Theta_t}\left(\chi_N,\Theta_t\right)\Big|_{\Theta_t=\Theta_{t-1}^i} \end{bmatrix} \quad \text{(A.20)}$$

Where $f_\chi(.)$ and $f_{\Theta_t}(.)$ are respectively Jacobian matrices w.r.t the spatial coordinates of the target pixels and transformation parameters. Using (A.18) these Jacobian matrices are established as:

$$f_\chi\left(\chi,\Theta_t\right) = \begin{bmatrix} s_x\cos(\Phi) & -s_x\sin(\Phi) \\ s_y\sin(\Phi) & s_y\cos(\Phi) \end{bmatrix} \quad \text{(A.21)}$$

$$f_{\Theta_t}\left(\chi,\Theta_t\right) = \begin{bmatrix} \cos(\Phi)x-\sin(\Phi)y & 0 & -s_x\sin(\Phi)x-s_x\cos(\Phi)y & 1 & 0 \\ 0 & \sin(\Phi)x+\cos(\Phi)y & s_y\cos(\Phi)x-s_y\sin(\Phi)y & 0 & 1 \end{bmatrix} \quad \text{(A.22)}$$

The computation of the adaptive velocity estimate using previous particle set and predictions of the linear-Gaussian state variables is carried out as given in Algorithm A.1

Algorithm A.1: Adaptive Velocity Estimation and Particle Prediction

i. By using the previous particle states $\Theta_{t-1}^{p,i}$ geometrically transform the incoming image and determine the center of the hypothesized target region via the predictions of the linear-Gaussian states represented by $\Theta_{t|t-1}^{l,i}$. Then, compute the estimate of the adaptive velocity $v_t^i = \left[v_t^{p,i}\ v_t^{l,i}\right]$ for each particle as given in (A.17).

ii. Measure the quality of the estimation as explained in Chapter 5 then determine the best linear and nonlinear states. Note that if the adaptive velocity isn't good

enough to terminate the estimation, iterations are performed using these states until adaptive velocity estimation converges.

a. If the adaptive velocity estimation converges, terminate the iterations and set the predicted nonlinear states as

$$\Theta_t^{p,i} = \Theta_{t-1}^{p,*} + v_{t,K}^{p,*} + u_t^i \tag{A.23}$$

Where $\Theta_{t-1}^{p,*}$ denotes the state of the particle which gives the best estimate of the adaptive velocity $v_t^{i,*}$. The term u_t^i is the realization of the adaptive process noise that depends on the estimation quality as given in [5].

Meanwhile, using the summation of the shifts in displacement parameters the pseudo measurement of the linear Gaussian states is obtained as

$$z_t^l = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \Theta_{t|t-1}^{l,*} + V_{t,K}^{l,*} \tag{A.24}$$

Where $\Theta_{t|t-1}^{l,*}$ is the predicted linear Gaussian state associated with the particle which gives the best adaptive velocity estimation $v_t^{i,*}$.

b. Otherwise, if the number of iterations doesn't exceed a predetermined threshold, continue the iterative adaptive velocity estimation such that update the states represented by the selected particle via adaptive velocity estim in the Kth iteration and take the summation of the estimated displacement values as follows:

$$\Theta_t^{p,*} = \Theta_t^{p,*} + v_{t,K}^{p,*} \tag{A.25}$$

$$V_{t,K}^{l,*} = v_{t,K}^{l,*} + V_{t,K-1}^{l,*} \ \textit{where}\ V_{t,K-1}^{l,*} = \sum_{k=1}^{K-1} v_{t,k}^{l,*} \tag{A.26}$$

c. Otherwise, declare occlusion and set occlusion flag true

By using the particles in the estimation process of the adaptive velocity, we will benefit from the particles diversity on SSD surface and so the search becomes more robust. Otherwise, estimation is susceptible to be trapped in local optima and fail to converge to global optimum. Furthermore, estimating adaptive velocity iteratively indicates that samples are drawn from the distribution space denoted as $p\left(\Theta_t^p \middle| \Theta_t^l, \Theta_{t-1}^p\right)$. In this way new scale and rotation variables are obtained by propagating Θ_t^p according to its state transition model.

In this thesis the system dynamic equation for Θ_t^l is assumed to have a white noise acceleration model. This model is represented as

$$\Theta_t^l = F\Theta_t^l + \Gamma u_t^l \tag{A.27}$$

Where state transition matrix F and noise gain Γ are given by

$$F = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.28}$$

$$\Gamma = \begin{bmatrix} T^2/2 & 0 \\ T & 0 \\ 0 & T^2/2 \\ 0 & T \end{bmatrix} \tag{A.29}$$

The noise term Γu_k^l is zero mean Gaussian of covariance given by

$$C_t^l = block - \text{diag}\left[\Psi \mathrm{q}_{t_x} \;\; \Psi \mathrm{q}_{t_y}\right] \tag{A.30}$$

Where, q_{t_x} and q_{t_y} denote nominal variance of the corresponding state variables and matrix Ψ is written as

$$\Psi = \begin{bmatrix} T^4/4 & T^3/2 \\ T^3/2 & T^2 \end{bmatrix} \tag{A.31}$$

After the nonlinear state of the target is predicted, Kalman Filter updates the predictions of the linear states and so the posterior distribution of the linear states

denoted as $p\left(\Theta_t^l \middle| \Theta_t^p\right)$ is computed analytically. This abstract representation implies that the updated linear Gaussian states are obtained using the information from the prediction of nonlinear states. In other words it means that the linear Gaussian states are updated with pseudo-measurement defined in (A.24). The numerical counter part of this abstract notation is established as follows

$$\Theta_{t|t}^l = \Theta_{t|t-1}^l + K_t^l\left(z_t^l - H\Theta_{t|t-1}^l\right) \tag{A.32}$$

$$P_{t|t}^l = \left[\mathrm{I} - K_v^l H\right] P_{t|t-1}^{l,*} \tag{A.33}$$

$$K_t^l = P_{t|t-1}^{l,*} H^T \left(HP_{t|t-1}^l H^T\right)^{-1} \tag{A.34}$$

Where the pseudo measurement equation is defined as follows

$$z_t^l = H\Theta_t^l \tag{A.35}$$

with, H is given by

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{A.36}$$

Here, the pseudo measurement obtained from adaptive velocity estimation is assumed to be noiseless under the assumption that occlusion isn't declared. On the other hand, the variance of the noise entering the linear and Gaussian system is calculated in a similar way given in [5] thus the motion model is adapted to the changes in kinematic behavior of the target.

Before calculating the (A.32), (A.33) and (A.34) the process noise entering the linear Gaussians systems must be adapted.

First of all, the innovation between the predicted measurement and pseudo measurement is computed as

$$\tilde{z}_t^{l,i} = z_t^l - H\Theta_{t|t-1}^{l,i} \tag{A.37}$$

Then, making the use of the innovation and priori covariance matrix $P_{t|t-1}^{l,i}$ calculate the normalized innovation squared (NIS)

$$\varepsilon_t^{l,i} = \left(\tilde{z}_t^{l,i}\right)^T \left(HP_{t|t-1}^{l,i}H^T\right)^{-1}\left(\tilde{z}_t^{l,i}\right) \tag{A.38}$$

If the system model is linear and Gaussian the normalized innovation squared becomes chi-square distributed with two degrees of the freedom. Then

$$E\left[\varepsilon_t^{l,i}\right] = 2 \tag{A.39}$$

Therefore, the upper and the lower threshold values ($\varepsilon_{\max}$ and $\varepsilon_{\min}$) for NIS can be selected from chi-square table [48]. These thresholds are used in maneuver detection i.e., if the target deviates from the route which is defined by its system model, chi-square test informs the algorithm to update the noise levels [48] and so the motion of the target can still be modeled via our state transitional model. The following inequalities point out how to determine these thresholds from the chi-square table

$$\mathrm{P}\left\{\varepsilon_t^l < \varepsilon_{\max}\right\} = 1-\mu \tag{A.40}$$

$$\mathrm{P}\left\{\varepsilon_t^l > \varepsilon_{\min}\right\} = 1-\mu \tag{A.41}$$

Where μ is a design parameter usually express a small tail probability thus in our work it is assigned to 0.05, which makes $\varepsilon_{\max} \approx 6$ and $\varepsilon_{\min} \approx 0.1$.

After calculating the thresholds the noise entering the system models of the linear and Gaussian states $\Theta_t^{l,i}$ is adapted as follows

$$\zeta_{t,j}^{l,*} = \max\left\{\min\left(\sqrt{\varepsilon_{\max}}, \sqrt{\varepsilon_t^{l,i}}\right), \sqrt{\varepsilon_{\min}}\right\}\sqrt{q_{\text{nominal},j}^l}, \quad j = x, y \tag{A.42}$$

Where the term $q_{\text{nominal},j}^l$ denotes the nominal variance of the translational motion parameters for the axis indicated by j and $\zeta_{t,j}^{l,*}$ is the updated standard deviation of the process noise for the corresponding axis.

Using the adjusted process noises recalculate the priori covariance matrix $P_{t|t-1}^{l,*}$ as.

$$P_{t|t-1}^{l,*} = FP_{t-1|t-1}^{l}F^{T} + C_{t}^{l,*} \tag{A.43}$$

Where $C_t^{l,*}$ denotes the updated system covariance matrix with adjusted process noises using (A.42) and it is given by

$$C_t^{l,*} = block - \text{diag}\left[\Psi \mathrm{q}_{t_x}^{l,*}\ \Psi \mathrm{q}_{t_y}^{l,*}\right] \tag{A.44}$$

Here, $\mathrm{q}_{t_x}^{l,*}$ and $\mathrm{q}_{t_y}^{l,*}$ are square of the adjusted standard deviations computed via (A.42) and designates the new noise variances for motion along the x and y axes.

As a result by updating the process noise the system model that defines the translational motion of the target is adjusted. Hence, if the target motion deviates from its predetermined model, the priori covariance matrix will be increased and so Kalman gain will get higher values to give more importance to the measurement innovation. Otherwise, if the model fits the motion of the target well enough, the priori covariance matrix will decrease up to some level that depends on $\varepsilon_{\min}$. In this sense Rao-Blackwellized particle can be seen as a Kalman Filter bank with adaptive system models whose parameters are statistically adjusted [49].

In this thesis the iteratively estimated adaptive velocity, which includes updates to the parameters related to the displacements, is used as a pseudo measurement for the linear Gaussian states. As it is mentioned in Chapter-5 one of the fundamental problems for the motion estimation algorithms is the great displacements of the target between successive frames. Our method handles this problem via Kalman predictions. Since convergence of the adaptive velocity principally depends on the estimate of the initial displacement, our method will also help to decrease the number of iterations in the computation of adaptive velocity estimation.

To update the predictions of the states represented by particles associated importance weights must be calculated. With optimum linear Gaussian state $\Theta_t^{l,*}$ and particle filter predictions $\Theta_t^{p,i}$ a nonlinear measurement model is employed to compute the likelihood of the states represented by particles. Thus, the new weight measuring the quality of the nonlinear states is obtained. Using the updated

covariances $P_{t|t}^{l,i}$ optimum linear state vector $\Theta_t^{l,*}$ is computed as a weighted sum of the updated linear states. The equation that computes the optimum linear state is established as

$$\hat{\Theta}_t^{l,*} = \sum_{i=1}^{N} \left(\tilde{P}_{t|t}^{l,i} \right)^{-1/2} \Theta_{t|t}^{l,i} \tag{A.45}$$

Where $\left(\tilde{P}_{t|t}^{l,i} \right)^{-1/2}$ denotes the normalized inverse square root of the posteriori covariance matrix computed by Kalman Filter associated to the ith particle. To decrease the computational complexity inverse square root of the posteriori covariance matrix is computed by neglecting the off-diagonal elements and then inverting square root of the obtained diagonal matrix. In order to provide these weights to represent a true pdf, we normalize them as follows

$$\left(\tilde{P}_{t|t-1}^{l,i} \right)_k^{-1/2} = \frac{\left(P_{t|t}^{l,i} \right)_k^{-1/2}}{\sum_{i=1}^{N} \left(P_{t|t}^{l,i} \right)_k^{-1/2}} \tag{A.46}$$

Here $\left(\tilde{P}_{t|t-1}^{l,i} \right)_k^{-1/2}$ is the kth diagonal of the corresponding matrix.

Using the optimum linear state the computation of the importance weights are obtained as

$$w_t^i = \int_{\Theta_t^l} p\left(\mathrm{I_t}, \Theta_t^l \middle| \Theta_t^p \right) d\Theta_t^l \tag{A.47}$$

Here, since $p\left(\mathrm{I_t}, \Theta_t^l \middle| \Theta_t^p \right)$ is a Gaussian density by taking the integral over the linear state we obtain the measurement likelihood denoted as $p\left(\mathrm{I_t} \middle| \Theta_t^p \right)$.

Thus, as given in [18] the importance weights of the particles can be shown to be

$$w_t^i \cong p\left(I_t, \Theta_t^{l,*} \middle| \Theta_t^{p,i}\right)$$

$$= \prod_{n=1}^{N} \left\{ \sum_{j=s,w} m_{j,k-1} \frac{1}{\sqrt{2\pi\sigma_{j,t}^2}} e^{\frac{\left(I(w(\chi_n,\Theta_t^l),t)-\mu_{j,t-1}\right)^2}{2\sigma_{j,t}^2}} \right\} \Bigg|_{\Theta_t^i = \left[\Theta_t^{p,i}, \hat{\Theta}_t^{l,*}\right]} \tag{A.48}$$

After calculating the importance weights of each particle the expected mean of the nonlinear state is computed in either minimum mean square error sense (MMSE),

$$\hat{\Theta}_t^{p,MMSE} = \sum_{i=1}^{N} \omega_t^{p,i} \Theta_t^{p,i} \tag{A.49}$$

or the maximum a posteriori sense (MAP)

$$\hat{\Theta}_t^{p,MAP} = \underset{\omega_t^{p,i}}{\arg\max}\left(\Theta_t^{p,i}\right) \tag{A.50}$$

Finally, using the estimated nonlinear states and linear Gaussian states ($\left\{\hat{\Theta}_t^p, \hat{\Theta}_t^{l,*}\right\}$) the most probable target region is extracted from the current image by the geometric transformation and then, with the pixel intensities within that region target appearance model is updated as given in Chapter 4.

So far three probability density functions (pdf), one obtained for Kalman Filter measurement update with pseudo-measurements and remaining ones are the Particle Filter prediction and measurement likelihood, are used in order to estimate the target state. Justification of why these pdfs are used in Particle Filter and Kalman Filter are explained by deriving the filtering density $p\left(\Theta_{t,}^l \Theta_t^p \middle| \mathrm{I}_{1:t}\right)$ where $\mathrm{I}_{1:t}$ denotes the history of the observed target images in the video frame. In Bayesian framework this pdf is recursively estimated as

$$p\left(\Theta_t^l, \Theta_t^p \middle| \mathrm{I}_{1:t}\right) \propto p\left(\mathrm{I}_t \middle| \Theta_t^l, \Theta_t^p\right) \iint p\left(\Theta_t^l, \Theta_t^p \middle| \Theta_{t-1}^l, \Theta_{t-1}^p\right) p\left(\Theta_{t-1}^l, \Theta_{t-1}^p \middle| \mathrm{I}_{1:t-1}\right) d\Theta_{t-1}^l d\Theta_{t-1}^p$$

$$= p\left(\mathrm{I}_t \middle| \Theta_t^l, \Theta_t^p\right) \iint p\left(\Theta_t^l \middle| \Theta_{t-1}^l, \Theta_{t-1}^p\right) p\left(\Theta_t^p \middle| \Theta_t^l, \Theta_{t-1}^p, \Theta_{t-1}^l\right) p\left(\Theta_{t-1}^l, \Theta_{t-1}^p \middle| \mathrm{I}_{1:t-1}\right) d\Theta_{t-1}^l d\Theta_{t-1}^p \tag{A.51}$$

$$= p\left(\mathrm{I}_t \middle| \Theta_t^l, \Theta_t^p\right) \iint p\left(\Theta_t^l \middle| \Theta_{t-1}^l\right) p\left(\Theta_t^p \middle| \Theta_t^l, \Theta_{t-1}^p\right) p\left(\Theta_{t-1}^l, \Theta_{t-1}^p \middle| \mathrm{I}_{1:t-1}\right) d\Theta_{t-1}^l d\Theta_{t-1}^p$$

Where the expressions on the right hand side of the above equation are evaluated by the following steps:

Time Updates:

$$p\left(\Theta_t^l \middle| \Theta_{t-1}^p\right) = \int p\left(\Theta_t^l \middle| \Theta_{t-1}^l\right) p\left(\Theta_{t-1}^l \middle| \Theta_{t-1}^p\right) d_{\Theta_{t-1}^l} \tag{A.52}$$

Where $p\left(\Theta_{t-1}^l \middle| \Theta_{t-1}^p\right)$ is obtained from $p\left(\Theta_{t-1}^l, \Theta_{t-1}^p \middle| \mathrm{I}_{1:t\text{-}1}\right)$ by using Bayes Theorem

The above probability density function (pdf) implies that the linear states evolve according to their system model before new measurement becomes available. Once the time update of the linear states has been performed they are utilized to predict the nonlinear states according to

$$p\left(\Theta_t^p, \Theta_t^l \middle| \mathrm{I}_{1:t-1}\right) = \int p\left(\Theta_t^p \middle| \Theta_t^l, \Theta_{t-1}^p\right) p\left(\Theta_t^l \middle| \Theta_{t-1}^p\right) p\left(\Theta_{t-1}^p \middle| \mathrm{I}_{1:t-1}\right) d_{\Theta_{t-1}^p} \tag{A.53}$$

Where the pdf $p\left(\Theta_t^p \middle| \Theta_t^l, \Theta_{t-1}^p\right)$ can be interpreted as a prediction of the nonlinear states conditioned on the previous ones and predicted Kalman states. As a result of (A.53) the system state proceeds to next time t.

In this step both equations indicate the time update steps of the Rao-Blackwellized Particle Filter. The first equation is the regular Kalman Filter time update and the second one is the Particle Filter prediction.

Measurement Update:

By incorporating the information from the priori distribution given in (A.52) into the adaptive velocity estimation or in other words the particle filter prediction the posterior distribution of the linear states is obtained via (A.53) as well. When the pdf given in (A.53) is conditionally separated as

$$p\left(\Theta_t^l, \Theta_t^p \middle| \mathrm{I}_{1:t\text{-}1}\right) = p\left(\Theta_t^l \middle| \Theta_t^p\right) p\left(\Theta_t^p \middle| \mathrm{I}_{1:t-1}\right) \tag{A.54}$$

The pdf $p\left(\Theta_t^l \middle| \Theta_t^p\right)$ designates the posterior distribution of the linear states. In (A.53) the relation between the pseudo-measurement and linear Gaussian states are

given by the pdf denoted as $p\left(\Theta_t^p \middle| \Theta_t^l, \Theta_{t-1}^p\right)$. This abstract notation implies that the pseudo-measurement is obtained by making use of the predicted linear Gaussian states and previous estimates of the nonlinear ones. That is what we have expected since the relation between the measurement and the states is defined by likelihood functions [5]. In this work pseudo-measurement is also obtained in the process of registering the target template to the most probable image region which can be thought as extracting the measurement from the image sequence.

Finally, the aim of Rao-Blackwellized particle filter is accomplished by obtaining the filter density of nonlinear states after marginalization is carried out through the integral over linear states:

$$\begin{aligned} p\left(\Theta_t^p \middle| \mathrm{I}_{1:t}\right) &= \frac{\overbrace{\int p\left(\mathrm{I}_t \middle| \Theta_t^l, \Theta_t^p\right) p\left(\Theta_t^l, \Theta_t^p \middle| \mathrm{I}_{1:t-1}\right) d\Theta_t^l}^{\text{Marginalization}}}{\iint p\left(\mathrm{I}_t \middle| \Theta_t^l, \Theta_t^p\right) p\left(\Theta_t^l, \Theta_t^p \middle| \mathrm{I}_{1:t-1}\right) d\Theta_t^l d\Theta_t^p} \\ &= \frac{\int p\left(\mathrm{I}_t, \Theta_t^p, \Theta_t^l \middle| \mathrm{I}_{1:t-1}\right) d\Theta_t^l}{\int p\left(\mathrm{I}_t, \Theta_t^p \middle| \mathrm{I}_{1:t-1}\right) d\Theta_t^p} \\ &= \frac{p\left(\mathrm{I}_t, \Theta_t^p \middle| \mathrm{I}_{1:t-1}\right)}{p\left(\mathrm{I}_t \middle| \mathrm{I}_{1:t-1}\right)} \end{aligned} \tag{A.55}$$

Exploiting Markov property of the nonlinear states the above equation can be rewritten as

$$p\left(\Theta_t^p \middle| \mathrm{I}_{1:t}\right) \propto p\left(\mathrm{I}_t \middle| \Theta_t^p\right) p\left(\Theta_t^p \middle| \mathrm{I}_{1:t-1}\right) \tag{A.56}$$

Here, the first pdf on the left hand side of (A.56) corresponds to the importance weight of the particles that is computed as given in (A.48).

Briefly, Rao-Blackwellized particle filter combines the ideas from Kalman Filter banks and Particle Filter [50]. Therefore, the recursively estimated posterior distribution is a combination of parametric and un-parametric distributions [50].

The Proposed RBPF Algorithm

The proposed RBPF algorithm is summarized as follows:

i. INITIALIZATION: For n=1,…,N initialize both particle set $\left\{\Theta_t^{p,n}, \omega_t^{p,n}\right\} = \left\{\Theta_o^{p,n}, \frac{1}{N}\right\}$ Where $\Theta_o^{p,n} \sim p\left(\Theta_o^p\right)$ and the Kalman state variables and their associated covariance matrices $\left\{\hat{\Theta}_{0|-1}^l, P_{0|-1}^l\right\}$.

Here, $p\left(\Theta_o^p\right)$ is a 3D Gaussian distribution with

$$E\left[\Theta_o^p\right] = \left[s_{x,0}\ s_{y,0}\ \theta_0\right]^T,\ \ \mathrm{cov}\left[\Theta_o^p\right] = \text{block-diag}\left[\sigma_{s_x}^2\ \ \sigma_{s_y}^2\ \ \sigma_\theta^2\right] \qquad \text{(A.57)}$$

At time t=0 scale and rotation parameters of the target is set to 1.0 and 0 degree respectively and they are accepted as mean values of these variables. Also we assumed that the dynamics of these state variables doesn't change too rapidly. So, small values are assigned to variances of these variables.

In a similar way, the Kalman states are distributed according to a Gaussian density. The parameters of this density are given by

$$E\left[\Theta_o^l\right] = \left[d_{x,0}\ \dot{d}_{x,0}\ d_{y,0}\ \dot{d}_{y,0}\right]^T,\ \ \mathrm{cov}\left[\Theta_o^p\right] = \text{block-diag}\left[\sigma_{d_x}^2\ \ \sigma_{\dot{d}_x}^2\ \ \sigma_{d_y}^2\ \sigma_{\dot{d}_y}^2\ \right] \qquad \text{(A.58)}$$

Since we don't know any information about the derivative of displacements the mean of these parameters are accepted as zero. But dynamics of these state variables change more rapidly than other variables in the state space. Also, it is crucial to predict the next position of the target as accurately as possible. Because of the uncertainty and dynamics of these variables the variance of these states are set to higher values than they are for scale and rotation variables

However, the values of parameters characterizing the Gaussian densities are design parameters and they depend on the dynamics of the target and frame rate of the camera. For example when frame rate is low the covariance matrix of these densities should consist of high variances.

ii. PF TIME UPDATE (Prediction): For n=1,…,N generate new nonlinear states represented by particles according to

$$\Theta_{t+1}^{p,n} \sim N\left(\Theta_t^{p,*} + v_t^{p,*}, C_t^p\right) \tag{A.59}$$

Here, $\Theta_t^{p,*}$ is a nonlinear state at time t which makes the best estimation of the adaptive velocity denoted as $v_t^{p,*}$.

iii. KALMAN PSEUDO MEASUREMENT UPDATE: After adapting linear and Gaussian motion models to the dynamics of the target by the adjusted process noises, by using the pseudo measurements obtained from adaptive velocity estimation update the Kalman predictions as given in (A.32), (A.33) and (A.34)

iv. OPTIMUM LINEAR STATE: Compute the optimum linear state $\hat{\Theta}_t^{l,*}$ from the updated ones as given in (A.45).

v. PF MEASUREMENT UPDATE: Rectify the target region using optimum linear state and particle predictions then evaluate and normalize the importance weights according to (A.48)

vi. EXPECTED MEAN Of NONLINEAR STATE: Either in MMSE or MAP sense computes the expected mean of the target's nonlinear state. Thus, together with optimum linear state the estimation of the full system state vector is obtained.

vii. KALMAN PREDICTION: Make one step ahead predictions for linear Gaussian states for each particle using their system models. Thus, obtain the predictions associated with the target position in the next frame

$$\hat{\Theta}_{t+2|t+1}^{l,i} = F\hat{\Theta}_{t+1|t+1}^{l,i} \tag{A.60}$$

with associated covariance of the predictions

$$P_{t+2|t+1}^{l,i} = FP_{t+1|t+1}^{l,i}F^T + C_{t+1}^{l,n} \tag{A.61}$$

APPENDIX B

RAO-BLACKWELL THEOREM

Theorem: Let's assume that $\hat{\Theta}$ is an unbiased estimator and T is a sufficient statistics both for Θ. Then the estimator $\hat{\Theta}^* = E\left[\hat{\Theta}|T\right]$ is at least good as $\hat{\Theta}$ in Mean Square Error (MSE) sense such that:

$$MSE\left(\hat{\Theta}^*\right) \leq MSE\left(\hat{\Theta}\right) \tag{B.1}$$

Proof: In order to prove the above inequality we use the following equality

$$Var\left(\hat{\Theta}\right) = E\left[\hat{\Theta}^2\right] - E\left[\hat{\Theta}\right]^2 \tag{B.2}$$

Since we know that $Var\left(\hat{\Theta}\right) \geq 0$ the following inequality can be established

$$E\left[\hat{\Theta}^2\right] \geq E\left[\hat{\Theta}\right]^2 \tag{B.3}$$

Using the this fact we have that

$$E\left[\left(\hat{\Theta}-\Theta\right)^2|T\right] \geq E\left[\left(\hat{\Theta}-\Theta\right)|T\right]^2 = \left(E\left[\hat{\Theta}|T\right]-\Theta\right)^2 \tag{B.4}$$

Where $E\left[\hat{\Theta}|T\right]$ corresponds to the conditional estimator $\hat{\Theta}^*$.

Then by taking the expectation of the both sides the following equation demonstrating the variance reduction in Rao-Blackwellized Particle Filter is obtained.

$$E\left[\left(\hat{\Theta}^*-\Theta\right)^2\right] \leq E\left(E\left[\left(\hat{\Theta}-\Theta\right)^2|T\right]\right) = E\left[\left(\hat{\Theta}-\Theta\right)^2\right] \tag{B.5}$$

Here the left side of the above inequality is MSE of conditional estimator, $MSE\left(\hat{\Theta}^*\right)$ and the other side gives the MSE of the unbiased estimator, $MSE\left(\hat{\Theta}\right)$. In consequence, according to (B.5) it can be said that estimator given by $\hat{\Theta}^*$ is improved by Rao-Blackwellization procedure.