Integral Image Based Probabilistic non-linear Subspace Visual Object Tracking

By

Iftikhar Majeed

NUST201362744MSEECS61313F

Supervisor

Dr. Omar Arif

Department of Computing

A thesis submitted in partial fulfillment of the requirements for the degree of Masters in Computer Sciences (MS CS)

In

School of Electrical Engineering and Computer Science,

National University of Sciences and Technology (NUST),

Islamabad, Pakistan.

(December 2015)

Approval

It is certified that the contents and form of the thesis entitled "**Integral Image Based Probabilistic non-linear Subspace Visual Object Tracking**" submitted by **Iftikhar Majeed** have been found satisfactory for the requirement of the degree.

Advisor: **Dr. Omar Arif**

Signature: _____

Date: _____

Committee Member 1: **Dr. Asad Anwar Butt**

Signature: _____

Date: _____

Committee Member 2: **Dr. Anis ur Rahman**

Signature: _____

Date: _____

Committee Member 3: **Dr. Muhammad Muneeb Ullah**

Signature: _____

Date: _____

Certificate of Originality

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics which has been acknowledged.

Author Name: **Iftikhar Majeed**

Signature: _____

Dedication

This thesis is dedicated to my parents and siblings for their love, endless support and encouragement.

Acknowledgment

I am thankful to ALLAH Almighty, the Most Beneficial and the Most Merciful, I was able to complete my thesis because of His countless blessings. I am happy to express my gratitude and special thanks to my advisor Dr. Omar Arif for his excellent support and guidance throughout my research. Beside my advisor, I am thankful to my thesis committee members; Dr. Asad Anwar Butt, Dr. Anis ur Rahman, and Dr. Muhammad Muneeb Ullah for their guidance and encouragement.

Abstract

Visual tracking is the process of estimating the location of an object in consecutive video frames. It is a major problem in the domain of computer vision with large set of applications such as traffic monitoring, robot planning, surveillance, vehicle navigation and human computer interaction. The problems related to object tracking are complex object shapes, noise in the image, occlusion, and background clutter. The aim of this work is to develop such technique which can track object in a complex environment. This work presents a visual object tracking algorithm using an eigenspace representation. Object appearance and spatial information is learned from a single template using a non-linear subspace projection to arrive at a eigenspace representation. This non-linear subspace representation provides a robust and compact representation of the object. Localization is performed using a similarity measure in non-linear eigenspace representation. A probabilistic search strategy, based on particle filter, is employed to find the region of an object in each frame of the video sequence that best models the target object in the subspace representation. Particle filter estimates the posterior distribution using weighted samples. Increasing the number of samples increases the estimation accuracy at the cost of increased computations. We, therefore

propose a novel kernel subspace integral image framework, which allows the tracker to densely sample the state space without loosing computational efficiency. The proposed tracker is tested on number of challenging sequences to demonstrate the performance.

Table of Contents

1 Introduction and Motivation **1**

1.1 Object Representations 2

1.2 Feature Selection 4

1.3 Object Tracking 5

1.4 Thesis Motivation 6

1.5 Thesis Contributions 7

1.6 Thesis Organization 8

2 Literature Review **9**

2.1 Generative and Discriminitive Methods 10

2.2 Template based Approaches 10

2.3 Histogram based Approaches 11

2.4 Variational Target Localization 12

2.5 Principal Component Analysis 13

3 Background **15**

3.1 Target Eigenspace Representation 15

3.1.1 Properties of the Eigenspace Representation 17

3.2 Similarity Measure in Target Eigenspace 18

4 Design and Methodology **21**

4.1 Particle Filter 21

4.2 Visual Object Tracker 23

4.2.1 Kernel Subspace Integral Image based Observation Likelihood 24

4.2.2 Overall Algorithm 29

5 Implementation and Results **30**

6 Conclusion **34**

List of Figures

3.1 The target object color values lie on the surface of the ellipse in eigenspace, while other color values lie interior to the ellipse. The distance from the origin can be used as a similarity function. 20

4.1 Kernel integral image based non-linear subspace region similarity. [First]: Template image. [Second]: Given image. [Third]: Region similarity of given image. Each pixel represents average of pixel similarity of the region centered at that pixel. 28

5.1 Sample tracking results of the proposed tracker, hand (first row), hand2 (second row), dinosaur (third row), torus (fourth row) and gymnastic (fifth row). 32

5.2 In the Caviar data set, the target object is occluded by people coming from the opposite direction. In Pets 2009 data set, the target object is successfully tracked in a crowded scene. 32

List of Tables

5.1 Tracking results. The first number indicates failure rate while the second number shows RMS between the ground truth center location and tracker estimate. The adaptive color tracker ACT [1] only tracked torus sequence and is not listed in the table. Last five columns are taken from Table 2 in [2] for easy comparison. X indicates that the tracker could not track the sequence. 31

Chapter 1

Introduction and Motivation

Object tracking can be defined as the process of locating an object of interest or the problem of estimating the trajectory of an object in video frames. It is fundamental problem in computer vision field with many applications traffic monitoring, robot planning, vehicle navigation, human computer interaction, motion based recognition, and automated surveillance. Object tracking becomes complex problem because of complex object shapes, noise in the image, loss of information while projecting 3D data to 2D image, illumination changes, partial and total occlusions. Many object tracking approaches have been proposed to track single or multiple objects and these approaches varied with each other depending on object shape, and its appearance. As different object representations are used based on its shape, and different features such as colour, texture, edges etc. are selected based on appearance of an object. We will explain different object representations, and feature selection methods used in object detection and will further explain various tracking methods used on the basis of object and motion representation.

1.1 Object Representations

Objects is anything that is of interest. Objects can be represented by their shape and appearance. Joint shape and appearance representations described as follows as in [3]

- Points: The object can be represented by single point, or set of points. Point or Centroid symbol is best fit for tracking objects that lodge minor regions in an image.

- Primitive geometric shapes: The object is represented by ellipse or rectangle etc. The motion of an object for this representation is commonly demonstrated by transformation, affine, translation or projective.

- Object silhouette and contour: Object can be represented by defining the boundary of an object; known as contour representation. The region within contour representation is known as silhouette. Contour and silhouette representation is best fit when objects are of complex shapes.

- Articulated shape models: Articulated shape model is formed when different parts of object represented separately using rectangle or ellipse etc. For example, human body is composed of different body parts, legs, feet, head, and hands etc., can be represented using this kind of representation.

- Skeletal models: Rigid as well as articulated objects can be represented in skeletal model representation.

There are many ways to characterize the appearance of objects. Most common representation based on object appearance are described as follows as in [3].

- Probability densities of object appearance: The probability density of the object appearance features is approximated either by parametric or nonparametric. It can be Gaussian or a mixture of Gaussians or can be nonparametric such as histograms or parzen windows. Appearance features such as colour, or texture can be calculated from image area displayed by the contour or silhouette models.

- Templates: Templates provide joint appearance and spatial information of an object can be used in learning as well as tracking of an object. This is mostly used when object poses do not change during tracking as information is generated based on single view only. Templates are shaped as contour or silhouette.

- Appearance models: Active appearance models can be made by modelling object appearance and shape simultaneously. Appearance vector of colour, texture, or gradient can be stored for each landmark which can exist on boundary of an object or inside region. In case of Multi-view appearance models, subspace can be generated based on different views of an object using subspace approaches like Principal component analysis (PCA).

Particular object representation is usually selected based on application and tracking algorithms. For smaller objects, point or centroid representation

is used. Primitive geometric representations are normally used when shapes are looked as similar to rectangle, or circle. For example, Comaniciu et al. [2] employs colour histograms using elliptical shape representation.

1.2 Feature Selection

Feature selection and Object representations are closely related to each. For example, in case of histogram based appearance representation, colour can be used as visual feature while edges can be used as feature in case of contour based representation. Object can be distinguished based on its uniqueness of its visual feature. Visual feature plays important role in tracking of particular object. In general, combination of common visual features described below are used in object tracking algorithms.

- Color: RGB values are used to represent color in domain of image processing.
- Edges: Object boundaries or edge detection is used to identify changes in image intensities. Edges are not as sensitive to illumination changes as colour is.
- Texture: Texture can measure intensity of surface to identify smoothness. It requires initial processing to generate descriptors and features are less sensitive as compared to colour of illumination changes.

Features can be selected based on their usefulness or they should not be correlated with each other. In some cases, for correlated feature, PCA can

be used to transform correlated features to uncorrelated features. However, color is mostly used as visual feature for object tracking. Comaniciu et al. [4] represent the object appearance using colour histogram. As described above, colour is most sensitive to illumination changes, therefore, combination of features are used in object tracking to improve tracker performance.

1.3 Object Tracking

The aim of visual object tracker is to locate object position in successive frame of the video or to generate trajectory or path of an object over time using joint shape or appearance information extracted from previous frames based on object representation. Point/centroid based representation, geometric representation using rectangle or ellipse is used to estimate motion of rigid objects. For non-rigid objects, silhouette or contour based representation can be used to approximate object motion in the scene. We briefly introduce main categories of object tracking.

- Point Tracking: Objects are represented by points, and correspondence of an object can be generated based on object state including position and motion of an object.

- Kernel Tracking: In kernel tracking, object shape and appearance information is characterized. Kernel can be in the form of rectangle or ellipse with colour histogram.

- Silhouette Tracking: Information within object region is used in silhouette tracking methods. Information inside region can be represented as

appearance density or shape models.

Primitive geometric shapes representation of an object is widely used in state-of-the-art methods. Parametric motion of the object is computed in tracking methods based on rigidity constraints. It can be in the form of affine, projective or translation. Object similarity is maximized using previous and current frame of video sequence to estimate parametric motion of the object. There are number of ways to estimate motion of the object. It can be in the form of gradient ascent/descent based maximization/minimization approach, or simple brute force approach. The main limitation in using gradient ascent/descent approach is that at least portion of the object must be visible to track it. To remove this limitation, Kalman filtering approach is used to predict object location in next frame. However, Kalman filter give poor results when state space is not Gaussian distribution. To overcome this issue, conditional probability based particle filtering approach is used to estimate the state of the object. Primitive geometric shapes representation may not give correct motion estimation using similarity measure because complete object may not be selected in defined shape, or it may include background noise. In this case, probability density estimation is used to give high weights to pixels which are inside shape/region using conditional probability of visual feature (colour, texture, etc.).

1.4 Thesis Motivation

In recent years, object tracking has been much studied and sufficient progress has been made. Object tracking still remains a very interesting and challeng-

ing problem in the domain of computer vision. Several factors such as object shape, background noise, illumination variations, partial as well as total occlusions make object tracking a very complex problem. No single tracker exist in the domain of object tracking which can handle all scenarios successfully. Still, there is much room for exploration and making contributions in the domain of object tracking. This work can be applied to improve security measures by tracking people or vehicles, and also in traffic monitoring and controlling. Also, it could be used in surveillance using Robots. Major advantages includes machine-machine interaction can be enhanced by sending data from robot to computers through video communication which could be further utilized to be processed and taking actions needed at specific place, security mechanisms can be improved by tracking people and vehicles and monitor suspicious activities, especially, in traffic controls system to adjust traffic signal based on number of vehicles.

1.5 Thesis Contributions

The work presented in thesis [5] uses feature vectors associated to pixels of the target template as observations. Deformable objects can be tracked as the learning is carried out pixel wise. At the same time individual pixels are tied together through non-linear subspace representation of the model, which provides a robust and compact representation of the object being tracked. Due to its ability to track rigid and non-rigid objects and robustness to noise, we base our method on [5].

The method proposed in [5, 6] uses gradient descent method to maximize

the similarity function with respect to the transformation parameters. An alternate mean-shift procedure is also given. These strategies can provide reliable solution and converge to local maximum but fail to recover from lost objects and to track fast moving objects. We therefore propose to use particle filtering approach [7] to sequentially estimate the state variables, which in our case are the transformation parameters of the rectangular target object region. Particle filter has the ability to recover from lost tracks and to track fast moving objects even with partial and total occlusions. Particle filter has been successfully employed for visual object tracking, see for example [8, 9]. Particle filter performance increases by increasing the number of samples. However, increasing the samples also increases the computational cost. In this thesis, we propose a novel kernel subspace integral image formulation that allows us to densely sample the state space without loosing computational efficiency. The speedup obtained over non integral image based implementation is of about 2.5 orders of magnitude over non-integral image based implementation.

1.6 Thesis Organization

This thesis is structured as following. Chapter 2 presents the previous work related to our method. Chapter 3 introduces the proposed methodology leading to results in Chapter 4. Chapter 5 provides conclusion and future work directions.

Chapter 2

Literature Review

Visual tracking is the process of estimating the location of an object in consecutive video frames. It is a major problem in the domain of computer vision with large set of applications such as traffic monitoring, robot planning, surveillance, vehicle navigation and human computer interaction. Some of the problems associated with object tracking are image noise, occlusion, background clutter, complex object shapes, etc.

Objects can be represented by their appearances, such as color, texture, edges, and shape information, which provide characteristic information about the target object. This characteristic information, gathered from single template or multiple templates of the target object, is encoded into a cost or similarity function.

There are different types of tracking algorithms which determine the correspondence of the object.

2.1 Generative and Discriminitive Methods

Methods which determine the correspondence of the object region in consecutive images by optimizing the pre-determined similarity functional are called *generative methods* [10, 11], as they search the image space to find the region most similar to the target model. Generative algorithms models input data using joint probability distributions, $p(x, y)$, and calculate conditional probaility $p(y|x)$ by making predictions using Bayes rules.

In contrast to generative methods, discriminative methods formulate the problem as a binary classification problem and models posterior density, $p(y|x)$ directly. From positive and negative samples, a classifier is trained to separate the foregorund from the background. This can be carried out at pixel level [12, 13] in which classifier is trained to distinguish features between actual object and background, or at region level [14, 15] in which part of the object is identified as rigion. The latter methods are also called *tracking by detection*. Tracking by detection methods are more suitable for tracking rigid methods while methods that pose tracking as pixel wise classification problem are more suitable for non rigid objects as they ignore the spatial information [8].

2.2 Template based Approaches

Different encoding strategies are employed to track rigid and non-rigid objects. Tracking rigid objects call for methods that encode the spatial geometry of the object in similarity function. In the simplest case of template

matching, the similarity function is reduced to per pixel difference between the template and the target region. More involved methods use multiple templates, sparse representations and subspace models [16, 17, 10, 18, 19, 20].

2.3 Histogram based Approaches

For deformable objects, histogram based approaches are more suitable [8], reducing the similarity between target and candidate region to similarity between color distributions. For example, Comaniciu et al [4] estimated probability density function of the region of an object using histograms weighted by a spatial kernel. At the region level, correspondence of an object is determined between consecutive frames using Bhattacharya coefficient which is optimized using meanshift iterative procedure between candidate distributions and the target object. Histogram becomes problematic as they discard spatial information especially when target object lies in the background or when object is passing through partial or total occlusions. Instead of using the Bhattacharya coefficient, Hager et al [21] calculated distance between modulated-kernel histograms using Matusita distance measure technique and Newton-style iterations are used to optimize Matusita distance. As compared to meanshift, Matusita distance provides faster convergence. Additionally, limitations of using a single kernel are provided and an extension using multiple spatially distributed kernels is developed. Multiple kernels are chosen so as to increase the rank of the resulting system. Another extension to multiple-kernel tracking is [22], in which target object can be found using multiple collaborative kernels. In [23], spatiograms are derived based on his-

tograms including spatial information. Each histogram bin is expanded to include covariances, and spatial means of pixels to derive spatiogram. The algorithms uses probability density functions (histograms) faced two issues; firstly, In case of higher dimensions, algorithms becomes computationally expensive and secondly, when sample set is small then feature space becomes sparse. The sparseness makes similarity measures, such as Bhattacharya coefficient, computationally unstable [24]. Methods, such as [24, 25, 26] define similarity functions between kernel density estimates of the template and target distribution in a joint feature-spatial space. The relationship between the appearance and spatial information is more fully exploited as compared to the previously discussed approaches. Since these methods employ non-parametric density comparison techniques, the intermediate step of estimating the density function is not carried out and the similarity functions are defined directly on the samples obtained from the target and the candidate regions.

2.4 Variational Target Localization

Appearance and spatial information of the target object is learned using eigenvalue decomposition of the covariance matrix in an alternate image descriptor. Find the distance between object region covariance matrix and model covariance matrix. Exhaustive search is performed in [27] over the image to find the region which has smallest distance between object and model covariance matrix. The algorithm can recover from partial as well as total occlusions, or during fast movements due to its global nature but

at the same time, it is vulnerable to false positives, background clutter and noise. In [28], Karasev et al extended the covariance tracker by considering spatially weighted mean and covariance descriptions of the target object, and provide a variational approach to target localization. Avidan [29] trains many binary classifiers to separate object pixels from background pixels and treats tracking problem as binary classification problem. A confidence map is generated when strong classifier is tested in the current image on all pixels. Then, object rectangle is found by running meanshift procedure on generated confidence map.

2.5 Principal Component Analysis

Essential to improved tracking is the derivation of a model that can capture the relationship between the purely image-based observations and the spatial content associated to said observations. Some generative methods use unsupervised learning techniques such as principal component analysis (PCA) to measure the correlation among the pixels of the templates of the target. The templates are vectorized to form a matrix $D = [I_1, I_2, ...I_n]$, where each column is a vectorized template. The covariance matrix obtained from the data in D is diagonalized to obtain a low dimensional eigenspace representation of the target. This representation has been used for tracking in[30] and [31]. In [31], the subspace is also incrementally updated to account for appearance and illumination change. Tsai et al. [32] perform PCA on a collection of vectorized signed distance maps of the training shapes to incorporate the shape model into the segmentation procedure. In these settings each vec-

torized template I_i is an observation with the implicit assumption that the image appearance will remain similar to the training templates. However, under partial or extensive occlusions this assumption will not hold and the tracker may give erroneous results. Instead of using vectorized images as observations, [6] use feature vectors associated to pixels of the target template as observations. Pixel vectors extracted from a single template are used for learning the target model. Deformable objects can be tracked as the learning is carried out pixel wise. At the same time individual pixels are tied together through non-linear subspace representation of the model, which provides a robust and compact representation of the object being tracked. Due to its ability to track rigid and non-rigid objects and robustness to noise, we base our method on [5, 6] described in next section.

Chapter 3

Background

This chapter explains non-linear eigenspace representation as described in [5] in which object appearance and spatial information is learned from a single template using a non-linear subspace projection to arrive at a eigenspace representation. This representation allows tracking pixel-wise but the pixels are tied together using non-linear subspace representation which provides a robust and compact representation of the object.

3.1 Target Eigenspace Representation

The feature/pixel vector associated to a given pixel is a d-dimensional concatenation of a p-dimensional appearance vector and a 2-dimensional spatial vector $\rho = [\mathcal{I}(x), x]^T$, where $\mathcal{I}(x)$ represents feature extracted at template pixel location $x = [\mathrm{x}, \mathrm{y}]^T$. The extracted feature can be color, gradient, texture, etc., or any combination of these. The pixel vectors are extracted from the segmented target template image(s). The set of all pixel vectors define

the target input space $\mathbb{D}$,

$$\mathbb{D} = \{\rho_1, \rho_2, ..., \rho_n\},$$

where n is the total number of pixel vectors extracted from the template images. The eigenspace representation of the input space comes from performing eigenvalue decomposition of the $n \times n$ kernel matrix, $K_{ij} = \boldsymbol{k}(\rho_i, \rho_j)$, where $\boldsymbol{k}$ is a Mercer kernel and n is the number of target object pixel vectors. An Example of the Mercer kernels is the Gaussian kernel, which is used in this paper.

$$\boldsymbol{k}(\rho_i, \rho_j) = \exp\left(-\frac{1}{2}(\rho_i - \rho_j)^T \Sigma^{-1}(\rho_i - \rho_j)\right), \tag{3.1}$$

where Σ is a $d \times d$ matrix appropriately chosen for rigid and non-rigid objects. The use of Mercer kernel allows for the computations in higher dimensional Hilbert space, $\phi : \mathbb{R}^d \to \mathcal{H}$, such that

$$\boldsymbol{k}(\rho_i, \rho_j) = \phi(\rho_i) \cdot \phi(\rho_j).$$

A test point ρ is represented in the eigenspace by the following projection equation:

$$P(\rho) = \Lambda^{-\frac{1}{2}} E^T \boldsymbol{k}_\rho, \tag{3.2}$$

where Λ is a diagonal matrix containing the eigenvalues of matrix K, E is a $n \times q$ matrix containing q eigenvectors, and $\boldsymbol{k}_\rho = [\boldsymbol{k}(\rho, \rho_1), \boldsymbol{k}(\rho, \rho_2), \ldots, \boldsymbol{k}(\rho, \rho_n)]^T$. This results in q dimensional eigenspace representation of the pixel vector, $P(\rho) = [P_1(\rho), \ldots, P_q(\rho)]^T$, where each component is, in expanded form,

written as

$$P_k(\rho) = \sum_{i=1}^{n} \alpha_i^k \boldsymbol{k}(\rho, \rho_i), \tag{3.3}$$

where $\alpha_i^k = \frac{E_i^k}{\lambda^k}$ with E_i^k begin the i^{th} element of the k^{th} eigenvector.

All further computations are performed in the eigenspace, which is related nonlinearly to the input space. This results in nonlinear algorithm.

3.1.1 Properties of the Eigenspace Representation

The eigenspace representation has several properties that are advantageous when tracking. In particular, the benefits described below will transfer to the similarity function to be defined shortly.

Enhanced clustering/discrimination: Our kernel based eigenspace representation is based on unsupervised clustering algorithm, kernel principal component analysis (KPCA). The algorithm can capture a number of clusters/features equal to the number of eigenvectors retained plus one [33]. In case of principal component analysis (PCA), the maximum eigenvectors are limited by the the dimension of the input space, $\mathbb{R}^d$. For KPCA, the maximum number of eigenvectors is given by the number of data points in the input space, n. Given that $n >> d$, KPCA can represent significantly more clusters [33], thus its representation capacity is richer. In case of a two-dimensional data, PCA can have maximum of two eigenvectors while the eigenvectors learned using KPCA capture different aspects of the two-dimensional dataset. The eigenvectors capture non-linear relations among the dataset. These properties can be very useful for developing an object tracking algorithm, where the relationship among three or more pixels pro-

vide useful information about the target [34].

Noise/outlier removal: There are two ways noise and outliers can effect the tracking process, either during the learning phase (the noise present in the sample set used for training) or during the tracking phase. The use of KPCA helps in reducing the effect of noise and outliers in both cases. Noise suppression is achieved by selecting less than the maximal eigenvalues possible. In case, when outliers are added to the training set, the isoclines of the top eigenvector coefficients remain the same. The reconstruction can show that the eighth eigenvector and beyond will capture noise and therefore can be ignored. In next section, we will show that the projections on to the eigenspace go to zero when data is far from the template samples. This helps in reducing the effect of noise/outliers during the tracking phase. The remaining sections exploit these properties for tracking.

3.2 Similarity Measure in Target Eigenspace

Object tracking in the target eigenspace requires defining a similarity function in the eigenspace. Here, the similarity function will measure the similarity of a feature vector to the learned model. As per Section 3.1, the similarity function defined in the eigenspace will capture nonlinear relationships between the feature vectors.

The similarity function comparing a given feature vector (or set of feature vectors) to the learned model will exploit the geometric properties of the eigenspace. Under Gaussian kernel, the eigenspace is a high-dimensional elliptical space. A feature vector ρ is represented by $P(\rho) = [P_1(\rho), \ldots, P_q(\rho)]^T$,

where each $P_k(\rho)$ is computed using Equation (3.3). It is evident from Equation (3.3) that the function $P_k(\rho)$ tends to zero as the vector ρ recedes from the input space $\mathbb{D}$. Similarly, when ρ is a feature vector representative of an input space element, the distance of the vector $P(\rho)$ from the eigenspace origin, is bounded and given by

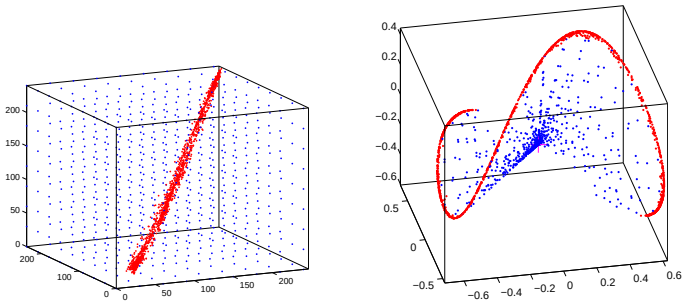
$$||P(\rho)|| = \langle P(\rho), P(\rho) \rangle^{\frac{1}{2}} = 1.$$

However, since the projection $P_k(\rho)$ is approximated using the Nystrom approximation method [35], the distance computation satisfies the inequality $||P(\rho)|| \leq 1$ [36].

All features in the input space are mapped non-linearly to high dimensional Hilbert space. During projection, a feature vector which corresponds to the data is mapped onto the hyper ellipse surface whereas noise or outlier, is mapped closer to the origin depending on the probability to which it is treated as outlier. Therefore, the objects that are onto the surface of ellipse has maximum distance from origin, used as similarity and ones which are closer to the origin has less similarity or zero if lies onto the origin. So the distance from the origin acts as similarity measure or observation likelihood. The similarity of a test feature vector ρ to the target model $\mathbb{D}$ is defined as

$$s(\rho) = ||P(\rho)||_1 = \sum_{k=1}^{q} |P_k(\rho)| \tag{3.4}$$

where q is the total number of eigenvectors retained. In Figure 3.1, the target color values shown in red are learned using the procedure defined



(a) Input space. Target color values are shown in red.

(b) Target object eigenspace representation

Figure 3.1: The target object color values lie on the surface of the ellipse in eigenspace, while other color values lie interior to the ellipse. The distance from the origin can be used as a similarity function.

above, while the blue points depict other points in the domain. The learned eigenspace is shown in Figure 3.1(b), for which both the red and blue points are mapped. The target object color values lie on the surface of the ellipse in the eigenspace, while all other color values lie within the ellipse. The distance from the origin can be used as a similarity function. Importantly, the similarity function rewards inliers and ignores outliers, thus lending robustness to outliers (such as target occlusions and background clutter). Further details on the statistical interpretation of this approach and its relation to robust density matching can be found in [37].

Chapter 4

Design and Methodology

A target eigenspace representation in which feature vector consisting of object appearance and spatial information is learned from a single template using a non-linear subspace projection is explained in Section 3.1. The similarity of feature vector is computed using similarity measure explained in Section 3.2, then feature vector similarity is summed up to define observation likelihood used in particle filter explained in Section 4.1 followed by object tracking methods described in Section 4.2.

4.1 Particle Filter

Particle filter method [38], also knowns as sequential Monte Carlo (SMC) method approximates the posterior density of the state-space variables of a dynamical system from sequence of observations. Unlike Kalman filter, the posterior density is not restricted to be Gaussian distributed. It can be of any form and is non-parametrically represented by a set of random samples (a.k.a

particles), and associated weights. Let $\boldsymbol{x}_t$ and $\boldsymbol{y}_t$ be the latent state and observation variables at time t. Evolution of the process is governed by two probability distributions: state transition probability $\boldsymbol{x}_{t+1}|\boldsymbol{x}_t \sim p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)$ and observation likelihood $\boldsymbol{y}_t|\boldsymbol{x}_t \sim p(\boldsymbol{y}_t|\boldsymbol{x}_t)$, with the assumption that state evolves according to the first order Markov process. The basic idea is to approximate the posterior density with a set of m weighted particles $\{\boldsymbol{x}_t^i, w_t^i\}_{i=1}^m$. The particles are drawn from a proposal distribution, which can be the same as the transition probability. The particles are sequentially updated using transition probability to obtain approximation to the the posterior density $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$ with a sum of m Dirac functions centered at particles $\{\boldsymbol{x}_t^i\}_{i=1}^m$ as:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) \approx \sum_{i=1}^{m} w_t^i \delta(\boldsymbol{x}_t - \boldsymbol{x}_t^i), \tag{4.1}$$

where weights are computed as $w_t^i \propto w_{t-1}^i p(\boldsymbol{y}_t|\boldsymbol{x}_t^i)$. This particular type of particle filter is called Sequential Importance Sampling (SIS). It suffers from particle degeneracy problem, where only a handful of particles contribute to the posterior density approximation. Rest of the particles tend to have very small weights. This problem can be avoided by using a resampling stage leading to Sequential Importance Resampling (SIR). At each iteration, current particles are dropped and new particles are added from the current dropped particle set with probability proportional to their weights.

4.2 Visual Object Tracker

The objective of visual object tracking is to provide an accurate and compact region that encompasses the target object in each frame of the video. The state space consists of location, width and height of the region, $\boldsymbol{x} = [\mathrm{x}, \mathrm{y}, w, h]^T$, where $\{\mathrm{x}, \mathrm{y}\}$ is the location of the top left corner of the region in image coordinates. Each dimension of the state space is modeled by an independent Gaussian distribution:

$$p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) \propto \mathcal{N}(\boldsymbol{x}_t, \Sigma). \tag{4.2}$$

If there is no scale change, the width and height of the region can be kept fixed. Initial state is provided by the user in the first frame. Feature vectors from the region defined by the initial state are used to create target object eigenspace representation using the method described in Section 3.1. m particles $\{\boldsymbol{x}_0^i, w_0^t\}_{i=1}^m$, are generated initially using Equation (4.2) with weights set as $w_0^i = 1/m$.

At each new frame of the video sequence, particles are sampled using Equation (4.2), and their weights updated by $w_t^i \propto w_{t-1}^i p(\boldsymbol{y}_t|\boldsymbol{x}_t^i)$. The key component is the observation likelihood, $p(\boldsymbol{y}_t|\boldsymbol{x}_t^i)$, which should reflect the similarity of the particle to the target template. We extract all the feature vectors from the image region corresponding to the particle and sum the feature vectors similarity (Equation (3.4)) to define the observation likelihood:

$$p(\boldsymbol{y}_t|\boldsymbol{x}_t^i) \propto S(r_{\boldsymbol{x}_t^i}) = \frac{1}{C} \sum_{x \in r_{\boldsymbol{x}_t^i}} s(\rho(x)), \tag{4.3}$$

where $r_{\boldsymbol{x}_t^i}$ is the region defined by the particle $\boldsymbol{x}_t^i$. Weight of the particles is normalized, $w_t^i = \frac{w_t^i}{\sum_{j=1}^{m} w_t^j}$. The particle with the maximum weight is used as the estimate of the object region in the current frame. Resampling of the particles is carried out to avoid the degeneracy problem.

One limiting factor of particle filtering approach is the large number of particles required to accurately estimate the posterior distribution. Increasing the particles also increases the computational cost. Specially, in our case computing observation likelihood Equation (4.3) is expensive as it requires iterating over all the target feature vectors for each feature vector of the particle region. In the next sub section, we provide a novel procedure to efficiently compute Equation (4.3) using integral image. This allows us to increase the number of samples without the computational overhead.

4.2.1 Kernel Subspace Integral Image based Observation Likelihood

Let ρ be a feature (e.g. color) extracted at each image location and s be a function that maps extracted feature ρ computed at location x to a real number: $s : \rho(x) \rightarrow \mathbb{R}$. Filtering of values of s over the region r is defined as a function $A : r \rightarrow \mathbb{R}$ and expressed as:

$$S(r) = \sum_{x \in r} s(\rho(x)). \tag{4.4}$$

In the simplest case, $\rho(x)$ is intensity image and $s(\rho(x)) = \rho(x)$. Filtering over multiple overlapping regions is computationally inefficient as many com-

putations will be repeated for each region. This calls for an efficient integral image [39] based framework. Once the integral image has been computed, filtering (4.4) can be computed in constant time, irrespective of the size of the region.

4.2.1.1 Integral Image

Let $\mathbb{I}$ be an integral image of the feature function s. The value of integral image at location $x = [\mathrm{x}, \mathrm{y}]^t$ is computed by taking the sum of all pixels values above and left of x in image s:

$$\mathbb{I}(x) = \sum_{\acute{\mathrm{x}} \leq \mathrm{x}, \acute{\mathrm{y}} \leq \mathrm{y}} s(\rho(\acute{x})). \tag{4.5}$$

Once, the integral image has been computed, filtering of values of s over a region $r = [\mathrm{x}, \mathrm{y}, w, h]$ can be computed in constant time using the following formula:

$$S(r) = \mathbb{I}(\mathrm{x}, \mathrm{y}) - \mathbb{I}(\mathrm{x} + w, \mathrm{y}) - \mathbb{I}(\mathrm{x}, \mathrm{y} + h) + \mathbb{I}(\mathrm{x} + w, \mathrm{y} + h). \tag{4.6}$$

There is one caveat in using integral image, i.e., the function $s(\rho(x))$ must be independent of the region definition. This is not true in the case of pixel similarity function (Equation (3.4)). As mentioned in Section 3.1, the feature vector is concatenation of appearance and spatial values, $\rho_r(x) = [\mathcal{I}(x), u]^T$, where $u = [x - x_b]$, with x_b being the origin of the candidate region under consideration. A feature vector extracted from same image location but for two different candidate regions will have different spatial component,

resulting in different pixel similarity to the learned model. This means that the integral image can not be formed for pixel similarity function $s(\rho_r(x))$. Hussein et al. [40] provide an approximate integral image solution for non-uniform filters. In the next section, we propose a method to compute the pixel similarity function efficiently using integral image. The pixel similarity (Equation 3.4) not only contains non-uniform kernel but it also contains non-linear subspace projection. We call the resulting image, kernel subspace integral image.

4.2.1.2 Kernel Subspace Integral Image

This section contains one of our main contributions. Similarity of a feature vector to the learned model is the distance from the origin of the projection of feature vector onto the learned eigenspace representation (Equation 3.2). We reproduce Equation (3.4) in expanded form below:

$$\begin{aligned} s(\rho_r(x)) &= \sum_{k=1}^{q} |P_k(\rho_r(x))| \\ &= \sum_{k=1}^{q} \left| \sum_{i=1}^{n} \alpha_i^k \boldsymbol{k}(\rho_r(x), \rho) \right| \end{aligned}$$

Feature vector is composed of appearance and spatial features. Appearance features are independent of region definition while the spatial components are not. We decompose $P_k(\rho_r(x))$ into region independent and dependent

terms as follows:

$$\begin{aligned} P_k(\rho_r(x)) &= \sum_{i=1}^{n} \alpha_i^k \boldsymbol{k}\left(\rho_r(x), \rho\right) \\ &= \sum_{i=1}^{n} \alpha_i^k \boldsymbol{k}\left(\mathcal{I}(x), \mathcal{I}(x)\right) \boldsymbol{k}\left(u, x\right) \\ &= \sum_{i=1}^{n} a_i(x) \boldsymbol{k}\left(u, x\right) \end{aligned}$$

where x is the i^{th} target object template pixel location, $a_i(x) = \alpha_i^k \boldsymbol{k}\left(\mathcal{I}(x), \mathcal{I}(x_i^t)\right)$ is the appearance part which is region independent and $\boldsymbol{k}\left(u, x_i^t\right)$ is region dependent as $u = [x - x_b]^T$. Following [40], we compute the taylor expansion of the region dependent component and retain only the first two components.

$$\begin{aligned} P_k(\rho_r(x)) &= \sum_{i=1}^{n} a_i(x) exp\left(-\frac{(u-x)^2}{2\sigma_s^2}\right) \\ &= \sum_{i=1}^{n} a_i(x) \sum_{j=0}^{\infty} \frac{1}{j!} \left(-\frac{(u-x)^2}{2\sigma_s^2}\right)^j \\ &\approx \sum_{i=1}^{n} a_i(x) \left(1 - \frac{(u-x)^2}{2\sigma_s^2}\right) \end{aligned}$$

Expanding it further, $P_k(\rho_r(x))$ can be expressed as a linear combination of region independent terms:

$$\begin{aligned} P_k(\rho_r(x)) \approx \sum_{i=1}^{n} a_i(x) - \frac{1}{2\sigma_s^2} \left(\sum_{i=1}^{n} a_i(x)(x - x_i)^2 + \right. \\ \left. 2x_b \sum_{i=1}^{n} a_i(x)(x_i - x) + x_b^2 \sum_{i=1}^{n} a_i(x) a_i(x) \right). \end{aligned} \tag{4.7}$$

The region independent terms are $s_1(x) = \sum_{i=1}^{n} a_i(x)$, $s_2(x) = \sum_{i=1}^{n} a_i(x)(x-$

Figure 4.1: Kernel integral image based non-linear subspace region similarity. [First]: Template image. [Second]: Given image. [Third]: Region similarity of given image. Each pixel represents average of pixel similarity of the region centered at that pixel.

$x_i)^2$, $s_3(x) = \sum_{i=1}^{n} a_i(x)(x_i - x)$ and $s_4 = \sum_{i=1}^{n} a_i(x)$. Filtering of region independent terms over a region r can be efficiently performed using the integral image. The observation likelihood (4.3) can therefore be computed in constant time, once the integral images have been computed. It should be pointed out that the cost of computing integral image is $\mathcal{O}(n_1 \times n_2)$, where $n_1 \times n_2$ is the size of the image. However, since we densely sample the state space, resulting in filtering over many overlapping regions, the speedup obtained is about 2.5 orders of magnitude over a non integral image based implementation.

Figure 4.1 shows a simple example. Feature vectors ($[RGBXY]^T$) are extracted from the template sunflower and learned using the method described in Section 3.1. The non-linear subspace region similarity of the given sunflowers image is computed using kernel integral image framework (Equation (4.7)) and shown in Figure 4.1c. Each pixel of this image depicts the similarity of the region centered at that pixel.

4.2.2 Overall Algorithm

Input: Rectangular region of the target object in the first frame taken as target template. Extract Feature vectors of the template to learn the object eigenspace representation (Section 3.1).
Generate m particles $\{\boldsymbol{x}_0^i, w_0^t\}_{i=1}^m$ using Equation (4.2) and set weights as $w_0^i = 1/m$.

t=0;

repeat

for *i=1:m* **do**

draw samples using the state transition equation (Equation (4.2)).

end

Crop the current image so that it contains all the particle regions.

Extract feature vectors and compute integral images (Section 4.2.1.2);

for *i=1:m* **do**

Update the importance weights of the particles

$$w_k^i = w_{k-1}^i p(y_k|\boldsymbol{x}_k^i),$$

where observation likelihood, $p(y_k|\boldsymbol{x}_k^i)$ is given by Equation (4.3) and computed using kernel integral image, Section 4.2.1.2

end

Normalize the weights, $w_t^i = \frac{w_k^i}{\sum_{j=1}^m w_k^j}$;

Perform resampling;

Choose the particle with the max weight as the current estimate of the object;

t=t+1;

until *end of video*;

Algorithm 1: Proposed Algorithms

Chapter 5

Implementation and Results

We have analyzed the performance of proposed Algorithm 1 on number of tracking videos with significant change in appearance, rotation, and deformation. The algorithm was implemented in Matlab on core i5 2.4GHz processor. The run time is 2 frames/sec. Once pixel similarity values are calculated for each individual frame, region similarity can be efficiently computed in $O(1)$ time using the proposed integral image framework. In all experiments, 2000 particles are used and feature vectors are built using color and spatial values of the pixels, (i.e [RGBXY]). The standard deviation in the Gaussian kernel for color component is set to $\sigma_a = 40$, and for spatial component, $\sigma_s = \sqrt{m_1^2 + m_2^2}/(2 * 0.99)$, where $m_1 \times m_2$ is the size of the template object. This is the lower limit on valid values for σ_s, such that Equation (4.7) gives valid approximation. The number of eigenvectors used in learning the target object nonlinear subspace representation is 25.

For comparison purposes, we have used the tracking dataset of [2]. The same testing procedure as described in [2] is followed. Each experiment is

	Baseline [6]	**LGT** [2]	**Our**	PAKT [44]	FOF [43]	PF [41]	BHMC [45]	OBT [42]
hand	1.0 6.5	0.2 9.1	**0.0** **4.8**	22.6 18.5	10.0 19.7	4.3 14.4	29.9 27.4	10.0 17.5
hand2	7.0 15.5	1.9 10.3	**1.1** **9.6**	40.0 18.7	13.1 17.4	10.1 16.6	45.4 26.1	31.0 22.5
gymnastics	X	**0.2** 11.3	0.9 **9.3**	3.0 17.1	3.7 23.1	4.7 22.8	9.7 27.6	4.0 21.3
diver	X	**1.2** **13.7**	5.9 16.8	2.4 18.1	2.2 19.2	4.3 16.5	3.9 21.1	7.0 17.1
dinosaur	4.0 27.3	**0.0** **11.5**	**0.0** 16.3	8.2 23.6	2.7 19.2	2.2 23.3	15.6 35.1	7.0 30.3
torus	0.0 5.25	0.0 5.1	**0.0** **2.8**	10.6 15.2	6.0 14.8	2.5 16.4	23.4 21.5	13.0 14.8

Table 5.1: Tracking results. The first number indicates failure rate while the second number shows RMS between the ground truth center location and tracker estimate. The adaptive color tracker ACT [1] only tracked torus sequence and is not listed in the table. Last five columns are taken from Table 2 in [2] for easy comparison. X indicates that the tracker could not track the sequence.

repeated multiple times for each sequence and recorded how many times each tracker failed and needs to be restarted. Root-mean-squared error (RMS) between ground truth center point location and tracker center point location is used to measure the tracker accuracy. The proposed tracker is compared with the baseline tracker [6], local global tracker (LGT) [2] and adaptive color tracker ACT [1]. Cehovin et.al [2] tested the said dataset on five trackers; a color-based particle filter [41] (PF), an online boosting tracker [42] (OBT), a flock-of-features tracker [43] (FOF), a piecewise-affine kernel tracker [44] (PAKT) and the basin-hopping Monte Carlo tracker [45] (BHMC). They are also listed in the table for easy comparison.

Table 5.1 summarizes the results of all sequences. The first number indi-

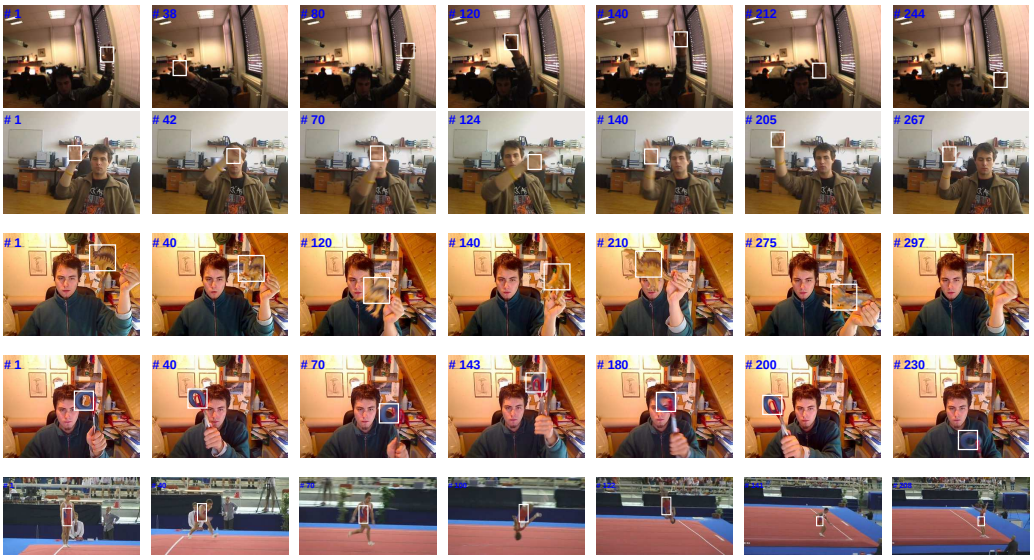
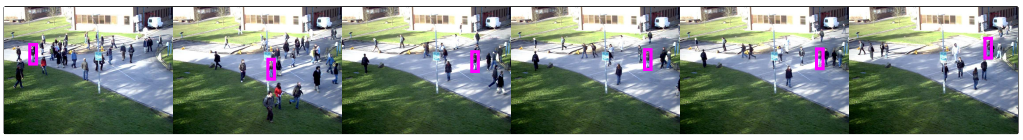


Figure 5.1: Sample tracking results of the proposed tracker, hand (first row), hand2 (second row), dinosaur (third row), torus (fourth row) and gymnastic (fifth row).

(a) Caviar data set.



(b) PETS 2009 data set.

Figure 5.2: In the Caviar data set, the target object is occluded by people coming from the opposite direction. In Pets 2009 data set, the target object is successfully tracked in a crowded scene.

cates number of failures per 30 runs of each sequence and the second number represents RMS error between the ground truth center point and tracker estimate. Figure 5.1 shows sample tracking results. In hand sequence, the

proposed tracker tracked the whole sequence without requiring reinitialization. All other comparison trackers had to be reinitialized at least once. In hand2 sequence, our tracker shows significant improvement as compared to other trackers in terms of both failure rate and error accuracy, even though it was very difficult and challenging sequence because its visual properties matches with person arm. The gymnastics and diver sequences were most challenging for our tracker as they contain rapid 360° rotation of the human body. Our tracker learns a holistic view of the target object, whereas a part based tracker may be more suitable for these sequences. Still, the performance was at par with the best tracker for gymnastic sequence. For diver sequence, the proposed tracker did not perform well. We think this is due to the use of same σ in the Gaussian kernel for both the horizontal and vertical spatial components. As the diver's aspect ratio is close to .15, this results in spurious subspace learning. However, proposed algorithm still performed better than baseline tracker which failed on both gymnastics and diver sequences.

The proposed algorithm was also tested on sequences containing partial occlusions such as the Caviar 2 sequence, whose sample frames are shown in Figure 5.2(a). The target object is occluded by people coming from the opposite direction. Similarly, the target object in Pets 2009 data set (Figure 5.2(b)) is successfully tracked in a crowded scene until the person is fully occluded.

Chapter 6

Conclusion

In this work, we have proposed a probabilistic object tracking algorithm based on particle filter. A non-linear subspace target representation is learnt using appearance and spatial information from a single target object template. Kernel eigenvalue decomposition is used to learn subspace representation due to its nonlinear characteristic, excellent discrimination capability, noise suppression, and outlier rejection. A kernel subspace integral image framework is derived to compute the observation likelihood of a candidate region in the subspace representation. This allowed the tracker to have large number of samples without loosing the computational efficiency. The proposed algorithm was tested on real-world challenging sequences and the results were compared to other state of art algorithms. Experimental results show improved tracking performance.

The tracker learns the target model in the first frame and later does not update the learnt model. Although, the tracker can handle appearance and shape changes, as the tracked sequences had considerable variations, yet the

performance can further be improved by adding an adaptive step, where the learnt model is updated to account for shape, appearance and illumination variations.

In this thesis, fixed size rectangle is used to represent object. Rectangle size is set manually for each sequence using hit and trial method to see which rectanlge size gives best results. However, rectanlge size can be set automatically by finding an object span and set appropriate width and height. During object motion with differnet rotation and pose, if rectangle size can be scaled based on visible portion of object. This can further enhance tracker performance.

Bibliography

[1] M. Danelljan, F. S. Khan, M. Felsberg, and J. v. d. Weijer, "Adaptive color attributes for real-time visual tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1090–1097.

[2] L. Čehovin, M. Kristan, and A. Leonardis, "An adaptive coupled-layer visual model for robust visual tracking," in *13th International Conference on Computer Vision*, Nov 2011.

[3] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," 2006.

[4] P. M. D. Comaniciu and V. Ramesh, "Kernel-based object tracking," in *proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, 2003, pp. 564–577.

[5] O. Arif, "Robust target localization and segmentation using statistical methods," in *Ph.D dissertation, Georgia Institute of Technology*, 2010.

[6] O. Arif and P. A. Vela, "Non-rigid object localization and segmentation using eigenspace representation," in *IEEE International Conference on Computer Vision*, 2009, pp. 803–808.

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

[8] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, "Locally orderless tracking," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1940–1947.

[9] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *IEEE Conference on Computer vision and pattern recognition*, 2012, pp. 1822–1829.

[10] J. Kwon and K. M. Lee, "Tracking by sampling trackers," in *IEEE International Conference on Computer Vision*, 2011, pp. 1195–1202.

[11] B. Liu, J. Huang, L. Yang, and C. Kulikowsk, "Robust tracking using local sparse appearance model and k-selection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1313–1320.

[12] M. Godec, P. M. Roth, and H. Bischof, "Hough-based tracking of non-rigid objects," *Computer Vision and Image Understanding*, vol. 117, no. 10, pp. 1245–1256, 2013.

[13] S. Avidan, "Ensemble tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 261–271, 2007.

[14] S. Hare, A. Saffari, and P. H. Torr, "Struck: Structured output tracking with kernels," in *IEEE International Conference on Computer Vision*, 2011, pp. 263–270.

[15] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *ECCV*. Springer, 2012, pp. 702–715.

[16] D. Ross, J. Lim, and M.-H. Yang, "Adaptive probabilistic visual tracking with incremental subspace update," in *ECCV*. Springer, 2004, pp. 470–482.

[17] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai, "Minimum error bounded efficient l1 tracker with occlusion detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1257–1264.

[18] D. Joseph Tan, S. Holzer, N. Navab, and S. Ilic, "Deformable template tracking in 1ms," in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.

[19] A. B. V. Gay-Bellile and P. Sayd., "Direct estimation of nonrigid registrations with image-based self-occlusion reasoning," in *proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[20] F. Jurie and M. Dhome., "Hyperplane approximation for template matching," in *proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.

[21] G. Hager, M. Dewan, and C. Stewart, "Multiple kernel tracking with SSD," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2004, pp. 790–797.

[22] F. Zhimin, W. Ying, and M. Yang, "Multiple collaborative kernel tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 1268–1273, 2007.

[23] S. Birchfield and S. Rangarajan, "Spatiograms versus histograms for region-based tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2005, pp. 1158–1163.

[24] C. Yang and L. D. R. Duraiswami, "Efficient mean-shift tracking via a new similarity measure," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 176–183.

[25] M. Singh, H. Arora, and N. Ahuja, "Robust registration and tracking using kernel density correlation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2004, p. 174.

[26] A. Elgammal, R. Duraiswami, and L. Davis, "Probabilistic tracking in joint feature-spatial spaces," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2003, pp. 781–788.

[27] F. Porikli, O. Tuzel, and P. Meer, "Covariance tracking using model update based means on Riemannian manifolds," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2006, pp. 728–735.

[28] P. Karasev, J. Malcolm, and A. Tannenbaum, "Kernel-based high-dimensional histogram estimation for visual tracking," in *IEEE International Conference on Image Processing*, 2008, pp. 2728–2731.

[29] S. Avidan, "Ensemble tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 494–501.

[30] M. Black and A. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *International Journal of Computer Vision*, vol. 26, no. 1, pp. 63–84, 1998.

[31] R. L. J. Lim, D. Ross and M. Yang, "Incremental learning for visual tracking," in *Advances in Neural Information Processing*. MIT Press, 2004, pp. 793–800.

[32] A. Tsai, A. Yezzi, W. Wells, C. Tempany, D. Tucker, A. Fan, W. Grimson, and A. Willsky, "A shape based approach to the segmentation of medical imagery using level sets," *IEEE Transactions on Medical Imaging*, vol. 22, no. 2, pp. 137–154, 2003.

[33] C. Ding and X. He, "K-means clustering via principal component analysis," in *International Conference on Machine Learning*, 2004, pp. 225–232.

[34] J. Meltzer, S. Soatto, M.-H. Yang, and R. Gupta, "Multiple view feature descriptors from image sequences via kernel principal component analysis," 2004, pp. 215–227.

[35] C. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *Advances in Neural Information Processing Systems*. MIT Press, 2001, pp. 682–688.

[36] P. Arias, G. Randall, and G. Sapiro, "Connecting the out-of-sample and pre-image problems in kernel methods," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 18–23.

[37] O. Arif and P. Vela, "Robust density comparison for visual tracking," in *British Machine Vision Conference*, 2009.

[38] A. Doucet, N. De Freitas, and N. Gordon, *An introduction to sequential Monte Carlo methods*. Springer, 2001.

[39] F. C. Crow, "Summed-area tables for texture mapping," *ACM SIGGRAPH computer graphics*, vol. 18, no. 3, pp. 207–212, 1984.

[40] M. Hussein, F. Porikli, and L. Davis, "Kernel integral images: A framework for fast non-uniform filtering," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[41] P. Perez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *ECCV*, 2002, pp. 661–675.

[42] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via online boosting," in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2006, pp. 6.1–6.10.

[43] J. Hoey, "Tracking using flocks of features, with application to assisted handwashing," in *British Machine Vision Conference*, 2006.

[44] B. Martinez and X. Binefa, "Piecewise affine kernel tracking for non-planar targets," *Pattern Recognition*, vol. 41, no. 12, pp. 3682–3691, 2008.

[45] J. Kwon and K. M. Lee, "Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping monte carlo sampling," in *CVPR*, 2009, pp. 1208–1215.