ADVANCED MACHINE LEARNING APPROACHES FOR TARGET DETECTION, TRACKING AND RECOGNITION

By

VIJAY VENKATARAMAN

Bachelor of Technology in Electrical Engineering
Indian Institute of Technology, Madras
Chennai, Tamil Nadu, India
2003

Master of Science in Electrical Engineering
Oklahoma State University
Stillwater, Oklahoma, United States
2005

Submitted to the Faculty of the
Graduate College of
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
December, 2010

ADVANCED MACHINE LEARNING APPROACHES FOR TARGET DETECTION, TRACKING AND RECOGNITION

Dissertation Approved:

Dr. Guoliang Fan

Dissertation Advisor

Dr. Martin Hagan

Dr. Qi Cheng

Dr. Jay Hanan

Dr. Mark Payton

Dean of the Graduate College

ACKNOWLEDGMENTS

The path leading to this dissertation has been a long one and I would like to take this opportunity to thank and express my sincere gratitude to all those who have helped me along the way to make this work possible.

First and foremost, I express my deep and sincere gratitude to my advisor Dr. Guoliang Fan for his unconditional support, patience, and encouragement throughout my graduate studies. His stimulating suggestions and encouragement have helped me greatly during the course of my research work and in the writing of this dissertation. I greatly appreciate his unwavering dedication through which he always made time to discuss ideas, help with papers and talk about the little problems that crop up in research at any time of the day or night. His enthusiasm for research was a constant motivational force for me and helped me sail through many difficult times in the course of my research work. I consider myself to be very lucky to have an advisor like Dr Fan. Finally, I would like to thank him and his family for considering the students of the lab to be part of their extended family and hosting us several times over the course of the years.

I would like to thank the Principal Investigator of this research project, Dr. Joseph Havlicek, University of Oklahoma, Norman for procuring the funding for this work and his active interest in my research. I am extremely grateful for his diligent efforts and the amount of time he spent helping me revise my journal manuscript. The additional experiments that he suggested were critical in strengthening our research idea and also helped me gain a deeper understanding of the research problem. I also gratefully acknowledge the funding sources that made my Ph.D. work possible.

The members of the VCIPL lab have contributed immensely to my personal and professional time at OSU. This group has been a source of friendship as well as good collaboration for my research. I am especially grateful to my peer group members who stuck it out in grad school with me: Xin Zhang and Yi Ding. I would like to acknowledge the former post doctoral member of the lab Dr. Xin Fan (Frank) for his contribution in our joint research work on tracking and recognition. I am thankful to Chunna Tian (Tina), Xin and Frank for sharing their program codes and taking time to explain the code to me. Other lab members, past and current, to whom I am thankful include: Jung-Jae (Brian) for his time in rendering 3D models and sharing his computer resources with me, Liangjiang for developing a nice interface to showcase my research work with a faster implementation of my code, Colby for developing the covariance tracker code and Micahel for several amusing conversations during lunch time. Other past and present group members that I have had the pleasure to work with or alongside of are Thitiporn, Cheng, Dr. Li Tang, Prof Liu, Nazmul, Ayesha, David, John and the numerous visiting scholars who have come through the lab. Thank you all for the time spent together, the happy times in our lab and group meetings will be always be cherished in my memory.

For this dissertation, I would like to thank my committee members Dr. Martin Hagan, Dr Jay Hanan and Dr. Qi Cheng for their time, interest and helpful comments. I also thank Dr. Fatih Porikli of Mitsubishi Electric Research Labs, Cambridge, Massachusetts for providing me an opportunity to work with him as an intern. This internship was greatly rewarding and exposed me to research outside the academic sphere. I am grateful to Dr. Fatih for his contributions of ideas, time and effort in the development of the detection algorithm presented in this dissertation. Finally, I would like to sincerely thank Dr.Yarlagadda Rao for all his valuable remarks that have contributed towards both my professional and personal life.

Last but not the least, I want to thank my family members and close friends for their support during the years I have been working on this research. My parents receive my deepest gratitude for their dedication and the many years of support during my undergraduate studies that provided the foundation for this work. I want to thank my wife, Vidya the most for her love, sacrifice and kind indulgence over the course of my research work. I am thankful for her help in making many of the figures and aiding me organize my thoughts into succinct sentences. Had I written this paragraph myself without her help; I couldnt have put it any better. Thank You !!!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 Motivation

In the present day world, the availability of affordable, high quality image and video capturing devices has generated an abundance of visual data and the need for automated visual data analysis. Over the past two decades a number of such techniques have been proposed ranging the topics of image enhancement, automated surveillance, vision aided navigation, automated object detection, tracking and recognition, human-computer interaction etc. Further, the field specific advantages of imaging in the non-visible band (infrared for night surveillance, synthetic aperture radar (SAR) for subterranean resource mapping) has resulted in the popularity of imaging in non-visible frequencies. In such cases, data analysis techniques are either adopted from those of the visual band or developed specifically to deal with the intricacies associated with the specific non-visible band.

Detection, tracking and recognition tasks are of great importance in military applications and are incorporated into a number of systems that aid in keeping track of a large number of targets over vast spaces in the battleground. They are also used in sensor based missile guidance systems. Civilian applications include highway traffic monitoring, restricted area surveillance, border security etc. The task of tracking and recognition is challenging because targets of interest do not always appear the same and can have different signatures based on their pose, camouflage etc. In addition, the sensors used to acquire information are affected by many intrinsic (focal length, IR sensitivity etc) and extrinsic factors (temperature, reduced visibility, atmospheric conditions etc). Further, there are problems associated with varying backgrounds, presence of clutter, occlusion, interaction between various targets etc. All of these issues necessitate the need for a robust system that can accomplish the tasks of detection, tracking and recognition under adverse field conditions.

The focus of this dissertation will be on the key components of an Automatic Target Recognition (ATR) system namely: target detection, tracking, learning and recognition with greater emphasis on infrared videos, however the techniques discussed can easily be extended to videos in the visual band. We begin with a formal definition of a few key terms that will be used through this dissertation. Detection refers to the process of locating an object of interest (target) in the image obtained from the imaging sensor such as a video camera, forward looking infrared (FLIR), synthetic aperture radar (SAR) etc. Tracking refer to the process of being able to continuously follow the movement of the target directly on the image plane or infer its location in in real world co-ordinates. Recognition is the task of associating the target with a class label pertaining to its class, type or identity.

1.2 Research goals and challenges

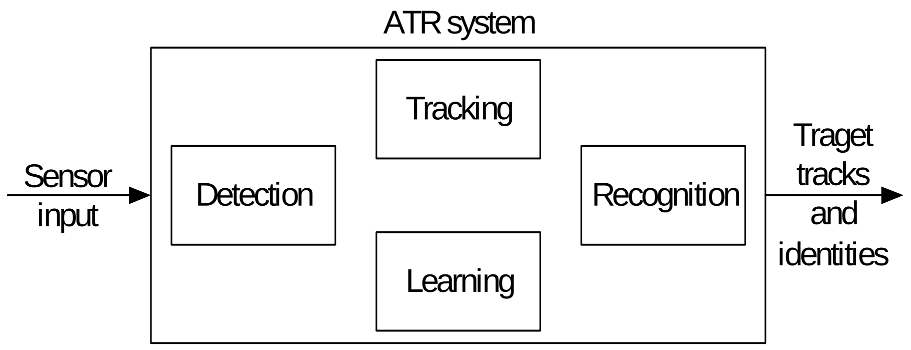


Figure 1.1: Components of an integrated ATR system.

The overall goal of this research is to develop an integrated ATR system capable of detecting, tracking and recognizing targets of interest in a given infrared video sequence. Fig. 1.1 shows the components that make up an ATR system and in the following we discuss the challenges related to each component

- **Objective 1: Target Detection** Target detection is the process of identifying areas in the image, that could possibly belong to a target of interest. Target detection is one of the most fundamental and challenging tasks in computer vision. Since detectors often form the first stage of the consecutive tracking and recognition tasks it is also vitally important the detector be both accurate and fast.

  Challenges

  1. Small target size. Low spatial resolution of the target region often implies there is very little information available about the target that can be used to distinguish it from the surrounding background and clutter.
  2. Variable thermal signatures, movable parts, pose variations. These variables result in many possible appearance for a single target making detection more difficult.
  3. Low signal-to-noise (SNR) ratio sensor images. The presence of strong sensor noise and background clutter make distinguishing the target from background very difficult.
  4. Environmental conditions. Environmental conditions like the time of the day, relative position of the sun, relative humidity etc can affect the amount of thermal radiation received by the sensor and thereby alter the appearance of the target.

  The problems faced by a detector are well illustrated by the sample frames shown in Fig.1.2. In Fig.1.2 (a) and (b) the military targets are inconspicuous and blend in easily with the surrounding clutter making it difficult to identify them. In Fig.1.2 (c) there are multiple targets of interest present and they are obscured by trees and surrounded by similar looking rooftop structures. A robust detector will require the use salient region descriptors that can accurately model the target appearance and

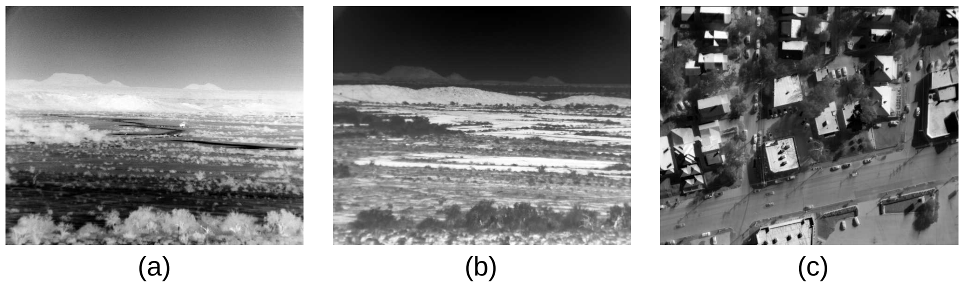


Figure 1.2: Examples of Medium Wave InfraRed (MWIR) images presented to the detector to identify (a) single military target at night time (b) single military target at day time (c) multiple civilian targets imaged from an UAV.

competent classifiers that can distinguish the large pool of target appearances from every possible background and clutter.

- **Objective 2: Target Tracking**

  Tracking refers to the process of determining the target position either on the image plane or inferring its position in the real world 3D co-ordinates using information from the image.

  Challenges

  1. Targets of interest often are non-cooperative and exhibit strong maneuvering action as an evasion tactic. Further, the imaging sensor platform is often times airborne and exhibits strong ego motion. The difficulty in modeling such unpredictable actions makes the tracking task challenging.
  2. Time varying target signatures. Due to motion of the sensor platform relative to the target, its appearance as observed by the sensor will be different at different time instances. Therefore the appearance description of the target has to be continuously updated to maintain a robust track.
  3. The task of inferring 3D position from 2D image observations is an ill-posed problem as information of distance along the sensors field of view is lost.

4. Small target size. The small target size makes it difficult to compute complex and powerful features to describe the target area.

5. Low signal-to-noise (SNR) ratios, poor target visibility, occlusions and presence of multiple targets make the task of maintaining stable and continuous track difficult.

Tracking in general depends on the robustness of two important models (1) the target appearance and (2) the target motion or kinematics. The appearance model provides a description of the target area usually in the form of extracted features from the target area. The motion model describes the typical motion of the target. The tracker then infers the true location of the target based on prior knowledge contained in the two descriptive models and the observed frames. In this work, we consider the problem of tracking maneuvering ground targets in infrared (IR) imagery acquired from an airborne platform. The challenges described above prohibit the straightforward extension of existing visible band tracking algorithms to FLIR images. Therefore there exists a need to develop appropriate appearance and kinematic models to suit the unique challenges imposed by infrared video sequences.

- **Objective 3: Target Appearance Learning**

  Target appearance learning is the process of updating and maintaining a valid description of the target appearance that is used to track the target.

  Challenges

  1. Maneuvering action and sensor platform motion. These two independent motions imply that the appearance of the target on the sensor plane continuously changes with time. Therefore the appearance model used by the tracker has to be updated over time for robust tracking.

  2. Choice of appearance representation. The procedure used in the update of the

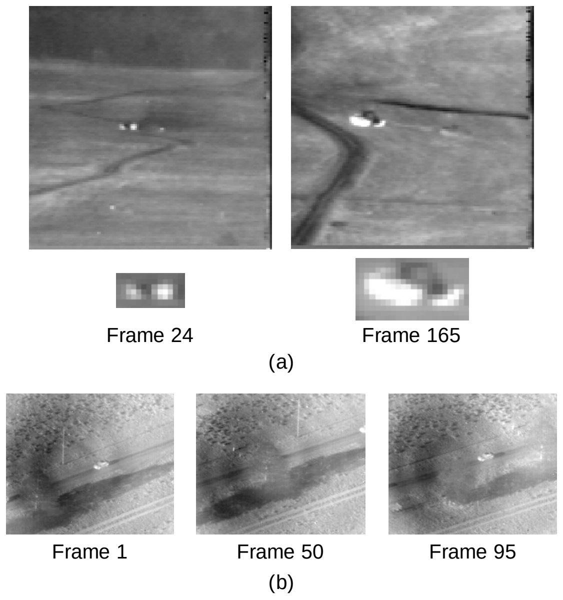


Figure 1.3: (a) Example of nonstationary target signature evolution in AMCOM LWIR run `rng18_17`. There are two vehicles. The lead vehicle is barely visible. The second vehicle, which is clearly visible, is the target of interest. Top row: Observed frames. Bottom row: closeup views of the target. (b) Example of target being occluded by foliage in the VIVID dataset.

appearance model greatly depends on the choice of the appearance representation.

3. Occlusion. When a target is occluded its appearance is usually replace by that of clutter. In such circumstances it is important to prevent update of the appearance model so that when the target reappears it is possible to resume tracking.

An example of the profound nonstationary variations in target appearance over relatively short time scales is shown in Fig. 1.3(a). Here, a longwave imaging sensor is situated on an airborne platform that closes on a pair of maneuvering ground vehicles. Profound changes in the target's appearance are observed between frames 24 and 165 over a time scale of only a few seconds and arise primarily from the relative motion between the sensor and the target. There is substantial magnification that results from the sensor closing on the target and pose change that results from the target executing an aggressive turning maneuver. While the second vehicle in Fig. 1.3(a) exhibits a strong signature, the lead vehicle is much dimmer and is barely visible amid the surrounding clutter, demonstrating that brightness alone cannot be used as the sole basis for reliable detection and tracking. Rather, more sophisticated techniques are generally required for representing the target appearance and for adapting to (*e.g.*, learning) complex appearance changes that occur over time.

Further, in many cases the target may move out of the sensor's view or may become occluded thereby significantly altering the observed appearance. An example of this is shown in Fig. 1.3(b) where the target being observed moves behind a tree along its path and thereby disappears from the sensor's view. While it is important to adapt the appearance model to accommodate variations in the target signature it is equally important to avoid learning the appearance of occluding objects or the background.

- **Objective 4: Joint Tracking and Recognition**

  Recognition is the process of identifying with a target with a class or identity label. In most cases recognition is intertwined with the tracking process and the identity associated with the target increases in confidence over time.

  Challenges

  1. Variability among different target types. The variability in appearance across different target types is vast and it is necessary to develop a model that is capable of compactly representing these variation.
  2. Appearance variability due to viewing angle and distance further necessitate a model with the above mentioned capabilities.
  3. Motion cues. Taking advantage of the peculiar motion characteristic of a target can help in categorizing it with higher confidence and therefore must be built into the joint tracking and recognition process.

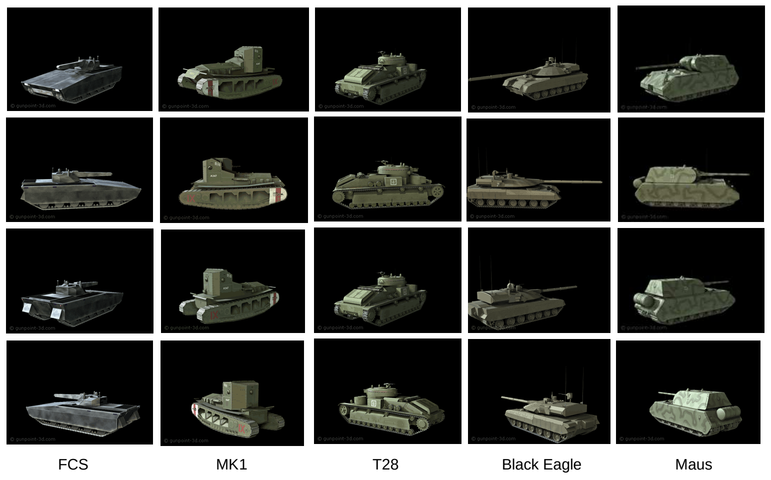


Figure 1.4: 3-D target models of five representative tanks illustrating the variability in target appearance.

In addition to time variations in the target appearance, recognition is greatly complicated by the immense variability among target types as shown in Fig. 1.4. It would be a daunting task to store all possible appearance of various targets for recognition purposes. Therefore we will need an efficient method that can be trained from a small training set to characterize shape variation due to the factors of class identity and viewing angle.

1.3 Contributions and outline

In the following we give a brief description of the specific contributions of our research with respect to the above mentioned objectives and challenges.

- **Contribution I**: A new sparse feature termed as relational combinatorics feature *RelCom* is defined, that encapsulates higher order spatial structure information within the target window. The RelCom feature is not limited to pixel features and can easily accommodate any vectorized descriptor of the target area. The feature can be thought of as a weak classifier that imposes a relational rule over the vector descriptor. A number of such weak classifiers are combined using boosting (AdaBoost) to define a strong classifier with acceptable performance that can distinguish between targets and non-targets. The advantage of the RelCom classifier is that it is fast, accurate and not limited to any particular feature type.

- **Contribution II**: A dual histogram appearance model with feature selection for tracking small maneuvering targets in FLIR images. We propose the use of a quad histogram based appearance model to represent the target area. Specifically we use histogram representations of the intensity and standard deviation information from both the target area (foreground) and the area surrounding the target (background). The histogram features are fast and easy to compute and are robust in cases where the target is extremely small in size. Further, a feature selection mechanism to assign varied importance to the histogram features during the tracking process is also presented.

- **Contribution III**: An appearance learning framework to account for variations in the target signature. Though histogram features are fairly robust, over time they may become invalid, due to movement of the target or imaging sensor, and result in track loss. To overcome this effect, we model the evolution of the histogram over time as a state space system and apply an Adaptive Kalman Filter (AKF) to estimate the states. The AKF makes use of the autocorrelation of the filter residues to estimate the unknown system and measurement noise variances, required in the filtering process.

- **Contribution IV**: A generative model framework for joint tracking and recognition. In order to account for the multitude of target appearance variations, we propose the use of a nonlinear tensor-based generative model that can synthesize a target signature given the target type and an arbitrary pose. In addition to aiding the tracker by accounting for inter-frame appearance changes, this model also facilitates recognition by generating distinct type-specific appearances with any pose. In addition, a target dependent generative motion model is proposed to account for the mechanical variability among the target types. These two generative models are coupled in a graphical model framework for joint tracking and recognition.

- **Contribution V**: The generative model mentioned in the previous step, deals with discrete identity labels. We extend this generative model by introducing the concept of a continuous valued identity manifold. This identity manifold allows us to recognize not only a known target, but also an unknown one by interpolating the shape meaningfully between two training targets. Additionally, we develop a new multi-view shape-based generative model that integrates a hemisphere-shaped view manifold with this identity manifold to provide simultaneous identity and pose estimation.

This work follows in the spirit of the evolution of ATR research over the past decade [1], and is fueled by several recent advances in the field of machine learning and computer vision. A tabular form of the contributions and the associated advantages is shown in Fig.1.5. It is seen that the solutions to the challenges in different components of the ATR system are dependent on a variety of broad technical areas related to machine learning, computer vision, signal processing etc. For example, boosted classifiers were mainly developed for use in pattern recognition, Adaptive Kalman filters were primarily developed for use in the field of signal processing and state estimation. In this research, these have been integrated together to develop a hybrid ATR system. The rest of this dissertation is organized as follows:

- In **Chapter 2** we provide an overview of the existing literature in the fields of target detection, tracking and recognition in the context of both visual and infrared imagery.
- In **Chapter 3** we discuss the development of a low false alarm and low computational complexity target detector based on a novel feature termed as Relational Combinatorial (RelCom) feature.

- In **Chapter 4** we present a particle filter tracker based on a quad histogram appearance model for target tracking in FLIR images under difficult field conditions along with the concept of feature selection to decide the relative importance of the four different histograms to maintain a robust track.
- In **Chapter 5** we greatly improve the tracker presented in Chapter 4 by considering the issue of appearance learning using Adaptive Kalman Filters (AKFs). We show the advantage of our proposed auto-covariance least squares technique over traditional methods by experiments on both real and simulated data.
- In **Chapter 6** we discuss a joint motion-appearance generative model for simultaneous 3D target tracking and recognition. This model uses both the target appearance and motion dynamics in determining the target identity. Further, we decompose the shape variability in the given training set into the two factors: discrete identity label and a continuous view angle by means of non-linear tensor decomposition.
- In **Chapter 7** we extend the generative model in Chapter 6 by removing the constraint of discrete target identity labels. We introduce the concept of identity manifold to account for both inter and intra class shape variations. In addition to being able to deal with arbitrary view variations, this model allows us to determine the identity of an unknown target at both the class and sub-class level.
- In **Chapter 8** we conclude the dissertation with the discussion of future work.

Objective	Detection	Tracking	Learning	Joint tracking and recognition	Joint pose and identity recognition
Methodology	Boosted classifier	Particle filter	Adaptive Kalman Filter (AKF)	Non-linear tensor decomposition, graphical model	Coupled view-identity manifolds for multi-view target modeling
Contribution	• RelCom features - a sparse representation that captures spatial structure of the target	• Dual foreground background histogram appearance model • Online feature selection	• Use a bank of AKFs for appearance histogram learning • Application of autocovariance least squares, to estimate unknown system noise variances	• Generative models for both target appearance and motion • Exploit both appearance and motion cues in an integrated graphical framework	• Continuous-valued identity manifold that captures both inter-class and intra-class shape variability combined with a hemispherical view manifold
Advantage	• Fast detection using lookup tables • Low false alarm rate • Extendable to all vectorized features	• Accurate tracking of target position and size • Handle small sized and hard to see targets	• Ability to track over long time periods • Handle appearance change • Occlusion handling • Improved accuracy of position and size estimates	• Joint estimation of 3D position, pose and target identity • Improved identity estimation by associating motion with target type	• Joint estimation of target identity and pose • Identity of previously unknown targets estimated at both class and sub-class levels.

Figure 1.5: The main contributions of this research and associated advantages.

CHAPTER 2

Literature Review

This chapter provides an overview of the existing literature in the fields of detection, tracking, appearance learning and joint tracking and recognition.

2.1 Detection

Detection is the first and a very important component of an ATR system. Based on the survey presented in [2], detection methods can be separated into four distinct categories as shown in Fig 2.1. A brief discussion of the different methods with insights from [2] is presented in the following.

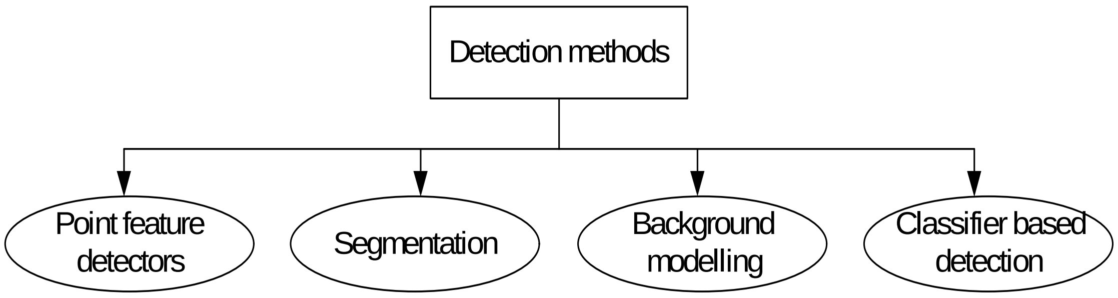


Figure 2.1: Broad classification of detection methods in literature.

2.1.1 Point feature detectors

Point feature detectors are used to find interest points on the image, and are mostly located on distinctly textured areas, strong gradient locations, corners etc. Commonly used point detectors include Harris interest points [3], SIFT features [4] etc. The Harris detector relies on the local gradient information and tries to identify points that show strong variations in the selected vicinity. SIFT feature points have the additional capability of identifying interest points across multiple scales and is more capable of handling image deformations.

2.1.2 Segmentation

Segmentation is the process of dividing a image into perceptually similar regions. Effectiveness of segmentation algorithms is largely dependent on the criteria for finding a good partition. Graph cuts [5] are a popular method for segmentation in visual band images where the targets are fairly large in spatial extent. In [6] it is shown that feature descriptors developed primarily for visible spectrum can be adopted in infrared in the case of larger targets like pedestrians. Active contours [7] and morphological operator [8, 9] based segmentation methods are commonly used in infrared images for detecting small targets.

2.1.3 Background subtraction

Background subtraction based detection works by building a representation of the background or scene and any region of the image that deviates from this model is labeled as object/foreground. Common models include a Gaussian model or a Gaussian mixture model of the pixels/regions over the intensity or color feature space. Over the years background subtraction algorithms have been successfully developed to mitigate the effects of changing illumination [10], noise and periodic background motion [11]. This makes these approaches very suitable for use in fixed camera systems where the background is relatively slow changing and any significantly moving object is the target.

2.1.4 Classifier methods

These methods require training a classifier to distinguish between targets and non-targets using exemplars from both categories. The choice of features used to represent the exemplars plays an important role in the effectiveness of the classifier and therefore must be chosen so as to be discriminative between the two classes. Haar wavelets have become popular due to their efficient computation [12]. More recently, histogram-based representations of image gradients in spatial context, including the histogram of oriented gradients (HOG) [13], the scale-invariant feature transform [4], the shape context [14], were shown to yield more distinctive descriptors. In [15] a region was represented by the covariance matrix of image attributes in addition to histograms. The list of common descriptors can be extended to Gabor filters, appearance templates, local binary patterns, etc. The explosion of available features has led to the application of data mining approaches [16, 17] for feature selection. The objective of the classifier is to determine a hypersurface that separates the two classes in hight dimensional feature space. Neural networks, boosting [18], support vector machines (SVM) [12] and decision trees are some of the most commonly used classifier methods.

2.2 Target tracking

In line with the comments by Cominiciu *et.al* in [19] a typical visual tracker comprises of two major components namely, *target representation* and *filtering*. The former is mainly associated with the appearance description of the target and adapting to changes in appearance. The latter mainly relates to the dynamics of the target and evaluation of different hypothesis. Fig. 2.2 provides a list of the common techniques used in visual tracking. In most cases the tracker is based on a chosen technique from each of the two components. This section analyzes some of the existing tracking algorithms and their comprising systems.

2.2.1 Interest points

When using interest points to represent a target, the points are usually the output of a detection stage. The objective in this approach is to associate the detected points at every frame with a unique target track. The problem becomes one of identifying the correspondence between the detected points in subsequent frames. There have been a number of proposed

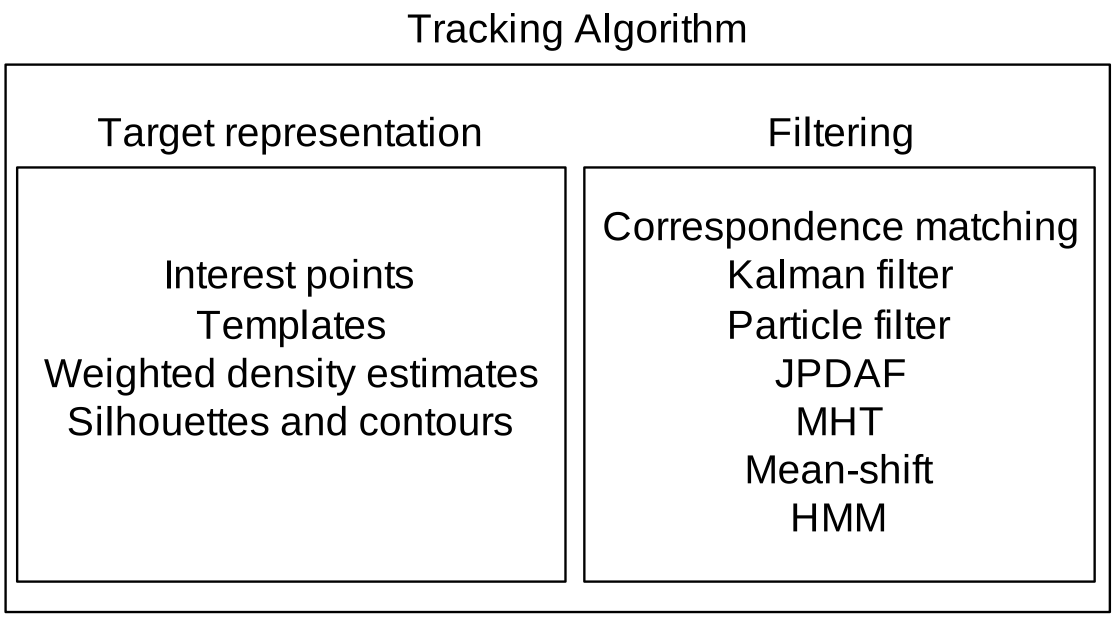


Figure 2.2: The two major components of a tracking algorithm target appearance modeling and filtering. Some of the commonly used methods in each component are also shown. A tracking algorithm may be constructed by choosing a method from each component.

solution to solve the **correspondence matching** problem of which the most notable is the Greedy Optimal Assignment (GOA) proposed by Veenman *et.al* [20]. In [21] detector results obtained from background subtraction are processed through a template mask and used by a **Kalman Filter** to estimate the position of the person. In cases were multiple targets are present, Joint Probability Data Association Filtering **(JPDAF)** [22] can be implemented to associate a single observation with each unique target. The Multi Hypothesis Tracker **(MHT)** [23] on the other hand maintains multiple hypothesis for each target in any given time frame and the final track of the target is the most likely hypothesis over a set of given observation frames. The use of interest points is most suitable in cases where the object is very small and can be represented by a point. For larger objects multiple points will be required to effectively track the target.

2.2.2 Templates

Templates that are often a small image of the target area, are a simple way to depict the target appearance. They carry both spatial and statistical information about the target area. Templates however are limited to a single view and size of the target and are most useful when the target does not vary considerably in the observed time frame. Examples of a simple template based **particle filter** tracker are discussed in [24, 25].

2.2.3 Weighted density estimates

Here appearance features of the target area such as greyscale intensity, color or gradient information is represented as densities. The form of the density can be parametric (gaussian, gaussian mixture) or non-parametric (histogram). Usually in such representations more weight is attached to pixels closer to the target center and less weights assigned to pixels near the background. This is done to reduce the effect of background information corrupting the target appearance when the target boundary is not perfect. Due to its invariance to scale and slow varying nature, intensity histograms are widely employed for target representation [19, 26, 27]. In [19] a **mean-shift** based tracker that searches for the target in the neighborhood of its previous location is presented. The mean-shift algorithm uses a histogram representation of the target and candidate areas are evaluated using the Bhattacharya distance.

2.2.4 Silhouettes and contours

Silhouette representation of a target can be thought of as a shape template and is usually obtained from determining the target edges. Shape matching is then performed to identify the location of the target. Shape matching can be explicitly dependent only on the edges or also include the information contained within the edges. Contours on the other hand, can be thought of as target boundaries that continuously evolve over time. In [28] a **particle filter** is used to optimize a set of spline and affine motion parameters to fit the target. Methods that represent the contours in parametric form (spline) do not allow operations like split and merge that are useful when dealing with multiple targets. In such cases, direct minimization of the contour energy functional is preferred and is achieved through gradient descent or greedy methods. The contour energy function is usually dependent either on optical flow or appearance statistics of the target and background [7].

2.3 Appearance learning

This section provides a brief review of the existing works on the important issues of appearance representation, learning and occlusion handling. Appearance learning strategies strongly depends on the appearance representation as shown in Fig. 2.3.

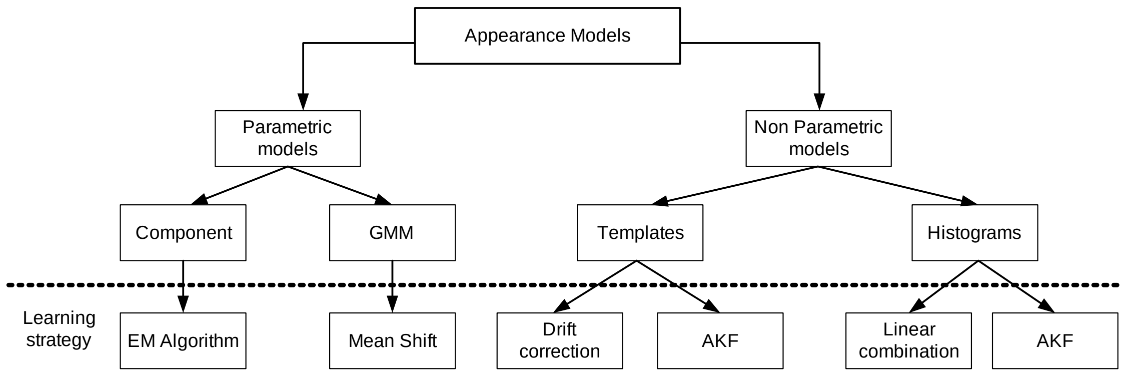


Figure 2.3: Different appearance learning strategies depending on the target appearance model of choice.

2.3.1 Parametric models

Parametric models are in general a statistical model that captures the key characteristics of the target appearance in a way that facilitates estimation of the model parameters continuously online [29]. A sophisticated model combining stable, wandering, and outlier components in a Gaussian mixture model (GMM) was proposed in [29], where the model was updated via an expectation maximization (EM) algorithm. GMM based appearance learning was also applied in [30], where a mean-shift algorithm was used to update the parameters online. These methods rely on elaborate parametric models and are effective for tracking extended targets with large spatial signatures. However, in case of infrared images where the targets are very small, there may not be enough pixels on the target to achieve robust and statistically significant parameter estimation.

2.3.2 Non-parametric models

Non-Parametric models are ones in which the target appearance is characterized by empirically derived features that represent certain characteristics of the target appearance. Such features may include simple templates, kernel-based windows [31–33], or local statistics [27, 32] including intensity histograms and their moments.

Drift correction strategies for template tracking were proposed in [34, 35]. Here the new template is drift corrected so as to match closely with the reference template given in the first frame. The adaptability of this technique in cases where there is significant change in the target's appearance over time is questionable. A robust Kalman filter was also developed for appearance learning in [36] where the intensity values of the target template are estimated by means of state space model. Here the process noise was assumed known and covariance matching was used to estimate the variance of the innovations.

For histogram-based target representations, appearance learning is generally accomplished by iteratively updating a reference histogram [37–39]. Typically, the new reference histogram at each iteration is given by a linear weighting of the previous reference histogram and the most recent observation, where the weighting coefficient may be based on an appropriate measure of histogram similarity. While such techniques are often effective for adapting the appearance model when the target has a large spatial extent, they can be susceptible to drifting problems, particularly when applied to smaller targets. Improved histogram estimation was achieved by modeling the temporal evolution of the reference histogram in an adaptive Kalman filtering (AKF) framework in [37]. In [40, 41], the AKF measurement noise variance was estimated from the first frame and was assumed stationary, while the process noise variance was estimated online using covariance matching [42]. A robust Kalman filter was also developed for appearance learning in [36], where the process noise was assumed known and covariance matching was used to estimate the variance of the innovations.

Various methods have been developed for handling temporary track loss, especially occlusion. Typically, occlusion can be detected by investigating the distance between candidates and reference representations. The distance between the contours of objects was used in [7]. Latecki and Miezianko incorporated motion cue into the definition of the template distance [43]. Wu *et al.* explicitly introduced a state variable as the indicator for occlusion into the dynamic Bayesian networks in order to estimate the probability of occlusion. In their approach, the likelihood is also defined based on the template distances [44]. For the histogram representation, the percentage of outliers are taken as the index to occlusion in [40], and the outliers are classified based on the residuals in the Kalman filtering of histograms.

2.4 Joint tracking and recognition

Joint vehicle tracking and recognition is a prevalent and challenging issue in many civilian and military surveillance applications. One major challenge in vehicle appearance modeling is that of representing appearance variability both within a vehicle class and across different vehicle classes. For example, appearance models in particular must accommodate the variation in appearance over time that occurs due to pose variation or partial occlusion. A brief overview of some of the common appearance models that lend themselves to target recognition are shown in Fig. 2.4.

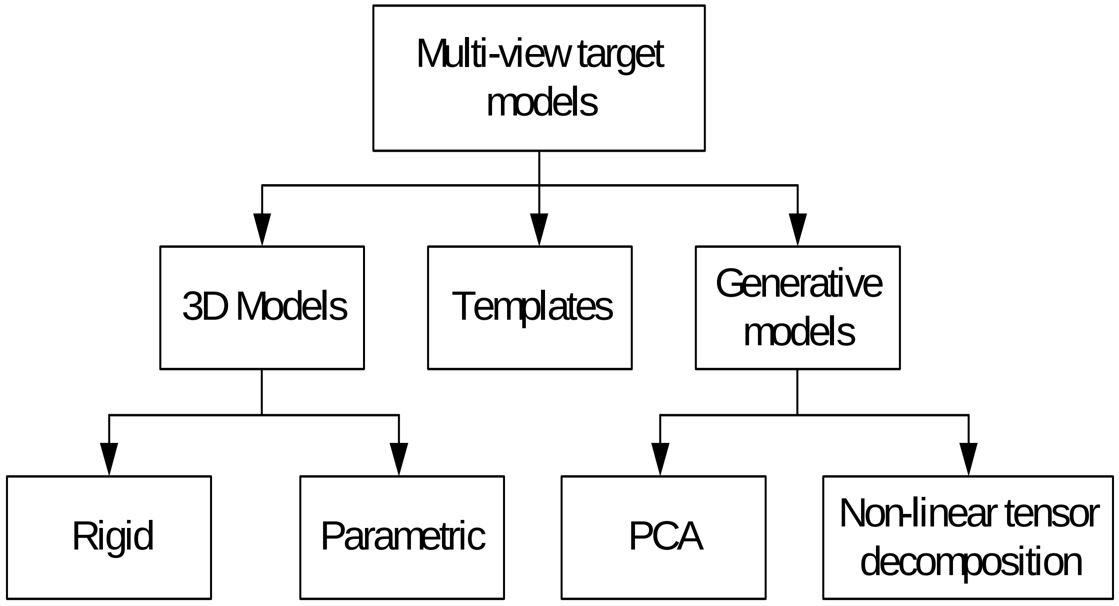


Figure 2.4: Different appearance learning strategies depending on the target appearance model of choice.

2.4.1 Multi-view target models

There are two theories on object representation. One suggests a set of representative 2D snapshots [45, 46] and the other involves a 3D object model [47]. In the first theory, unknown views can be interpolated from the given ones, and while in the second one, the 3D model is used to match the 2D observation via 3D-to-2D projection. Accordingly, most object recognition methods can be categorized into two groups, i.e., those involving 2D multi-view images [8, 48–52] and those supported by explicit 3D models [53–56]. Some make use of both the 3D shape and 2D appearances [57]. A variety of 2D features (e.g., silhouettes, edges, HOG, SIFT) or 3D models (e.g., meshes, polyheadrons) were used in these methods. The psychophysical evidence [58] motivates us to use 2D view-based silhouettes for multi-view object representation.

The ready availability of 3D CAD models of common vehicular targets has made their use very popular in tracking and recognition systems. Lou, et al., proposed the use of a predefined 3D shape model to track the position and pose of a vehicle in [53] by making use of the ground plane motion constraint. In [59], the shape and pose were characterized explicitly using a deformable model with multiple parameters that must be optimized continuously for localization and recognition.

Templates are the simplest way to characterize appearance difference among target types and views. In [48], shape information was exploited along with the object appearance in order to classify vehicles according to type, where only a few representative poses were considered for each type.

Generative models provide a way to parameterize the appearance variations caused by identity and pose variations. In the context of face tracking and recognition, appearance models adaptive to pose changes were also studied extensively, including Principal Component Analysis (PCA) based models [29, 60, 61] and non-linear manifold based methods [62]. One early work in [63] applied PCA to find two separate eigenspaces (one for all objects and one for a specific object under different poses) for joint identity and pose estimation. The bilinear models [64] and multilinear analysis [65] have provided more systematic multi-factor representation by decomposing HD observations into several independent factors. In [66], the view variable is related with the appearance through shape sub-manifolds which have to be learned for each object class. Recently, multi-linear analysis was combined with manifold learning to provide a generative model-based human shape representation that is specified by multiple factors including the identity (body shape), pose, and view [67]. The major advantage of this model is that it is able to synthesize an unseen observation under an arbitrary view given an identity and a pose, which can aid the tracker in accounting for inter-frame appearance variation that occurs due to changes in pose or view. Moreover, this approach also facilitates recognition with a built-in identity variable. This appearance generative model is one of the two major components in our research that can be easily connected with a type-dependent motion model by sharing a single identity variable.

Along another line of thinking, some methods can synthesize novel 3D shapes from a set of 3D objects belonging to the same class. For example, in [56], a 3D object is represented by mesh vertices that are matched to salient feature points on the observed image. The main challenge is to determine correspondences between the model and observation. A correspondence-free method was presented in [55], where multiple 3D objects are used to develop a generic 3D shape prior, and PCA is applied to generate novel 3D shapes. Then the shape and pose parameters can be optimized jointly by maximizing the degree of match between the 2D projection of the novel 3D model with given the segmentation boundary of the unknown object. In contrast, our generative model is controlled by two independent variables constrained on their own low dimensional manifolds, making the inference process very efficient and flexible.

2.4.2 Motion models

Motion modeling is a second important component of joint tracking and recognition. For example, multiple motion models have been widely adopted to accommodate different maneuvering actions in vehicle tracking [68]. In reality, a vehicle is equipped with a specific engine and mechanical system that generates a unique motion pattern and maneuverability. This inspires researchers to develop multiple type-dependent motion models to achieve joint vehicle tracking and recognition [69, 70]. However, these approaches require sensors capable of providing direct measurements of the kinematics (radar, for example), and do not consider the vehicle appearance captured by passive imaging sensors such as electro-optical (EO)/infrared (IR) sensors. We recently proposed a generative model-based maneuvering vehicle tracking approach that is able to capture the underlying physical constraints in the mechanical system of a maneuvering vehicle and requires only passive imaging sensors [71].

2.4.3 Integrating motion and appearance cues

Integrating motion and appearance cues for joint tracking and recognition is attractive because of their complementary nature. In the context of activity recognition, for example, object trajectories aggregated from past tracking history have been employed as important cues for classification of high-level object activities [72–74]. In addition, recent face tracking and expression recognition algorithms have incorporated multiple temporal models of facial features conditioned on the variable to be recognized, i.e. facial expression [75, 76]. However, in these methods there is no direct dependence between the identity (object activity or facial expression) and the object appearance; thus the appearance contributes only indirectly to recognition through tracking crucial signatures (trajectories and facial features) that determine the identity. On the other hand, identity-dependent appearance models are widely used in joint face tracking and recognition [62, 77, 78], but no identity-dependent multiple dynamic (motion) models have been used. Motion models are used for appearance update in [60, 62, 62], but these have no direct impact on recognition. In our work, we present a new approach where both appearance and motion models are dependent on the vehicle identity and are integrated into a unified probabilistic framework.

CHAPTER 3

Target Detection using Relcom Features [1]

3.1 Background

Small object detection still remains one of the most fundamental and challenging tasks in computer vision. On the core, it requires salient region descriptors that can accurately model object appearance and competent classifiers that can distinguish the large pool of object appearances from every possible background and clutter. Detection in infrared images is especially challenging due to the low spatial resolution of the object region. Variable thermal signatures, movable parts, combined with external illumination and pose variations, contribute to the complexity of the problem. Since detectors often form the first stage of the consecutive tracking and recognition tasks it is vitally important the detector be both accurate and fast.

Here we introduce the relational combinatorics features *RelCom*. We first generate combinations of low-level attribute coefficients, which may directly correspond to pixel coordinates of the object window or feature vector coefficients representing the window itself, up to a prescribed size n (pairs, triplets, quadruples, etc). We then apply relational operators such as margin based similarity rule over each possible pair of these operands. The space of relations constitutes a proposition space that divides the original feature space into discrete regions. From this space we define combinatorial functions of Boolean operators to form complex hypotheses as shown in Fig. 3.1. Therefore, we can produce any relational rule over the operands, in other words, any logical proposition over the low-level

[1] The work presented in this chapter was done in collaboration with Dr Fatih Porikli when the author was an intern at Mitsubishi Electric Research Labs (MERL).

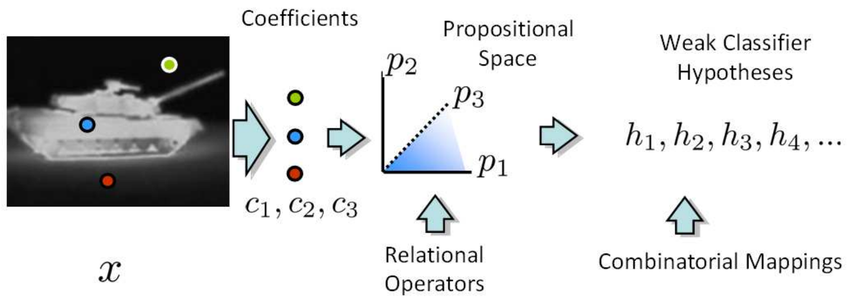


Figure 3.1: RelCom: After a number of coefficients are selected from the input image (or feature vector), a set of relational operators are imposed to generate a discrete proposition space, from which hypotheses are constructed by applying combinations of Boolean operators (conjunction, disjunction, etc.).

descriptor coefficients. In case these coefficients are associated with pixel coordinates, we encapsulate higher order spatial structure information within the object window. Using a descriptor vector instead of pixel values, we effectively impose feature selection without any computationally prohibitive basis transformations such as PCA. In addition to proposing a simple methodology to encode the relations between n pixels on an image (or n vector coefficients), we employ boosting to iteratively select a set of weak classifiers from these relations to perform faster target detection.

RelCom is significantly different from the body of work developed around n-tuples, as we explicitly use logical operators with a learned similarity thresholds as opposed to raw intensity (or gradient) values. Unlike the sparse features and associated pairings, it extends the combinations of the low-level attributes to multiples of operands to gain better object structure imposition on the classifier. Instead of mining of compositional features [17], which can split the feature space only along the dimensions as k-trees, RelCom partitions the space into margin regions along the hyperplanes and constructs higher level hypotheses, thus, it can provide much better granularity using the same number of primitive classification rules.

3.2 RelCom features

Consider a dataset $\mathscr{D}_N = \{\mathbf{x}_t, c_t\}_{t=1}^N$ with N training samples where each sample is characterized by its feature vector $\mathbf{x}_t \in \mathbb{R}^d$ and has an associated binary class label $c_t \in \{-1, 1\}$. The traditional classification problem is to find a classifier function $\mathbf{g}(.) : \mathbf{x} \to c$ that provides a mapping between the feature space and class labels. $\mathbf{g}(.)$ is usually determined by minimizing classification error over a representative training set. Instead of a direct mapping from the feature space to class labels, we define a binary valued propositional feature space $\{\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_K\}$ where each $\mathbf{f}_k : \mathbf{x} \to \{0, 1\}$. In effect this is a transformation from the continuous valued scalar space to a binary valued space and possibly a reduction in dimension if $K < d$. The mapping function $\mathbf{f}_k$ can take on a multitude of forms such as a simple decision stump in a single dimension, a multi-dimensional hyperplane, a threshold based match filter etc. For any given classification problem there are a plethora of possible feature representations of the objects involved. Therefore, the choice of $\mathbf{f}_k$ will be dependent on the semantic meaning of the features $\mathbf{x}$ and the problem at hand.

Once we have obtained the K-bit binary string $\mathbf{F} = \{\mathbf{f_1}, \mathbf{f_2}, \cdots, \mathbf{f_K}\}$ by choosing an appropriate mapping function, it is easy to see that there are 2^{2^K} possible ways to assign binary class labels to any given test sample $\mathbf{x}$. An example for the case of $K = 3$ is shown in Tab.3.1 where the left column represents all possible binary string patterns and each hypothesis column $h_i(\mathbf{F})$ on the right represents one possible class label assignment pattern. Though the number of possible hypothesis increases greatly with K we have found in our experiments $K = 2, 3$ was adequate to meet the detection challenge. The value of h_i indicate whether a sample is classified as positive (1) or negative (-1) for a given propositional binary pattern.

To illustrate further the concept of combinatorial features consider a d dimensional feature descriptor $\mathbf{x} = [\mathbf{x}(1) \;\; \mathbf{x}(2) \cdots \; \mathbf{x}(d)]^T$ and an associated 3-bit propositional mapping $\{\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3\}$ using simple decision stumps. Fig.3.2 represents a hypothetical projection of these decision stumps along the first two dimensions of the feature space. We observe that

$\mathbf{f}_1$	$\mathbf{f}_2$	$\mathbf{f}_3$	$h_1(\mathbf{F})$	$h_2(\mathbf{F})$	$h_3(\mathbf{F})$	$\cdots$	$h_i(\mathbf{F})$	$\cdots$	$h_{256}(\mathbf{F})$
0	0	0	-1	-1	-1	$\cdots$	1	$\cdots$	1
0	0	1	-1	-1	-1	$\cdots$	1	$\cdots$	1
0	1	0	-1	-1	-1	$\cdots$	1	$\cdots$	1
0	1	1	-1	-1	-1	$\cdots$	-1	$\cdots$	1
1	0	0	-1	-1	-1	$\cdots$	-1	$\cdots$	1
1	0	1	-1	-1	-1	$\cdots$	1	$\cdots$	1
1	1	0	-1	-1	1	$\cdots$	-1	$\cdots$	1
1	1	1	-1	1	-1	$\cdots$	1	$\cdots$	1

Table 3.1: Illustration of the 2^{2^K} possible class label assignments for a propositional binary string of length $K = 3$.

the entire feature space has been divided into a small number of discrete region each with a binary string label. Fig.3.3 represents the decision boundaries corresponding to a few possible hypothesis from Tab.3.1 where data samples falling within the shaded region are classified as positives. It is observed that simple logical boundaries in the propositional form complex decision boundaries in the original feature space. This enables us to define complex decision boundaries by combining the results of individual simple decision stumps in a multitude of combinations. From Fig.3.3 it is easy perceive that the decision regions resulting from the combinations are more likely to be beneficial in classification problems than those of any individual decision stumps. However, the regions of the individual decision stumps are a subset of the larger set of all possible combinations. Though combinational features allow for complex decision boundaries we still consider each of these to be a weak classifier and perform boosting to select an informative subset from these combinations.

Some of the hypotheses in Tab.3.1 are degenerate and are logically invalid such as the first and last columns. Half of the remaining are complements of a different column and

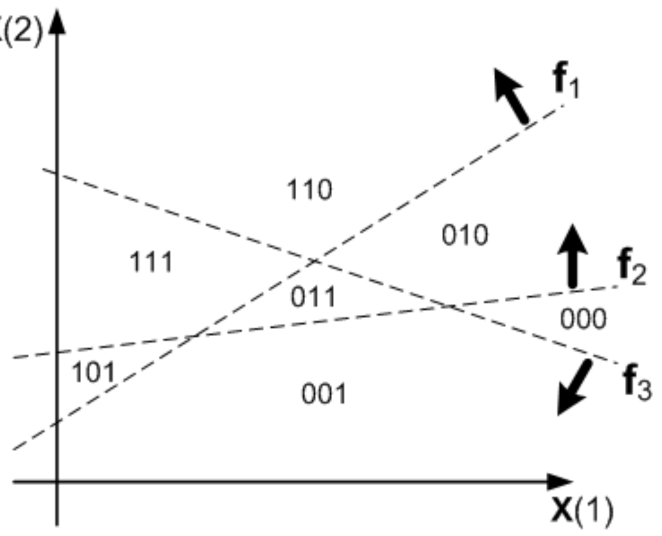


Figure 3.2: Example of one possible mapping from feature space to the propositional space spanned by a 3-bit binary string. The dotted lines represent three simple decision stumps. Data points that lie on the positive normal side (represented by dark arrows) of a decision stump map to a binary 1 in the propositional space.

need not be evaluated explicitly. Based on the definition of $\mathbf{f}_k$ it is possible that some of the patterns in the left column never occur and this further reduces the number hypothesis to be evaluated. An example of this is seen in Fig.3.2 where the string 100 is not a possibility. Thus, when we search within the hypotheses it is not necessary to evaluate all of 2^{2^K} possibilities.

Fast target detection invariably requires the computational load imposed by features and the propositional mapping to be minimized. In this chapter we primarily consider the simplest possible feature - raw image pixel values. The feature vector $\mathbf{x}$ is taken to be a raster scan of the pixel values making its dimension d equal to the number of the pixels in the target window. Experiments with other feature descriptors computed in a target window, e.g. HOG feature are also considered.

Inspired by the n-tuple classifier and other recent works [79, 80] that capture pairwise feature variations in a small subset of the entire feature space, we define our propositional mapping function to be simple a margin based similarity rule that operates on two feature

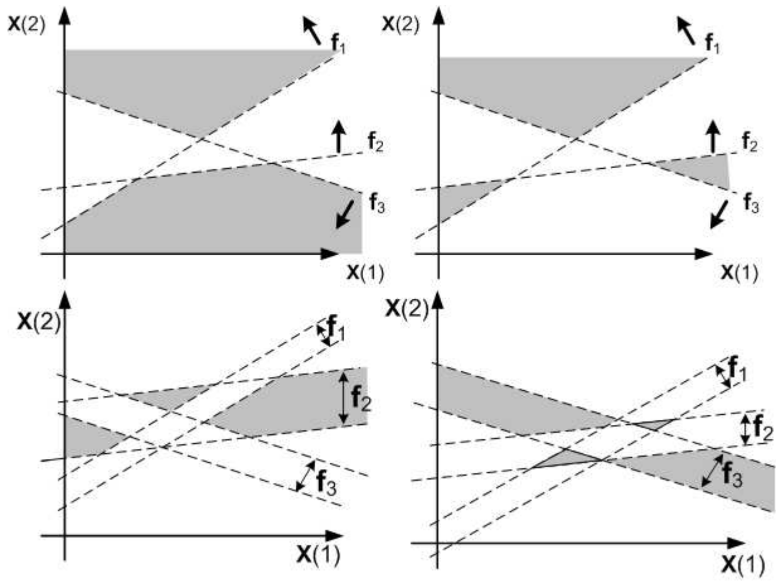


Figure 3.3: Illustration of a few possible complex decision boundaries using combinatorial features. Data points that lie in the shaded regions are classified as positives. Propositional mapping using Top: simple decision stumps and Bottom: margin based similarity rule.

dimensions chosen from a set of n randomly sampled feature dimensions. For a given d dimensional feature vector $\mathbf{x}$, we randomly select $n,\ (n < d)$, of the possible dimensions and represent it by $P_n = \{p_1, p_2, \cdots, p_n\}$ where each p_k is unique and $p_k \in \{1, 2, \cdots, d\}$. Given an arbitrary n-tuple P_n, for each unique pair (p_i, p_j), $p_i, p_j \in P_n$ we can define a propositional mapping $\mathbf{f}_k$ of the form

$$\mathbf{f}_k(\mathbf{x}) = \begin{cases} 1 & |\mathbf{x}(p_i) - \mathbf{x}(p_j)| \leq \tau_k \\ 0 & otherwise, \end{cases} \tag{3.1}$$

where $\mathbf{x}(p_i)$ represents the value of the feature along the p_i'th dimension. The margin value τ_k indicates the acceptable level of variation and it can be chosen so as to maximize the classification performance of a particular hypotheses if prior knowledge of the feature space is available. Given an n-tuple and the definition in Eq.3.1 the number of unique propositional mappings $\mathbf{f}_k,\ k \in \{1, \cdots, K\}$ that can be defined is limited to $K = \binom{n}{2}$ corresponding to the number of possible unique pairs (p_i, p_j). We denote the resulting binary string by $\mathbf{F}(\mathbf{x}) = \{\mathbf{f_1}, \cdots, \mathbf{f_K}\}$.

When the propositional mapping in Eq.3.1 is applied to raw image pixel values with n-tuples, we are effectively analyzing the intensity variation patterns over the image n pixels at a time. This draws attention to the extremely large number of n-tuples that can be selected for any given image vector of dimension d, $T_n = d!/(d-n)!$. In this work we mostly deal with the cases of $n = 2, 3$ which we refer to as 'doublet' and 'triplet' respectively. For the case of even a 10×10 template and triplets there exists ≈970k unique choices of triplets. A second point of concern is the selection of K different threshold τ_k, a continuous variable, that is difficult to optimize without prior knowledge of the feature space. The vastness of this parameter space of n-tuples and thresholds makes determining optimal values for either of them impossible. However, each n-tuple, threshold pair can be thought of as a weak classifier and these can be combined by boosting to produce a strong classifier. Since we explore different sparse combinations of the original feature space using relational operators we refer to our feature as 'RelCom'.

3.2.1 Boosting

To select the most discriminative RelCom features from a large pool of candidates we use the discrete AdaBoost algorithm. Since the output of each RelCom hypothesis is binary it can easily be adapted into the discrete AdaBoost framework. AdaBoost works iteratively to combine a number of weak classifiers linearly to produce a strong classifier with acceptable classification performance. In each iteration a single weak classifier is selected that minimizes the weighted error over the training set. The weights of the misclassified samples are increased (and the weights of each correctly classified example are decreased), so that in the next iteration the new weak classifier focuses more on the misclassified examples. It has been shown [81] that for a binary classification problem the error of the final hypothesis decreases exponentially with the number of boosting rounds (i.e additional weak classifier).

Our adaptation of the Discrete AdaBoost for RelCom features (shown in Fig. 3.4) is similar to original AdaBoost, except differences at the level of weak learners. In this case, domain of the weak learners is in the combinatorial n-tuple hypotheses and threshold space. In each iteration random samples of a set of n-tuples P_n^s and associated thresholds $\mathscr{T}^s$ are drawn for more efficient spanning of the enormous search space. These define the mapping from the input feature space $\mathbf{x}_t$ to the propositional space $\mathbf{F}_s^t(\mathbf{x}_t)$. Next we identify the n-tuple pattern, associated threshold and the hypothesis pattern that minimizes the weighted error on the training set and update the training sample weights. The identified patterns, threshold and hypothesis are added to the classifier pool with associated weight α_i. Once the classifier is trained, given a test feature vector, we can identify the propositional mapping from a lookup table. The hypothesis corresponding to the test pattern is pre-stored in a second lookup table and also requires no computation. The output of the strong classifier is the sign of the sum of the weighted RelCom feature responses as shown in Fig. 3.5.

In order to illustrate the competence of the RelCom features with boosting we demonstrate the proposed method on two infrared datasets 1)CSUAV (civilian vehicles from aerial view) and 2) SENSIAC (military targets from planar view) using raw pixel values as the feature vector. Figure 3.6 shows the position of the top 10 RelCom features for the case of triplets. Interestingly, the features are distributed on the target and background to gather clues about the target shape. Figure 3.7 presents the map of the weighted locations of the RelCom features on the SENSIAC dataset when the number of weak learners is varied. As visible, the target edges are are found to be more discriminative. Increasing the number of the features helps to concentrate attention on the salient regions of the target especially in the night time images. For the day time images the effect is less pronounced due to the presence of considerable clutter.

3.2.2 Computational load

Note that, the relational operator that we use to map from the feature space to propositional space has a very simple margin based distance form. Therefore, it is possible to construct a 2D lookup table of the responses for n-tuples and then combine them into separate hypotheses lookup tables. This can be achieved without any loss of information for the intensity features, and an insignificant adaptive quantization loss for other low-level features. Particularly, this lets us masterfully trade in the computational load with the memory imprint of the algorithm, which itself is relatively small (as many 100×100 or 256×256 binary tables as the number of features). In case of 500 triplets, the memory for the lookup tables is approximately 100MB. After obtaining the propositional binary string a secondary lookup table of the hypothesis is used to identify the class label. We can then multiply these binary values with their corresponding weak classifiers' weights and aggregate the sum to determine the response. In other words, in the testing stage of the algorithm we will only employ array access operations instead of any arithmetic operations, which results in a very fast detector. Due to vector multiplications, neither SVM radial basis functions nor linear kernels can be implemented in such a manner.

Further, in the context of boosted classifiers it is possible to implement a rejection cascade that significantly reduces the computational load in scanning window based detection. As an example, for Haar wavelet based face detection the classifier becomes 750$\times$ faster (reported in [18]) by decreasing the effective number of features to be tested from 6000 (of the original boosted strong classifier) to a mere 8 on average! In other words, a cascaded implementation of the boosted RelCom has every potential to further speed up the detection.

We present the computational load and the performance of several classifiers including the boosted RelCom doublet and triplet versions in Table 3.2. As shown, RelCom boosting provides one of the fastest classifiers whose complexity only depends on the number of weak classifiers even without a cascade implementation. When compared to SVM-RBF, it would demand only a fraction of the load ($\sim$600$\times$ speed up for the INRIA dataset) and at the same time outperforms the SVM-RBF.

Algortihm	Computational complexity	INRIA	
		$\sim$ Operations	% of FA
SVM-Linear	$10d$	21,000	4.58
SVM-RBF	$57dN_{sv} + 10N_{sv}$	90,600,000	0.38
RelCom Doublet	$11N_c$	16,500	0.22
RelCom Triplet - 1	$15N_c$	**7,500**	0.23
RelCom Triplet - 2	$15N_c$	150,000	**0.02**

Table 3.2: Computational complexity and performance of different algorithms given the input vector of dimension d, the number of learnt support vectors N_{sv}, the number of weak classifiers N_c, the number of principal components N_{pc}. The relative costs of processor operations are measures against the cost of memory access, which is taken to be unity. The above expressions assume the cost of a addition to be 3, multiplication to be 5, exponential to be 35. For INRIA dataset using 64×32 intensity images, $d = 2048$, $N_{sv} = 776$. We set $N_c = 1500, 500, 10k$ weak learners for the RelCom doublet, triplet-1 and triplet-2 respectively.

3.3 Experimental results

To demonstrate the capability of the proposed RelCom features we perform detection on three different datasets. 1) INRIA Person dataset - human detection in visual images, 2) SENSIAC ATR dataset - military vehicle detection in midwave infrared images and 3) CSUAV dataset - car detection in midwave infrared images taken from an UAV. We compared the performance of three different algorithms including SVM-Linear, SVM-RBF and RelCom triplets. The SVM-RBF parameters were set to maximize cross validation accuracy on the training set. LibSVM toolbox was used for training and testing. The basis of comparison are the ROC curves that plot the probability of true detections *vs* probability of false alarms and visual detection results.

For the standard INRIA dataset we obtained 2416 pictures of mirrored and centered images and a further 12180 samples of random backgrounds. Of these only a fifth of the positive samples and a tenth of the negative samples were used for training. For testing purposes 24360 random background images and 1126 positives we used. All images were of size $32{\times}64$. Here we test algorithm performance for two different feature type: greyscale intensity values and HOG features. For HOG feature calculations, we used [-1 1] filter in orthogonal directions and adapted integral histogram for fast evaluation. HOG features were computed for 8 directions in non-overlapping blocks of size 16×8 resulting in a $8 \times 4 \times 4 = 128$ dimensional feature vector.

Figure 3.8(a) shows the detection performance curves for INRIA dataset when using intensity features. The boosted RelCom triplets with 10k classifiers significantly outperforms SVM-RBF to our surprise by a factor of 13.8 at the 50% true detection level. At the same time it outperforms SVM-Linear by almost a factor of 25. Figure 3.8(b) presents the ROC curves in the case the HOG feature. Performance of the RelCom is at par with the SVM-RBF and 12 times better than the SVM-Linear. We are able to achieve performance comparable to SVM-RBF at significantly lower computational cost. This illustrates that the proposed method is applicable to any given feature and not limited to spatial features.

The SENSIAC dataset consists of midwave image sequences acquired both during day time and night time for eight different targets. The targets are imaged at multiple distances and poses. We select 3 of those targets (Pickup, BTR70 and BRDM2) at a distance of 2000 meters to create a training set of 60 positive samples and 5900 negative samples each for both day and night time images. The testing set consisted of 200 positive samples and 45800 negative images each for day and night time images. The images were histogram equalized before sample templates of size 15×45 were extracted. We train two separate classifiers for day and night time data on greylevel intensity features. The ROC performance curves are shown in Fig.3.9 for both day and night time detection. We observe that for the night time data the performance all of the algorithms is very similar as the target is distinctly visible. The advantage of the RelCom features is greatly emphasized in the day time images where there is a lot of confusing clutter. In addition results of detection in three different scenarios are shown in Fig.3.10, Fig.3.11 and Fig.3.12. Note that in each scenario either the target or the imaging distance is previously unseen (not present in training set). We observe that the RelCom based classifier is able to clearly identify the target even when the other methods fail entirely or result in excessive false alarms.

The CSUAV dataset contains MWIR images acquired from an UAV flying over a civilian locality. From this dataset we selected 1050 positive images and 13000 negative images for training. For testing purposes 1050 positives and 65000 negatives were used. The template size was 20×30. Here again it was found that the RelCom triplet greatly outperforms the SVMs in detection performance. Fig 3.13 shows the result of using RelCom triplets in three different scenarios. Since we trained the classifier with a general template irrespective of orientation, during the detection phase the image was scanned for targets at orientations of $0^o, 45^o$ and 90^o to detect targets oriented along different directions. The results were combined using non-maximum suppression. We can see that majority of the vehicles including those in shade and near trees where correctly detected even though some roof tops were falsely detected. These experiments establish the competence of the RelCom detector for small target detection using intensity features in infrared images.

In conclusion we have shown that high-level combinations of basic relational features can be used in a boosting framework to construct very fast classifiers that are as competitive as SVM-RBF while requiring only a fraction of the computational load. To summarize the advantages of our method:

- RelCom can speed up detection several orders of magnitude because it does not require any complex computations thanks to the two-layer lookup tables.
- It can accommodate both basic features including pixel intensities and other complex descriptors vector computed for the object window.
- It utilizes simple relational operators to capture the spatial structure within the object window effectively.
- It can be applied to very small object windows unlike HOG features.

Given:

$*$ Training dataset with feature vectors, class labels $\mathscr{D} : \{(\mathbf{x}_1, c_1), \cdots, (\mathbf{x}_N, c_N)\}$, where $c_t = \pm 1$ indicates the class label.

$*$ N_c the required number of weak classifiers.

$*$ S weak classifiers pool size.

Initialize:

$*$ Sample weights $W_1(t) = \frac{1}{2N^+}, \frac{1}{2N^-}$ for $c_t = 1, -1$ respectively, where N^+ and N^- are the number of positive and negative samples.

AdaBoost:

$*$ For $i = 1, \cdots, N_c$

- Randomly sample S n-tuples $P_n^1, \cdots, P_n^S$. For each P_n^s also sample threshold values $\mathscr{T}^s = \{\tau_k^s\}$, $k = \{1, \cdots, \binom{n}{2}\}$.
- For each of the S n-tuples compute the propositional mapping $\mathbf{F}_1^t, \mathbf{F}_2^t, \cdots, \mathbf{F}_S^t$ over the training samples $t = \{1, \cdots, N\}$ using Eq.3.1.
- For each of the S n-tuples compute error for all valid hypothesis in the set $h_j(\mathbf{F})$ $j = \{1, \cdots, 2^{2^K}\}$ as $\lambda_s^j = \sum_{t=1}^N W_i(t)[c_t \neq h_j(\mathbf{F}_s^t)]$.
- Set $P_{sel,n}^i = P_n^{smin}$, $\mathscr{T}_{sel}^i = \mathscr{T}^{smin}$, $h_{sel}^i(\mathbf{F})$=$h_{jmin}(\mathbf{F})$ and $\epsilon_i = \lambda_{smin}^{jmin}$. Where $smin$ and $jmin$ are indices : $\lambda_{smin}^{jmin} < \lambda_s^j \; \forall s \neq smin, j \neq jmin$.
- Calculate $\alpha_i = \frac{1}{2} \cdot \ln \left[\frac{1-\epsilon_i}{\epsilon_i}\right]$.
- Update the sample weights for $t = \{1, \cdots, N\}$, $W_{i+1}(t) = W_i(t) \exp[-\alpha_i c_t h_{sel}^i(\mathbf{F}_{smin}^t)]$.
- Normalize the weights $\sum_{t=1}^N W_{i+1}(t) = 1$.

Output: Selected n-tuples $P_{sel,n}^i$, threshold $\mathscr{T}_{sel}^i$, hypothesis h_{sel}^i and classifier weight α_i for $i = \{1, \cdots, N_c\}$.

Figure 3.4: Training RelCom features with discrete AdaBoost.

Given:

$*$ A single sample feature vector $\mathbf{x}_{test}$

$*$ RelCom classifier consisting $P^i_{sel,n}$, $\mathscr{T}^i_{sel}$, h^i_{sel} and α_i for $i = \{1, \cdots, N_c\}$

Testing:

$*$ Identify $\mathbf{F}_1$,$\mathbf{F}_2, \cdots$,$\mathbf{F}_{N_c}$ using $\mathbf{x}_{test}$,$P^i_{sel,n}$ and $\mathscr{T}^i_{sel}$.

$*$ final classifier $H(\mathbf{x}_{test})$ =sign $[\sum_{i=1}^{N_c} \alpha_i h^i_{sel}(\mathbf{F}_i)]$.

Figure 3.5: Testing with RelCom features.

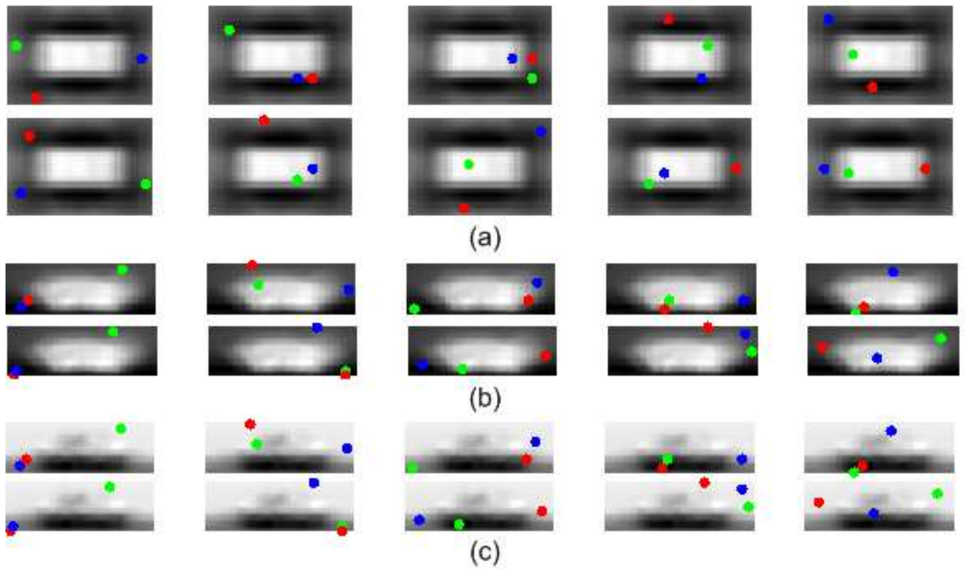


Figure 3.6: Top ten RelCom triplet feature locations shown on the mean positive images for (a) CSUAV dataset, (b) SENSIAC night time and (c) SENSIAC day time. The features identified compare the object with its background aiming to distinguish the silhouette.

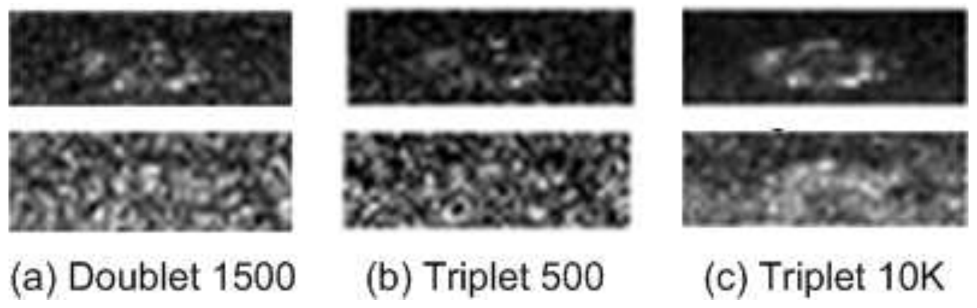
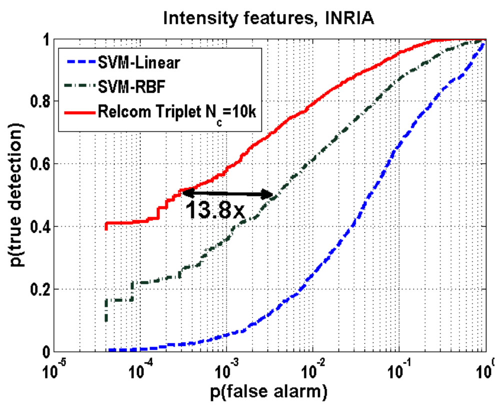
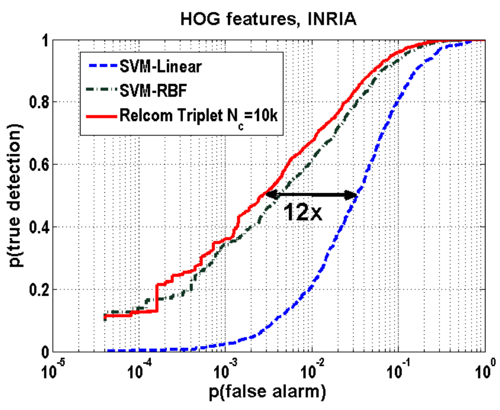


Figure 3.7: Map of normalized weighted location of RelCom features for the SENSIAC dataset. Top: Night time and Bottom: Day time. The target edges, as expected, are found to be more salient by RelCom features.

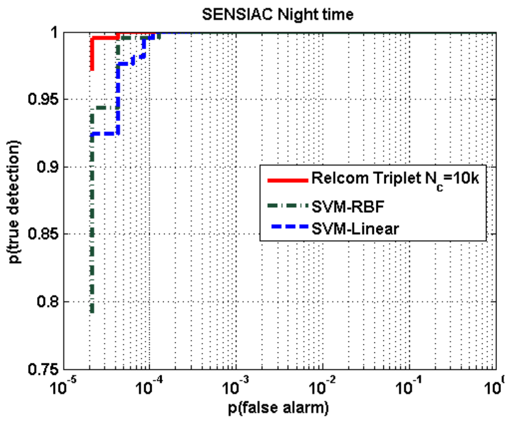


(a)

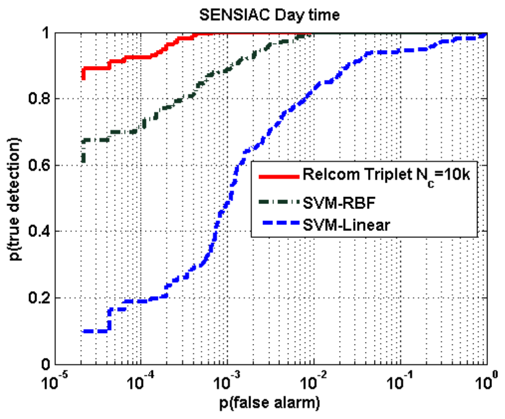


(b)

Figure 3.8: (a) Detection-error trade off curves for INRIA dataset. As visible, RelCom outperforms SVM-RBF, (b) Detection-error trade off curves for INRIA dataset using HOG features.

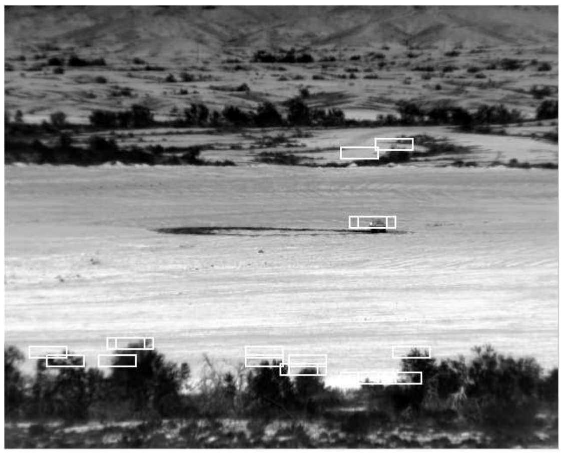


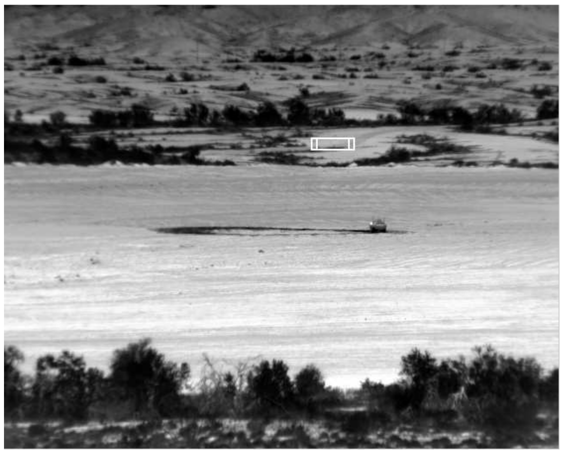
(a)



(b)

Figure 3.9: ROC curves for SENSIAC dataset (a) Night time (b) Day time.


(a)


(b)

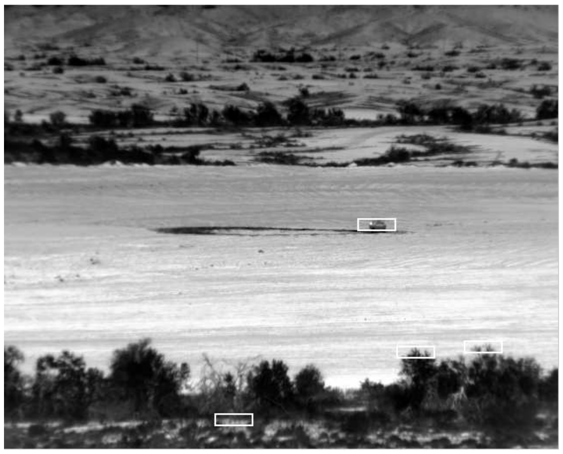
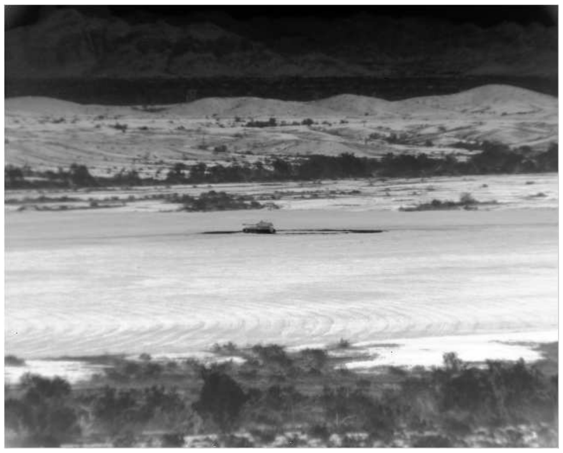
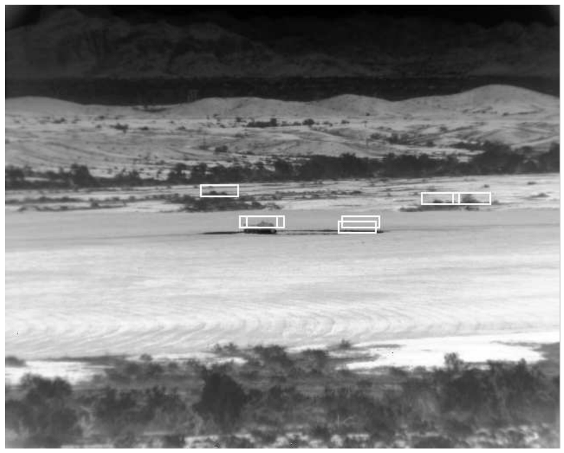

(c)

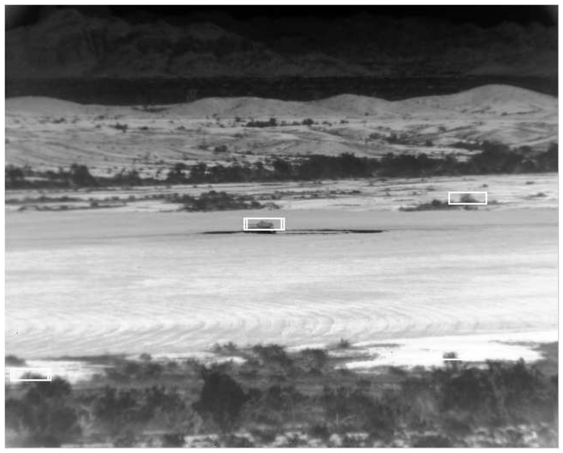
Figure 3.10: Sample detection results at FA rate of 10^{-4} on Day time SENSIAC data at Distance:1500m (unseen) and Target: BRDM2 (seen) for (a) SVM-Linear (b) SVM-RBF and (c) RelCom triplet.



(a)

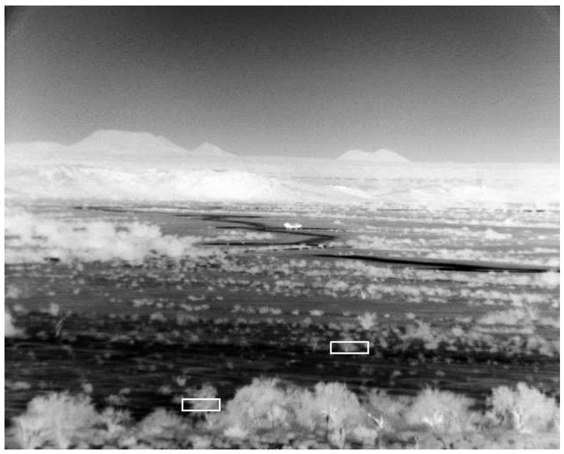


(b)

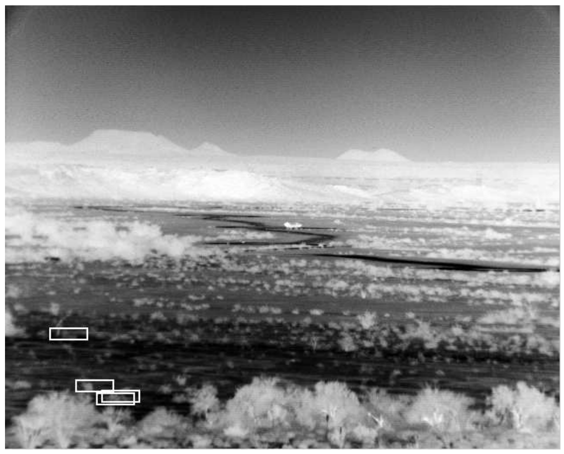


(c)

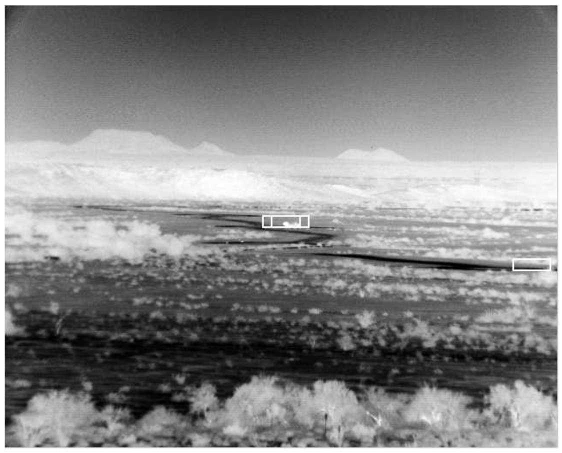
Figure 3.11: Sample detection results at FA rate of 10^{-4} on Day time SENSIAC data at Distance:2000m (seen) and Target: ZSU23 (unseen) for (a) SVM-Linear (b) SVM-RBF and (c) RelCom triplet.



(a)

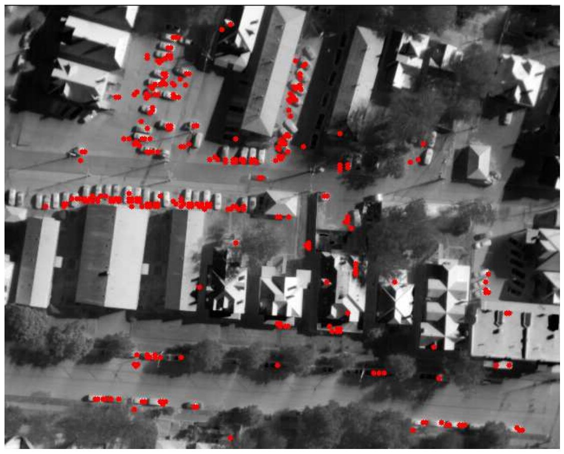


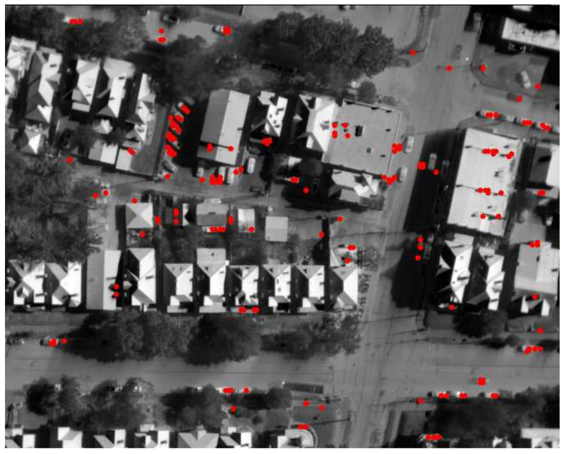
(b)



(c)

Figure 3.12: Sample detection results at FA rate of 10^{-4} on Night time SENSIAC data at Distance:4000m (unseen) and Target: BMP2 (unseen) for (a) SVM-Linear (b) SVM-RBF and (c) RelCom triplet.


(a)


(b)

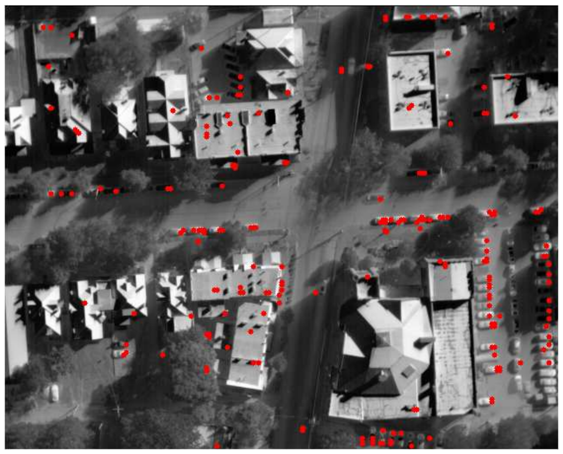

(c)

Figure 3.13: Sample detection results of RelCom triplet at FA rate of 10^{-4} on three different scenarios from the CSUAV dataset.

CHAPTER 4

Target Tracking with Online Feature Selection

4.1 Background

Target tracking in the forward looking infrared (FLIR) imagery has remained a challenging problem in the field of computer vision. Usually, FLIR images are characterized by low signal-to-noise (SNR) ratios, poor target visibility, and time varying target signatures. These factors prohibit the straightforward extension of existing optical tracking algorithms to FLIR images. In addition strong ego motion of the FLIR sensor makes it difficult to characterize the target's kinematics. Two issues will be discussed in this chapter. One is how to develop an elaborate target appearance model for superior tracking performance? How to select optimal features to characterize the appearance model? Despite the fact that the issue of size estimation is especially important in automatic missile guidance systems where the size may act as a cue for distance, it is usually not considered in most FLIR tracking algorithms [27], partially because of the lack of robust and reliable target appearance model. For example, to achieve a robust position localization, traditional tracking algorithms assign less confidence to the target's boundary pixels compared with the center ones [26, 27]. However, this biased confidence assignment leads to inaccurate size estimation.

In this chapter, we propose *a dual appearance model* that accounts for both foreground and background appearances. Unlike the classifier [82, 83] approach, we track the target by matching both its foreground and background statistics with their corresponding reference model. These statistics are used as features for appearance modeling. As the target appearance changes with time, the contribution of each feature towards tracking will vary. It is desired to provide a feature selection scheme that can adaptively assign weights to different features indicating their relative importance to the tracking process. We formulate the online weights update as a probabilistic estimation problem that is directly integrated with the tracker. In addition to accurate target localization and size estimation, the proposed algorithm also indicates the optimal feature combination for target representation while tracking.

4.2 Probabilistic formulation

In this section, we devise a probabilistic framework to formulate the research problem. Let $\mathbf{x}_k$ denote the state vector to be estimated that includes the position and size at time instant k. In addition, our tracking approach is able to update weights $\mathbf{v}_k$ of different features associated with the appearance model in an online fashion. Therefore, target tracking and feature selection are formulated as a state space problem where we need to estimate posterior densities $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ and $p(\mathbf{v}_k|\mathbf{y}_{1:k})$ given the observations $\mathbf{y}_{1:k}$.

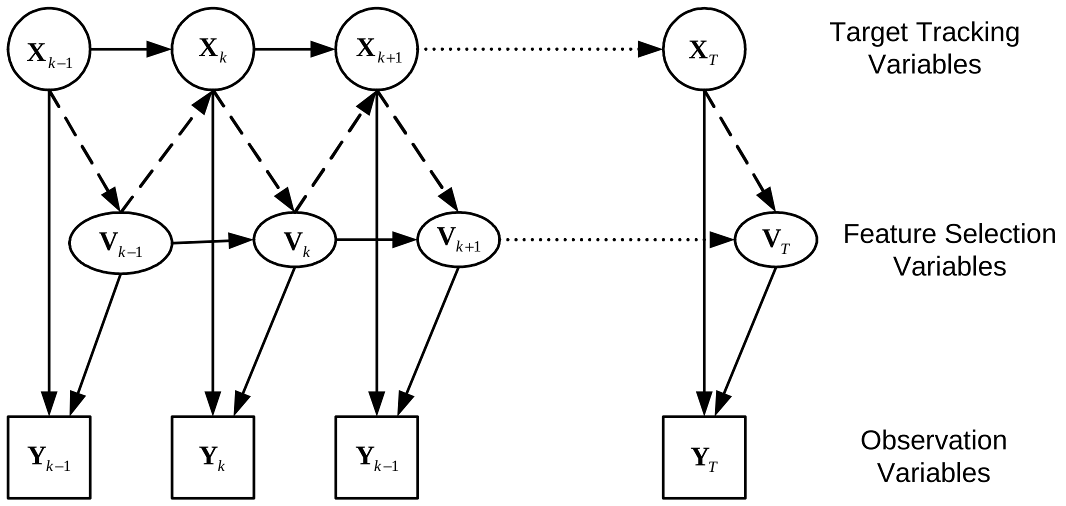


Figure 4.1: The diagram of the proposed tracking algorithm.

The conditional dependencies between the variables are graphically depicted in Fig. 4.1. Noticing these dependencies, the estimation can be obtained by recursive Bayesian filters. For $p(\mathbf{x}_k|\mathbf{y}_{1:k})$, we have

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) \propto \int_{\mathbf{v}_{k-1}} p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{v}_{k-1})p(\mathbf{v}_{k-1}|\mathbf{y}_{1:k-1})\mathrm{d}\mathbf{v}_{k-1} \cdot \int_{\mathbf{x}_{k-1}} p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})\mathrm{d}\mathbf{x}_{k-1}. \tag{4.1}$$

We define the kinematic model of the target, $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ in Sec 4.3 and the weighted likelihood $p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{v}_{k-1})$ in Sec 4.4. Given $p(\mathbf{x}_k|\mathbf{y}_{1:k})$, the posterior density of the feature weights $p(\mathbf{v}_k|\mathbf{y}_{1:k})$ can be represented by:

$$p(\mathbf{v}_k|\mathbf{y}_{1:k}) = \int_{\mathbf{x}_k} p(\mathbf{v}_k|\mathbf{x}_k, \mathbf{y}_{1:k})p(\mathbf{x}_k|\mathbf{y}_{1:k})\mathrm{d}\mathbf{x}_k. \tag{4.2}$$

Let

$$L(\mathbf{v}_k) = \int_{\mathbf{x}_k} p(\mathbf{x}_k|\mathbf{y}_{1:k})p(\mathbf{v}_k|\mathbf{x}_k)p(\mathbf{y}_k|\mathbf{v}_k, \mathbf{x}_k)\mathrm{d}\mathbf{x}_k, \tag{4.3}$$

where the weighted likelihood $p(\mathbf{y}_k|\mathbf{v}_k, \mathbf{x}_k)$ has the same form as $p(\mathbf{y}_k|\mathbf{v}_{k-1}, \mathbf{x}_k)$ and $p(\mathbf{v}_k|\mathbf{x}_k)$ describes how likely the feature weights fit the given tracking estimation, which will be defined in Sec 4.5. Thus (4.2) becomes

$$p(\mathbf{v}_k|\mathbf{y}_{1:k}) = L(\mathbf{v}_k) \int_{\mathbf{v}_{k-1}} p(\mathbf{v}_k|\mathbf{v}_{k-1})p(\mathbf{v}_{k-1}|\mathbf{y}_{1:k-1})\mathrm{d}\mathbf{v}_{k-1}, \tag{4.4}$$

where $p(\mathbf{v}_k|\mathbf{v}_{k-1})$ is the evolution prior of the feature weights to be discussed in Sec 4.5.

Motivated by the idea of particle filters [84], we approximate the posterior densities $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ and $p(\mathbf{v}_k|\mathbf{y}_{1:k})$ using two weighted sample sets $\{\mathbf{x}_k^j, w_k^j\}_{j=1}^{N_p}$ and $\{\mathbf{v}_k^i, w_{v,k}^i\}_{i=1}^{N_v}$. The integrals in (4.1) and (4.4) can be approximated by summations. To avoid the high computational expense brought by integrating $\mathbf{v}_{k-1}$ and $\mathbf{x}_k$ out from (4.1) and (4.3), we replace the variables $\mathbf{v}_{k-1}$ and $\mathbf{x}_k$ by their expectations $E(\mathbf{v}_{k-1})$ and $E(\mathbf{x}_k)$ estimated from the weighted sample sets. The detailed algorithm is summarized in Table 4.1.

4.3 Kinematic models

In our formulation the state vector at any time instant k is defined as $\mathbf{x}_k$=$[\mathrm{x}_k, \mathrm{s}_k]$, where x_k=$[x_k, y_k]$ contains the position information and s_k=$[s_k^x,\ s_k^y]$ shows the size in x and y

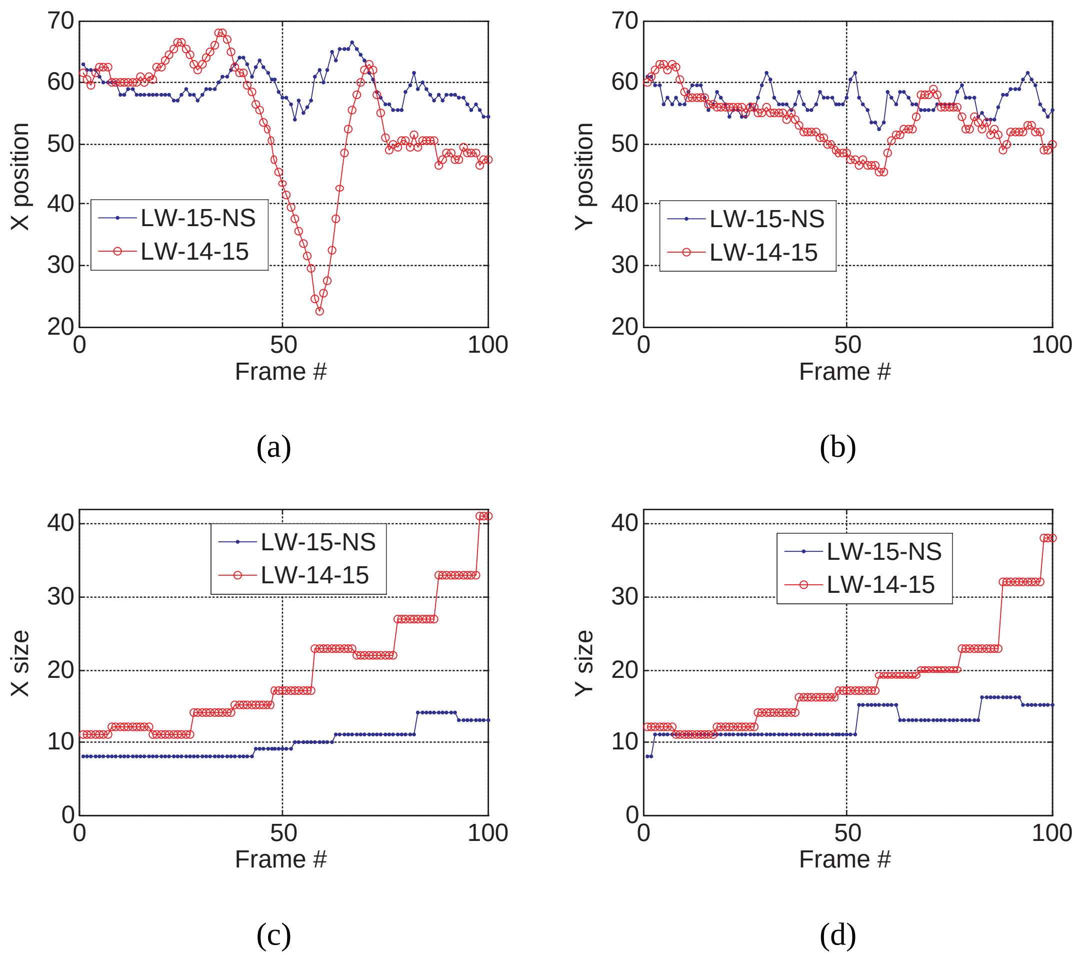


Figure 4.2: The ground truth values of X-position, Y-position, X-size and Y-size of the sequences LW-15-NS and LW-14-15 over 100 frames.

directions. The foreground area $N_F(\mathbf{x_k})$ is defined a rectangle box whose top-left corner corresponds to coordinate (x_k,y_k) and the length and width are given by s_k^x and s_k^y respectively. In the following, we will analyze the position and size dynamics of the target based on the ground truth data, and design appropriate dynamic models for both position and size variations. These dynamic models determine the state transition probabilities, i.e., $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ that play an important role in particle filtering.

4.3.1 Target position

First let us consider the dynamics of the target's position. Fig. 4.2 shows the evolution of the position and size of the sequences LW-14-15 and LW-15-NS from the AMCOM dataset[1]. From Fig. 4.2 (a) and (b) we observe that for the LW-15-NS sequence, the target stays around the center of the image. This is due to the fact that the sensor is mounted on an airborne platform which is homing in on the target. However the sequence LW-14-15 is characterized by strong ego-motion of the sensor platform. Therefore, we need a model which can account for the low variability of the position when there is no ego motion and at the same time provide more variability when there is strong ego motion. An adaptive framework which can adjust the variability of the position is apt to deal with this requirement.

We employ a first order model which can adapt to the change in the variability of the target state based on the idea in [26]. Such a model requires an estimate of the velocity of the target based on the previous n frames. The velocity at any time instant k is given by equation (4.5). In the cases when $k < n$ the estimate is made based on all available frames up to time k.

$$\mathbf{E}_n[\triangle \mathrm{x}_k] = \frac{1}{n} \sum_{l=k-n-1}^{k-1} |\mathrm{x}_l - \mathrm{x}_{l-1}|. \tag{4.5}$$

Then the state transition model for the position vector $\mathrm{x_k}$ is defined as

$$\mathrm{x}_k = \mathrm{x}_{k-1} + C_k v_k, \tag{4.6}$$

where $C_k \propto \mathbf{E_n}[\triangle \mathrm{x}_k]$ and $v_k \sim N(0, I)$. In this model if the target is moving with a low velocity then the variance of the process noise is low, thereby reducing the variability of the target state and vice versa. This increased variability physically reflects the spread of the particles over a larger area in the state space, thereby increasing the probability of locating the target whose position is affected by strong ego motion of the sensor.

[1]http://cis.jhu.edu/data.sets/AMCOM/amcom.html

4.3.2 Target size

As the next step we now consider the dynamics of the target size. From Fig. 4.2 (c) and (d) we observe that the target size has a tendency to increase in steps over time. Though the LW-15-NS sequence shows gradual increase in size, the sequence LW-14-15 is characterized by rapid size changes. In the design of a model for the size dynamics we do not favor an adaptive variance model due to size increments in single frames. Therefore for the size dynamics we employ a simple first order model with fixed variance. The state transition model for the size vector s_k is defined in (4.7),

$$\mathrm{s}_k = \mathrm{s}_{k-1} + Dw_k, \tag{4.7}$$

where D is a fixed constant, and $w_k \sim N(0, I)$. The state transition probabilities can be derived from (4.6) and (4.7).

4.4 Target appearance model

We now discuss the issue of target representation. This model describes the appearance of the target in the image in relation to the underlying states $\mathbf{x}_k$ =[x_k, y_k, s_k^x, s_k^y] and therefore defines the likelihood $p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{v}_k)$. We characterize our target using a non-parametric model based on the intensity and local standard deviation histograms of the target area and its local background. We call it the 'Dual model' since in addition to the information from the target area we also model its background.

4.4.1 Dual foreground-background model

At any given time instant k assume the intensity image $\mathbf{I}_k$ and local standard deviation image $\mathbf{S}_k$ are available. Given $\mathbf{x}_k$ we can determine foreground area $N_F(\mathbf{x}_k)$ in the image. We next define a local background area $N_B(\mathbf{x}_k)$ using the "center-surround" approach. For a target area defined by a rectangle of size $s_k^x \times s_k^y$ another rectangle of size $(2 \times s_k^x) \times (2 \times s_k^y)$ defines the extent of the local background. This is illustrated in Fig.4.3. It is a common practice to use a weighted histogram for the target's representation [26, 27]. This is due to the fact that often boundaries between the target and the background are not precise and therefore pixels from the background regions may easily corrupt the histogram of the target area. So a weighting kernel is often used to obtain a weighted histogram by assigning higher trust to pixels closer to the center of the target. The areas corresponding to the foreground, background and the placement of the kernel are illustrated in Fig.4.3.

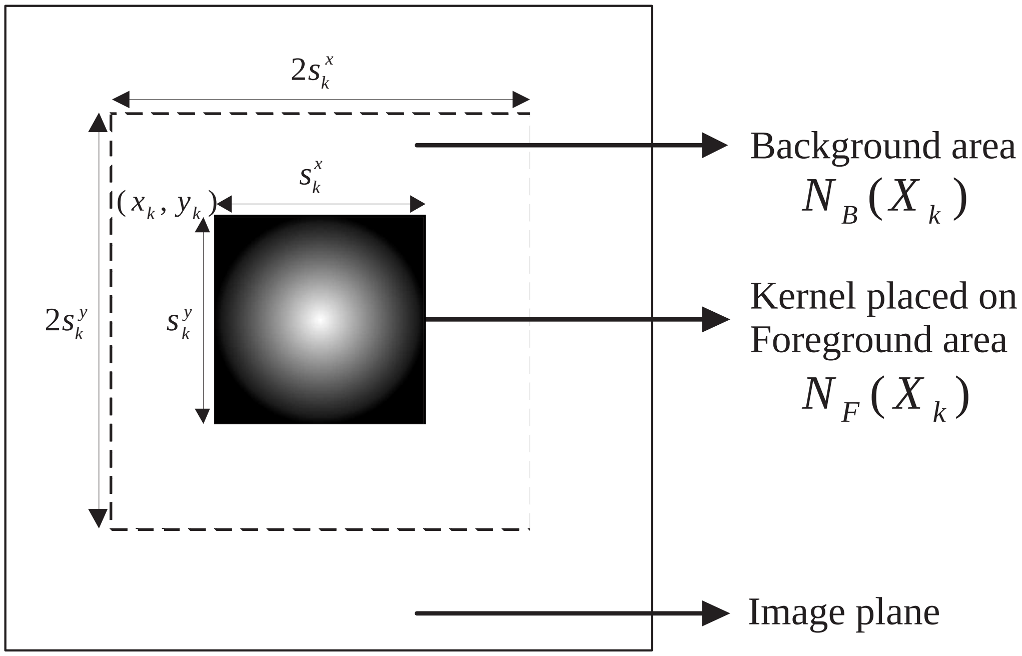


Figure 4.3: Foreground region $N_F(\mathbf{x}_k)$ with overlapped kernel and background area $N_B(\mathbf{x}_k)$ defined based on $\mathbf{x}_k = [x_k, y_k, s_k^x, s_k^y]$.

Let $\boldsymbol{p}$ denote the location of the target's centroid based on the position and size information contained in $\mathbf{x}_k$. The function $b : \mathbb{R}^2 \rightarrow \{1, \cdots, m\}$ maps the intensity value of the pixel at a position given by $\boldsymbol{r}$, to its bin index in the quantized feature space. The probability of the feature $u = 1 \cdots m$ is given by [85]

$$p_{fi}^u(\mathbf{x}_k) = \lambda_1 \sum_{\boldsymbol{r} \epsilon N_F(\mathbf{x}_k)} \mathbf{K}_{\boldsymbol{H}}(\boldsymbol{r} - \boldsymbol{p}) \delta[b(\boldsymbol{r}) - u], \tag{4.8}$$

where λ_1 is a normalization constant obtained such that $\sum_{u=1}^{m_1} p_{fi}^u(\mathbf{x}_k) = 1$ and δ is the

Kronecker delta function. We use the triangular kernel for $\mathbf{K}_{\boldsymbol{H}}$, where the width of the kernel is determined by the target size information contained in $\mathbf{x}_k$. Now we can define a m dimensional vector $f_{fi}(\mathbf{x}_k) = [p^1_{fi}(\mathbf{x}_k), \cdots, p^m_{fi}(\mathbf{x}_k)]$ called the foreground histogram. In a similar manner the background area histogram may be obtained as $f_{bi}(\mathbf{x}_k) = [p^1_{bi}(\mathbf{x}_k), \cdots, p^m_{bi}(\mathbf{x}_k)]$ where

$$p^u_{bi}(\mathbf{x}_k) = \lambda_2 \sum_{\boldsymbol{p}_i \epsilon N_B(\mathbf{x}_k)} \delta[b(\boldsymbol{p}_i) - u], \tag{4.9}$$

and λ_2 is a normalization constant obtained such that $\sum_{u=1}^{m_1} p^u_{bi}(\mathbf{x}_k) = 1$. Analogously we can obtain the foreground histogram $f_{fs}(\mathbf{x}_k)$and the background histogram $f_{bs}(\mathbf{x}_k)$ from the local standard deviation image $\mathbf{S}_k$. Therefore given $\mathbf{x}_k$, $\mathbf{I}_k$ and $\mathbf{S}_k$ the candidate region is characterized by $\mathbf{F}(\mathbf{x}_k)$ defined as follows,

$$\mathbf{F}(\mathbf{x}_k) = \{f_{fi}(\mathbf{x}_k), f_{bi}(\mathbf{x}_k), f_{fs}(\mathbf{x}_k), f_{bs}(\mathbf{x}_k)\}, \tag{4.10}$$

which is composed of four different histograms corresponding to each of: target area intensity $f_{fi}(\mathbf{x}_k)$, background intensity $f_{bi}(\mathbf{x}_k)$, target area standard deviation $f_{fs}(\mathbf{x}_k)$ and background standard deviation $f_{bs}(\mathbf{x}_k)$. The tracker then evaluates any given candidate area by comparing the similarity of the above four histograms from the known reference model. Thereby we use the information both from the foreground and background area directly in the tracking process.

4.4.2 Distance measure

During the tracking process we need to evaluate candidate areas based on their distance from a known appearance model of the target denoted by $\mathbf{F}'_k$. The reference model $\mathbf{F}'_k$ also has a structure similar to $\mathbf{F}(\mathbf{x}_k)$ and is given by $\mathbf{F}'_k = \{f'_{fi,k}, f'_{bi,k}, f'_{fs,k}, f'_{bs,k}\}$. The tracker in essence will compare candidate models $\mathbf{F}(\mathbf{x}_k)$ against $\mathbf{F}'_k$ based on the histogram intersection (HI) metric first suggested by Swain and Ballard in [86] to measure the similarity between two histograms. The HI metric between any two normalized histograms $\boldsymbol{p}$

and $\boldsymbol{q}$ with m bins each is given by (4.11)

$$\mathrm{d}(\boldsymbol{p}, \boldsymbol{q}) = \sum_{i=1}^{m} \min(\boldsymbol{p}(i), \boldsymbol{q}(i)). \tag{4.11}$$

Every candidate region $\mathbf{F}(\mathbf{x}_k)$ comprises four different histograms and therefore we can compute four HI metrics, one corresponding to each pair of histograms in the candidate and the model $\mathbf{F'}_k$. Consequently we define our distance metric $\mathrm{D}(\mathbf{F}(\mathbf{x}_k), \mathbf{F'}_k)$ as

$$\mathrm{D}(\mathbf{F}(\mathbf{x}_k), \mathbf{F'}_k; \mathbf{v}_{k-1}) = \sum_{z \in Z} v_z^{k-1} \cdot \mathrm{d}(f_z(\mathbf{x}_k), f'_{z,k}), \tag{4.12}$$

where $Z = \{fi, bi, fs, bs\}$ and v_z^{k-1} are chosen such that $\sum_{z \in Z} v_z^{k-1} = 1$. The implication of the v_z^{k-1} term is to allow for a particular histogram to be more or less dominant in the distance calculation. The values of v_z^{k-1} are adaptively selected online and this concept is discussed in Sec 4.5 on feature selection. The likelihood $p(\mathbf{y}_k | \mathbf{x}_k, \mathbf{v}_{k-1})$ is defined based on the distance measure in (4.12).

4.4.3 Model update

Since the target is continuously changing with time it is quite intuitive that the reference model $\mathbf{F'}_k$ has to be updated to account for this change, otherwise it may result in tracking error. We update our reference model using a simple strategy where past observations are forgotten with time. The reference model $\mathbf{F'}_{k+1}$ is obtained based on $\hat{\mathbf{x}}_k$ the mean estimate of the states at time step k, and the reference model $\mathbf{F'}_k$ by using (4.13) for all $z \in Z$.

$$f'_{z,k+1} = \xi_{z,k} \cdot f'_{z,k} + (1 - \xi_{z,k}) \cdot f_z(\hat{\mathbf{x}}_k), \tag{4.13}$$

where

$$\xi_{z,k} = \mathrm{d}(f_z(\hat{\mathbf{x}}_k), f'_{z,k}). \tag{4.14}$$

$\xi_{z,k}$ gives the similarity between the model and the estimated histogram. Therefore a sudden change in the target appearance warrants a more aggressive update of the model histograms whereas slower changes do not affect the reference models dramatically.

4.5 Online feature selection

There are two issues related to online feature selection for target tracking. One is how to evaluate the effectiveness of a certain linear combination of four histogram-based features, and how how to evolve the feature weights over time to accommodate the variation of the target appearance. In the following, we will discuss these two issues.

4.5.1 Feature evolution

The tracker in our model estimates the underlying states based on the distance defined in (4.12) where the weights v_z^{k-1} determines the relative importance of the feature z in the tracking process. A single set of fixed v_z^{k-1} may not be the best choice for effective tracking in every sequence. Our idea is that if these weights can be adjusted adaptively based on the sequence at hand, the tracking process will be more robust. So we propose a method in which the v_z^{k-1} are updated online in order to adapt to the changing circumstances.

We formulate our feature selection in a state space form, where at any time k the state $\mathbf{v}_k$ containing the individual feature importance is defined as

$$\mathbf{v}_k = [v_{fi}^k, v_{bi}^k, v_{fs}^k, v_{bs}^k], \tag{4.15}$$

and is subject to the constraint $\sum_{z\in Z} v_z^k = 1$. The constraint implies that we have only three independent variables. Therefore it will be easier to decompose $\mathbf{v}_k$ into three independent components and then develop a dynamic model for the new transformed variables.

We define a new vector $\mathbf{\Gamma}_k = [\alpha_k, \beta_k, \gamma_k]$ such that the individual elements of $\mathbf{v}_k$ can may be expressed as

$$\begin{aligned} v_{fi}^k &= \alpha_k \beta_k, \\ v_{bi}^k &= (1-\alpha_k)\gamma_k, \\ v_{fs}^k &= \alpha_k(1-\beta_k), \\ v_{bs}^k &= (1-\alpha_k)(1-\gamma_k). \end{aligned} \tag{4.16}$$

Therefore $\mathbf{\Gamma}_k$ uniquely determines $\mathbf{v}_k$ and is just constrained by the fact $0 \leq \alpha_k, \beta_k, \gamma_k \leq 1$. We then model the dynamics of each component of $\mathbf{\Gamma}_k$ as a first order Markov chain with a common predefined step size δ and equal probability of transition. The dynamics of α_k is given by

$$\alpha_{k+1} = \alpha_k + \epsilon, \tag{4.17}$$

where $\epsilon \in \{-\delta, 0, \delta\}$ with equal probability. The dynamics of the other parameters (β and γ) can be obtained in a similar way and together they determine the transitional probabilities of the feature weights $p(\mathbf{v}_k|\mathbf{v}_{k-1})$ that is used to generate possible feature hypotheses for next time step.

4.5.2 Feature evaluation

Our feature selection is based on the concept that a good feature will result in a higher *confidence* for the state estimation. Confidence may be described as the measure of how peaky (small variance) the posterior density is at the ground truth state, or for a given state estimation that is assumed to be accurate. Consider the example in Fig. 4.4, the estimate of the state x is given by the solid line and it has a mean around 0. Now we need to select features that would best estimate x. The feature that maximizes the belief of the current state estimation is considered to be the best feature. Based on this idea, all feature hypotheses i.e. different linear combinations of the four histograms, are ranked based on the Mahalanobis distance between the posterior estimate produced by that feature and the one from the tracking result.

Let $\{\mathbf{x}_k^j, w_k^j\}$, $j = 1 \cdots N_p$ represent the state samples and corresponding weights at time k and let $w_{v,k}^i(j)$ represent the weights for the same set of samples $\mathbf{x}_k^j$ when evaluated with feature i, $i = 1 \cdots N_v$. Assume all the weights have been normalized. First we compute the weighted mean of the states as $\hat{\mathbf{x}}_k$=$\sum_{j=1}^{N_p} w_k^j \mathbf{x}_k^j$. Then we compute the covariance

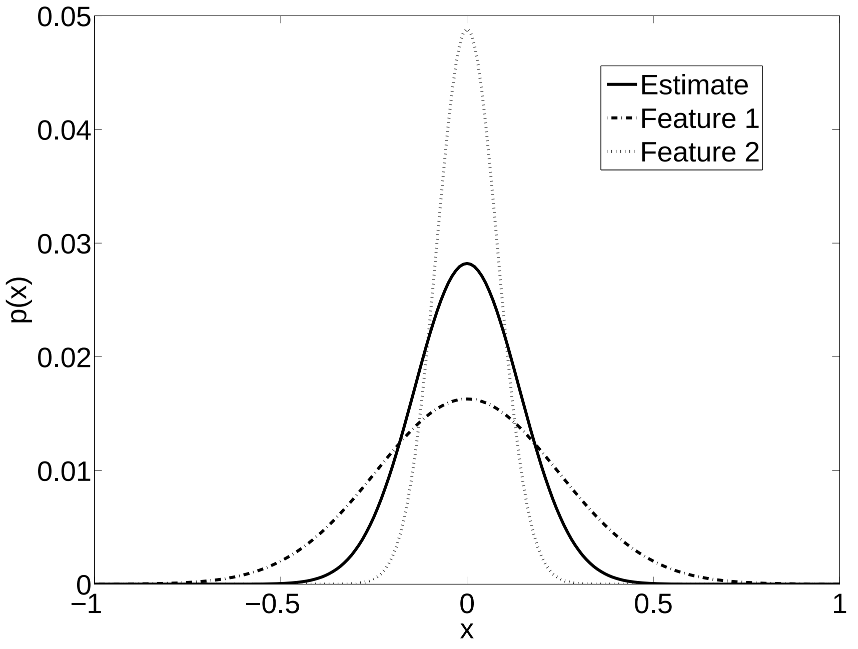


Figure 4.4: The solid line indicates the current belief of x. The other lines correspond to the state estimate using two different features. Feature 2 is considered better than Feature 1 since it shows more confidence on the given state estimation.

matrix associated with weighted samples as

$$\mathbf{\Sigma}_k = \frac{\sum_{j=1}^{N_p} w_k^j (\mathbf{x}_k^j - \hat{\mathbf{x}}_k)(\mathbf{x}_k^j - \hat{\mathbf{x}}_k)'}{1 - \sum_{j=1}^{N_p} (w_k^j)^2}. \tag{4.18}$$

Now to evaluate each feature we use the information in the weights $w_{v,k}^i(j)$. The likelihood of the ith feature at time k, i.e., $p(\mathbf{v}_k|\mathbf{x}_k)$, is denoted as MD_k^i that is defined as

$$MD_k^i \propto -\sum_{j=1}^{N_p} w_{v,k}^i(j)(\mathbf{x}_k^j - \hat{\mathbf{x}}_k)\mathbf{\Sigma}_k^{-1}(\mathbf{x}_k^j - \hat{\mathbf{x}}_k)'. \tag{4.19}$$

The $\mathbf{\Sigma}_k^{-1}$ term helps us to take into account the variance of the individual state variables and the covariances among them. The $w_{v,k}^i(j)$ term in (4.19) ensures that a feature hypothesis which assigns higher weights to samples closer to the mean will get a better fitness value. Based on the fitness value of every feature hypothesis, we can compute a mean feature $\hat{\mathbf{v}}_k$ to be used in the next frame.

4.5.3 The complete tracking algorithm

We use two particle filters to obtain the sequential estimation of the target's states and the feature weights. Once the particle set of $\mathbf{v}_{k-1}$ is available at time $k-1$, its mean estimate $\hat{\mathbf{v}}_{k-1}$ is passed on and defines the feature weights at time k. The particle set corresponding to the tracking states are evaluated based on $\hat{\mathbf{v}}_{k-1}$ to determine the mean state estimate $\hat{\mathbf{x}}_k$. Given the mean estimation, the particle sets of the features $\mathbf{v}_k$ are then updated. The tracking and feature selection are achieved by recursively performing the above process. The pseudo code of the algorithm is provided in Table4.1, and the algorithm diagram is shown in Fig. 4.1.

4.6 Experimental results

The proposed algorithm is evaluated on the AMCOM FLIR dataset. The dataset comprises FLIR sequences in grayscale format (128$\times$128 pixels). Information about the target position, size and type are also made available. We perform the tracking on 3 different sequences namely, LW-14-15 from frames 160 through 230, LW-15-NS from frames 160 through 230 and LW-17-01 from frames 1 through 70. We estimate the position and size of the targets at every frame using a particle filtering framework with adaptive feature selection. We present both visual results of tracking and for the first time, quantitative results for the position and size estimation for this dataset.

4.6.1 Experimental setup

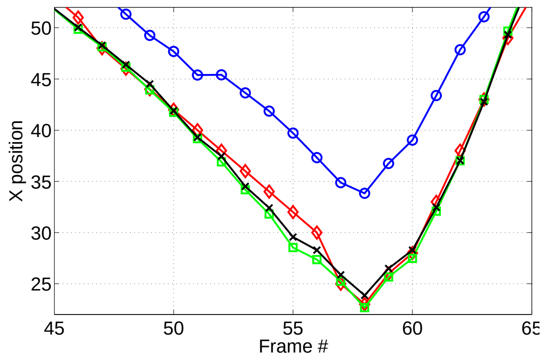
We begin with testing a particle filter algorithm, PF_{simple} which uses the dynamic models described in Sec 4.3 and considering only the foreground intensity histogram for target representation. The second tracking algorithm, PF_{dual} extends PF_{simple} by including the proposed dual target model that has equal weights (0.25) for all four histograms. The third algorithm, $\mathrm{PF}_{feature}$ further extends PF_{dual} by incorporating online feature selection. For every sequence the target appearance is initialized based on the ground truth in the first frame. Initialization methods similar to those presented in [27] may also be used.

The particles for tracking are initialized as a gaussian distribution with unit variance and mean as the true value. We set the number of particle for tracking N_p to be 100 and the number of particles for feature selection N_v as 200. N_v determines the number of linear combination of the features at every time step. The computational time of PF_{dual} is about three times as that of PF_{simple}, as three additional histograms are used along with the foreground histogram. $\mathrm{PF}_{feature}$ requires only a few extra seconds when compared to PF_{dual} since it uses most of the data precomputed in PF_{dual}. In our experiments we set $C_k = 3\mathbf{E}_{\mathbf{n}}[\triangle \mathrm{x}_k]$ in (4.6), $D = \sqrt{3}$ in (4.7), the step size for the feature evolution δ is set as 0.05 and, the parameter λ used in the evaluation of the weights of the tracking particles is set to be 200. The number of bins for the intensity and standard deviation histograms are set to be 64 and 16 respectively.

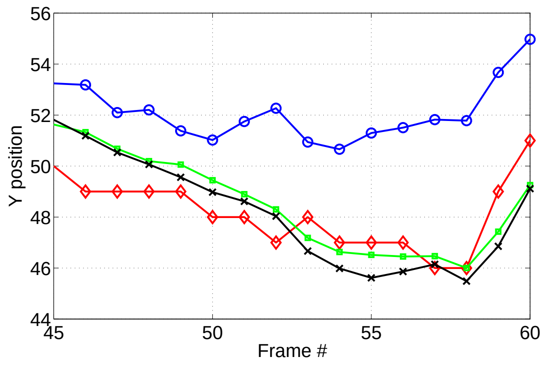
4.6.2 Results and discussion

The partial tracking results for the sequence LW-14-15 are shown in Fig. 4.5 and Table 4.2 compares the performance of the algorithms on three different FLIR sequences. From these, we observe that the $\mathrm{PF}_{feature}$ and PF_{dual} algorithms are able to track the states more accurately when compared to PF_{simple}. This result confirms our theory that, including background information in the target appearance model improves the tracking performance. The importance of the adaptive motion model is also clearly observed, wherein the algorithms are able to cope with the strong global motion of the sensor in frames 45 through 65. Though the $\mathrm{PF}_{feature}$ and PF_{dual} algorithms perform quite similarly in the size estimation, the $\mathrm{PF}_{feature}$ is consistently able to produce size estimates closer to the ground truth value.

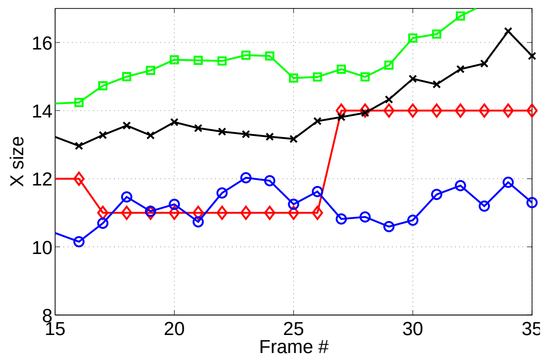
From the table we infer that mostly the use of the dual model improves the position and size estimates. Further improvement is achieved using the feature selection and this trend



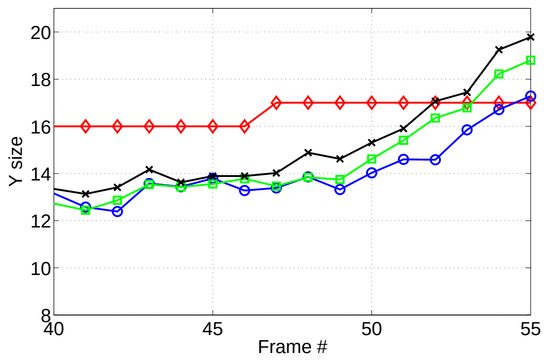
(a)



(b)



(c)



(d)

Figure 4.5: -◇-:Ground truth,-o-:PF_{simple},-□- :PF_{dual},-×-:$\mathrm{PF}_{feature}$. Partial tracking results (15-20 out of 70 frames) of the state vectors (a) X position (b) Y position (c) X size and (d) Y size for the sequence LW-14-15 using the three different algorithms.

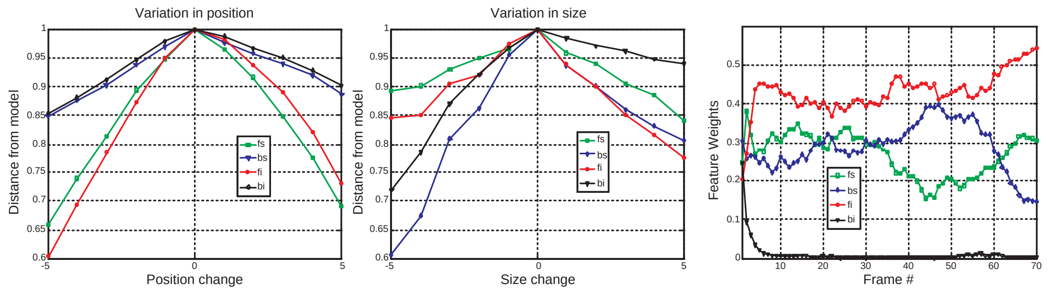


Figure 4.6: Variations of the HI distance defined in (5.6) with respect to the change in positions (a) and sizes (b) for four different features. (c) Variations of the relative importance between four different features during the tracking process of the FLIR sequence LW-14-15.

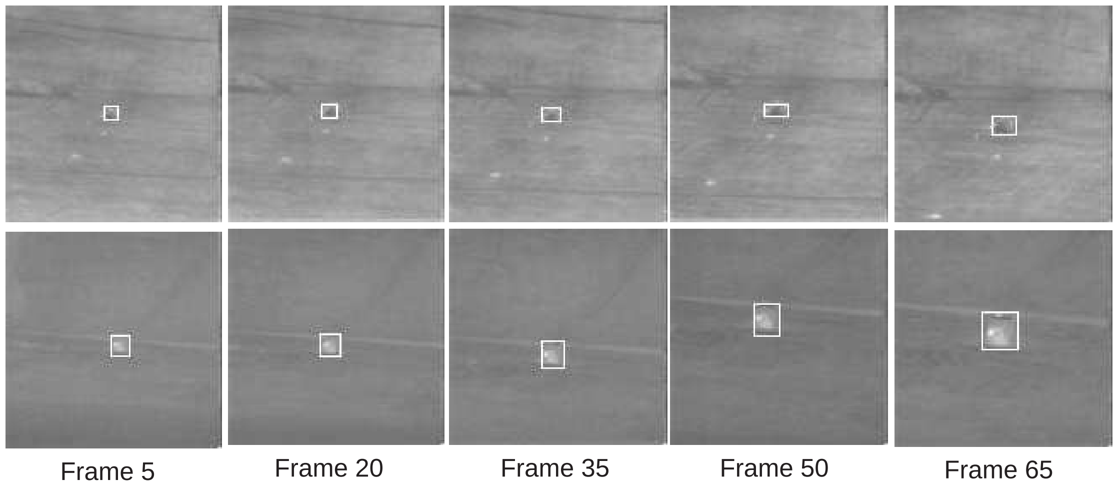


Figure 4.7: The tracking gates produced by algorithm $\mathrm{PF}_{feature}$ for sequences LW-15-NS (top row) and sequence LW-14-15 (bottom row).

is clearly observed for the sequence LW-14-15. However in the sequence LW-15-NS the estimation of s^y has deteriorated with the use of the dual model in comparison to PF_{simple}, which is due to the fact that the choice of equal weights for the features may not be optimal for this sequence. This is ascertained by the fact that both the position and size estimation improve in comparison to PF_{dual} when feature selection is incorporated. Also we note that the state estimation for the sequence LW-17-01 has degraded with the use of feature selection, which is due to the extremely small maximum size of the target (6$\times$9 pixels) where the histograms may not contain sufficient information to result in effective feature evaluation.

In Fig.4.6 (a) and (b) we illustrate the sensitivity of the HI distance to variation in size and position for the different features. We observe that the fi is very sensitive to both position and size changes. The fs feature is sensitive to position but does not show large variations with change in size. Feature bs shows the maximum sensitivity to any size change. Feature bi is the least sensitive to position change and is slightly affected by smaller sizes. Fig.4.6 (c) represents the relative weights of the features during the tracking for the sequence LW-14-15. We observe that feature fi is given the most importance followed by bs, fs and bi. This is expected, since features fi and bs show the maximum sensitivity to the variation of size or position. This result confirms that our feature selection criterion effectively selects the most relevant features for tracking purposes.

```
Initialization: Draw $\mathbf{x}_0^j \sim N(X_0, 1)$, and set $\mathbf{F}'_1 = \mathbf{F}(X_0)$,
  where $X_0$ is the ground truth of the states in the initial frame.
  Draw $\mathbf{\Gamma}_0^i \sim Unif(0,1)$. Set $\hat{\mathbf{v}}_0$=[0.25 0.25 0.25 0.25].
For $k$=1,$\cdots$,T
    For $j$=1,$\cdots$, $N_p$
        Draw $\mathbf{x}_k^j \sim p(\mathbf{x}_k^j|\mathbf{x}_{k-1}^j)$ using (4.6) and (4.7).
        Compute $w_k^j = \exp\left(\lambda \cdot \mathrm{D}(\mathbf{F}(\mathbf{x}_k^j), \mathbf{F}'_k; \hat{\mathbf{v}}_{k-1})\right)$.
    End
    Normalize the weights such that $\sum_{j=1}^{N_p} w_k^j = 1$.
    Compute the mean of the states $\hat{\mathbf{x}}_k$=$\sum_{j=1}^{N_p} w_k^j \mathbf{x}_k^j$.
    Compute the covariance matrix $\mathbf{\Sigma}_k$ using (4.18).
    For $i$=1,$\cdots$,$N_v$
        Draw $\mathbf{\Gamma}_k^i \sim p(\mathbf{\Gamma}_k^i|\mathbf{\Gamma}_{k-1}^i)$ using (4.17).
        Compute $\mathbf{v}_k^i$ based on $\mathbf{\Gamma}_k^i$ using (4.16)
        For $j$=1,$\cdots$,$N_p$
            Compute $w_{v,k}^i(j)$= exp $\left(\lambda \cdot \mathrm{D}(\mathbf{F}(\mathbf{x}_k^j), \mathbf{F}'_k; \mathbf{v}_k^i)\right)$.
        End
        Normalize the weights such that $\sum_{j=1}^{N_p} w_{v,k}^i(j) = 1$.
        Compute $MD_k^i$ for feature $i$ using (4.19).
    End
    Normalize the feature weights such that $\sum_{i=1}^{N_v} MD_k^i = 1$.
    Compute the mean feature vector $\hat{\mathbf{v}}_k$=$\sum_{j=1}^{N_p} MD_k^i \mathbf{v}_k^i$.
    Set $\mathbf{\Gamma}_k^i$=resample($\mathbf{\Gamma}_k^i, MD_k^i$).
    Set $\mathbf{x}_k^j$ =resample($\mathbf{x}_k^j, w_k^j$)
    Update the reference model to obtain $\mathbf{F}'_{k+1}$ using (5.4).
  End
```

Table 4.1: The pseudo-code of the proposed tracking algorithm.

Algorithm	PF_{simple}				PF_{dual}				$\mathrm{PF}_{feature}$			
Sequence	x	y	s^x	s^y	x	y	s^x	s^y	x	y	s^x	s^y
LW-14-15	3.78	1.86	4.17	1.41	1.33	**0.89**	2.12	1.58	**0.95**	0.93	**1.40**	**1.36**
LW-15-NS	1.26	1.04	1.60	**1.23**	**0.80**	1.15	1.02	2.03	0.88	**0.92**	**0.95**	1.33
LW-17-01	0.64	1.44	0.99	1.45	0.75	**0.95**	**0.84**	**1.25**	**0.62**	1.10	0.87	1.28

Table 4.2: Mean error of the state variables over 70 frames averaged over 20 Monte Carlo runs using three different algorithms.

CHAPTER 5

Target Tracking with Online Appearance Learning

5.1 Background

Target tracking in infrared imagery has remained a challenging problem in the field of image processing and understanding. Infrared imagery acquired under actual field condition is typically characterized by strong, structured background clutter, strong sensor noise, poor SNR, and strong ego-motion of the sensor relative to the target. In addition to these characteristics arising from sensors and environment, the targets of interest are highly maneuverable. Therefore, their observed signatures may exhibit profound non-stationary variations over relatively short time scales making it difficult to maintain a robust track lock over long time scales. This phenomenon of the target representation deviating from its true signature due to accumulated tracking errors has been referred to variously as the "drifting problem" in [41,87], the "template update problem" in [34,43,88,89], and a "stale template condition" in [90]. These are challenges that are exemplified by the well-known AMCOM [1] closure sequences [2, 91–96] and the VIVID dataset [2], which we will use as illustrative examples in this work.

An example of this non-stationary variation is shown in Fig. 5.1(a). Here, a longwave imaging sensor is situated on an airborne platform that closes on a pair of maneuvering ground vehicles. Profound changes in the target's appearance are observed between frames 24 and 165 over a time scale of only a few seconds and arise primarily from the relative

[1]This data is available from the Johns Hopkins University Center for Imaging Science (http://cis.jhu.edu) and elsewhere

[2]https://www.sdms.afrl.af.mil/main.php

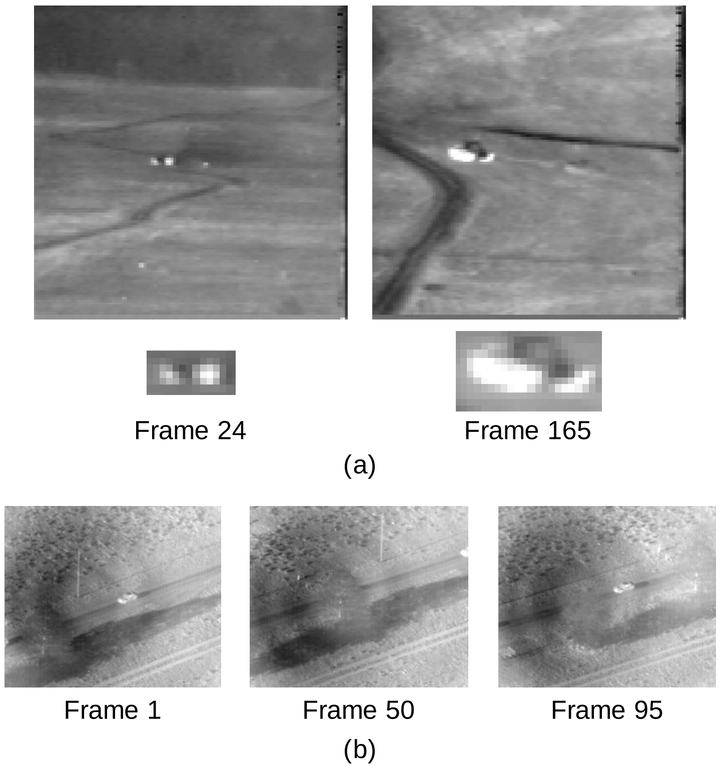


Figure 5.1: (a) Example of nonstationary target signature evolution in AMCOM LWIR run `rng18_17`. There are two vehicles. The lead vehicle is barely visible. The second vehicle, which is clearly visible, is the target of interest. Top row: Observed frames. Bottom row: closeup views of the target. (b) Example of target being occluded by foliage in the VIVID dataset.

motion between the sensor and the target. The is substantial magnification that results from the sensor closing on the target and pose change that results form the target executing an aggressive turning maneuver. While the second vehicle in Fig. 5.1(a) exhibits a strong signature, the lead vehicle is much dimmer and is barely visible amid the surrounding clutter, demonstrating that brightness alone cannot be used as the sole basis for reliable detection and tracking. Rather, more sophisticated techniques are generally required for representing the target appearance and for adapting to (*e.g.*, learning) complex appearance changes that occur over time.

Further, in many cases the target being tracked may move out of the sensor's view or may become occluded thereby significantly altering the observed appearance. An example of this is shown in Fig. 5.1(b) where the target being observed moves behind a tree along its path and thereby disappears from the sensors view. While it is important to adapt the appearance model to accommodate variations in the target signature it is equally important to avoid learning the appearance of occluding objects or the background. A robust tracker must be able to quickly adapt to profound variations in the target signature and must suspend adaptations when the target of interest is occluded or has left the scene.

In this work we address the issues of appearance learning and robust tracking in an unified *co-inference* framework proposed by [97]. A histogram-based appearance learning is explicitly combined with the estimation of the target position and size, achieving sub-pixel accuracy. Specifically, we utilize a dual foreground-background appearance representation that involves a total of four histograms, including histograms of the pixel intensities and of the local standard deviations computed over both the target region and the immediately surrounding background. The intensity histograms for both the target and background are estimated in each frame along with the target position and size. This coupled estimation forms the *co-inference* process, in which the inference of histograms relies on that of target kinematics, and vice versa. The estimation of the histograms, i.e., appearance learning, is achieved by a bank of adaptive Kalman filters (AKFs), where the unknown process and measurement noise variances are estimated simultaneously using the recently developed autocovariance least squares (ALS) method [98, 99]. The main tracker is a particle filter where the state vector gives the target position and magnification and the likelihood function depends on the adaptive appearance model. Further, we devise a track loss detection scheme embedded into the appearance learning process, which provides for robust track loss detection in the VIVID dataset. Aside from the accurate estimation of the target appearance, temporary track losses can also be detected by examining the Kalman filter residuals, a byproducts of AKFs at each step. Hence, we are able to tackle both the "drifting problem" and "temporary track losses" in this *co-inference* framework.

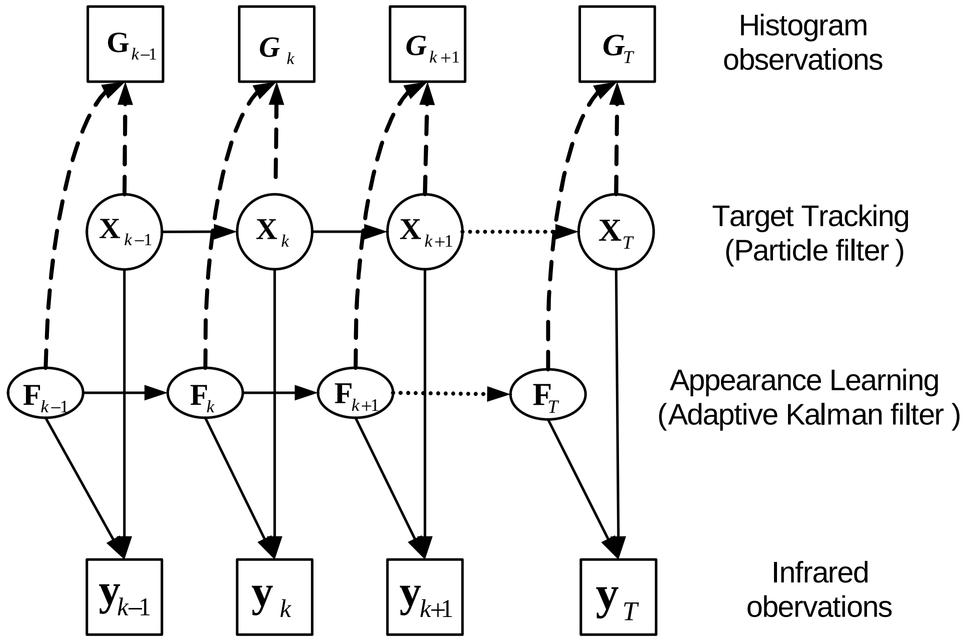


Figure 5.2: Appearance learning for target tracking in a co-inference paradigm.

5.2 Problem formulation

In contrast to most traditional appearance updating schemes, we explicitly model the evolution and observation processes of the target's appearance represented by multiple histograms. Thus, appearance learning, i.e., the updating of target histograms, is formulated as a sequential state estimation problem. In this section, we introduce a graphical model, as shown in Fig. 5.2, which integrates the estimation of target appearance (histogram bins) $\boldsymbol{F}_{1:T}$ with that of target kinematics states, $\boldsymbol{X}_{1:T}$. The target kinematics $\boldsymbol{X}_{1:T}$ and appearance $\boldsymbol{F}_{1:T}$ are coupled by their observations, $\boldsymbol{y}_{1:T}$ and $\boldsymbol{G}_{1:T}$. The intensity statistics within the region given by $\boldsymbol{X}_k$ provides the observation for appearance histograms G_k at the time k, and the appearance states $\boldsymbol{F}_{k-1}$ determines the likelihood of the observed image frame $\boldsymbol{y}_k$. Wu and Huang [97] show that the estimation of hidden states interacted with common observations invokes a *co-inference* process, denoted as:

$$\begin{aligned} P_{X_k} &\approx \mathcal{X}(\boldsymbol{Z}_k, E[\boldsymbol{F}_k|\boldsymbol{Z}_k]), \\ P_{F_k} &\approx \mathcal{F}(\boldsymbol{Z}_k, E[\boldsymbol{X}_k|\boldsymbol{Z}_k]), \end{aligned} \tag{5.1}$$

where $\boldsymbol{Z}_k$ includes the observations of kinematics and appearance histograms, P_{X_k} and P_{F_k} are the probability distributions of $\boldsymbol{X}_k$ and $\boldsymbol{F}_k$, and $\mathcal{X}(\cdot)$ and $\mathcal{F}(\cdot)$ represent the inference processes for these two distributions, respectively. $E(\cdot)$ denotes mathematical expectation, which can be approximated be statistical estimate in practice. Equation 5.1 suggests that the probability of $\boldsymbol{X}_k$ relies on the expectation of $\boldsymbol{F}_k$, and the expectation of $\boldsymbol{X}_k$ is used to calculate the distribution of $\boldsymbol{F}_k$.

We take the advantage of this co-inference in the joint estimation of target kinematic and appearance states, and are able to separately apply two inference algorithms with the estimate of one type of hidden state involved in the other. Recursive Bayesian filters are well suited to both inferences when the dynamic evolution and observation processes for $\boldsymbol{X}_k$ and $\boldsymbol{F}_k$ are explicitly modeled. A Bayesian filter for the kinematics estimation at a single time step k, $p(\boldsymbol{X}_k|\boldsymbol{y}_k)$, from its previous distribution $p(\boldsymbol{X}_{k-1}|\boldsymbol{y}_{k-1})$ can be denoted as [84]:

$$p(\boldsymbol{X}_k|\boldsymbol{y}_{k-1}) = \int_{\boldsymbol{X}_{k-1}} p(\boldsymbol{X}_k|\boldsymbol{X}_{k-1})p(\boldsymbol{X}_{k-1}|\boldsymbol{y}_{k-1})d\boldsymbol{X}_{k-1}, \tag{5.2}$$

$$p(\boldsymbol{X}_k|\boldsymbol{y}_k) \approx p(\boldsymbol{y}_k|\boldsymbol{X}_k, \bar{\boldsymbol{F}}_{k-1})p(\boldsymbol{X}_k|\boldsymbol{y}_{k-1}). \tag{5.3}$$

It should be noted that the estimate of appearance, $\bar{\boldsymbol{F}}_{k-1}$, is embedded in the inference of $\boldsymbol{X}_k$. Analogously, the inference of histograms can also be obtained by the recursive Bayesian filter with the estimate of $\boldsymbol{X}_k$ involved. These filters are numerically implemented by the particle filtering and adaptive Kalman filters due to different physical characteristics and mathematical assumptions on the dynamic processes of target kinematics and appearance, respectively.

As strong ego motion and maneuvering actions present in typical infrared imagery, the kinematic transition, i.e., $p(\boldsymbol{X}_k|\boldsymbol{X}_{k-1})$ in Eq. 5.2, can hardly be characterized by linear equations. Moreover, the calculation of histogram observations of a target in Eq. 5.3 requires non-linear operations from image pixels in the region given by $\boldsymbol{X}_k$. Hence, a particle filtering based technique is necessary for the estimation of $\boldsymbol{X}_k$ by Eqs. 5.2 and

5.3, which is detailed in Section 5.4. The estimate of appearance histograms is applied to the evaluation of the likelihood, $p(\boldsymbol{y}_k|\boldsymbol{X}_k, \bar{\boldsymbol{F}}_{k-1})$ rather than to the proposal density that generates the samples of $p(\boldsymbol{X}_k)$ in [97].

On the other hand, a Kalman filter is applicable to appearance learning. The dynamics of appearance histograms, $p(\boldsymbol{F}_k|\boldsymbol{F}_{k-1})$, can be reasonably assumed as Gaussian, which reflects the actual appearance variation in infrared imagery. Interestingly, it is possible to deal with abrupt appearance variations due to occlusion through the byproduct of the Kalman estimation, which is elaborated in Sec. 5.5. The appearance histograms obtained from the image pixels within the region given by the estimate of kinematics, $\bar{\boldsymbol{X}}_k$, can be regarded as the direct observation of the true state of appearance histograms corrupted by Gaussian noise. The estimate of $\boldsymbol{X}_k$ is naturally brought into the measurement updating in the Kalman filtering process in order to achieve *co-inference*. However, the robustness and generalization of the estimation is questionable if the noise parameters are set to known in an *ad hoc* way, because targets may present significant appearance variations captured in different physical conditions (e.g., weather and temperature) even by IR sensors with identical specifications. The adaptive estimation of these noise parameters turns out to be the key to the success of this AKF-based appearance learning. Two specific AKFs will be discussed that are compared with the traditional histogram similarity method.

5.3 Histogram-based appearance learning

Let $\mathbf{y}_k$ be a sequence of video frames acquired from an imaging sensor at discrete time instants $k \in \mathbb{N}$. For simplicity, we suppose throughout Section 5.3 that there is a single object of interest, which could be, *e.g.*, a target or a patch of background. Let $\mathbf{g}_k = \{g_k^b\}_{b=1,\ldots,N_b}$ be the observed normalized histogram of the object computed from the frame $\mathbf{y}_k$, where $\sum_{b=1}^{N_b} g_k^b = 1$ and where the histogram is discretized to N_b bins. Similarly, let $\mathbf{f}_k = \{f_k^b\}_{b=1,\ldots,N_b}$ be the reference histogram, which provides an idealized model of the object appearance at time k. The objective of histogram learning is to estimate the present appearance model $\mathbf{f}_k$ by incorporating the current observation $\mathbf{g}_k$ into the previous appearance model $\mathbf{f}_{k-1}$. This is typically formulated as a time-varying linear filter

$$\mathbf{f}_k = \boldsymbol{\xi}_k \cdot \mathbf{g}_k + (\mathbf{1} - \boldsymbol{\xi}_k) \cdot \mathbf{f}_{k-1}, \tag{5.4}$$

where $\mathbf{1}$ is a vector with all entries equal to one and "$\cdot$" represents the Hadamard (or Schur) product. The vector $\boldsymbol{\xi_k} = \{\xi_k^b\}_{b=1,\dots,N_b}$ controls the balance between the previous reference model $\mathbf{f}_{k-1}$ and the new observation $\mathbf{g}_k$, where $0 \leq \xi_k^b \leq 1$ is the time dependent filter coefficient for the bth histogram bin. Accurate tuning of $\boldsymbol{\xi_k}$ is the key to effective appearance learning.

In this section we discuss three different learning techniques that share the form (5.4) and differ only in how $\boldsymbol{\xi_k}$ is computed. The first is the traditional histogram similarity based method where all bins are updated with the same weight ($\xi_k^b = \xi_k; b = 1, ..., N_b$). We shall refer to this method as HS. After briefly reviewing the basic Kalman filter, we turn our attention to two AKF methods that use different approaches for estimating the process and measurement noise variances. The first, which we will call $\mathrm{AKF}_{\mathrm{cov}}$, uses covariance matching where the same weight is applied to all bins. The second, which we refer to as $\mathrm{AKF}_{\mathrm{als}}$, uses the recent ALS technique [98,99] and maintains a separate weight ξ_k^b for each histogram bin.

5.3.1 Histogram similarity method (HS)

In the widely used HS method, the weight vector $\boldsymbol{\xi}_k$ in (5.4) is updated based on histogram similarity [39,86]. All N_b entries of $\boldsymbol{\xi}_k$ share a common value given by the metric

$$\xi_k = 1 - \mathrm{h}(\mathbf{f}_{k-1}, \mathbf{g}_k), \tag{5.5}$$

where h is a normalized similarity measure. One popular similarity measure is derived from the Bhattacharyya coefficient [19]. In practice, however, we find that the histogram intersection defined by [86]

$$\mathrm{h}(\mathbf{f}_{k-1}, \mathbf{g}_k) = \sum_{i=1}^{N_b} \min(f_{k-1}^i, g_k^k) \tag{5.6}$$

is more useful for quantifying histogram similarity in IR imagery. With (5.6), if the observed and reference histograms are nearly identical then $\mathrm{h}(\mathbf{f}_{k-1}, \mathbf{g}_k) \approx 1$ and ξ_k is small, implying that very little information from the observation will be incorporated into the learning process at time step k. Alternatively, if the two histograms are almost mutually exclusive then $\mathrm{h}(\mathbf{f}_{k-1}, \mathbf{g}_k) \approx 0$ and $\xi_k \approx 1$, implying that the new reference histogram will be heavily dependent on the observation and will largely discard the historical information in $\mathbf{f}_{k-1}$. Thus, the observation is weighted strongly when there is a sudden change in the object appearance. Note that the similarity metric (5.5), (5.6) depends on all N_b histogram bins and is scalar-valued, implying that a common weight ξ_k is applied to all bins with the HS method.

As with most dynamic appearance learning strategies, the HS method can potentially overadapt in the presence of strong measurement noise and/or rapidly evolving target and clutter signatures causing track loss due to the target appearance model becoming corrupted with background information. Explicit outlier rejection measures were implemented in [37, 40] to control this problem.

To reformulate appearance learning as a Kalman filtering problem, we model corresponding bins f_k^b and g_k^b from the reference and observed histograms in state space according to

$$f_{k+1}^b = f_k^b + w_k^b, \tag{5.7}$$

$$g_k^b = f_k^b + v_k^b, \tag{5.8}$$

where w_k^b and v_k^b are mutually uncorrelated process and measurement noises, both assumed zero-mean, white, and Gaussian with variances $\sigma_{wb}^2(k)$ and $\sigma_{vb}^2(k)$ that are time-varying in general. The Kalman filter state prediction and update equations for the system under consideration are given by

$$\text{State prediction:}\quad \widehat{f}^b_{k|k-1} = \widehat{f}^b_{k-1} \tag{5.9}$$

$$\text{Covariance prediction:}\quad p^b_{k|k-1} = p^b_{k-1} + \sigma^2_{wb}(k) \tag{5.10}$$

$$\text{Kalman gain:}\quad K^b_k = \frac{p^b_{k|k-1}}{p^b_{k|k-1} + \sigma^2_{vb}(k)} \tag{5.11}$$

$$\text{Innovation:}\quad r^b_k = g^b_k - \widehat{f}^b_{k|k-1} \tag{5.12}$$

$$\begin{aligned}\text{State update:}\quad \widehat{f}^b_k &= \widehat{f}^b_{k|k-1} + K^b_k r^b_k \\ &= K^b_k g^b_k + (1 - K^b_k)\widehat{f}^b_{k-1}\end{aligned} \tag{5.13}$$

$$\text{Covariance update:}\quad p^b_k = (1 - K^b_k) p^b_{k|k-1}. \tag{5.14}$$

There is a direct correspondence between (5.4) and (5.13), where the Kalman gain K^b_k in (5.13) may be associated with the coefficient ξ^b_k in (5.4); hence, with the Kalman filtering formulation we obtain $\xi^b_k \equiv K^b_k$.

The Kalman filter balances the relative contributions to appearance learning from the reference and observed data based on the estimated variances $\sigma^2_{wb}(k)$ and $\sigma^2_{vb}(k)$. When $\sigma^2_{wb}(k) \gg \sigma^2_{vb}(k)$, for example, we have $K^b_k \approx 1$ implying that the observation will be weighted much more heavily than the historical reference data. Under the linearity and Gaussianity assumptions applied here, the state estimates (5.9) and (5.13) are optimal in the minimum mean squared error sense.

However, computing the Kalman gain (5.11) requires knowledge of $\sigma^2_{wb}(k)$ and $\sigma^2_{vb}(k)$, both of which are usually unknown in practice. This leads to the adaptive Kalman filter (AKF), which seeks to estimate the unknown noise variances on the fly. A brief overview of AKF methods was given in [42] and more recent surveys appear in [100, 101]. In [42], these techniques were broadly divided into four categories: Bayesian, maximum likelihood, correlation, and covariance matching methods. The former two are computationally expensive in general. In the following, two different AKF-based appearance learning algorithms are presented that rely on the covariance matching and correlation approaches.

5.3.2 AKF: Covariance Matching (AKF$_{cov}$)

Covariance matching techniques [42, 102] are based on the relationship that exists between the process and measurement noise variances and the autocorrelation of the innovations process (5.12). Since the innovations are observable, their autocorrelation can be estimated by an empirical sample variance under suitable ergodicity assumptions. Thus, if one of the two variances $\sigma_{wb}^2(k)$ and $\sigma_{vb}^2(k)$ is known, then the other can be estimated by matching the empirically calculated innovations autocorrelation to its theoretical value. Here, we adopt the specific technique used in [37, 40, 41] where $\sigma_{vb}^2(k)$ is known and $\sigma_{wb}^2(k)$ is obtained by covariance matching

It follows easily from (5.7)-(5.14) that the autocorrelation of the innovations process is given by [103, Section V.B]

$$E[r_k^b r_j^b] = [p_{k-1}^b + \sigma_{vb}^2(k) + \sigma_{wb}^2(k-1)]\delta(k-j), \tag{5.15}$$

where $\delta(\cdot)$ is the Kronecker delta. With $\sigma_{vb}^2(k)$ known and p_{k-1}^b given by (5.14), an obvious empirical approach for solving $\sigma_{wb}^2(k-1)$ from (5.15) is to approximate $E[(r_k^b)^2]$ by computing the sample variance of (5.12) over the last L_{cov} frames $\mathbf{y}_{k-L_{\text{cov}}+1}, \ldots, \mathbf{y}_k$. However, because the process noise could be time varying in general, there is a delicate tradeoff between choosing L_{cov} large enough to obtain statistically significant estimates while simultaneously choosing L_{cov} small enough to track nonstationary changes in $\sigma_{wb}^2(k)$.

In appearance learning for visual target tracking, this problem has been addressed previously by assuming identical statistics across variables in order to increase the sample size to larger than L_{cov} while still sampling from only the L_{cov} most recent frames. In [37], it is assumed that $\sigma_{vb}^2(k)$ is independent of both b and k and that $\sigma_{wb}^2(k)$ is independent of b, so that all N_b bins of the histogram in each frame share identical noise statistics. The innovations sample variance may then be computed across bins as well as over time. The same assumptions on $\sigma_{vb}^2(k)$ are made for the template-based appearance model of [40], where b indexes pixels in the template rather than bins in the histogram. By assuming a common value $\sigma^2_{wb}(k)$ for all template pixels in the current frame, the innovations sample variance can be averaged across both pixels and time. A similar strategy was employed in [41] with the principle difference that $\sigma^2_{vb}(k)$ was assumed time varying and estimated by an auxiliary algorithm independent of the covariance matching. Similar covariance matching was used to estimate the scale matrix in [36].

To formulate this class of covariance matching algorithms in our present setup, we assume that $\sigma^2_{vb}(k)$ is independent of both k and b and that $\sigma^2_{wb}(k)$ is independent of b (as in [37,40]). Let $\mathcal{B}$ be the set of nonzero histogram bins ($|\mathcal{B}|$ denotes the number of nonzero bins) and estimate $E[(r^b_k)^2]$ with the sample variance

$$\widehat{C}_r(k) = \frac{1}{|\mathcal{B}|L_{\text{cov}}} \sum_{i=0}^{L_{\text{cov}}-1} \sum_{b\in\mathcal{B}} (r^b_{k-i})^2. \tag{5.16}$$

Under these assumptions p^b_{k-1} is independent of b. Thus, we arbitrarily choose p^1_{k-1} and use (5.16) in (5.15) to obtain the approximate solution

$$\sigma^2_{wb}(k-1) \approx \widehat{C}_r(k) - \sigma^2_{vb}(k) - p^1_{k-1}. \tag{5.17}$$

As in [37,40], the initialization at $k=1$ is given by

$$\sigma^2_{vb}(k) = \tfrac{1}{2}\widehat{C}_r(1)\ \forall\, b, k; \quad p^b_0 = \tfrac{1}{2}\widehat{C}_r(1)\ \forall\, b, \tag{5.18}$$

which implies $\sigma^2_{wb}(0) = 0$. We refer to this algorithm as AKF_{cov} and use it in the following primarily as a baseline for comparison with the AKF_{als} technique given in the next section.

5.3.3 AKF: Autocovariance Based Least Squares (AKF*als*)

The ideal expression (5.15) for the autocorrelation of the innovations holds when there are no modeling errors and the filter gains K^b_k in (5.11) are optimal. However, if the process and measurement noise variances are unknown, then the gains will be suboptimal and the innovations process will generally exhibit a nontrivial correlation structure. The main idea of autocovariance based methods is to exploit any observed nonzero correlations at lags other than zero to obtain solutions for the unknown noise variances and/or the optimal gains. Pioneering work in this area was given by Mehra in [42, 104] where the residual autocorrelation was used for adaptive Kalman filtering. Mehra's method involves a three-step iterative process where a Lyapunov-type equation must be solved at every time step. Under the assumption that the process and measurement noises are wide sense stationary (WSS), Carew and Bélanger [105] developed an improved algorithm that estimates the optimal Kalman gains directly using one matrix inversion and several matrix multiplications, eliminating the need to estimate the process and measurement noise variances explicitly and avoiding the requirement to iteratively solve the Lyapunov equation associated with Mehra's method. Neethling and Young [106] introduced a related weighted least squares technique that improves the statistical efficiency of the methods in [42, 104, 105] and incorporates a side constraint to guarantee positive semi-definite (PSD) estimates for the unknown noise variances.

Recently, Odelson, *et al.*, developed a new Autocovariance Least Squares (ALS) method capable of providing PSD estimates for both the process and measurement noise variances simultaneously [98]. In addition, the ALS variance estimates are more stable than those delivered by Mehra's method and converge asymptotically to the optimal values with increasing sample size. However, the proof of convergence given in [98,107] depends explicitly on assumptions that the system is time invariant and that the process and measurement noises are WSS (extension to a time varying system with WSS noises was given in [108]). The ALS algorithm in [98] is primarily meant for identifying the system noise properties in an offline learning process under WSS assumptions. First, the filter innovations are obtained from the observations using a suboptimal Kalman gain over an extended period of time. Then the autocovariance structure of these innovations is used to reliably estimate the noise variances. Once the noise variances are known, the optimal Kalman gain can be determined and applied for filtering during run time using the standard Kalman filtering equations (5.9)-(5.14).

For appearance learning, our interest in this paper is primarily in real-time, online scenarios where, for the first time, we consider application of the ALS method under the much weaker assumption that the noise variances $\sigma^2_{wb}(k)$ and $\sigma^2_{vb}(k)$ are only *piecewise stationary*. In order to extend the ALS method to this case, we consider the evolution of the target appearance to be a piecewise stationary process with non-stationary transitions. The piecewise stationarity assumption can be justified by the high frame rate of the imaging sensor compared to the rate at which the target appearance changes. Such assumptions are common in, *e.g.*, the context of audio and video compression [109–115]. The nonstationary characteristics of $\sigma^2_{wb}(k)$ and $\sigma^2_{vb}(k)$ directly correlate with the rate at which the target appearance and sensor noise are changing. The piecewise stationary formulation allows us to apply the ALS algorithm to each stationary block individually and thereby allows us to adapt to the varying nature of the target appearance histogram over time. In effect, we adapt the filter gain ξ^b_k at the end of each stationary block depending on the observed variation trend in that block. This raises the issue of determining the block boundaries. Most existing methods that determine the block intervals require *a priori* knowledge of the observations; since this is not the case in our real-time application, we consider equal length blocks. We study the effect of block size by performing experiments using the ALS method on a simulated nonstationary system in Section 5.3.4.

In this section, we extend the ALS method for application to a piecewise stationary process in the context of histogram-based appearance learning, which we refer to as $\mathrm{AKF}_{\mathrm{als}}$ in this paper. As before, the state model is given by (5.7) and (5.8). We assume that w^b_k and v^b_k are mutually uncorrelated and that $\sigma^2_{wb}(k)$ and $\sigma^2_{vb}(k)$ depend on b and k and are piecewise constant. With this setup, the noise statistics are generally different for each bin of the histogram and there is a separate coefficient ξ^b_k for each $b \in [1, N_b]$. The size of each piecewise stationary block is assumed to be N_d frames. We also define a block index p, and then the pth block is represented by

$$\boldsymbol{Y}(p) = \{\mathbf{y}_k | k \in \mathbb{K}(p)\}, \tag{5.19}$$

where

$$\mathbb{K}(p) = \{k|(p-1)N_d + 1 \leq k \leq pN_d\}. \tag{5.20}$$

Using this framework, we update the noise variances of the appearance histogram corresponding to each bin at the end of every block. In effect, we are adapting the filter gain (learning rate) for the current block based on the observed variations in the preceding block.

We now briefly present the least squares formulation to determine the system noise variances of a particular histogram bin at the end of a single block. For the remainder of the section, we drop the bin index b for brevity. We assume that the asymptotic Kalman gain K_{p-1} estimated from the previous block is available. Given K_{p-1}, the state estimates in (5.13) for all frames in $\mathbb{K}(p)$ are given by $\widehat{f}_k = \widehat{f}_{k|k-1} + K_{p-1}r_k$. The error in the predicted state (5.9) is defined as $\varepsilon_k = f_k - \widehat{f}_{k|k-1}$. Then, for all frames $k \in \mathbb{K}(p)$, this prediction error along with the innovation (5.12) can be formulated together in a state model according to [98, 99]

$$\begin{aligned} \varepsilon_{k+1} &= \overbrace{(1-K_{p-1})}^{\overline{A}_p}\varepsilon_k + \overbrace{\begin{bmatrix} 1 & -K_{p-1} \end{bmatrix}}^{\overline{G}_p} \overbrace{\begin{bmatrix} w_k \\ v_k \end{bmatrix}}^{\overline{W}_p}, & (5.21) \\ r_k &= \varepsilon_k + v_k. & (5.22) \end{aligned}$$

The ALS method aims to observe the filter innovations and exploit any observed nonzero correlations at different lags to obtain solutions for the unknown noise variances and/or the optimal gains. The autocorrelation of the innovations in the pth block at any lag j is given by

$$\mathscr{C}_j(p) = E[r_k r_{k+j}]; \quad 0 \leq j < L_{\text{als}}, \tag{5.23}$$

where $k, k+j \in \mathbb{K}(p)$ and $L_{\text{als}} < N_d$ is the order of the autocorrelation lags we wish to consider in formulating the ALS problem. We assume $E[\varepsilon_0] = 0$ and cov$(\varepsilon_0) = \pi_0$ and define

$$\begin{aligned}\overline{Q}_p &= E[\overline{W}_p \overline{W}_p{}^T] = \begin{bmatrix} \sigma_w^2(pN_d) & 0 \\ 0 & \sigma_v^2(pN_d) \end{bmatrix}, \qquad (5.24)\\ \chi_p &= E[\overline{W}_p v_k] = \begin{bmatrix} 0 \\ \sigma_v^2(p) \end{bmatrix} \qquad (5.25)\end{aligned}$$

for $k \in \mathbb{K}(p)$. Note in (5.25) that although $E[\overline{W}_p v_k]$ contains the time index k, this expectation is constant over $\mathbb{K}(p)$ due to the piecewise stationarity assumption.

In the interest of clarity and to illustrate the form of the relevant relations, we assume $L_{\text{als}} = 3$ in the following; generalization to other L_{als} is straightforward. The least squares estimation problem is formulated in terms of an autocovariance matrix $R_p(L_{\text{als}})$ that, for $L_{\text{als}} = 3$ and $k, k+1, k+2 \in \mathbb{K}(p)$, is given by

$$R_p(3) = E \begin{bmatrix} (r_k)^2 & r_k r_{k+1} & r_k r_{k+2} \\ r_k r_{k+1} & (r_{k+1})^2 & r_{k+1} r_{k+2} \\ r_k r_{k+2} & r_{k+1} r_{k+2} & (r_{k+2})^2 \end{bmatrix}. \qquad (5.26)$$

The individual elements of $R_p(3)$ are functions of $\pi_0, \bar{A}_p, \bar{G}_p, \bar{Q}_p$ and $\bar{\chi}_p$. Let "vec" be the vectorization operator which transforms a matrix into a vector by stacking the columns upon one another. Then the vectorization of $R_p(3)$ is given by

$$\begin{aligned}\text{vec}[R_p(3)] &= (\Theta_p \otimes \Theta_p)\pi_0 \\ &\quad + \Gamma_p \otimes \Gamma_p \text{vec}(I_3)\text{vec}(\overline{G}_p \overline{Q}_p \overline{G}_p{}^T) \\ &\quad + (\Psi_p \oplus \Psi_p + I_3)\text{vec}(I_3)\sigma_v^2(p), \qquad (5.27)\end{aligned}$$

where I_n denotes the $n \times n$ identity matrix, $\otimes$ denotes the Kronecker product, $\oplus$ denotes direct sum, and the matrices Θ_p, Γ_p and Ψ_p are given by

$$\Theta_p = \begin{bmatrix} 1 \\ \overline{A}_p \\ \overline{A}_p^2 \end{bmatrix}, \Gamma_p = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ \overline{A}_p & 1 & 0 \end{bmatrix}, \Psi_p = \Gamma_p \oplus_{i=1}^{3} K_{p-1}. \qquad (5.28)$$

Using the Lyapunov equation to eliminate the π_0 term in (5.27), one obtains

$$\overbrace{\text{vec}[R_p(3)]}^{\mathscr{R}_p(3)} = \overbrace{\left[\begin{array}{c} D_p \,|\, D_p K_{p-1}^2 + (\Psi_p \oplus \Psi_p + I_9)\text{vec}(I_3) \end{array}\right]}^{\mathscr{A}_p} \overbrace{\begin{bmatrix} \sigma_w^2(p) \\ \sigma_v^2(p) \end{bmatrix}}^{\mathbf{x}_p}, \tag{5.29}$$

where

$$D_p = (\Theta_p \otimes \Theta_p)(1 - \overline{A}_p \otimes \overline{A}_p)^{-1} + (\Gamma_p \otimes \Gamma_p)\text{vec}[I_3], \tag{5.30}$$

and $\mathscr{R}_p(3)$ may be represented in terms of the autocorrelation terms defined in (5.23) as

$$\mathscr{R}_p(3) = \text{vec}\begin{bmatrix} \mathscr{C}_0(p) & \mathscr{C}_1(p) & \mathscr{C}_2(p) \\ \mathscr{C}_1(p) & \mathscr{C}_0(p) & \mathscr{C}_1(p) \\ \mathscr{C}_2(p) & \mathscr{C}_1(p) & \mathscr{C}_0(p) \end{bmatrix}. \tag{5.31}$$

Provided that the innovations process is reasonably locally ergodic, the quantities $\mathscr{C}_j(p)$ in (5.31) may then be estimated by

$$\widehat{\mathscr{C}_j}(p) = \frac{1}{N_d - j} \sum_{i=(p-1)N_d+1}^{pN_d - j} r_i r_{i+j}. \tag{5.32}$$

We define an estimated vectorized correlation matrix $\widehat{\mathscr{R}}_p(3)$ by replacing the theoretical correlations $\mathscr{C}_j(p)$ in (5.31) with the empirical estimates $\widehat{\mathscr{C}_j}(p)$ given by (5.32). From this definition and (5.29), we write

$$\mathscr{A}_p \mathbf{x}_p = \widehat{\mathscr{R}}_p(3). \tag{5.33}$$

The expression (5.33) forms the core of the ALS method: it relates the observed correlations contained in $\widehat{\mathscr{R}}_p(3)$ and defined in (5.32) to the desired variances $\sigma_w^2(p)$ and $\sigma_v^2(p)$ contained in $\mathbf{x}_p$. Also note that $\mathscr{A}_p$ is dependent only on the asymptotic Kalman gain K_{p-1} from the previous block. Thus, the least squares problem for the unknown noise variances

Table 5.1: Pseudo-code to implement $\mathrm{AKF}_{\text{als}}$ for a single bin of the appearance histogram during the pth time block and kth frame.

```
1. Predict bin value $\widehat{f}_{k|k-1} = \widehat{f}_{k-1}$.
2. Acquire observation $g_k$ based on tracker output.
3. Compute innovation $r_k = g_k - \widehat{f}_{k|k-1}$.
4. Update bin value $\widehat{f}_k = \widehat{f}_{k|k-1} + K_{p-1} r_k$.
5. If $k = pN_d$
   • Find $\widehat{\mathscr{R}}_p(L_{\text{als}})$ from $\widehat{\mathscr{C}}_j(p)$ for $0 \leq j \leq L_{\text{als}} - 1$ using (5.32)
   • Determine $\mathscr{A}_p$ using (5.29) to setup the ALS problem in (5.33).
   • Perform the optimization in (5.35) to obtain $\sigma_v^2(p)$ and $\sigma_w^2(p)$.
   • Compute asymptotic Kalman gain $K_p$ from the estimated
     noise variances for use in the next block.
  End
```

$\sigma_w^2(p)$ and $\sigma_v^2(p)$ can be expressed as

$$\Phi_p = \min_{\sigma_w^2(p),\sigma_v^2(p)} \left\| \mathscr{A}_p \begin{bmatrix} \sigma_w^2(p) \\ \sigma_v^2(p) \end{bmatrix} - \widehat{\mathscr{R}}_p(3) \right\|^2 \tag{5.34}$$

subject to $\sigma_w^2(p), \sigma_v^2(p) \geq 0$. The positive semi-definite requirements on $\sigma_w^2(p)$ and $\sigma_v^2(p)$ are enforced by appending a logarithmic barrier function to (5.34), resulting in

$$\begin{aligned} \Phi_p \quad = \quad & \min_{\sigma_w^2(p),\sigma_v^2(p)} \left\| \mathscr{A}_p \begin{bmatrix} \sigma_w^2(p) \\ \sigma_v^2(p) \end{bmatrix} - \widehat{\mathscr{R}}_p(3) \right\|^2 \\ & - \mu \log[\sigma_w^2(p)\sigma_v^2(p)], \end{aligned} \tag{5.35}$$

where μ is the barrier parameter. The least squares problem (5.35) has been shown to be convex and can be solved using a Newton recursion [98]. Pseudo-code to implement this $\mathrm{AKF}_{\text{als}}$ algorithm for a single bin of the histogram is given in Table 5.1.

5.3.4 Numerical simulations

Having extended the ALS method to the piecewise stationary case, we perform two numerical experiments on simulated data. The first compares AKF$_{\text{cov}}$ and AKF$_{\text{als}}$ in terms of their capability of noise estimation on a system with WSS noise characteristics. The second examines the performance of the proposed piecewise stationary ALS method against piecewise stationary and more general nonstationary system dynamics.

5.3.4.1 Comparison between AKF_{als} and AKF_{cov}

The objective of this experiment is to estimate the unknown noise covariance matrices from simulated data using AKF$_{\text{cov}}$ and AKF$_{\text{als}}$. Consider a system of the form

$$\boldsymbol{x}_k = \mathbf{A}\boldsymbol{x}_{k-1} + \boldsymbol{w}_{k-1}, \tag{5.36}$$

$$\boldsymbol{y}_k = \mathbf{C}\boldsymbol{x}_k + \boldsymbol{v}_k, \tag{5.37}$$

where $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ are zero mean, iid Gaussian noise processes with fixed covariances $\mathbf{Q}$ and $\mathbf{R}$, respectively. Let

$$\mathbf{A} = \begin{bmatrix} 0.1 & 0 & 0.1 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.3 \end{bmatrix}, \qquad \mathbf{C} = \begin{bmatrix} 1 & -0.1 & 0.2 \\ -0.2 & 1 & 0 \\ 0 & -0.4 & 1 \end{bmatrix},$$

$$\mathbf{Q} = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.75 & 0 \\ 0 & 0 & 0.25 \end{bmatrix}, \qquad \mathbf{R} = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.25 & 0 \\ 0 & 0 & 0.75 \end{bmatrix}.$$

During estimation, the diagonal elements of $\mathbf{Q}$ and $\mathbf{R}$ were initialized with random values uniformly distributed between zero and one. The asymptotic filter gain for the initialized noise covariances was then computed. This gain was used for filtering over 5000 data points to obtain innovations that were used, along with the initial estimates of $\mathbf{Q}$ and $\mathbf{R}$ by the AKF$_{\text{cov}}$ and AKF$_{\text{als}}$ methods to estimate the unknown noise covariances. Results from repeating the simulation 200 times are shown in Fig. 5.3, where each data point corresponds to the estimate from a single trial. It is observed that AKF$_{\text{als}}$ produces estimates that are positive semi-definite and more precise than those delivered by AKF$_{\text{cov}}$. The estimates produced by AKF$_{\text{cov}}$ seem to depend on the initial values of the unknowns. Since AKF$_{\text{cov}}$ assumes that at least one of the noise covariances is known *a priori*, an erroneous initial value can greatly distort the estimation. Further, there is no guarantee that the estimates (5.17) are PSD, as seen by the occasional negative estimates of the AKF$_{\text{cov}}$ method in Fig. 5.3. Unlike AKF$_{\text{cov}}$, AKF$_{\text{als}}$ (1) estimates both process and observation noise parameters simultaneously, (2) formulates a least squares problem based on multiple constraints obtained by considering the autocorrelation of the innovations at different lags and (3) enforces PSD constraints on the estimates.

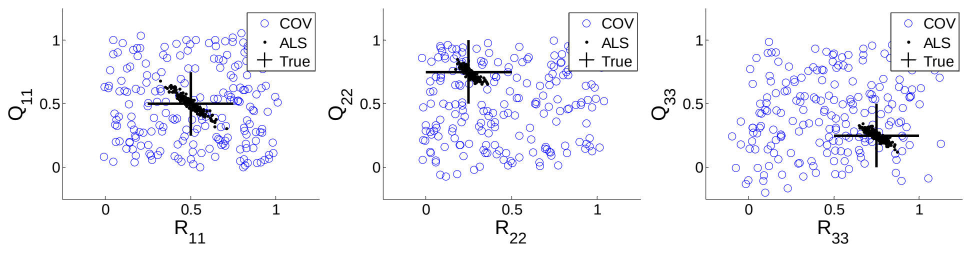


Figure 5.3: Diagonal elements of the noise covariance matrices $\mathbf{Q}$ and $\mathbf{R}$ as estimated by AKF$_{\text{cov}}$ and AKF$_{\text{als}}$ for WSS system dynamics.

5.3.4.2 Piecewise treatment of non-stationary systems by AKF$_{als}$

Here, we examine the performance of AKF$_{\text{als}}$ and its inherent block stationarity assumptions against the linear state model (5.36), (5.37) for the case of nonstationary noise processes $\boldsymbol{w}_{k-1}$ and $\boldsymbol{v}_k$ with diagonal covariance matrix entries that exhibit jump transitions and linear ramps. Let

$$\mathbf{A} = \begin{bmatrix} 0.9 & 0 & 0.7 \\ 0 & 0.95 & 0 \\ 0 & 0 & 0.7 \end{bmatrix}, \mathbf{C} = \begin{bmatrix} 1 & -0.1 & 0.2 \\ -0.2 & 1 & 0 \\ 0 & -0.4 & 1 \end{bmatrix},$$

and let $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ be zero mean, iid Gaussian noise processes with time varying diagonal covariance matrices $\mathbf{Q}$ and $\mathbf{R}$ having main diagonal entries given by the dotted blue lines in Fig. 5.4. As indicated in the figure, the noise covariances are block stationary during the first portion of each simulation and increase or decrease linearly with small-scale additive noise during the second portion. The transition times between these characteristics for all six diagonal covariance matrix entries are mutually independent.

The objective is to estimate the six unknown covariances using the AKF_{als} algorithm developed in Section 5.3.3. In the absence of any *a priori* knowledge about the transition times between piecewise stationary and linear characteristics in the noise variances, we set the block length N_d in the AKF_{als} algorithm to a constant. Choosing N_d small results in a paucity of data points being available to perform statistically significant least squares estimation, whereas choosing N_d large limits the ability of the algorithm to adapt to the nonstationary changes. The experiment is designed to study the performance of AKF_{als} as a function of the chosen block size N_d.

The estimates of the diagonal elements of $\mathbf{Q}$ and $\mathbf{R}$ are initialized with random values distributed uniformly in [0, 1]. The asymptotic Kalman gain corresponding to this initialization is used for filtering over the first block of length N_d to obtain innovations. These innovations are then used to formulate the least squares problem (5.35), the solution of which yields estimates for the six unknown noise covariances and an asymptotic Kalman gain K_1. In an offline application, this Kalman gain could be used to re-process the first block. For a real-time implementation, however, we instead use the asymptotic gain K_1 obtained from the first block to process the data in the second block. This approach is effective for achieving real-time performance provided that the jump transitions are not too large and the ramp characteristics are not too steep. The procedure is repeated recursively with the gain K_{p-1} from block $\boldsymbol{Y}(p-1)$ (defined in (5.19)) being used to process the data in block $\boldsymbol{Y}(p)$ and generate innovations.

Since the number of constraints in the least squares problem should be larger than the number of unknowns (six in this case), we set the number of autocorrelation lags considered by the AKF_{als} algorithm to $L_{\text{als}} = 10$. We performed simulations against the covariances shown in Fig. 5.4 with block sizes $N_d = 15$, 45, and 135, where 100 trials with different random initializations were run for each block size. The average estimated covariance values for the three different block sizes are shown as solid red lines in Fig. 5.4(a), (b), and (c).

In Fig. 5.4, we see that a small block size ($N_d = 15$) affords the opportunity to adapt quickly to abrupt nonstationary changes in the dynamics, but the estimation errors are generally large due to the limited observations available in each block. With the largest block size ($N_d = 135$), the algorithm is slower in adapting to nonstationary changes, especially those that occur in the middle of a block, but the estimation errors are generally much smaller than the ones with the small block size. Box plots describing the distribution of the estimation errors are given in Fig. 5.5, where we observe that, with increasing the block size, the median error decreases and the probability of a large estimation error diminishes as shown by the whiskers of the box plots. Overall, we find that AKF_{als} is able to cope reasonably well with both the jump and ramp nonstationarities depending on the block size.

5.4 Particle Filter-based Tracking

This section details a particle filtering-based tracking algorithm where histogram-based appearance learning is involved at each time step. We first present a target appearance model used for IR imagery, followed by the dynamic models used in IR tracking. The algorithm implementation is also discussed.

5.4.1 Dual foreground-background appearance model

We present a target model that involves the local statistics of both the target and its surrounding areas, as shown in Fig. 4.3. The use of background information for target track-

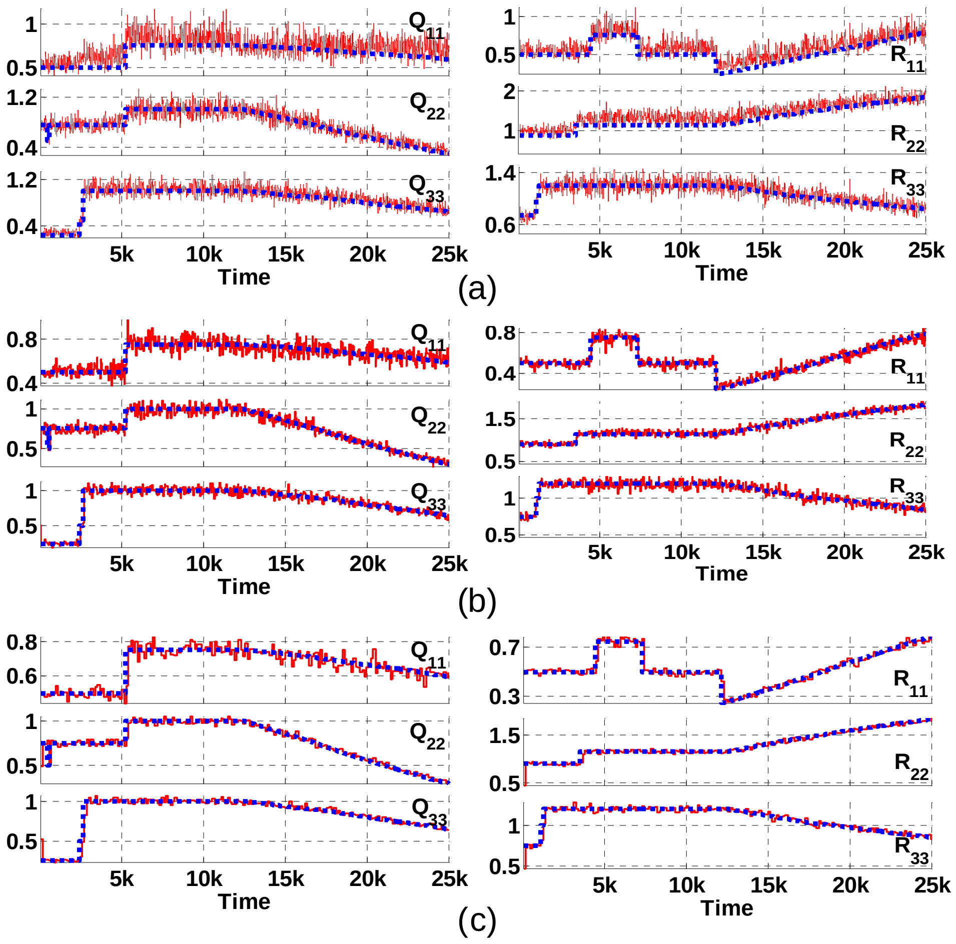


Figure 5.4: Simulation of AKF$_{\text{als}}$ against nonstationary noise statistics for three different block sizes. The dotted blue lines give the true values of the main diagonal entries of the process noise covariance matrix $\mathbf{Q}$ (left column) and measurement noise covariance matrix $\mathbf{R}$ (right column). The AKF$_{\text{als}}$ covariance estimates are shown as solid red lines for (a) $N_d = 15$, (b) $N_d = 45$, and (c) $N_d = 135$.

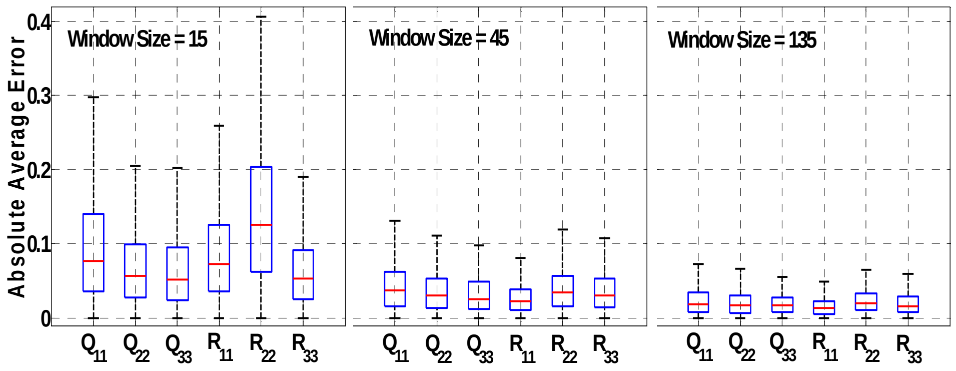


Figure 5.5: Box plots of AKF_{als} absolute estimation errors for three different block sizes against the nonstationary covariances shown in Fig. 5.4. Box bounds indicate the 25th and 75th percentile, while the centerline indicates the median error. The whiskers extend to approximately 99% coverage of the error data.

ing was discussed in many tracking algorithms, [82, 83, 116, 117]. In these methods, target tracking is performed on an intermediate classification image, usually called a confidence map [116], a likelihood image [117] or a weighted image [82], where each pixel is assigned with the probability belonging to the background or the foreground. Here we have a different point of view using background for target modeling. Our target model is motivated by the "hit-and-miss" morphological transform that uses both foreground and background for object detection. In practice, the background information is found to be of great assistance in localizing the target and determining its size. Specifically, the proposed target model involves four histograms to represent local statistics.

Let $\mathbf{y}_k$ represent the kth frame, and $\mathbf{x}_k$=$[x_k, y_k; s^x_k, s^y_k]$ the state to be estimated during target tracking, where (x_k, y_k) and (s^x_k, s^y_k) are the position (top-left corner) and size of the target area, respectively. As shown in Fig. 4.3, the target appearance, denoted by $\mathbf{G}(\mathbf{x}_k)$, is composed of four histograms: the foreground/background intensity $\mathbf{g}_A(\mathbf{x}_k)$/$\mathbf{g}_B(\mathbf{x}_k)$, foreground/background local standard deviation (stdev) $\mathbf{g}_C(\mathbf{x}_k)$/$\mathbf{g}_D(\mathbf{x}_k)$, which are extracted from $\mathbf{y}_k$ by using the kernel-based method in [85, 96] as follows.

Let $\boldsymbol{c}$ denote the location of the target's centroid determined from (x_k, y_k). The function

$\rho : \mathbb{R}^2 \rightarrow \{1, \cdots, m\}$ maps the intensity value of the pixel at position $\boldsymbol{r}$, to a bin index in the histogram. The probability of each bin $b = 1 \cdots N_b$ is computed as follows:

$$p_A^b(\mathbf{x}_k) = \lambda_1 \sum_{\boldsymbol{r} \varepsilon N_F(\mathbf{x}_k)} \mathbf{K}_{\boldsymbol{H}}(\boldsymbol{r} - \boldsymbol{c}) \delta[\rho(\boldsymbol{r}) - b], \tag{5.38}$$

where λ_1 is a normalization constant obtained such that $\sum_{b=1}^{N_b} p_A^b(\mathbf{x}_k) = 1$ and δ is the Kronecker delta function. We use the triangular kernel for $\mathbf{K}_{\boldsymbol{H}}$, where the width of the kernel is determined by the target size (s_k^x, s_k^y). Now we can define a m-dimensional vector $\mathbf{g}_A(\mathbf{x}_k) = [p_A^1(\mathbf{x}_k), \cdots, p_A^{N_b}(\mathbf{x}_k)]$ called the foreground histogram. In a similar manner the background area histogram may be obtained as $f_B(\mathbf{x}_k) = [p_B^1(\mathbf{x}_k), \cdots, p_B^{N_b}(\mathbf{x}_k)]$ where

$$p_B^b(\mathbf{x}_k) = \lambda_2 \sum_{\boldsymbol{r} \varepsilon N_B(\mathbf{x}_k)} \delta[\rho(\boldsymbol{r}) - b], \tag{5.39}$$

and λ_2 is a normalization constant obtained such that $\sum_{b=1}^{N_b} p_B^b(\mathbf{x}_k) = 1$. Similarly, we can obtain the foreground stdev histogram $\mathbf{g}_C(\mathbf{x}_k)$ and the background stdev histogram $\mathbf{g}_D(\mathbf{x}_k)$. Therefore given $\mathbf{x}_k$, the corresponding candidate region in frame $\mathbf{y}_k$ is characterized by $\mathbf{G}(\mathbf{x}_k)$ as follows,

$$\mathbf{G}(\mathbf{x}_k) = \{\mathbf{g}_A(\mathbf{x}_k), \mathbf{g}_B(\mathbf{x}_k), \mathbf{g}_C(\mathbf{x}_k), \mathbf{g}_D(\mathbf{x}_k)\}. \tag{5.40}$$

A reference target model learned from previous frames is also available that is composed of four histograms, i.e., $\mathbf{F}_{k-1} = \{\mathbf{f}_{A,k-1}, \mathbf{f}_{B,k-1}, \mathbf{f}_{C,k-1}, \mathbf{f}_{D,k-1}\}$. This reference model is updated online and used to evaluate any given candidate area in frame k represented by $\mathbf{G}(\mathbf{x}_k)$ as:

$$\mathrm{D}(\mathbf{G}(\mathbf{x}_k), \mathbf{F}_{k-1}) = \sum_{z \in Z} v_z \cdot \mathrm{d}(\mathbf{g}_z(\mathbf{x}_k), \mathbf{f}_{z,k-1}), \tag{5.41}$$

where $Z = \{A, B, C, D\}$ and d is defined in (5.6). v_z is used to adjust the significance of four histograms. In this work, all four histograms are given equal importance during tracking. It can also be adaptively selected as discussed in [96].

5.4.2 Target Dynamics

Two dynamic models are needed for IR tracking, one each for the position and size. In most IR sequences, the target predominantly exhibits relatively stable ground motion accompanied by strong ego-motion of the airborne sensor. Previous works in [27, 32] used a separate global motion model to compensate the sensor ego-motion. Inspired by [26], we use an adaptive motion model to capture both ground motion and ego-motion for IR tracking:

$$\mathbf{p}_k = \mathbf{p}_{k-1} + \mathbf{c}_k v_k, \tag{5.42}$$

where $\mathbf{p}_k = [x_k, y_k]$, $\mathbf{c}_k \propto \mathbf{E_n}[\triangle \mathbf{p}_k]$ and $v_k \sim \mathcal{N}(0, I)$. $\mathbf{E_n}[\triangle \mathrm{x}_k]$ is the velocity (in the image plane) estimated over the past n frames. In essence, this model controls the search area in proportion with the observed target velocity.

To account for the magnification effect of an infrared target, we need a dynamic model to increase or decrease the target size at each time step. Also, this model needs to control the magnification change to be proportional with the previous size. Thus the dynamic model for the size vector s_k is defined as

$$\mathbf{q}_k = D\mathbf{q}_{k-1}, \tag{5.43}$$

where $\mathbf{q}_k = [s_k^x, s_k^y]$ and $D \sim U[1-\epsilon, 1+\epsilon]$. In the case of dealing with the AMCOM closure sequences, we chose $\epsilon = 0.2$ in this work.

5.4.3 Tracking Algorithm

We develop a SIR (sequential importance re-sampling)-based tracking algorithm adapted from [84] that involves three steps: *particle propagation*, *particle evaluation*, and *appearance learning*. The complete tracking algorithm is shown in Table 5.2. First, particles are drawn according to the state dynamics defined in (5.42) and (5.43). Second, particle weights are computed by the likelihood function $p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{F}_{k-1})$, which is defined based on the distance measure in (5.41) as

$$p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{F}_{k-1}) \propto \exp(\lambda \cdot \mathrm{D}(\mathbf{G}(\mathbf{x}_k), \mathbf{F}_{k-1})), \tag{5.44}$$

where λ is a constant to control the sensitivity of the likelihood function. $\mathbf{G}(\mathbf{x}_k)$ and $\mathbf{F}_{k-1}$ are the observed target appearance given hypothesis state $\mathbf{x}_k$ and the previous reference appearance, respectively. Then after weight normalization and re-sampling, the state estimate is updated, followed by online appearance learning discussed in Section 5.3.3.

Table 5.2: Pseudo-code of the particle filter algorithm with online appearance learning for target tracking.

```
• Initialization: Draw $\mathbf{x}_0^j \sim N(X_0, 1)$, and set $\mathbf{F}_0 = \mathbf{G}(X_0)$,
  where $X_0$ is the ground truth of the state in the initial frame.
• For $k$ = 1,$\cdots$, $T$ (number of frames)
    1. For $j$ = 1,$\cdots$ , $N_p$ (number of particles)
        1.1 $\mathbf{x}_k^j \sim p(\mathbf{x}_k^j|\mathbf{x}_{k-1}^j)$ using (5.43) and (5.42)
        1.2 Compute $w_k^j = p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{F}_{k-1})$ using (5.44)
      End
    2. Normalize the weights such that $\sum_{j=1}^{N_p} w_k^j = 1$.
    3. Compute the mean of the states $\widehat{\mathbf{x}}_k = \sum_{j=1}^{N_p} w_k^j \mathbf{x}_k^j$.
    4. Set $\mathbf{x}_k^j$ =resample($\mathbf{x}_k^j, w_k^j$).
    5. Update reference model $\mathbf{F}_k$ based on state estimate
      $\widehat{\mathbf{x}}_k$ according to Table 5.1.
• End
```

5.5 Detecting occlusions and track losses

The task of excluding outlier observations is quite common in Kalman Filtering applications and is referred to as gating. The gating criterion is often based on the statistical properties of the residues. In [36] an error norm was defined on the residues to prevent outlier pixels from corrupting the appearance information. In [37] a hypothesis test was defined based on the standard deviation of the residues. This was possible because residues from each bin of the histogram were assumed to be statistically similar. Other rigorous methods that employ more sophisticated statistical tests also exist [118].

In this work, we make use of the fact that the residues computed in (5.12) for each histogram bin b follows a Gaussian distribution $r_k^b \sim N(0, \sigma_{vb}^2)$. Note that the residues provide information about the mismatch between the predicted and observed bin value. In our model, since each of the histogram bins is characterized by different distributions we associate with each bin an uncertainty term U_b defined as

$$U_b = \frac{1}{\sqrt{2\pi\sigma_{vb}^2}} \int_{-r_k^b}^{r_k^b} exp(\frac{-x^2}{2\sigma_{vb}^2})dx. \tag{5.45}$$

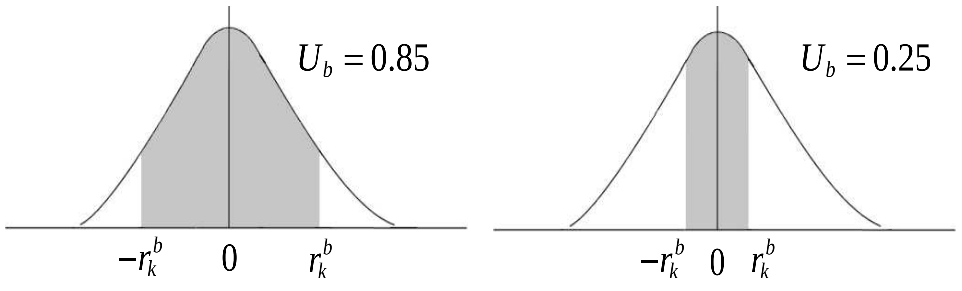


Figure 5.6: Illustration of the uncertainty associated with two different residual values for the underlying distribution $N(0, \sigma_{vb}^2)$.

Ideally if the observed appearance exactly matches the reference model (r_k^b=0), then there is very little chance of the observation being erroneous. The uncertainty values for two different r_k^bs for the same underlying distribution are shown in Fig. 5.6. We see that a value of r_k^b closer to zero results in lower uncertainty and a value farther away leads to higher uncertainty. During the filter operation we refrain from updating the appearance model when the average uncertainty over the non-zero bins rises above a threshold of 0.7 and declare a temporary "track loss". A prolonged track loss is indicative of an occlusion or movement of the target outside the sensor view.

Once in lost mode, in each subsequent frame, attempts are made to reacquire the target by using a histogram matching based detector. The detection is performed in window area around the last known target position and an average of the last few appearance histograms is used as the reference. Being dependent on only the last known appearance can be problematic in cases of partial occlusion. This is because a significant portion of the target appearance may already be lost before a track loss is detected. The target is declared as found and normal tracking resumes when the detected candidate region has a uncertainty value below the threshold 0.7. If no acceptable candidates are found within 100 frames of a temporary track loss, a complete "track loss" is declared and tracking is terminated. This simple methodology works very well in recovering the target after short periods of occlusions and scene absence and is discussed further in the experiments section. In case of a prolonged absence from view or occlusions, when the target reappears it may significantly differ from its previous known appearance. This makes it difficult for the detector to identify the target with high certainty and may require re-initialization.

5.6 Experimental Results

Our tracking algorithm was tested on the AMCOM IR dataset. This dataset comprises of sequences in grayscale format (128$\times$128 pixels). Ground truth information about the target position, size and type is available in the dataset and serves as a benchmark for performance evaluation. Ten representative IR sequences used in the experiment are given in Table 5.3. These sequences exemplify the challenges of IR tracking such as poor target visibility, strong ego-motion, small targets, size variations, dust clouds, significant clutter and background noise, etc.

Table 5.3: List of sequences used in experiments.

	Frame			Size	
Sequences	Starting frame	Ending frame	Length	Starting size	Ending size
LW-15-NS	20	270	250	5 x 8	16 x 16
LW-17-01	1	350	350	5 x 8	16 x 29
LW-21-15	235	635	400	3 x 4	10 x 10
LW-14-15	1	225	225	4 x 5	23 x 19
LW-22-08	50	300	250	5 x 8	17 x 24
LW-20-18	120	420	300	4 x 7	10 x 17
LW-18-17	1	190	190	5 x 9	11 x 25
LW-19-06	40	260	220	3 x 4	6 x 11
MW-14-10	1	450	450	6 x 11	12 x 28
LW-20-04	10	360	350	3 x 4	12 x 15

5.6.1 Experimental setup

Three appearance learning algorithms, namely HS, $\mathrm{AKF}_{\mathrm{cov}}$ and $\mathrm{AKF}_{\mathrm{als}}$ are integrated with the same tracking algorithm given in Table 5.2. It is worth mentioning that all three algorithms share the same linear filtering form defined in (5.4). HS determines ξ_k according to histograms similarity, while $\mathrm{AKF}_{\mathrm{cov}}$ and $\mathrm{AKF}_{\mathrm{als}}$ uses the Kalman gain. The detailed setting of the three tracking algorithms is listed in Table 5.4. In practice, the Kalman filter-based appearance learning algorithms were applied only to the two intensity histograms ($\mathbf{f}_A$ and $\mathbf{f}_B$). Because the dynamics of stdev histograms do not have a well-defined structure, the stdev histograms ($\mathbf{f}_C$ and $\mathbf{f}_D$) in all cases were updated using the HS method.

In addition to the tracking errors, we adopt an overlap metric proposed in [119] to quantify the degree of overlap between the tracking gate with the actual target area. Let A

Table 5.4: Description and value of the experimental parameters

Variables	Description	Values
$N_b^{(1)}$	bin number of the intensity histogram	32
$N_b^{(2)}$	bin number of the stdev histogram	16
L_{cov}	number of frames used for AKF$_{\text{cov}}$ in (5.16)	3
N_d	block size in frames	7
L_{als}	number of autocorrelation lags	5
c_k	dynamics of position used in (5.42)	$3\mathbf{E_n}[\triangle \mathrm{x}_k]$
N_p	number of particles used for tracking	200

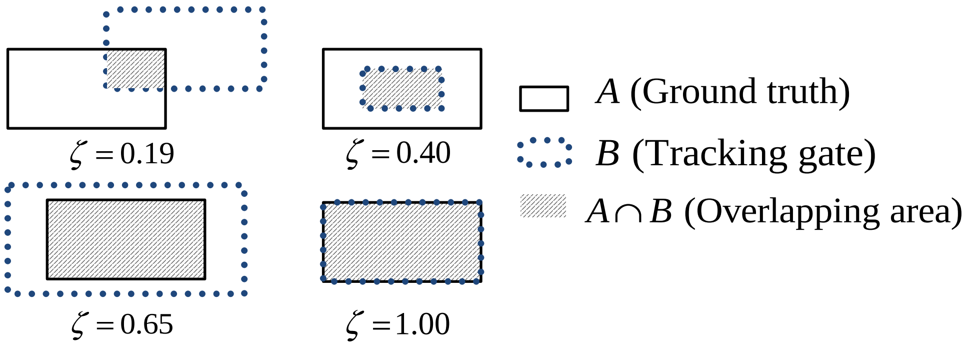


Figure 5.7: Illustration of the overlap metric for a few different tracking cases.

and B represent the tracking gate and the ground-truth bounding box respectively, then the overlap ratio ζ is defined as

$$\zeta = \frac{\#(A \cap B) \times 2}{\#(A) + \#(B)}, \tag{5.46}$$

where $\#$ is the number of pixels. A few representative examples of the metric are shown in Fig. 5.7.

5.6.2 Experimental Analysis

Three IR tracking algorithms (50 Monte Carlo runs) were evaluated and compared in terms of their performance of appearance learning (Fig. 5.8), the overlap metric ζ (Fig. 5.9) and the tracking error (Fig. 5.11 and Table 5.6).

5.6.2.1 Appearance learning

As shown in Fig. 5.8, it can easily be observed that the results of $\mathrm{AKF}_{\mathrm{als}}$ closely match the ground-truth. Closer examination reveals that HS and $\mathrm{AKF}_{\mathrm{cov}}$ result in the histograms that slowly deviate or "drift" from the true ones. This is clearly evident in Fig. 5.8(c), where the intensity variation in the latter part of the sequence (around frame 300) is not captured by HS and $\mathrm{AKF}_{\mathrm{cov}}$. Therefore, the tracker includes a large portion of the background into the tracking gate as seen in frames 320, 360 of Fig. 5.11 (c).

5.6.2.2 Overlap metric

The improvements of appearance learning can be further reflected by the overlap metric in Fig. 5.9(a)(b), which compares ζ_{als}, ζ_{cov} and ζ_{HS} in pairwise. For example, the improvement of $\mathrm{AKF}_{\mathrm{als}}$ over $\mathrm{AKF}_{\mathrm{cov}}$ or HS can be demonstrated by seeing most data points are above the diagonal lines. The comparable result of $\mathrm{AKF}_{\mathrm{als}}$ and $\mathrm{AKF}_{\mathrm{cov}}$ in sequence LW-22-08 is also shown in the similar appearance learning performance in Fig. 5.8(e) where the histogram-based appearance lacks strong modes and has widespread and small bin values. The average value of ζ corresponding to different algorithms is given in Table 5.5. The $\mathrm{AKF}_{\mathrm{als}}$ method has the largest value that indicates its superior performance of target tracking when compared to the other two algorithms.

5.6.2.3 Tracking error

Table 5.6 provides quantitative results of the tracking performance. In most cases, $\mathrm{AKF}_{\mathrm{als}}$ produces the least errors in terms of both position and size. The HS approach loses track of the target in sequences LW-20-18 (6 runs) and LW-19-06 (2 runs) as indicated by the large errors. The $\mathrm{AKF}_{\mathrm{cov}}$ also loses track of the target in the sequence LW-20-18 (1 run) due to the high similarity between foreground and background. More visual comparisons are shown in Fig. 5.11. We can see that $\mathrm{AKF}_{\mathrm{als}}$ offers the best position and size estimation except sequence LW-22-08, where $\mathrm{AKF}_{\mathrm{cov}}$ is slightly better due to the lack of well defined

Table 5.5: The overlap metric values of the three tracking algorithms.

Sequences	HS	$\mathrm{AKF}_{\mathrm{cov}}$	$\mathrm{AKF}_{\mathrm{als}}$
LW-15-NS	0.669	0.707	**0.714**
LW-17-01	0.547	0.596	**0.720**
LW-21-15	0.601	0.578	**0.620**
LW-14-15	0.676	0.682	**0.708**
LW-22-08	0.751	**0.770**	0.758
LW-20-18	0.689	0.753	**0.758**
LW-18-17	**0.704**	0.702	0.703
LW-19-06	0.670	0.685	**0.713**
MW-14-10	**0.802**	0.797	0.799
LW-20-04	0.715	0.711	**0.720**
Average	0.682	0.698	**0.721**

structure in the histogram-based appearance, as shown in Fig. 5.8(e).

5.6.3 Tracking performance of covariance descriptor

We also tested the covariance descriptor for IR tracking. The covariance descriptor was found to be robust and effective for object tracking in optical images and plays an important role in several state-of-the-art tracking algorithms [120, 121]. It was first proposed in [122] for object detection. This descriptor has several advantages: (1) it is able to fuse together many different features; (2) it is invariant to illumination conditions and rotation, contains both statistical and spatial information and is fast to compute; (3) it can be updated incrementally and systematically by some manifold learning methods. In IR tracking, the covariance descriptor involves local intensity, stdev, gradient, orientation and Laplacian information of the target area. This descriptor was combined with the particle filter whose dynamics were described in Section 5.4. The tracking results of using the covariance descriptor are shown in Fig. 5.10 where no learning is involved. It is observed that the covariance tracker is able to maintain a reasonable track of the target in LW-17-01, but fails to track the dark target in LW-15-NS. In both sequences, the tracker encounters difficulty in size estimation. The small size of the target, weak texture and absence of color significantly reduce the effectiveness of the covariance descriptor for tracking small targets in IR images.

5.6.4 Further Discussion

In summary, the HS method is usually encumbered by the drifting problem during incremental appearance learning. The $\mathrm{AKF}_{\mathrm{cov}}$ method, which assumes the same noise statistics for all histogram bins and estimates only the process noise without considering PSD conditions, results in a suboptimal Kalman gain. Its performance is marginally better than that of HS. The $\mathrm{AKF}_{\mathrm{als}}$ algorithm, which estimates both process and observation noises with PSD conditions for each individual bin in the histogram, is able to follow the modes and variations of the histogram during tracking and supports effective appearance learning. However, when a histogram lacks some strong modes and has widespread and small bin values, such as LW-22-08 and MW-14-10, all three methods are comparable. This is mainly because the poor structure of the histogram evolution may invalidate Kalman filter assumptions, while HS is still effective to incorporate the most recent tracker's observation for appearance learning when the histogram is less well defined. This justifies the use of HS for learning the stdev histograms which normally have weak structures.

5.6.5 Experiments on the VIVID dataset

In the VIVID dataset the targets are larger compared to the AMCOM dataset and the foreground information is robust enough to represent the target. Therefore we predominantly depend on the foreground information for tracking by setting the histogram importance as $v_{fi} = 0.45$, $v_{fi} = 0.05$, $v_{fv} = 0.45$ and $v_{bv} = 0.05$ in (5.41). The sequences tested

Table 5.6: Mean error of the state variables over averaged over the length of the sequence from 50 Monte Carlo runs using three different algorithms.

Algorithm	HS				$\mathrm{AKF}_{\mathrm{cov}}$				$\mathrm{AKF}_{\mathrm{als}}$			
Sequence	x	y	s^x	s^y	x	y	s^x	s^y	x	y	s^x	s^y
LW-15-NS	1.019	1.817	1.906	2.732	0.860	1.511	1.644	2.396	**0.801**	**1.461**	**1.423**	**2.339**
LW-17-01	2.406	3.415	2.104	**3.016**	2.145	3.005	2.101	3.163	**1.213**	**2.110**	**1.376**	3.033
LW-21-15	0.970	1.653	**2.624**	2.941	1.135	1.812	2.799	3.113	**0.893**	**1.300**	2.786	**2.575**
LW-14-15	**0.889**	0.815	3.160	2.137	0.932	**0.787**	2.981	2.157	1.099	0.801	**2.660**	**1.787**
LW-22-08	**1.167**	0.868	1.684	**2.049**	1.202	0.843	**1.070**	2.232	1.200	**0.839**	1.363	2.175
LW-20-18	3.230	1.831	1.657	1.953	0.901	1.095	**1.307**	1.766	**0.599**	**1.084**	1.439	**1.754**
LW-18-17	**1.269**	1.722	**0.733**	2.949	1.303	1.838	0.859	2.611	1.425	**1.679**	1.087	**2.252**
LW-19-06	1.977	1.545	1.566	1.544	0.797	0.764	1.681	1.454	**0.694**	**0.709**	**1.536**	**1.279**
MW-14-10	**0.628**	0.789	1.648	1.691	0.756	0.806	1.638	1.789	0.775	**0.778**	**1.629**	**1.607**
LW-20-04	0.702	0.954	**0.940**	1.528	0.697	0.937	1.071	1.614	**0.688**	**0.907**	1.006	**1.357**
Average	1.426	1.541	1.802	2.254	1.073	1.340	1.715	2.230	**0.939**	**1.167**	**1.630**	**2.016**

are typical of aerial surveillance videos and are affected by ego-motion of the sensor, occlusion by foliage, targets exiting scene and reappearing. Due to the absence of explicit ground truth information we only present visual evidence of the tracking performance. In all the sequences, the targets were manually initialized with an appropriate bounding box. We compare the performance of the tracking algorithm with $\mathrm{AKF}_{+\mathrm{tld}}$ and without $\mathrm{AKF}_{\mathrm{als}}$ track-loss detection on a few representative sequences with and without track-loss detection.

A few sample frames from three different sequences, the uncertainty associated with the $\mathrm{AKF}_{\mathrm{als+tld}}$ tracker and the corresponding foreground appearance variations are shown in Fig. 5.12, Fig.5.13 and Fig. 5.14 respectively.In SEQ1 corresponding to the top row of Fig. 5.12, the target is occluded by some trees around frame 50 and reemerges from behind them around frame 65. Both the $\mathrm{AKF}_{\mathrm{als+tld}}$ and $\mathrm{AKF}_{\mathrm{als}}$ algorithms perform similarly till the time of occlusion. During the period of occlusion the uncertainty associated with the appearance increases above the set as shown in Fig.5.13(a). Therefore, the $\mathrm{AKF}_{\mathrm{als+tld}}$ tracker stops all update to the appearance and goes into detection mode. The $\mathrm{AKF}_{\mathrm{als}}$ tracker on the other hand continues to learn new appearance corresponding to the background. By the time the target is occlusion free, the detector is able to locate the target with acceptable level of certainty and the $\mathrm{AKF}_{\mathrm{als+tld}}$ tracker begins tracking the target. The $\mathrm{AKF}_{\mathrm{als}}$ tracker by this time, has learnt the appearance of the background and loses track of the target. From Fig. 5.14 we can observe the appearance of significant peaks in the appearance histogram of the $\mathrm{AKF}_{\mathrm{als}}$ tracker as it shifts its focus toward the background. The $\mathrm{AKF}_{\mathrm{als+tld}}$ tracker suspends all updates until a reliable target is found, this is indicated by the missing values between the occlusion frames in Fig. 5.14 (a). In this sequence, the increased uncertainty around frames 30, 80 and 90 maybe attributed to the lens halo effect interacting with the target and can be clearly seen in the observed images.

In SEQ2, the target moves outside the sensors view range for a few frames and reenters the scene. By frame 195 the target is only partially visible and this is reflected in the increased uncertainty around those frames in Fig.5.13 (b). A subsequent exit from the scene triggers the detector mode of the $\mathrm{AKF}_{\mathrm{als+tld}}$ tracker. The detector is able to quickly detect the target upon re-entry and passes it on to the tracker when a uncertainty value below the threshold is achieved. The $\mathrm{AKF}_{\mathrm{als}}$ tracker slowly deviates from the target and begins to concentrate on the background as shown in Fig. 5.14(b).

In SEQ3, the target becomes absent from the scene for an extended period of time before re-entering. The exit of the target around frame 263 is easily picked up by the uncertainty indicator. Note the gradual increase in uncertainty corresponding to the slow exit of the target from the scene. When the target re-enters the scene, the detector is quick to move on to the true target, however, the uncertainty associated with it still remains high as seen in Fig.5.12 and Fig.5.13 (c). This suggests that, though the uncertainty criterion is robust enough to detect track-losses, it may not be a strong indicator of track-acquisition. In cases of long absences it may be necessary to re-initialize the tracker with a more robust detector using feature descriptors more complex than intensity histograms or manually.

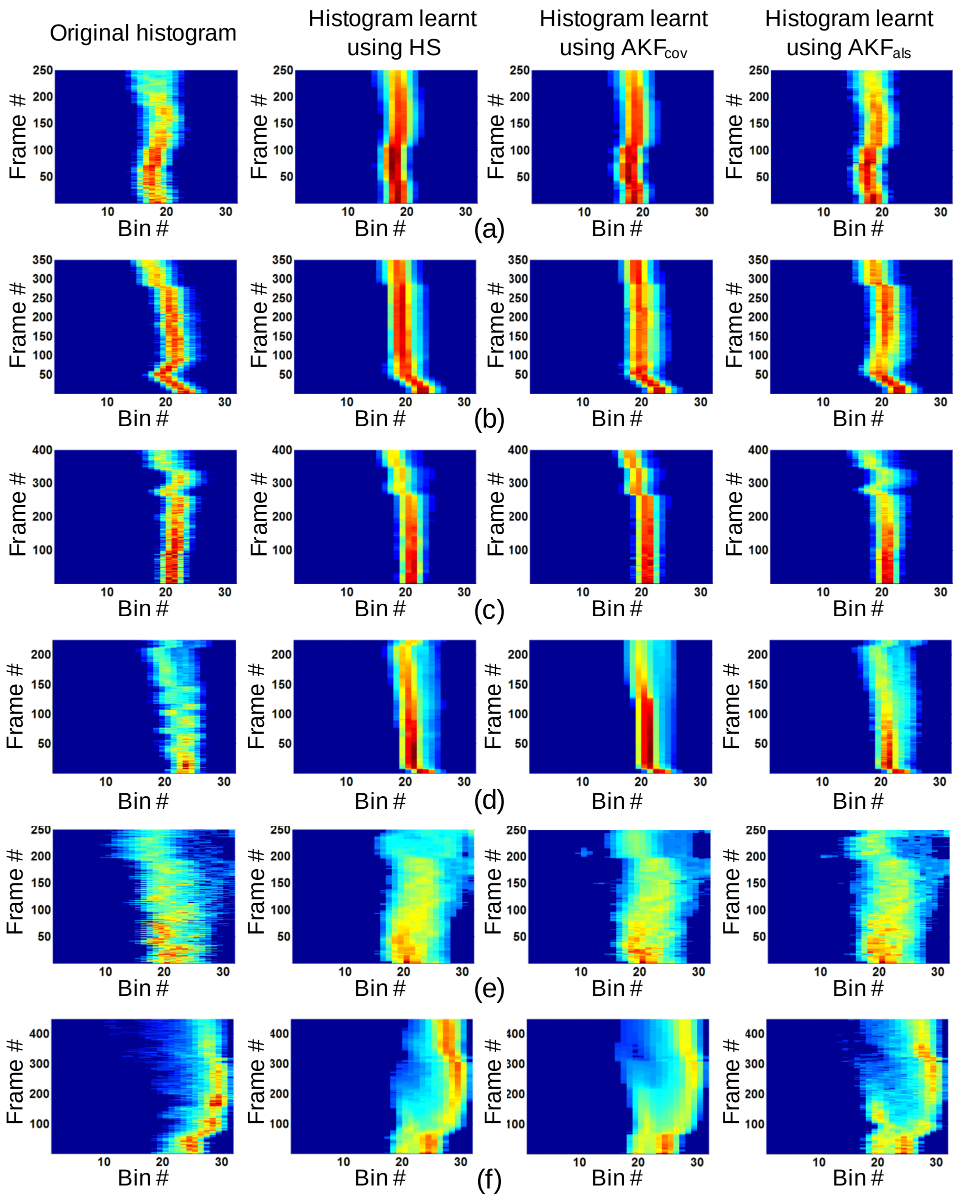


Figure 5.8: Comparison of appearance learning for the AMCOM sequences: (a) LW-15-NS (b) LW-17-01 (c) LW-21-15 (d) LW-14-15 and (e) LW-22-08 and (f) MW-14-10.

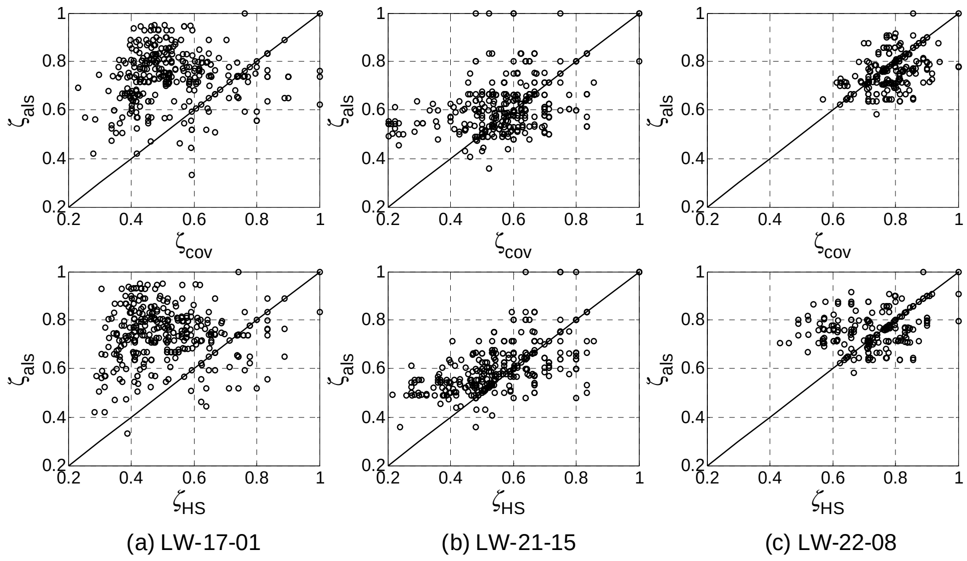


Figure 5.9: Pairwise overlap comparison ζ_{ALS} vs. ζ_{COV} (top row) and ζ_{ALS} vs. ζ_{HS} (bottom row) for LW-17-01 (a), LW-21-15 (b) and LW-22-08 (c).

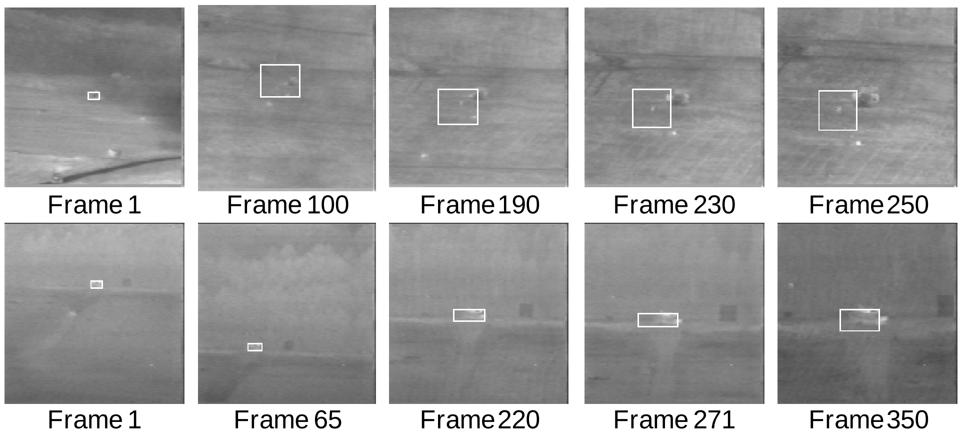


Figure 5.10: Tracking results for two AMCOM sequences using the covariance tracker. Top: LW-15-NS and Bottom: LW-17-01.

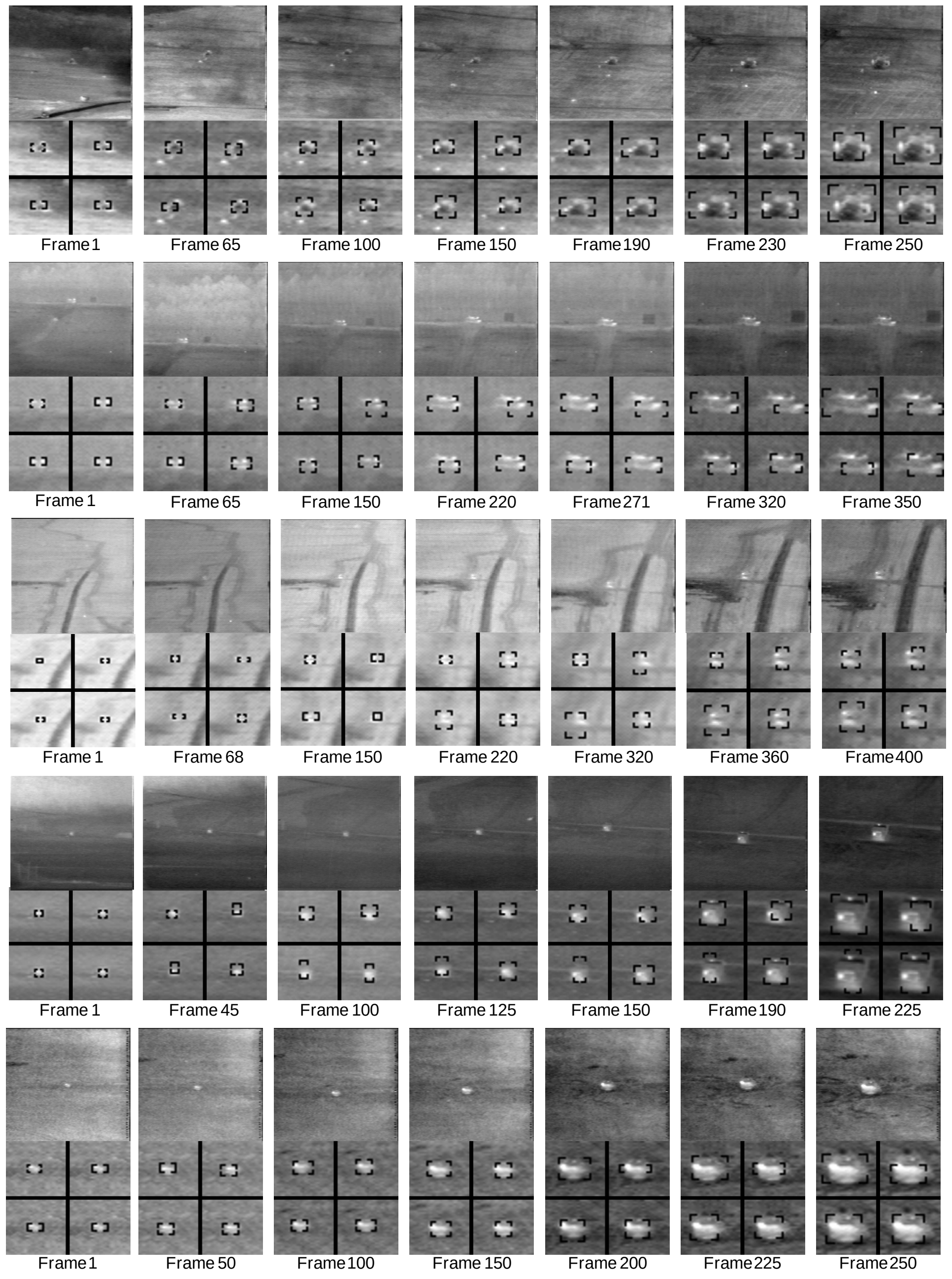


Figure 5.11: Tracking results of the three algorithm on five AMCOM sequences. The top row of each image shows the observed frame and and the bottom row depicts the tracking gates corresponding to the Ground truth (Top-Left), HS (Top-Right), $\mathrm{AKF}_{\mathrm{cov}}$ (Bottom-Left), $\mathrm{AKF}_{\mathrm{als}}$ (Bottom-Right). The sequences from top to bottom are LW-15-NS, LW-17-01, LW-21-15, LW-14-15 and LW-22-08

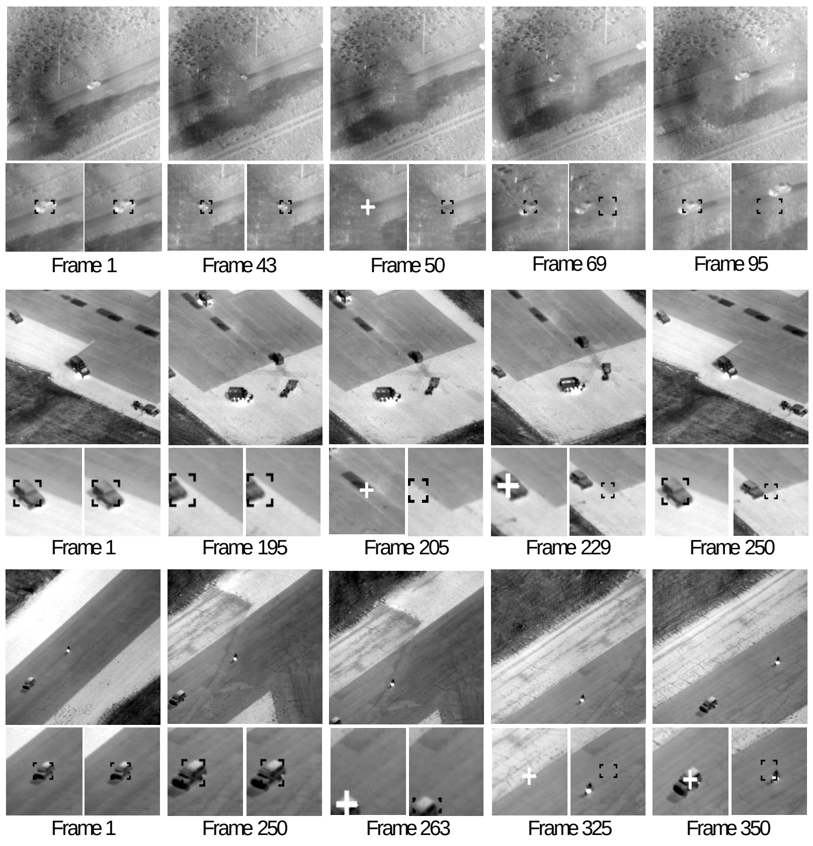


Figure 5.12: A few sample frames from three different sequences are shown in each of the rows. In each sub-image, the top image represents the observed frame. Bottom left is the result of $\mathrm{AKF_{als+tld}}$ and bottom right: $\mathrm{AKF_{als}}$. The black bounding boxes represent the tracking result. The white "+" sign represents the output of the detector in the $\mathrm{AKF_{als+tld}}$ method.

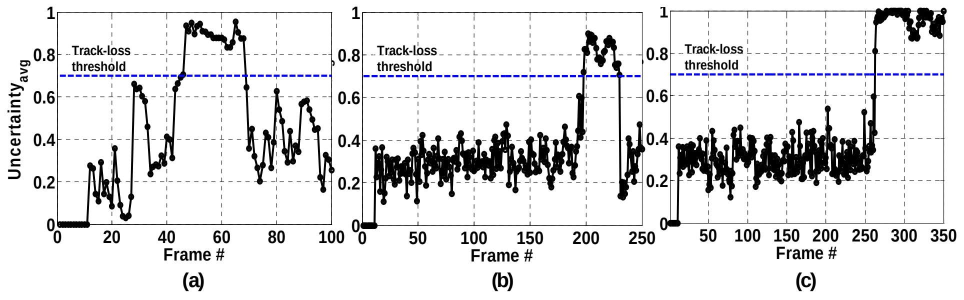


Figure 5.13: The appearance uncertainty associated with the foreground histogram by the $\mathrm{AKF_{als+tld}}$ tracker for (a)SEQ1, (b)SEQ2 and (c)SEQ3.

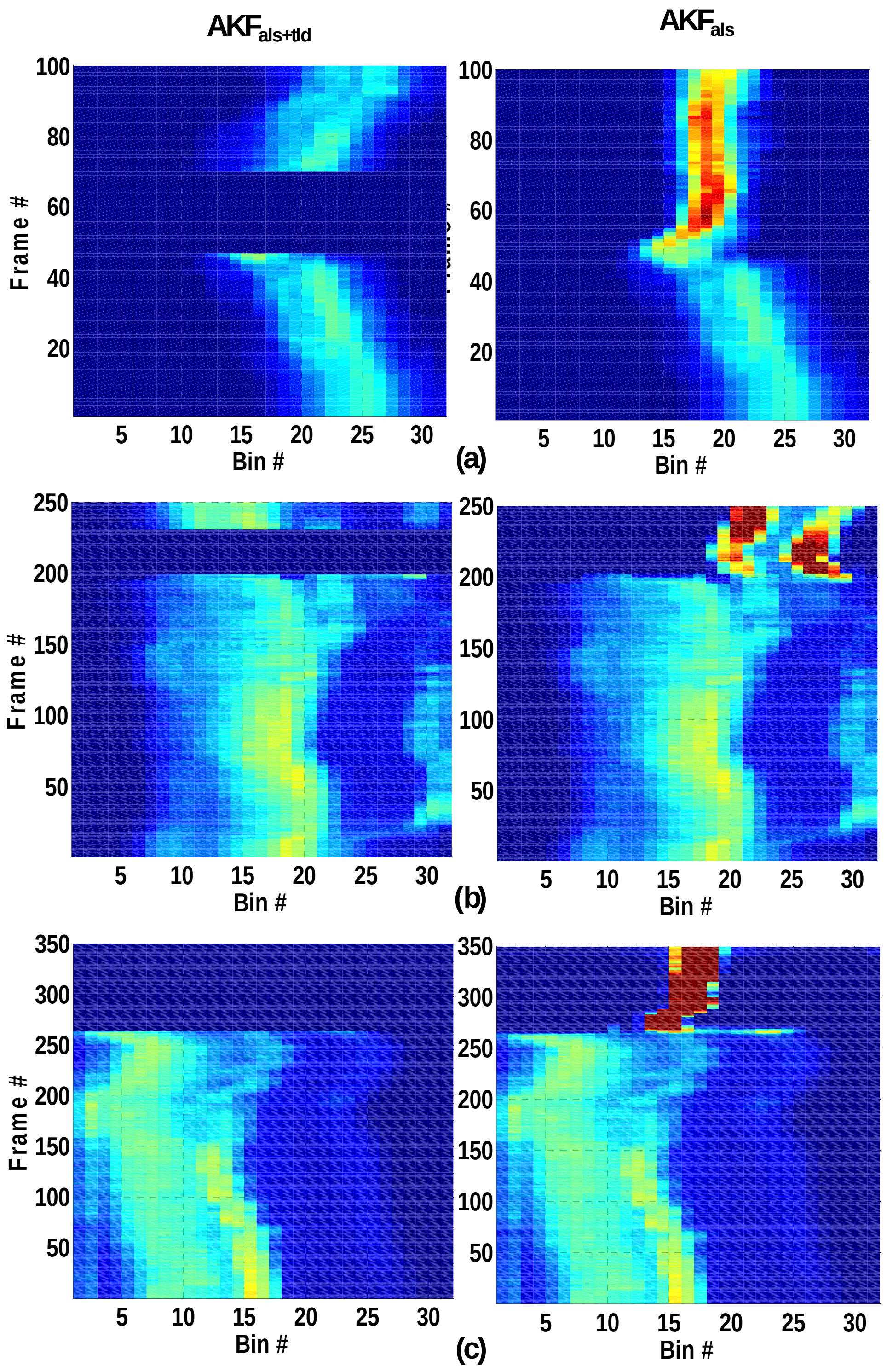


Figure 5.14: Foreground histogram variation associated with the $\mathrm{AKF}_{\mathrm{als+tld}}$ and $\mathrm{AKF}_{\mathrm{als}}$ trackers for (a)SEQ1, (b)SEQ2 and (c)SEQ3.

CHAPTER 6

Integrated Tracking and Recognition [1]

6.1 Background

Joint tracking and recognition is a challenging issue that is of great interest to both computer vision and signal processing communities. Though the nature of sensor measurements and research focuses are quite different, they share some common issues when the problem is formulated as a dynamic state-space system involving observation and system models. Traditionally, vision research focuses on the development of powerful appearance models [31, 67] (i.e. observation model), while signal processing research places emphasis on the system model by focusing on the dynamics of maneuvering targets [69, 70]. Recently, there is a trend to combine both appearance and motion cues into a joint tracking and recognition flow [60, 62]. We address the problem of joint target tracking and recognition by incorporating both appearance and motion information via two generative models. *In this chapter, we exploit the synergy between the two cues by systematically fusing them in an integrated probabilistic framework that enables their mutual interaction for joint tracking and recognition.*

Target appearance modeling is vital in any tracking and recognition algorithm since target appearance varies widely with pose changes. Most existing methods [31, 123] only deal with limited pose variations as they often use templates or other image features to accommodate appearance variability. In addition, such models need continuous update to ensure robust tracking and recognition [123]. To overcome these issues we suggest the use of a nonlinear tensor-based generative model similar to the one proposed in [67] that

[1]The work presented in this chapter was done in collaboration with Dr Xin Fan

can synthesize a target signature given the target type and an arbitrary pose. In addition to aiding the tracker by accounting for inter-frame appearance changes, this model also facilitates recognition by generating distinct type-specific appearances with any pose.

Traditionally, multiple motion models have been widely adopted to accommodate maneuvering actions in target tracking [68]. In reality, different targets would be equipped with customized engines and mechanical systems that generate distinct motion patterns. This inspires researchers to develop multiple type-dependent motion models to achieve joint target tracking and recognition [69, 70]. However, these approaches require sensors giving direct kinematics measurements, e.g., radar, and do not consider the target appearance. We employ target-dependent generative motion models coupled with the generative appearance model for joint tracking and recognition.

We develop an integrated graphical model to encapsulate all relevant factors, i.e., the target type, the motion and appearance models, as well as their cause-and-effect dependencies. The type-dependent motion and appearance models work synergistically for tracking and recognition, unlike the approaches in [60, 62] where motion models are adaptive to appearance changes but have no direct impact on recognition. We resort to a particle filter based inference method, in which a Kalman filter using the motion cues is embedded to improve the identity proposal generation as well as kinematics estimation. The experiments on simulated infrared sequences demonstrate the advantages and potential of the proposed approach for joint tracking and recognition.

6.2 Problem formulation

A graphical model is used to integrate all the factors along with their conditional dependencies into a probabilistic framework as shown in Fig.6.1, where three latent variables are estimated, i.e., a discrete valued target identity variable $I_t \in \{1, ..., N_T\}$, the continuous valued zero-order kinematics $\boldsymbol{X}_t$ (position $[p_x, p_y]'$ and pose ϕ) and the continuous valued first order kinematics $\boldsymbol{K}_t$ (linear velocity v and angular velocity w_v). $\boldsymbol{Z}_t$ denotes the current observation frame and $\boldsymbol{S}_t$ a intermediate variable represents the hypothesized target appearance. Here joint tracking and recognition simplifies to an estimate of the posterior probability $p(I_t, \boldsymbol{X}_t, \boldsymbol{K}_t|\boldsymbol{Z}_t)$.

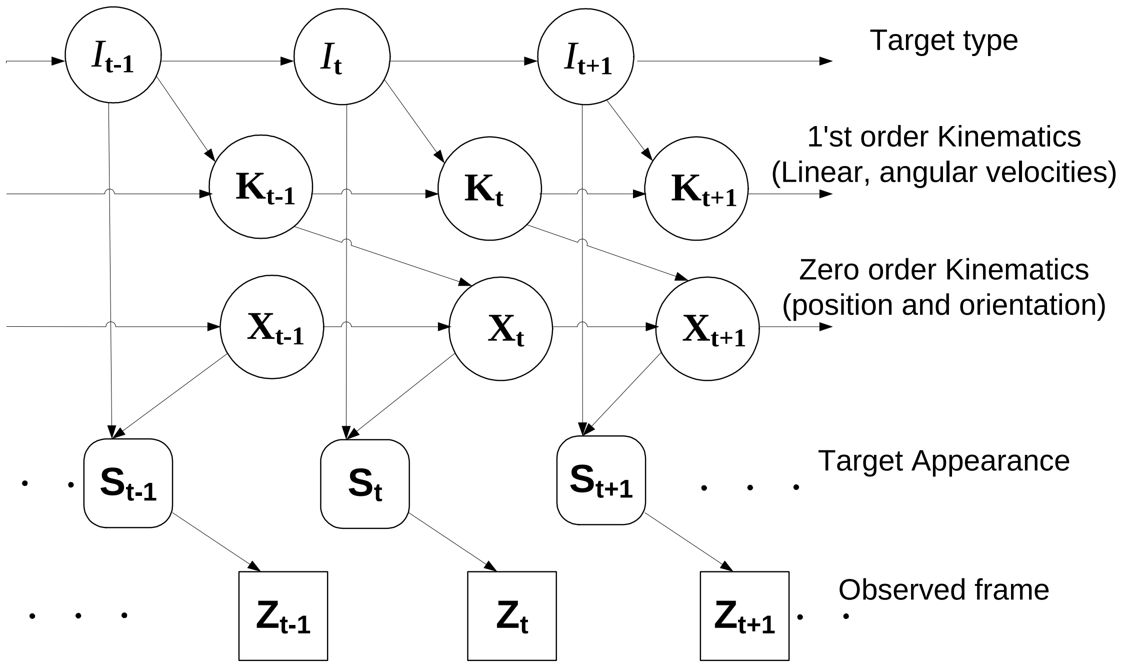


Figure 6.1: Graphical model for integrated tracking and recognition.

Fig.6.1 implies the probabilistic dependencies of the joint distribution $p(I_t, \boldsymbol{K}_t, \boldsymbol{X}_t, \boldsymbol{Z}_t|\Theta_{t-1})$, where Θ_{t-1} denotes the previous observation and latent states. The arcs between I_t,$\boldsymbol{X}_t$ and $\boldsymbol{S}_t$ as well as those between I_t,$\boldsymbol{K}_t$ and $\boldsymbol{X}_t$ show the cause-and-effect relationships that correspond to the generative appearance and motion models respectively. Specifically, the appearance of the target $\boldsymbol{S}_t$ in a given frame is dependent on its identity, position and pose with respect to the camera. The change in pose and positions between frames is dependent on the linear and angular velocities of the target which in turn depend on its identity. From the conditional independence in Fig.6.1, the joint probability can be factorized as:

$$p(I_t, \boldsymbol{K}_t, \boldsymbol{X}_t, \boldsymbol{Z}_t|\Theta_{t-1})$$
$$= p(\boldsymbol{Z}_t|I_t, \boldsymbol{X}_t)p(I_t, \boldsymbol{X}_t, \boldsymbol{K}_t|\Theta_{t-1}), \qquad (6.1)$$

where $p(\boldsymbol{Z}_t|I_t, \boldsymbol{X}_t)$ is the observation likelihood dependent on the appearance model. Applying the conditional independence between two time steps, the last term in 6.1 is:

$$p(I_t, \boldsymbol{X}_t, \boldsymbol{K}_t | \Theta_{t-1})$$
$$= p(\boldsymbol{X}_t | \boldsymbol{X}_{t-1}, \boldsymbol{K}_{t-1}) p(\boldsymbol{K}_t | I_t, \boldsymbol{K}_{t-1}) p(I_t | I_{t-1}), \tag{6.2}$$

where $p(\boldsymbol{X}_t | \boldsymbol{X}_{t-1}, \boldsymbol{K}_{t-1})$ and $p(\boldsymbol{K}_t | I_t, \boldsymbol{K}_{t-1})$ are related to the generative motion model and $p(I_t | I_{t-1})$ is defined by a target type transition matrix. Due to the non-Gaussian property of I_t and non-linearity of $\boldsymbol{X}_t$, we resort to the particle filtering approach where the posterior probability is maintained by a weighted sample set. The samples at time t can be generated from those at time $(t - 1)$ by 6.2, and weights are assigned by 6.1. A better proposal is derived by incorporating the current appearance observation in the spirit of APF [124] and the estimation can be further improved by utilizing the Gaussian probability of $p(\boldsymbol{K}_t | I_t, \boldsymbol{K}_{t-1})$.

6.3 Tensors: A brief review

As pointed out by the authors in [65] natural images are a result of a number of interacting factors related to illumination, scene structure etc. For example in [65] the authors consider a face dataset with variations in identity, illumination, expression, pose etc. In our problem, we want to develop a appearance model capable of handling variations in target shape due to viewing angle and target identity. The algebra of higher-order tensors provides a effective framework to separate out (decompose) the constituent factors of image ensembles. In this chapter, tensor decomposition is a primary component of the proposed generative appearance model. The aim of this section is to provide a brief review of the relevant tensor algebra concepts and is based on the works presented in [65, 125, 126].

Common linear algebra techniques like Principal Component Analysis (PCA) and Independent Component Analysis (ICA) have been successfully applied to several image analysis and representation problems primarily in the context of face recognition [127]. However, by their nature these methods are most suited for analysis of a single varying factor. In the case of face recognition this varying factor is most often the identity of the individual. These methods have difficulty in effectively decomposing other variations such as illumination or pose if they are present in the dataset. In this chapter we develop a target appearance model with the decomposable factors of target identity and view angle using multi-linear algebra.

A tensor is defined as a multidimensional matrix or a n-way array or n-mode matrix. They are higher order generalizations of a vector (first order tensor) and a matrix (second order tensor). In this section we represent scalars using lower case letters $(a, b, ..)$, vectors using bold lower case letters $(\boldsymbol{a}, \boldsymbol{b}, ..)$, matrices using bold upper case letters $(\boldsymbol{A}, \boldsymbol{B}, ..)$ and higher order tensors by script upper case letters $(\mathscr{A}, \mathscr{B}, ..)$. The order of a tensor $\mathscr{A} \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$ is N and a element of this tensor is denoted by $a_{i_1 i_2 \ldots i_N}$ where $1 \leq i_n \leq I_n$.

Tensor Flattening: One of the most important and useful operation that can be performed on a tensor is flattening (matricization or unfolding). It is the process of reordering the elements of a tensor into a matrix. For example, a $2 \times 3 \times 4$ tensor can be rearranged into a 4×6 matrix, 3×8 matrix or a 2×12 matrix. The mode-n matricization of a tensor $\mathscr{A} \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$ is denoted by $\boldsymbol{A}_{(n)}$. The example provided in [125] is repeated here for clarity of the flattening concept. Let the frontal slices of $\mathscr{A} \in \Re^{3 \times 4 \times 2}$ be represented by

$$\boldsymbol{A}_1 = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}, \boldsymbol{A}_2 = \begin{bmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{bmatrix}. \tag{6.3}$$

Then the three mode-n flattened matrices are given by

$$\boldsymbol{A}_{(1)} = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 \end{bmatrix}, \tag{6.4}$$

$$\boldsymbol{A}_{(2)} = \begin{bmatrix} 1 & 2 & 3 & 13 & 14 & 15 \\ 4 & 5 & 6 & 16 & 17 & 18 \\ 7 & 8 & 9 & 19 & 20 & 21 \\ 10 & 11 & 12 & 22 & 23 & 24 \end{bmatrix}, \tag{6.5}$$

$$\boldsymbol{A}_{(3)} = \begin{bmatrix} 1 & 2 & 3 & \cdots & 11 & 12 \\ 13 & 14 & 15 & \cdots & 23 & 24 \end{bmatrix}. \tag{6.6}$$

Note that different works use different ordering of the columns for the mode-n flattening operation. In general, the specific permutation of the columns is not important as long as it is consistent across all related calculations [125].

Mode-n product: The next important operation involving tensors that is of interest to us is the mode-n product. The mode-n product of a tensor $\mathscr{A} \in \Re^{I_1 \times I_2 \times \ldots \times I_n \times \ldots \times I_N}$ with a matrix $\boldsymbol{Q} \in \Re^{J \times I_n}$ is denoted by $\mathscr{B} = \mathscr{A} \times_n \boldsymbol{Q}$. The tensor $\mathscr{B} \in \Re^{I_1 \times I_2 \times \ldots \times I_{n-1} \times J \times I_{n+1} \ldots \times I_N}$ and elementwise we have

$$b_{i_1 \ldots i_{n-1} j i_{n+1} \ldots i_N} = \sum_{i_n=1}^{I_n} a_{i_1 i_2 \ldots i_N} q_{j i_n}. \tag{6.7}$$

The mode-n multiplication maybe represented using flattened matrices as follows

$$\mathscr{B} = \mathscr{A} \times_n \boldsymbol{Q} \Leftrightarrow \boldsymbol{B}_{(n)} = \boldsymbol{Q} \boldsymbol{A}_{(n)}. \tag{6.8}$$

Another important property of the mode-n product is

$$\mathscr{A} \times_n \boldsymbol{P} \times_m \boldsymbol{Q} = \mathscr{A} \times_m \boldsymbol{Q} \times_n \boldsymbol{P} \quad (m \neq n). \tag{6.9}$$

Tensor decomposition: Let us consider the singular value decomposition (SVD) of a matrix $\boldsymbol{P} \in \Re^{I_1 \times I_2}$. The SVD operation orthogonalizes the row space and the column space of the matrix $\boldsymbol{P}$ and provides the decomposition $\boldsymbol{P} = \boldsymbol{U}_1 \boldsymbol{\Sigma} \boldsymbol{U}_2^T$. Here $\boldsymbol{U}_1 \in \Re^{I_1 \times J_1}$ is the orthogonal columns space of the matrix $\boldsymbol{P}$, $\boldsymbol{\Sigma} \in \Re^{J_1 \times J_2}$ is the diagonal singular value matrix and $\boldsymbol{U}_2 \in \Re^{I_2 \times J_2}$ is the orthogonal row space of the matrix $\boldsymbol{P}$. In terms of the mode-n product discussed previously, SVD can be expressed by

$$\boldsymbol{P} = \boldsymbol{U}_1 \boldsymbol{\Sigma} \boldsymbol{U}_2^T \Leftrightarrow \boldsymbol{P} = \boldsymbol{\Sigma} \times_1 \boldsymbol{U_1} \times_2 \boldsymbol{U}_2. \tag{6.10}$$

Extending the concept of SVD to n-mode matrices it is possible to obtain

$$\mathscr{A} = \mathscr{C} \times_1 \boldsymbol{U}_1 \times_2 \boldsymbol{U}_2 \cdots \times_N \boldsymbol{U}_N. \tag{6.11}$$

Here the tensor $\mathscr{C}$ is known as the core-tensor and is analogous to the diagonal singular value matrix in the SVD decomposition of a matrix. Note that the core tensor $\mathscr{C}$ does not have a diagonal structure and contains information about the interaction between the mode matrices $\boldsymbol{U}_n$ for $n = 1, 2, \cdots, N$. The "N-mode SVD" operation expresses the tensor $\mathscr{A}$ as the mode-n product of N orthogonal spaces. The mode matrices $\boldsymbol{U}_n$ contain the orthonormal column space of the matrix $\boldsymbol{A}_{(n)}$ that result from the mode-n flattening of the tensor $\mathscr{A}$. Once the mode matrices are identified using regular SVD on the flattened matrices $\boldsymbol{A}_{(n)}$, the core tensor $\mathscr{C}$ maybe obtained by

$$\mathscr{C} = \mathscr{A} \times_1 \boldsymbol{U}_1^T \times_2 \boldsymbol{U}_2^T \cdots \times_N \boldsymbol{U}_N^T. \tag{6.12}$$

The implication of the decomposition in (6.11) can better understood by considering the face dataset example presented in [65]. Assume we have images face images of 20 different people imaged under 5 different views and 3 different expressions and let each image be of size 80×60. It is possible to represent this dataset as a tensor $\mathscr{A}$ of dimension $20 \times 5 \times 3 \times 4800$. We can perform tensor decomposition as described above to obtain

$$\mathscr{A} = \mathscr{C} \times_1 \boldsymbol{U}_{people} \times_2 \boldsymbol{U}_{views} \times_3 \boldsymbol{U}_{expression} \times_4 \boldsymbol{U}_{pixels}. \tag{6.13}$$

Here the matrix U_{people} contains the basis vectors of the space of identity of the subjects, U_{views} spans the space of viewpoint parameters, $\boldsymbol{U}_{expression}$ spans the space of expression

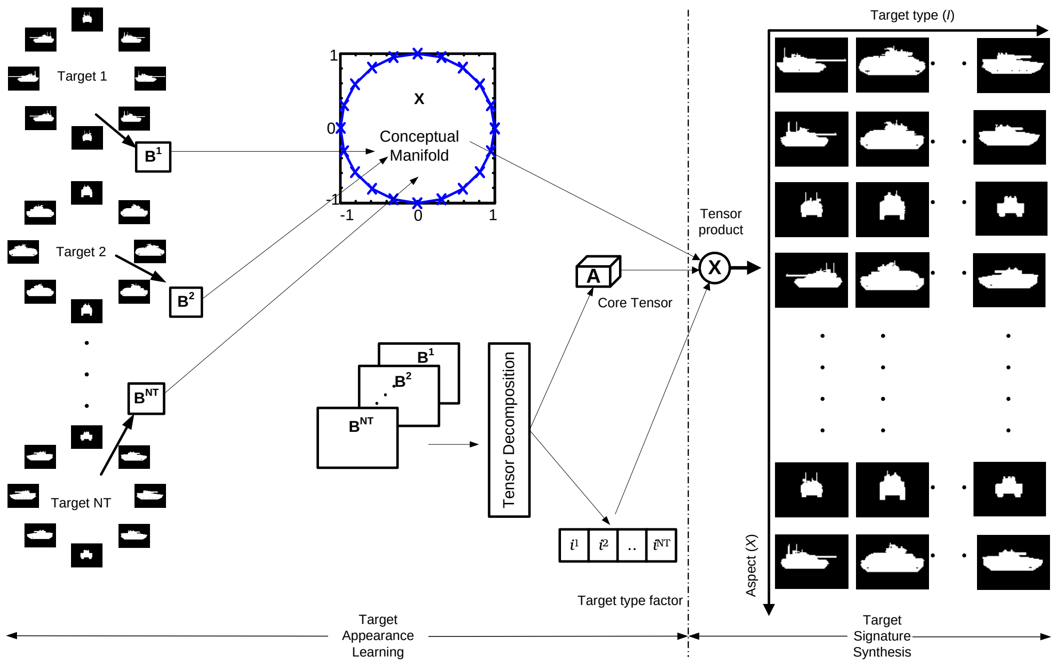


Figure 6.2: Tensor-based generative model for multi pose target representation.

parameters, $\boldsymbol{U}_{pixels}$ spans the space of images and the core tensor $\mathscr{C}$ governs the interaction between the mode matrices. The tensor decomposition provides for a way to meaningfully separate out the constituent factors affecting images in the ensemble. In our work, we apply a similar decomposition to separate out target identity and view related factors as outlined in the section below.

6.4 Generative models

There are two generative models involved in Fig.6.1, namely the appearance and motion generative models. The former relates the latent states with observation, whereas the latter specifies the evolution of latent states over time. The conditional dependencies among the variables in Fig.6.1 are derived from the generative models that encode the cause-and-effect relationships between variables. These two models that share the same identity (cause) variable are jointly called *joint appearance-motion generative models*.

6.4.1 Appearance model

One major challenge in tracking and recognition is that a target can continuously change its appearance due to pose variations. It is practically impossible to train an algorithm with all possible target appearances. Inspired by [67, 128], we use a generative model approach to overcome this problem. We assume that the appearances of different poses lie on a conceptual pose manifold represented by a 2-D circle common to all targets, as shown in Fig.6.2. Our objective is to have a functional mapping from this low dimensional conceptual space to the higher dimensional image space. Radial Basis Functions (RBFs) are often used in function interpolation and can easily provide a mapping between a higher dimensional and a low dimensional space. In addition, such a mapping can be learned from a small set of training data which include signatures (i.e.,silhouettes) of multiple targets under different poses. The learning results in a mapping function from a point on the low dimensional manifold to a high dimensional silhouette image and also provides a way for us to generate silhouettes corresponding to view points on the low-dimensional manifold that were not used for learning the mapping. Details of the learning of such a mapping function using radial basis functions is clearly presented in [129].

We use non-linear Gaussian Radial Basis Functions (GRBF) to obtain the following mapping [130]:

$$y_m^k = \mathbf{B}^k \psi(x_m), \tag{6.14}$$

where y_m^k is the high dimensional $(d \times 1)$ row vectorized silhouette image of target k under pose m; x_m is the point corresponding to pose m on the conceptual manifold. $\mathbf{B}^k$ is a $d \times N_c$ linear mapping corresponding to target k. $\psi(\cdot)$ is a non-linear kernel mapping to the embedding space, composed of N_c GRBFs along the manifold. For N_T different targets, based on 6.14 we may obtain their corresponding mapping functions $\mathbf{B}^k$ for $k = \{1, 2, ..., N_T\}$ that can be stacked together to form the matrix $\mathbf{C} = [\mathbf{B}^1\ \mathbf{B}^2\ \ldots\ \mathbf{B}^{N_T}]$. $\mathbf{C}$ contains information about target-dependent signatures pertaining to different poses. From $\mathbf{C}$, we can decompose the target type factor i^k by using the asymmetrical bilinear model to perform tensor decomposition as in [64]. The tensor-based target representation may then be written as

$$y_m^k = \mathscr{A} \times_3 i^k \times_2 \psi(x_m), \tag{6.15}$$

where $\mathscr{A}$ is called the core tensor with dimensionality $d \times N_c \times N_T$; i^k is the target type factor of dimensionality N_T; and $\times_j$ denotes mode-j tensor product.

As shown in Fig. 6.2, there are two steps involved in the generative model, *appearance learning* and *signature synthesis*. The former is accomplished by finding $\mathbf{B}^k$ in 6.14 followed by a bilinear decomposition to obtain $\mathscr{A}$ and i^k. The latter is characterized by 6.15, using which we can reconstruct a target signature of any type and pose. Once the learning is accomplished, we only need to store $\mathscr{A}$ and i^k to provide a general target representation. In addition to aiding the tracker by accounting for inter-frame appearance changes, this model can also facilitate recognition by generating distinct type-specific appearances with any pose.

6.4.2 Motion models

Motion models play an important role in target tracking [68]. In practice, different targets have widely varying kinematics maneuverability due to the nature of engines and mechanical systems. Using type-specific motion models will better capture the kinematics and maneuvering actions of different targets. Therefore, we consider multiple 3-D rigid motion models that are associated with different target types. Moreover, the motion cue provides additional evidence for recognizing target types.

For ground targets, we introduce a type dependent variable into the dynamics of linear velocity v along the direction of target motion (x' in Fig.6.3).

$$v_t = v_{t-1} + f(I_t) + s(I_t)\omega_v, \tag{6.16}$$

where ω_v is independent identically distributed (iid) Gaussian noise; $\omega_v \sim \mathcal{N}(0,1)$; $f(I_t)$ and $s(I_t)$ are the two variables controlling the target-dependent acceleration. It is worth

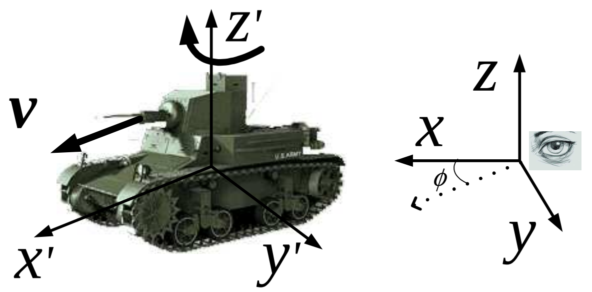


Figure 6.3: The 3-D coordinate systems of a ground vehicle. The axes $x'y'z'$ defines the body frame, where the vehicle is constrained to move along the x' direction and rotate about the z' axis to avoid unrealistic motion. The xyz axes define the observer's frame of reference where the plane xoy is parallel to $x'o'y'$.

noting that the dynamics of v_t is a linear Gaussian model given I_t, making it possible to update the related probabilities using a Kalman filter.

The dynamics for zero-order kinematics, i.e., pose and position, are introduced in addition to that of velocity. The pose variable ϕ follows a simple dynamic model given by

$$\phi_t = \phi_{t-1} + u + \omega_\phi, \tag{6.17}$$

where u denotes a fixed angular velocity, and $\omega_\phi \sim \mathcal{N}(0, \sigma_\phi^2)$. This model is capable of capturing subtle rotational dynamics of a rigid target. The target is assumed to move only on the ground plane (p_z=0). The position $\boldsymbol{p} = [p_x, p_y]'$ in observer's coordinate system is related to velocity v and pose ϕ as:

$$\boldsymbol{p}_t = \boldsymbol{p}_{t-1} + \boldsymbol{R}(\phi_{t-1})v_{t-1} + \boldsymbol{\omega}_p, \tag{6.18}$$

where $\boldsymbol{R}$ is a rotational vector defined as $\boldsymbol{R}(\phi) = [\cos\phi \;\; \sin\phi]'$, and $\boldsymbol{\omega}_p \sim \mathcal{N}(0, \text{diag}(\sigma_x^2, \sigma_y^2))$. Nonlinearity introduced in the state equation 6.18 requires a sampling based inference. Since the type dependent velocity appears in 6.18, N_T models are needed to characterize the type-specific motion patterns.

6.5 Inference algorithms

In this section, we detail two inference algorithms for the graphical model described in Section 6.2. One is the APF-based particle filter that takes into account the current observations about $\boldsymbol{X}_t$ (position/pose), and other is an enhanced APF algorithm where the current observation of $\boldsymbol{K}_t$ (velocity) is involved for proposal generation.

6.5.1 APF algorithm

Dynamics: The dynamics of kinematic variables corresponding to velocity v_t, pose variations ϕ_t and position $\boldsymbol{p}_t$ are defined in 6.16,6.17 and 6.18 respectively. In order to maintain multiple hypothesis about the target identity, we define the dynamics of the identity variable I_t as

$$p(I_t = i | I_{t-1} = j) = \boldsymbol{T}_t(i,j); i,j \in \{1, 2, \cdots, N_T\} \tag{6.19}$$

where $\boldsymbol{T}_t(i,j)$ is a transition matrix that defines the transition probability between identities i and j at time t. Using an annealing-like strategy, $\boldsymbol{T}_t$ is defined as follows:

$$\boldsymbol{T}_t(i,j) = \begin{cases} 1 - \exp(-at) & \text{if } i = j, \\ \frac{\exp(-at)}{N_T - 1} & \text{if } i \neq j, \end{cases}$$

where a is a fixed positive constant. This empirical setting of $\boldsymbol{T}_t$ gradually reduces the identity switch frequency based on the belief that the state inference is increasingly confident about the identity estimation with time. However, the value of a depends on the reliability of the observations.

Observation likelihood: Assume a target with certain identity moving in a 3-D scene according to its dynamics and we observe it via a stationary pre-calibrated perspective camera with known parameters $\mathbf{T}_{camera}$. Given the target type, pose and position we can synthesize the required target silhouette using the generative model 6.15. This silhouette is then appropriately placed on the 2-D image plane with specific scale and position, using $\mathbf{T}_{camera}$. Assuming that the observed image sequences are corrupted by additive i.i.d.

Gaussian noise $\mathcal{N}(0, \sigma_{obs}^2)$, the likelihood function $p(\boldsymbol{Z}_t|I_t, \boldsymbol{X}_t)$ of the current observation $\boldsymbol{Z}_t$ is given by:

$$p(\boldsymbol{Z}_t|I_t, \boldsymbol{X}_t) \propto \exp\Big[- \frac{\|\boldsymbol{Z}_t - g(I_t, \boldsymbol{X}_t, \mathbf{T}_{camera})\|^2}{2\sigma_{obs}^2} \Big], \tag{6.20}$$

where $g(\cdot)$ is a mapping function that results in a hypothesized target silhouette based on I_t, $\boldsymbol{X}_t$, and $\mathbf{T}_{camera}$; $\| \cdot \|$ gives the mean square error between the observation and the synthesized target appearance in the tracking gate.

Algorithm pseudo-code: Having defined the required system and observation models, sequential state estimation can be performed using an Auxiliary Particle Filter (APF). The pseudo code for a single time step of the APF-based inference is given in **Table 6.1** where the current observation is considered for drawing samples in step 4.

6.5.2 Kalman filter enhanced APF (KAPF)

Discussion on APF: Due to the lack of velocity measurements, there is no direct inference about the velocity v_t in the APF, making its estimate less accurate. However, velocity plays an important role in the estimation of $\boldsymbol{X}_t$ as well as I_t, when targets exhibit different motion patterns. In the following, we enhance the velocity estimation and apply it to improve the efficiency of state inference in two ways. Firstly, the estimated velocity is involved to generate more samples of the most likely target type and secondly, in generating improved position hypothesis for the next time step. The underlying idea of the APF is to use the *effect* (current observations) to generate more plausible *cause* samples (position/pose hypothesis). Similarly, we consider the *effect* (position estimation) to generate more likely *cause* samples (velocity). Unlike [60], where the change in high dimensional appearance is used for velocity update, we use position estimates to update velocity using a 1-D Kalman filter.

New proposal: A new proposal generation scheme is inserted between the steps 8 and 9 in **Table 6.1** for target type and velocity, in the spirit of RBPF using the derivation

```
Input: The current observation $\boldsymbol{Z}_t$ and the particle set at time
    $t-1$ represented by $\{\Phi_{t-1}^j\}_{j=1}^{N_p} = \{X_{t-1}^j, v_{t-1}^j, I_{t-1}^j\}_{j=1}^{N_p}$.
1. Propagate the particle set $\{\Phi_{t-1}^j\}_{j=1}^{N_p}$ using the motion
    model equations 6.16,6.17,6.18 and 6.19 to obtain the particle set
    $\{\widetilde{\Phi}_t^j\}_{j=1}^{N_p} = \{\widetilde{X}_t^j, \widetilde{v}_t^j, \widetilde{I}_t^j\}_{j=1}^{N_p}$.
2. Assign weights $\{\widetilde{w}_t^j\}_{j=1}^{N_p} \propto p(\boldsymbol{Z}_t|\widetilde{\Phi}_t^j)$ as in (6.20).
3. Normalize weights $\{\widetilde{w}_t^j\}_{j=1}^{N_p}$ such that $\sum_{j=1}^{N_p} \widetilde{w}_t^j = 1$.
4. Draw auxiliary variable $\lambda^j \in \{1, 2, \cdots, N_p\}$ such that
    $p(\lambda^j = i) = \widetilde{w}_t^i$ where $i = 1, 2, \cdots, N_p$.
5. Propagate the particle set $\{\Phi_{t-1}^{\lambda^j}\}_{j=1}^{N_p}$ using the motion
    model equations 6.16,6.17,6.18 and 6.19 to obtain the new particle set
    $\{\Phi_t^j\}_{j=1}^{N_p} = \{X_t^j, v_t^j, I_t^j\}_{j=1}^{N_p}$.
6. Assign weights $\{w_t^j\}_{j=1}^{N_p} \propto \frac{p(\boldsymbol{Z}_t|\Phi_t^j)}{p(\boldsymbol{Z}_t|\widetilde{\Phi}_t^{\lambda^j})}$ as in 6.20
7. Normalize weights $\{w_t^j\}_{j=1}^{N_p}$ such that $\sum_{j=1}^{N_p} w_t^j = 1$.
8. Estimate mean position $\widehat{\boldsymbol{p}}_t$ and pose $\widehat{\phi}_t$ from $\{\Phi_t^j, w_t^j\}_{j=1}^{N_p}$.
9. Set $\{\Phi_t^j\}_{j=1}^{N_p}$ = Re-sample $[\{\Phi_t^j, w_t^j\}_{j=1}^{N_p}]$.
Outputs: The particle set at time $t$ represented by $\{\Phi_t^j\}_{j=1}^{N_p}$ and
    the mean estimates $\widehat{\boldsymbol{p}}_t$ and $\widehat{\phi}_t$.
```

Table 6.1: Pseudo-code for one time step of the APF algorithm.

in [131, 132]. Since the velocity follows a linear Gaussian process (6.16), a Kalman filter is incorporated to generate identity samples and update the velocity samples. In addition to the particle set $\{\Phi_t^j\}_{j=1}^{N_p}$ in **Table 6.1**, we maintain a second set of particles $\{\varphi_t^j\}_{j=1}^{N_p}$ = $\{I_t^j, \mu_{v,t}^j, \Sigma_{v,t}^j\}_{j=1}^{N_p}$, where type samples I_t^j are from$\{\Phi_t^j\}_{j=1}^{N_p}$; the mean $\mu_{v,t}^j$ and variance $\Sigma_{v,t}^j$ characterize the velocity probability density for each particle. In generating target type samples, the type whose motion model fits the current velocity closely is given higher preference. Thus the predictive density of the type variable is assigned as

$$p(I_t|I_{t-1}, y_{1:t}) \propto p(y_t|y_{1:t-1}, I_{0:t-1}), \tag{6.21}$$

where y_t denotes the velocity observation at time step t obtained from the effect variables of velocity using

$$y_t = \|\widehat{\boldsymbol{p}}_t - \widehat{\boldsymbol{p}}_{t-1}\|. \tag{6.22}$$

We assume $p(y_t|v_t) \sim \mathcal{N}(v_t, b^2(I_t))$, where $b^2(I_t)$ is the variance of y_t. Then the predictive density of y_t is also Gaussian, i.e., $p(y_t|y_{1:t-1}, I_t) \sim \mathcal{N}(\mu_{y,t}, \Sigma_{y,t})$, and can be evaluated by 1-D Kalman filtering equations as:

$$\begin{aligned} \mu_{y,t}(I_t) &= \mu_{v,t-1} + f(I_t), \\ \Sigma_{y,t}(I_t) &= \Sigma_{v,t-1} + s^2(I_t) + b^2(I_t), \end{aligned} \tag{6.23}$$

where $f(I_t)$ and $s(I_t)$ appear in 6.16 characterizing the maneuverability. In addition, the Kalman filter can also be used to update the density of velocity by using the conditional Gaussian. For simplicity, we omit the identity variable I_t and the Kalman equations in the 1-D case becomes

$$\begin{aligned} \mu_{v,t} &= \mu_{v,t-1} + (\Sigma_{v,t-1} + s^2)\Sigma_{y,t}^{-1}(y_t - \mu_{y,t}), \\ \Sigma_{v,t} &= (\Sigma_{v,t-1} + s^2)b^2\Sigma_{y,t}^{-1}. \end{aligned} \tag{6.24}$$

Therefore, new velocity samples are generated from $\mathcal{N}(\mu_{v,t}, \Sigma_{v,t})$ and used for position prediction.

Algorithm pseudo-code: The KAPF algorithm is given in **Table 6.2** that is embedded between steps 8 and 9 in **Table 6.1**. The output containing the identity and velocity samples $\{I_t^j, v_t^j\}_{j=1}^{N_p}$ replace the ones generated by step 9 in **Table 6.1**, as priors for position prediction in the next time step. The elements of the transition matrix used in 6.19 are set to $1/N_T$ to avoid any possible bias.

6.6 Experimental results

The generative models and inference algorithms are evaluated in three experiments based on simulated sequences where the background is a real IR image with additive Gaussian

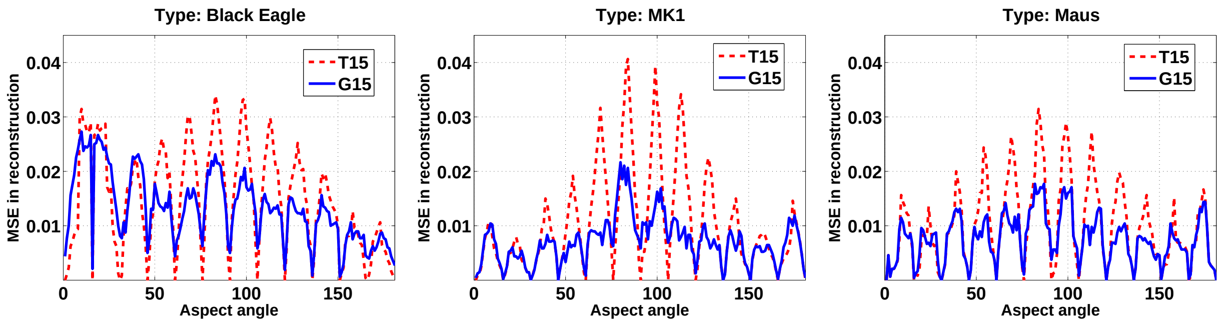


Figure 6.4: Comparison of reconstruction MSE for the generative model (G15) and the template model (T15). Only a pose range of 180^o is shown due to symmetric nature of the reconstruction error.

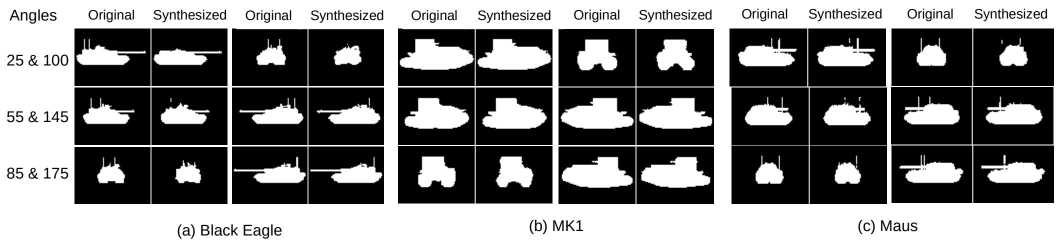


Figure 6.5: Illustration of the generative model based reconstruction.

```
Inputs: Position estimates $\widehat{\boldsymbol{p}}_t$ and $\widehat{\boldsymbol{p}}_{t-1}$ from Table 6.1 and the
  particle set $\{\varphi_{t-1}^j\}_{j=1}^{N_p}$=$\{I_{t-1}^j, \mu_{v,t-1}^j, \Sigma_{v,t-1}^j\}_{j=1}^{N_p}$.
  Compute $y_t$ based on 6.22.
  For $j = 1, \cdots, N_p$
    For $I = 1, \cdots, N_T$
      Calculate $\mu_{y,t}^j(I)$ and $\Sigma_{y,t}^j(I)$ using 6.23
      Evaluate p$(I|I_{t-1}^j, y_t) \propto N(y_t; \mu_{y,t}^j, \Sigma_{y,t}^j)$
    End
    Sample $I_t^j \sim p(I|I_{t-1}^j, y_t)$
    Update $\mu_{v,t}^j$ and $\Sigma_{v,t}^j$ using 6.24
  End
  Draw samples $\{v_t^j\}_{j=1}^{N_p} \sim N(\mu_{v,t}^j, \Sigma_{v,t}^j)$
Outputs: The particle set at time $t$ represented by $\{\varphi_t^j\}_{j=1}^{N_p}$ and
  the velocity samples $\{v_t^j\}_{j=1}^{N_p}$
```

Table 6.2: Pseudo-code of motion cue based proposal generation.

noise (SNR=20dB). The first experiment examines the generative appearance model in terms of its capability of synthesizing unseen target signatures. The second experiment compares the joint appearance-motion generative model against a template model for integrated tracking and recognition. The last experiment demonstrates the advantages of the KAPF algorithm for velocity, position and identity estimation.

6.6.1 Target signature generation

A comprehensive target database that includes 3-D models of many ground vehicles, mainly tanks was collected for this research. In the following experiments, we use five tank models from this database, i.e., FCS, MK1, T28, Eagle, and Maus, as shown in Fig. 6.6. From these 3-D models we obtain silhouettes of dimension 60×80 corresponding to different poses (1° to 360°) of a particular tank. These silhouettes are transformed into gray-scale images using signed distance transform as in [67, 128], to impose smoothness of the distance between poses. For each target k, the mapping ($\mathbf{B}^k$) is found using 24 silhouettes corresponding to poses 15° apart. The generative model is then learnt as in Sec 4.1.

Once the model is learnt we can synthesize a target silhouette of any pose. Fig.6.4 shows the mean square error (MSE) of the generative model (G15) when compared to the case of using templates (T15) sampled 15 degrees apart. We observe that synthesized silhouettes have much lower MSE than the template-based approach. Fig.6.5 illustrates the reconstructed silhouettes for a few untrained poses. The reconstruction closely resembles the true template for most view angles even for those with higher reconstruction error. However in both figures note the significantly large errors around 90° due to the fact that there is more perceived change per degree of rotation around these angles. This suggests that an uniformly sampled circular manifold may not be an optimal representation of the true underlying structure of the view manifold.

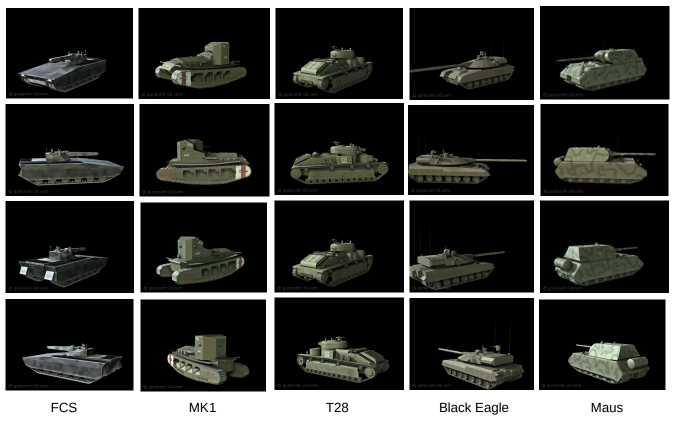


Figure 6.6: 3-D target models of the five tanks used in simulations.

6.6.2 Tracking and recognition with joint models

Five sequences corresponding to five targets were generated using the dynamics in 6.16, 6.17 and 6.18 and imaged through a virtual camera. When the video sequences were generated, we set $f(I_t) = 0.025$ to simulate a constant forward force and $s(I_t) = 0.01 * I_t$

to simulate type dependent acceleration; $u = \pi/180$ and $\sigma_\phi = \pi/60$ in 6.17. Few frames of a simulated IR sequence are shown in Fig. 6.7. The inference is done using the APF algorithm outlined in **Table 6.1**.

The tracking performance of two different appearance models i.e., the generative model (G15) and the template model (T15) are compared without considering type dependent motion, i.e., $s(I_t)$ is a constant in the APF inference algorithm. Sample tracking results for SEQ2 are shown in Fig.6.8. It is seen that the T15 model can lead to loss of track due to its limitation in matching intermediate poses absent in the training set. On the other hand, the G15 model that can create intermediate poses improves both position and pose estimation as shown in **Table 6.3**.

Next, the tracking performance of the G15 model along with type dependent motion i.e., the joint appearance-motion generative model (G15-M), is evaluated. To accommodate different motion models, we set $s(I_t) = 0.01 * I_t$ in the inference algorithm. **Table 6.3** lists the tracking performance of all three approaches. In most cases, G15-M shows the best tracking performance proving the usefulness of the joint generative model in pose and position estimation. The three methods result in comparable recognition performance, and for all five sequences the identity estimation converges to the true target type within a few time steps.

6.6.3 KAPF based inference and proposal generation

Five more sequences corresponding to each of the five targets were generated with $f(I_t) = 0.0025$, $s(I_t) = 0.01 * I_t$ in 6.16 and random turning actions $u = 0$, $\sigma_\phi = \pi/60$ in 6.17. In this experiment, the appearance model is trained from 36 silhouettes corresponding to 10° pose changes for each of five targets. The tracking performance of the KAPF is compared to that of APF* (APF with fixed transition matrix as defined in Sec 5.2) to avoid any prior bias.

As seen from Tab. 6.4, the KAPF algorithm shows improvement over the APF* method

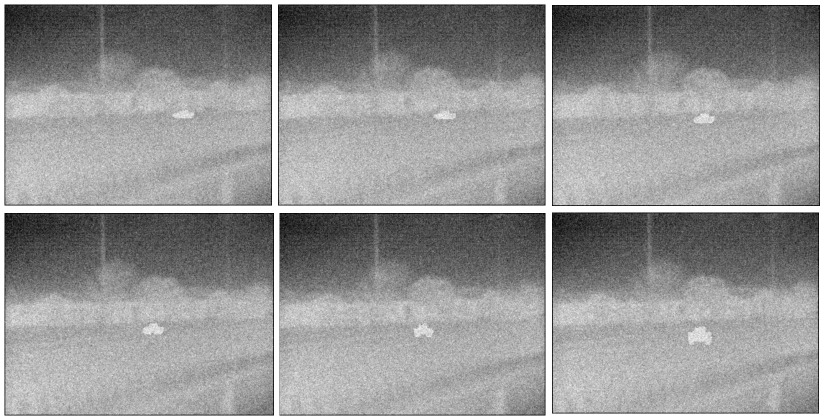
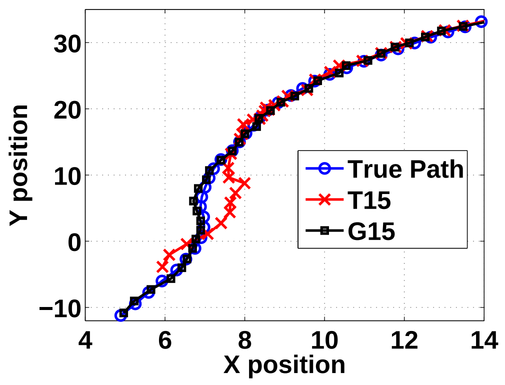


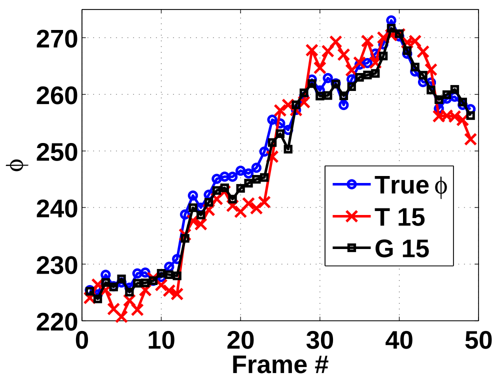
Figure 6.7: Sample frames of sequence SEQ1 (Tank FCS). Frames 1, 15, 28, 35, 40 and 50 left to right, top to bottom.

since the Kalman filter used for velocity estimation is optimal in the MSE sense under Gaussian assumptions. The appearance based likelihood defined in 6.20 is not very sensitive to the variation of scene depth (p_y) (that is only reflected by scale of the target signature) when compared to that of translation (p_x). In addition to the appearance likelihood, the KAPF also exploits motion cues to improve position estimates especially for depth (p_y). These improvements imply that the KAPF is able to generate velocity samples closer to the true states and thereby provide better prediction for position samples.

The KAPF also performs better than APF* in terms of recognition with fewer misclassification. The KAPF that uses both appearance and velocity information to generate identity samples, has an advantage over APF* which uses only appearance information, especially when two targets appear similar in a certain pose. Moreover, the KAPF maintains the hypotheses of other target types during tracking, despite strong preference towards the right type. This provides a good balance between *diversity* and *focus* of sample distribution during inference, and prevents the type samples from being trapped into incorrect type hypotheses.



(a)



(b)

Figure 6.8: Sample tracking result for sequence SEQ2: (a) path estimate and (b) pose estimate.

Algorithms		SEQ1	SEQ2	SEQ3	SEQ4	SEQ5
T15	ϕ	3.271	3.824	2.948	5.331	5.428
	p_x	0.115	0.248	0.106	0.263	0.231
	p_y	0.541	1.573	**0.208**	1.192	1.413
G15	ϕ	3.397	2.325	2.519	**3.468**	3.057
	p_x	0.106	0.153	0.116	**0.145**	**0.200**
	p_y	0.637	0.870	0.337	0.605	1.222
G15-M	ϕ	**2.768**	**2.258**	**2.007**	3.801	**2.785**
	p_x	**0.094**	**0.117**	**0.099**	0.185	0.232
	p_y	**0.323**	**0.560**	0.238	**0.454**	**1.174**

Table 6.3: RMSE of pose and position estimates over 50 frames averaged upon 20 Monte Carlo runs.

Algorithms		SEQa	SEQb	SEQc	SEQd	SEQe
APF*	ϕ	6.307	**2.195**	3.874	6.021	**2.188**
	p_x	0.146	0.105	0.378	0.132	0.135
	p_y	0.729	0.697	1.265	0.692	0.319
	v	0.035	0.054	0.239	0.075	**0.106**
KAPF	ϕ	**4.659**	2.209	**3.412**	**5.355**	2.314
	p_x	**0.089**	**0.098**	**0.161**	**0.106**	**0.128**
	p_y	**0.412**	**0.681**	**0.653**	**0.524**	**0.273**
	v	**0.030**	**0.036**	**0.096**	**0.054**	0.127

Table 6.4: RMSE of the pose, position and velocity estimates over 30 frames averaged upon 20 Monte Carlo runs.

CHAPTER 7

Recognition based on identity and view manifolds

7.1 Background

The human visual system (HVS) has remarkable ability to understand a scene by effortlessly recognizing individual objects present in the scene. The need to replicate this ability, at least in part, in real world applications has triggered intensive research in the area of object recognition. Some of this research is object-specific, like detecting humans or cars [133, 134], while others try to distinguish between multiple object classes [57]. There are two key issues of interest which is how to detect and recognize objects under different viewpoints and the second is how to account for both inter-class and intra-class variations. Usually, both the view and identity variables are considered to be discrete valued for view-independent object modeling. In this chapter, we study the case of vehicle recognition where we model both these variables to be continuous making it more flexible to handle unseen views or unknown class variants.

According to [50], the object recognition research could be roughly grouped into three categories, i.e., single-view 2D models, single instance 3D models, and multi-view models. The methods of the first group focus on object detection rather than pose estimation by modeling the appearances of multiple objects in a single, discrete or limited range of views [18, 135] without relating features across multiple views. Those of the second group estimate the pose/view by matching local features under rigid transformations [136, 137], making extensions to other object classes difficult. Those of the third group aim to build a coherent object model by relating descriptive features over multiple views [49, 52, 57, 134]. Our research belongs this group, and we propose a new shape-based generative model for

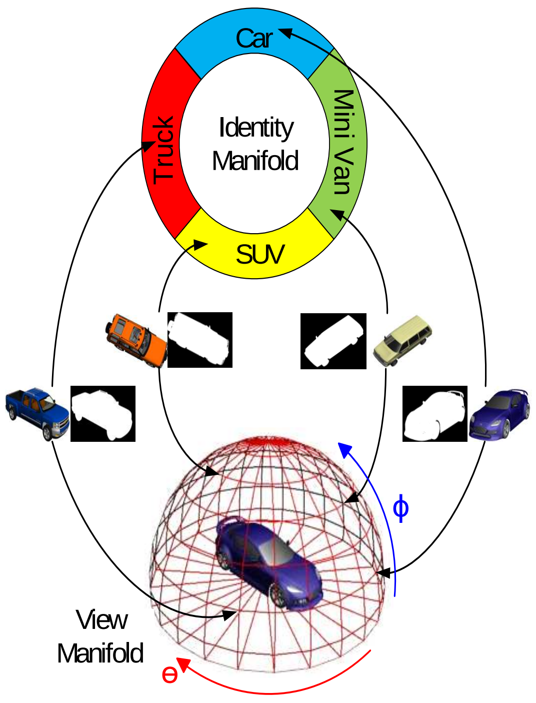


Figure 7.1: Coupled view-identity manifolds for vehicle modeling. We decompose the shape variability in the given training set into the two factors, identity and view, and both can be mapped to a low dimensional manifold. Then by choosing a point each manifold, a new shape can be interpolated that can be used for joint pose estimation and identity recognition.

general vehicle appearances that supports simultaneous pose and identity estimation.

To illustrate our approach, we choose four classes of road vehicles, i.e., cars, pickups, SUV's and mini-vans, each of which include several sub-classes for model training, as shown in Fig.7.1. There are two important continuous-valued manifolds supported by the proposed model. First, we learn a 1-D identity manifold to model both inter-class and intra-class shape variability among all training vehicles. An important issue is how to specify the manifold topology to order all classes and sub-classes along the identity manifold, and we propose a *class-constrained shortest-closed-path* to find the optimal manifold topology for identity modeling. Second, we define a 2-D hemisphere-shaped view manifold to cope with arbitrary view variations. A non-linear tensor decomposition technique is used to

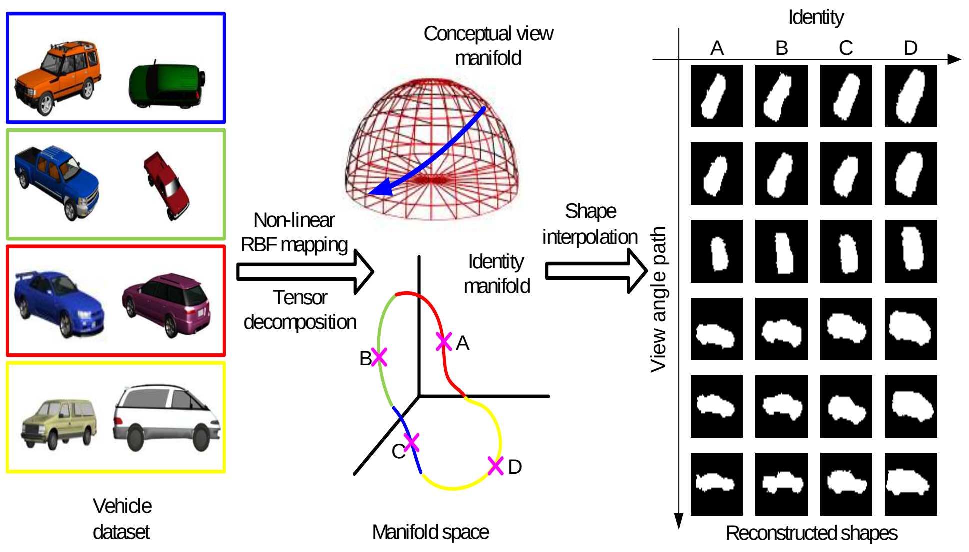


Figure 7.2: Illustration of the generative model for view and identity based shape appearance synthesis. Reconstruction results of the shape are shown for the blue path traversed along the view manifold and for four different points on the identity manifold that do not correspond to any of the training data. In each case the shape reconstructed has strong characteristics of the view and vehicle class.

integrate two manifolds into a unified generative model that is directly controlled by two variables. Although the simple silhouette-based shape feature is used in this work, the proposed model could be extended to other more powerful features for better modeling capability and recognition performance.

Though the proposed approach is not limited to a specific object recognition application, we use vehicle recognition as a case study in this work where the silhouette-based shape representation is adopted to represent vehicle appearances. In the following, we will discuss the identity and view manifolds first, followed by the development of shape-based generative model where the two manifolds are integrated for multi-view vehicle appearance modeling.

7.2 Identity manifold

The identity manifold plays the central role in our work that has the ability to capture both inter-class and intra-class shape variability. Unlike other methods in terms of the way that the identity is handled, its continuous nature makes it possible to interpolate an unknown subclass based on limited training data. However, there are two important questions to be addressed in order to learn such manifold. The first one is where or in which space we can learn this identity manifold. It should be in a LD latent space with only the identity factor rather than the HD observation space where the view and identity factors are coupled together, which will be addressed in Section 3.3. The second question is how to learn a *semantically valid* identity manifold that supports meaningful sub-class interpolation. Or what kind property we want to impose on the learned identity manifold.

To learn the identity manifold, we need to define a manifold topology first that includes the dimension or LD structure as well as the ordering relationship of all vehicle identities from different classes or sub-classes. If the training data are sparse (e.g., 20-30 in this work), we suggest a 1-D closed-loop structure to ensure valid identity interpolation, and this structure can also facilitates the inference process by a simple 1D identity variable. Although there is no clear ordering relationship among different vehicles, we do want those form the same class or of similar shapes to stay closer along the identity manifold compared with dissimilar ones. Thus we introduce a *class-constrained shortest-closed-path* method to find the unique ordering relationship across all vehicle identities. This method needs to specify a view-independent *distance* or *dissimilarity* measure between two identities. Ideally, we should use the shape dissimilarity between two 3D models. For simplicity, we use the accumulated mean square errors of multi-view silhouettes (after the distance transform [128]) to compute this distance.

Given N individual objects, suppose that we can find a set of N view-independent identity vectors in a LD latent space $\boldsymbol{i}^k, k \in \{1, \cdots, N\}$ along with the associated class label $\boldsymbol{L}_k$. Let us use $\boldsymbol{y}_m^k$ to denote the vectorized silhouette of object k under view m.

The similarity between objects u and v, represented by $\boldsymbol{i}^u$ and $\boldsymbol{i}^v$ respectively, over all M training views is given by

$$D(\boldsymbol{i}^u, \boldsymbol{i}^v) = \sum_{m=1}^{M} ||\boldsymbol{y}_m^u - \boldsymbol{y}_m^v|| + \alpha \cdot \epsilon(\boldsymbol{L}_u, \boldsymbol{L}_v), \tag{7.1}$$

where

$$\epsilon(\boldsymbol{L}_u, \boldsymbol{L}_v) = \begin{cases} 0 & \text{if } \boldsymbol{L}_u = \boldsymbol{L}_v, \\ 1 & \text{otherwise}, \end{cases} \tag{7.2}$$

where $||.||$ represents the Euclidean distance and α is a constant. The function $\epsilon(\boldsymbol{L}_u, \boldsymbol{L}_v)$ is a penalty term that ensures objects belonging to the same class are grouped together along the identity manifold. Let the manifold topology be represented by $\mathbf{T} = [t_1 \; t_2 \; \cdots \; t_{N+1}]$ where $t_i \in [1, N]$, $t_i \neq t_j$ for $i \neq j$ and $t_1 = t_{N+1}$. The class-constrained shortest-closed-path can be written as

$$\mathbf{T} = \arg\min_{\mathbf{T}} \sum_{i=1}^{N} D(\boldsymbol{i}^{t_i}, \boldsymbol{i}^{t_{i+1}}). \tag{7.3}$$

This manifold topology tends to group those objects of similar 3D shapes and/or within the same class together, enforcing the best local *semantic smoothness* along the identity manifold to be learned, which is essential for valid identity interpolation of an unknown sub-class.

7.3 Conceptual view manifold

In addition to the identity manifold, we also need to specify a view manifold to accommodate the view-induced shape variability. A common way is to use some non-linear dimensionality reduction techniques, such as LLE, Laplacian eigenmaps, to find the LD view manifold from HD observations for a given object [128]. There are two main limitations. One is that they are identity-dependent, and multiple view manifolds are involved which may have to be aligned together in the same latent space for object recognition. The other is that they are normally constrained by a 1D structure that may not accurately reflect all possible object poses in the real-world. In this paper, we design the view manifold to be a hemisphere that embraces almost all possible viewing angles around an object as shown in Fig. 7.1 and is characterized by two parameters: the azimuth angle θ and the elevation angle ϕ. Such conceptual manifold helps us avoid the issue of learning and aligning multiple view manifolds for each individual object. At the same time, it provides a unified and physically meaningful representation of the view space that supports efficient dynamic view estimation.

7.4 Non-linear tensor decomposition

In the following, we extend the non-linear tensor decomposition technique discussed in [67] to develop a shape-based generative model that can represent an object by a view and identity variables. This involves learning a non-linear mapping function from the data space to the unified view manifold and then linearly factoring out the identity vectors, giving a view-independent space for identity representation.

Let the d-dimensional observation of object k under view m be $\boldsymbol{y}_m^k$ and the corresponding LD point on the view manifold be $\boldsymbol{x}_m$. For each object k, we can learn a non-linear mapping between these two spaces using the generalized radial basis function (GRBF) kernel $\phi(.)$ according to

$$\boldsymbol{y}_m^k = \sum_{l=1}^{N_c} w_l^k \phi\left(||\boldsymbol{x}_m - \boldsymbol{z}_l||\right) + [1 \;\; \boldsymbol{x}_m] b_l, \tag{7.4}$$

where $\{\boldsymbol{z}_l | l = 1, ..., N_c\}$ are the N_c kernel centers on the view manifold; w_m^k are the object specific weight of each kernel and b_l is the mapping coefficient of the linear polynomial $[1 \; \boldsymbol{x}_m]$. This mapping maybe written in matrix form as

$$\boldsymbol{y}_m^k = \mathbf{B}^k \psi(\boldsymbol{x}_m), \tag{7.5}$$

where $\boldsymbol{x}_m$ is a point corresponding to view m on the view manifold. $\mathbf{B}^k$ is a $d \times N_c$ linear mapping corresponding to object k composed of the weight terms w_m^k in (7.4), $\psi(\boldsymbol{x_m}) = [\phi(||\boldsymbol{x_m} - \boldsymbol{z}_1||, \cdots, \phi||\boldsymbol{x_m} - \boldsymbol{z}_{N_c}||, 1, \boldsymbol{x_m})]$ is a non-linear kernel mapping that contains the regularization term $[1\ \boldsymbol{x}_m]$. Since $\phi(x_m)$ is dependent only on the view angle we can reason that the identity related information in contained within the term $\boldsymbol{B}^m$. For K different objects, based on (7.5) we may obtain their corresponding mapping functions $\mathbf{B}^k$ for $k = \{1, 2, ..., K\}$ that can be stacked together to form the tensor $\mathscr{A} = [\mathbf{B}^1\ \mathbf{B}^2\ \ldots\ \mathbf{B}^K]$. The tensor $\mathscr{A}$ can be thought of to contain information about identity-dependent signatures pertaining to different poses. Application of HOSVD to $\mathscr{A}$ abstracts K identity vectors $\boldsymbol{i}^k \in \Re^K$. This decomposition allows us to synthesize the appearance corresponding to an identity vector $\boldsymbol{i}^k$ and a view point $\boldsymbol{x}_m$ according to

$$\boldsymbol{y}_m^k = \mathscr{C} \times_3 \boldsymbol{i}^k \times_2 \psi(\boldsymbol{x}_m), \tag{7.6}$$

where $\mathscr{C}$ is called the core tensor with dimensionality $d \times N_c \times K$; and $\times_j$ denotes mode-j tensor product.

The identity vectors $\{\boldsymbol{i}^k | k = 1, ..., K\}$ from training objects maybe interpreted as the basis vectors of the exemplar identity space. Such an interpretation gives us an impression that any normalized linear combination of these basis vectors would form a valid new identity vector, which may lead to a meaningful shape interpolation. However the reconstruction produced in this manner normally does not resemble a real world object. To ensure valid shape reconstruction, we should first learn a smooth continuous-valued identity manifold via B-spline curving fitting in the tensor coefficient space according to the manifold topology defined in 7.3). It is expected that an arbitrary identity vector along this identity manifold will be more semantically meaningful due to its proximity to identity vectors from training objects, and should support valid shape interpolation. Thus (7.6) gives a compact generative model for multi-view shape modeling that is controlled by two continuous-valued variables each of which follows a LD manifold. Since the identity manifold is a closed-loop spanned in the N-dimensional tensor coefficient space, we can to *map* it to a circle along which inference is much easier.

7.5 Inference of view and identity

A graphical model is used to integrate all the factors along with their conditional dependencies into a probabilistic framework as shown in Fig. 7.3, where view $\boldsymbol{X}_t$ and identity I_t are two latent variables to be estimated and $\boldsymbol{Z}_t$ the observed shape. $\boldsymbol{S}_t$ is the synthesized object shape given view and identity hypotheses. The problem of joint identity and view estimation becomes the estimation of the posterior probability $p(I_t, \boldsymbol{X}_t|\boldsymbol{Z}_t)$. Due to the non-linearity nature of this inference problem, we resort to the particle filtering approach that involves a likelihood function and the dynamics of two latent variables to sequentially update $p(I_t, \boldsymbol{X}_t|\boldsymbol{Z}_t)$.

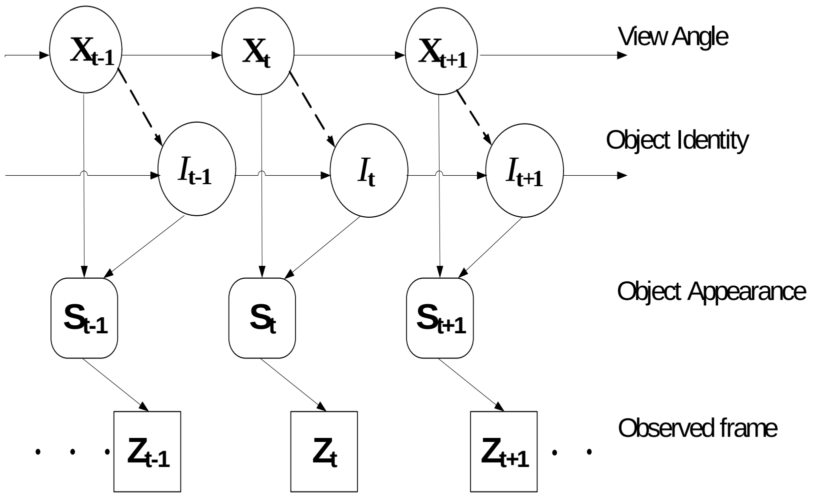


Figure 7.3: Graphical model for view and identity inference.

At the tth frame, given new hypotheses of generated along their own manifolds, the corresponding hypothesized appearance $\boldsymbol{S}_t$ can be reconstructed by the generative model defined in (7.6) given hypothesized I_t and $\boldsymbol{X}_t$. Then we can define the likelihood function by matching $\boldsymbol{S}_t$ with the observed shape $\boldsymbol{Z}_t$ as

$$p(\boldsymbol{Z}_t|I_t, \boldsymbol{X}_t) \propto \exp\Big[-\frac{\|\boldsymbol{Z}_t - \boldsymbol{S}_t\|^2}{2\sigma^2}\Big] \tag{7.7}$$

where σ^2 controls the likelihood sensitivity and $\|\cdot\|$ gives the mean square error between the observed and hypothesized object shapes.

We also need to specify two dynamic models, one along the circular-shaped identity manifold, and the other along the view manifold. Since the two latent variables have been decoupled in the generative model, we could define the two separate dynamic models to propagate the view and identity hypotheses along their own manifolds independently. However, it is perceivable that the identity dynamics should be influenced by the current view estimation (as indicated by the dotted line between the two variables in Fig. 7.3. It is mainly because that in some views (such as side views) the identity is much more distinguishable than other views (such as the rear/front/top views). Thus we define a view-sensitive prior $p(I_t|\boldsymbol{X}_{t-1}, I_{t-1})$ that propagates the identity hypothesis adaptively along the identity manifold according the current view/identity estimation, as shown in Fig. 7.5, which shows the amount of identity variation as a function of a view angle. While the view variable can follow a 2D random walk on the view manifold given by $p(\boldsymbol{X}_t|\boldsymbol{X}_{t-1})$. Then the joint dynamics of two variables can be defined by

$$p(I_t, \boldsymbol{X}_t|\Theta_{t-1}) = p(I_t|\boldsymbol{X}_{t-1}, I_{t-1})p(\boldsymbol{X}_t|\boldsymbol{X}_{t-1}), \tag{7.8}$$

where Θ_{t-1} is the previous state estimate. Then from (7.7) and (7.8), we can estimate posterior probability of latent variables sequentially via particle filtering as

$$p(I_t, \boldsymbol{X}_t|\boldsymbol{Z}_t) \propto p(\boldsymbol{Z}_t|I_t, \boldsymbol{X}_t)p(I_t, \boldsymbol{X}_t|\Theta_{t-1}). \tag{7.9}$$

The most computational demanding step is the likelihood calculation that involves online shape reconstruction from the generative model for each hypothesis during inference.

7.6 Experimental results

In this section, we first introduce the dataset involved in this work. Then we take a detailed look of the learnt identity manifold from the generative model, followed the recognition and pose estimation results under three cases.

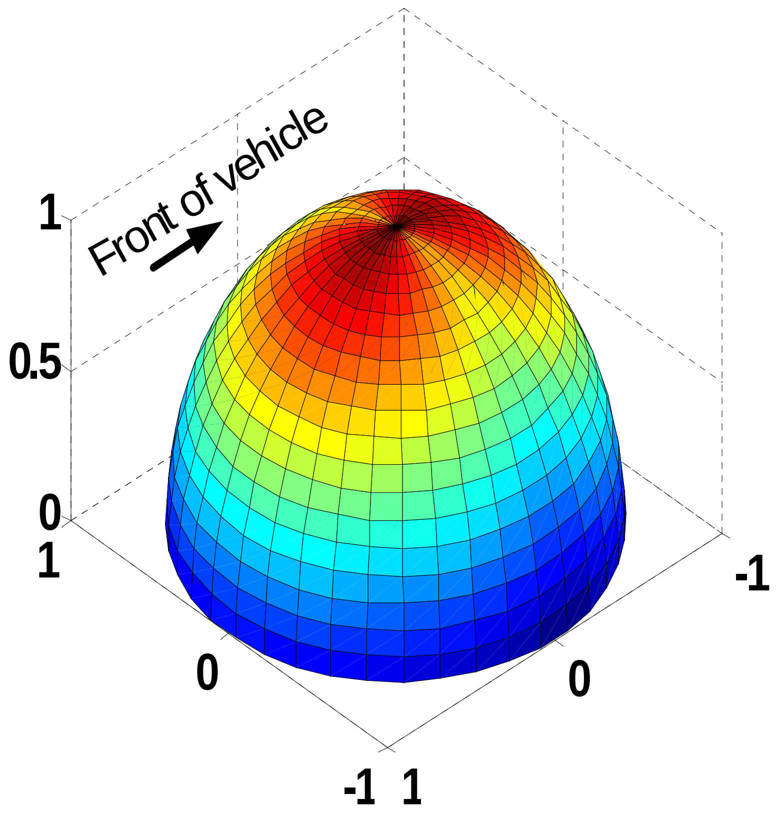


Figure 7.4: View-dependent identity dynamics where the color indicates the amount of variation allowed during sampling.

7.6.1 Dataset

We collected 3D models of 36 different vehicles (10 cars, 10 pickups, 10 SUVs and 6 mini-vans). Of these 29 models (8 cars, 8 pickups, 8 SUVs and 5 minivans) were used in the training phase and the rest used for testing. For each 3D model, we acquired a set of silhouettes corresponding to 200 training view points distributed on a view hemisphere. These training views are relatively sparser near the top of the view hemisphere and denser at the bottom. This is because there is less distinguishable shape variability in a top down view compared with that in a profile view. Using the silhouettes we can learn a generative model as described in Section 7.4 to obtain the identity vectors associated with all training examples according to (7.6).

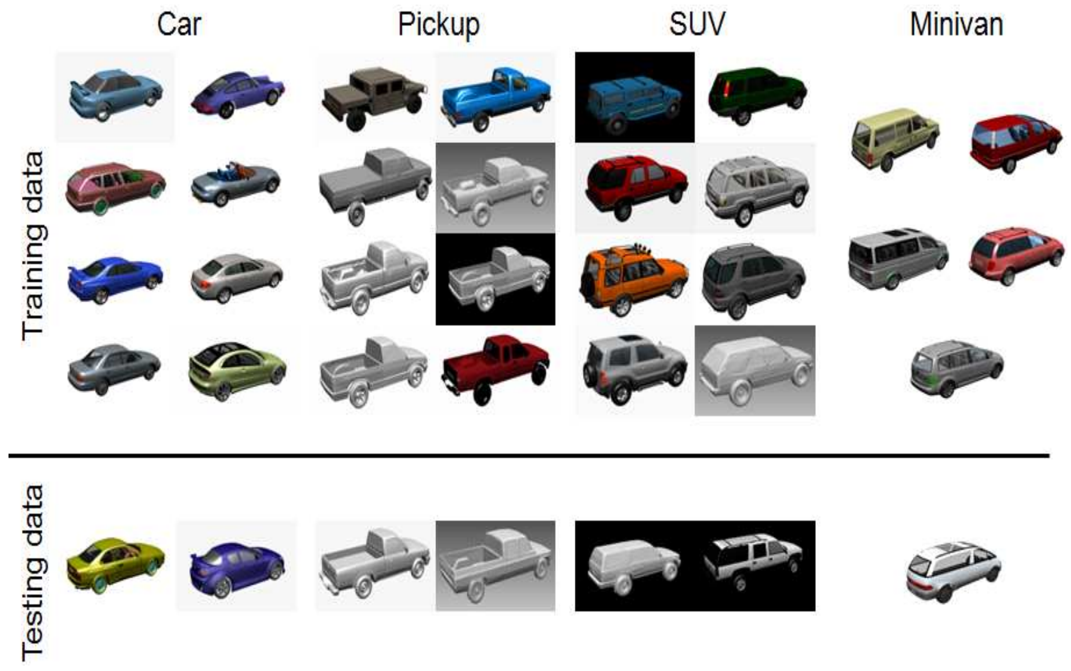


Figure 7.5: Illustration of the synthetic 3D vehicle dataset used in the experiments. Models from four different vehicle categories were used and these were divided into training and testing sets as shown.

7.6.2 Traversal along the identity manifold

Here we discuss the construction of the identity manifold and the reconstructed shapes when traversing along it. Given the manifold topology discussed in Section 7.2, wee can span a closed identity manifold by fitting a B-spline in the identity coefficient space defined in (7.6). Then a mapping is determined from this curve to a circle that allows us to parameterize the identity manifold by a 1D parameter.

To examine the validity of this identity manifold, we want to reveal the variation in each dimension of the identity coefficients as we traverse along the identity manifold, as shown in Fig. 7.6. Within each class, a relatively smooth transition can be observed in each dimension among sub-classes in the same class, while sharper transitions are noticeable between classes. For each class under a few representative views, we can interpolate new shapes by traversing along the identity manifold, as shown in Fig. 7.7, where the first and last columns are two adjacent training sub-classes for each of four classes while other columns show the interpolated shapes between the two sub-classes. Irrespective of the view angle, the identity manifold provides a smooth and meaningful transition of the interpolated shapes between every two training vehicles. For example, in row two, the pickup with a long-bed deforms slowly into one with a short-bed and larger cab. In row five, the height of the minivan reduces and its length increases from left to right, while all interpolated shapes maintain a discernible van shape.

7.6.3 Recognition and pose estimation

The proposed method was examined under three cases: (1) simulated video sequences with unknown vehicles , (2) real video sequences with a previously unseen vehicle, and (3) still images with silhouettes obtained by a segmentation process that is initialized manually.

In the first two experiments with simulated/real video sequences, we can obtain the continuous estimation of the identity and view variables. Specifically, the former one is represented by an angular parameter α ($0 \leq \alpha \leq 2\pi$) along the circular-shaped identity manifold, while the latter one includes the elevation angle ϕ ($0 \leq \phi \leq \pi/2$) and the azimuth angle θ ($0 \leq \theta \leq 2\pi$). In the first frame all the variables are uniformly distributed. In subsequent frames the particles are propagated and evaluated in accordance with (7.8) and (7.7), respectively. In the experiments with the still images, the identity and view are estimated by the exhaustive search along their own manifolds.

7.6.3.1 Simulated video data

Several simulated video sequences were created by moving the virtual camera along a smooth path on the view manifold which is centered with a previously unseen 3D vehicle model. Typical frames from one of these sequences shown in the first row of Fig7.8. In this example, the camera begins at a top-down view and progressively moves around the vehicle by reducing the elevation. The result of identity recognition and view estimation are shown in Fig.7.8 and Fig.7.9 respectively. Fig.7.8, we show the reconstructed shapes (the second row) and the corresponding view-based shapes (the last two rows) from the two nearest training vehicles which belong to the same class, showing the accurate identity estimation result. In Fig.7.9, there is an error of 180^o in the estimate of the azimuth angle (θ) in the first frame due to the shape ambiguity in the top-down view, this problem is quickly resolved as more frames become available. Actually, the car chosen for this experiment was longer than others in the “car” class and is easily confused with a pickup in the first few frames. However, as we progress towards lower elevation angles the identity estimate consistently lies between a four-door sedan and a two-door sedan.

7.6.3.2 Real video data

In this experiment we try to estimate the pose and identity of a toy vehicle in an indoor environment whose 3D model not available in the training data. A few closed-up view of this toy vehicle is shown in Fig. 7.10(a). A few sample image frames are shown in the first row Fig.7.11. Background subtraction was used to obtain the silhouettes of the moving vehicle as shown in the second row of Fig.7.11. The first 140 frame of this video contain only the background and is used for initializing the background subtraction algorithm. The two adjacent training vehicles from the same SUV class are correctly identified in most instances. In frame 238 the vehicle partially moves out of the scene resulting in a distorted silhouette and then a slightly wrong pose estimate, which leads to some changes in the identity compared to previous frame. However, as the vehicle moves back into full view the pose and identity estimate stabilize to nominal values. The estimated view points shown in Fig.7.10(b) are also consistent with the motion pattern of the toy vehicle that moves away from the camera, makes a left turn and then moves in reverses to face the camera. This experiment demonstrates the ability of our model to correctly identify unknown vehicle classes and relate them with existing models in the training set.

7.6.3.3 Still images

In this experiment, several different still images of vehicles were selected and each image was initialized with a few seed points manually to obtain a segmentation based on graph cuts. Sample images along with their segmentation results are shown along the first three columns in Fig. 7.12. In all the examples shown here, the identification of the vehicle class was found to be accurate. The result of pose estimation is also fairly accurate with only the car in the last row having its pose mis-estimated as facing backward. Being able to identify the vehicle class and pose even when the segmentation result is noisy, deformed or missing parts proves the robustness of the proposed approach.

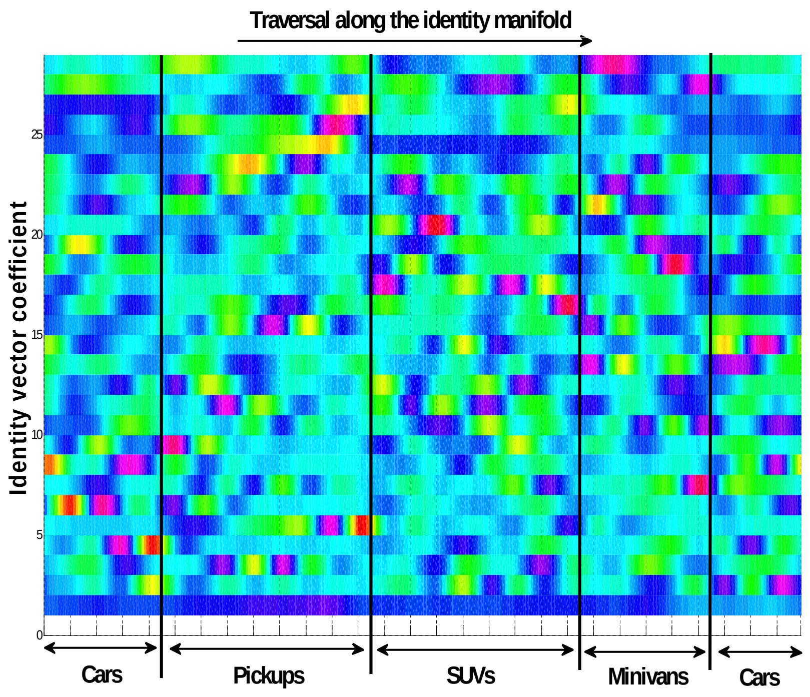


Figure 7.6: Variation in each dimension of the identity coefficient as we travel along the identity manifold between different classes are shown along each row. Class boundaries are indicated by steep transitions smoothed by a spline along one or more of the rows. The bottom row has the lowest variance and the top row has the maximum variance across different classes. Further, a clustering phenomenon is also observed with vehicles of one class having dominant peaks along certain dimensions in the coefficient space. For example, the cars have peaks concentrated along dimensions of lower variance indicating that the car models used for training do not vary significantly within their class, whereas the minivans with only 5 training examples have peaks along dimensions of larger variance indicating larger intra-class variation.

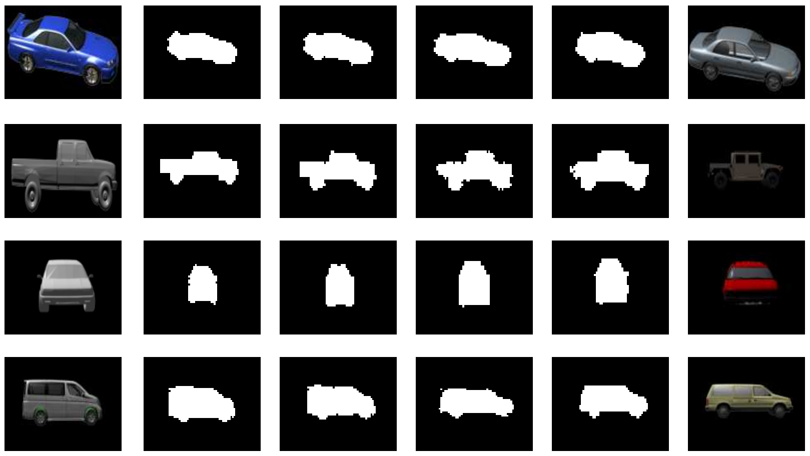


Figure 7.7: First and last columns: 3D models of two adjacent training vehicles along the identity manifold. Columns two through four: Interpolated shape when moving along the identity manifold.

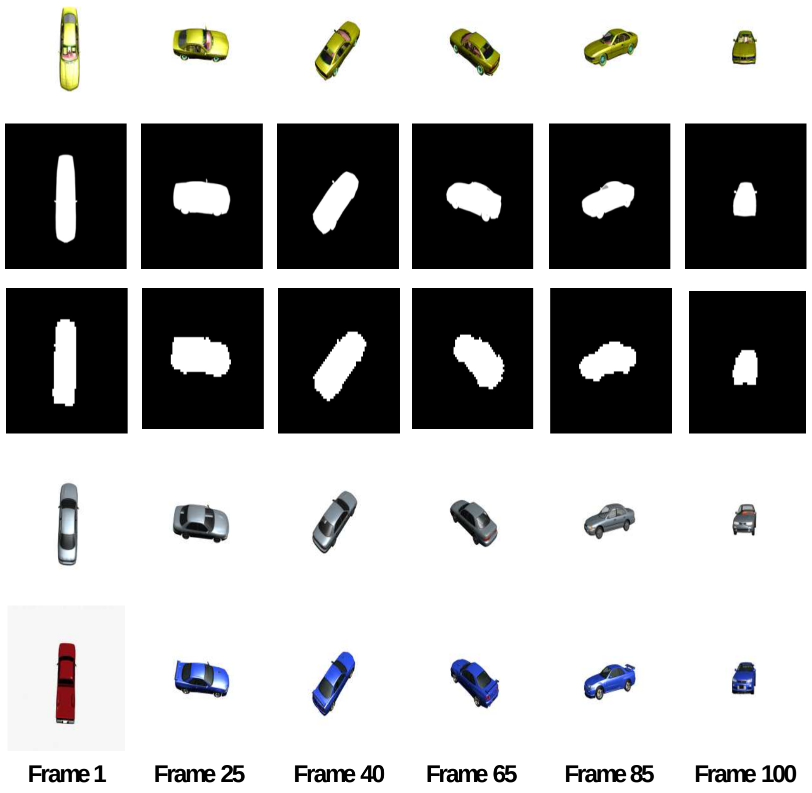


Figure 7.8: Results of pose and identity estimation in a simulated video sequence with a previously unknown car. First row: Observed silhouette. Second row: The closest reconstructed shape match. Third and fourth rows : Two of the closest neighbors along the identity manifold, shown in the estimate pose.

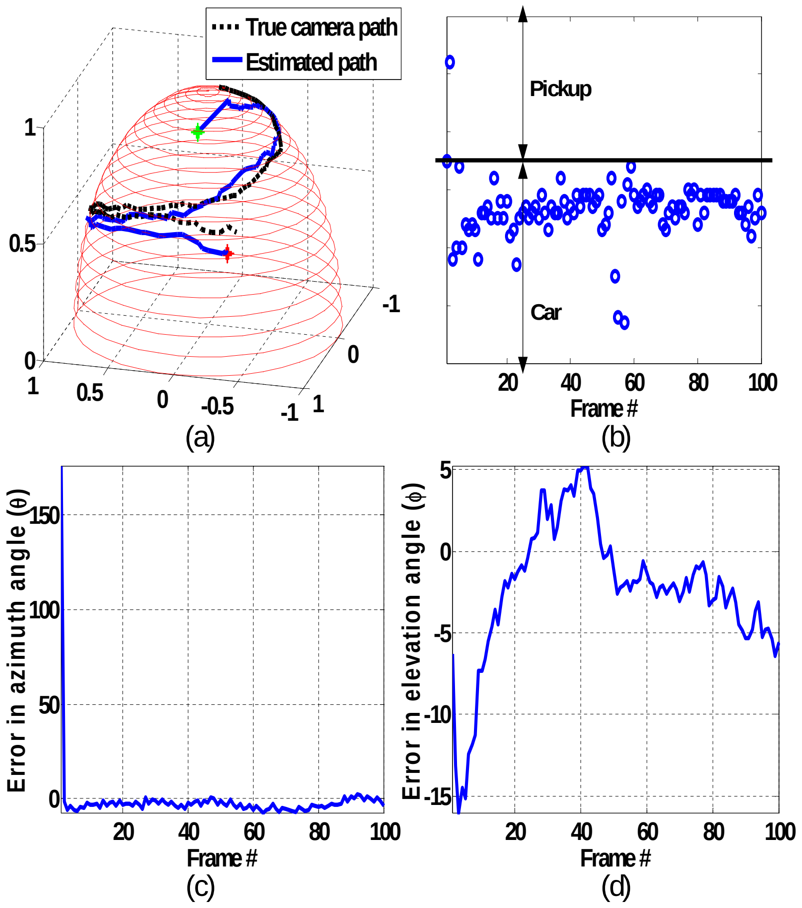
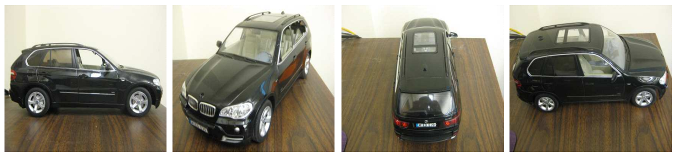
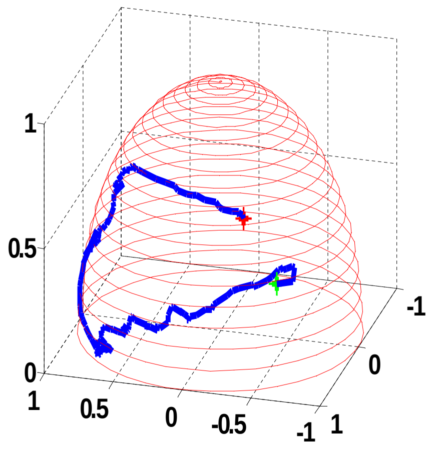


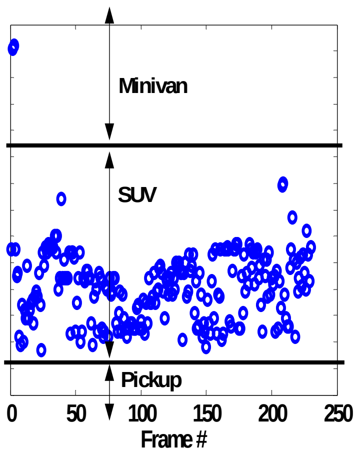
Figure 7.9: (a) Plot of the true and estimated camera path along the view manifold. The green marker indicates the first frame and the red marker the last frame. (b) Estimate of the object identity shown in a subsection of the identity manifold (c) and (d) Error in the estimated azimuth (θ) and elevation (ϕ) angles. The average errors were found to be 8.4^o for the azimuth and 3.6^o for the elevation angles



(a)



(b)



(c)

Figure 7.10: (a) Sample views of the toy vehicle used in this experiment; (b) The estimated camera path along the view manifold. The green and red marker indicates the first and last frames respectively. (c) Estimate of the vehicle identity shown in a subsection of the identity manifold.

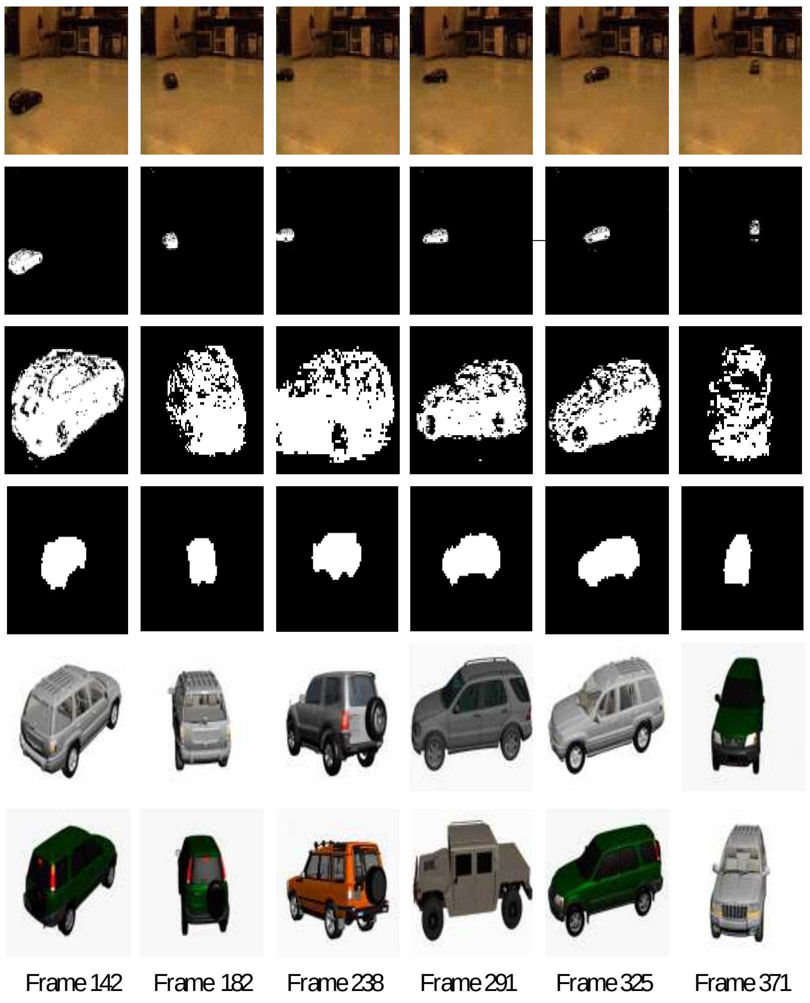


Figure 7.11: Results of pose and identity estimation in video acquired using a toy vehicle. From the first to last rows: (1) original video frame, (2) segmented object shape using background subtraction, (3) closer view of the segmented result in row two, (4) the best matching interpolated shapes (re-scaled), (5) and (6) two of the closest training vehicles along the identity manifold shown in the estimated pose.

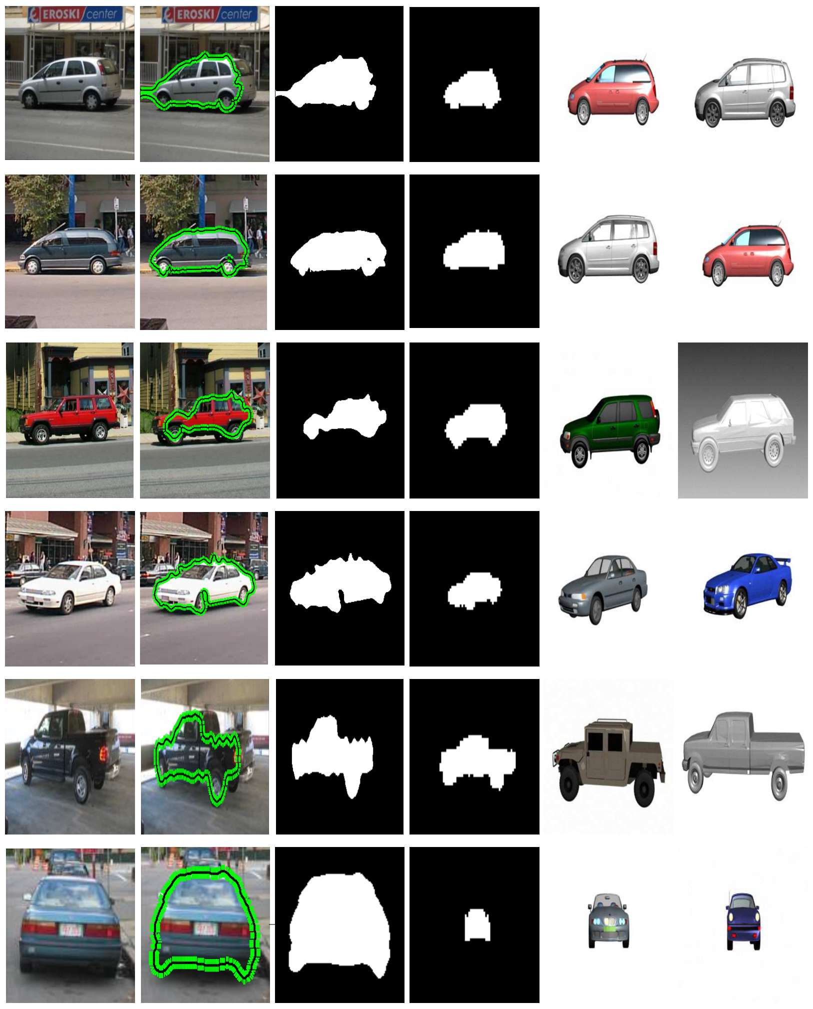


Figure 7.12: Example of pose and identity recognition in some real images. From the first to the last columns: original images, segmented results, segmented silhouettes, reconstructed shapes, two adjacent training vehicles along the identity manifold.

CHAPTER 8

Conclusions and Future Work

8.1 Conclusions

This dissertation presented the components of an integrated ATR system namely: target detection, target tracking, appearance learning and joint tracking and recognition.

- **Detection:** We have shown that high-level combinations of basic relational features can be used in a boosting framework to construct very fast classifiers that are as competitive as SVM-RBF while requiring only a fraction of the computational load. The proposed RelCom classifier was able to successfully detect small targets in complex environments.

- **Tracking:** A new target tracking algorithm for FLIR imagery was developed, that supports accurate target localization and size estimation. Specifically, a dual foreground-background target appearance model was proposed, that integrates local statistics of both background and foreground to enhance the tracker's sensitivity. Moreover, an online feature selection technique was presented that can select optimal features by maximizing the confidence of the state estimation. Both target tracking and feature selection are unified in a probabilistic framework where a coupled particle filtering approach is involved for sequential state estimation.

- **Appearance Learning:** We have discussed infrared tracking under a unified co-inference framework where both Kalman filtering and particle filtering are used to update target appearance and to estimate target kinematics respectively. The contribution of this research is how to robustly and reliably update the target appearance represented by multiple histograms during the tracking process. Particularly, we proposed a new AKF-based appearance learning method, $\mathrm{AKF}_{\mathrm{als}}$, which was compared with the two existing techniques ($\mathrm{AKF}_{\mathrm{cov}}$, HS) and was found to be superior.

- **Integrated Tracking and Recognition:** A unified framework for integrated target tracking and recognition was discussed, where joint appearance-motion generative models are incorporated in a graphical model. The target type and kinematics are jointly estimated by a particle filter based inference algorithm embedded with Kalman filters. The experimental results demonstrate that the joint appearance-motion models improve both tracking accuracy and recognition performance when compared to the ones using the template-based appearance model and a single motion model. Also, it was shown that the embedded Kalman filters provide better samples for both kinematics and the target type by exploiting the type-dependent motion models, leading to further improved tracking and recognition performance.

- **Recognition using Identity and View Manifolds:** Finally, we presented a continuous-valued identity manifold for object recognition that captures both inter-class and intra-class shape variability for a set of similar objects, i.e., road vehicles, which can be grouped into a few major classes each of which has several sub-classes. This identity manifold allows us to recognize not only a known vehicle, but also unknown ones by meaningfully interpolating between two training vehicles. Additionally, we develop a new multi-view shape-based generative model that integrates a hemisphere-shaped view manifold with this identity manifold to provide simultaneous identity and pose estimation.

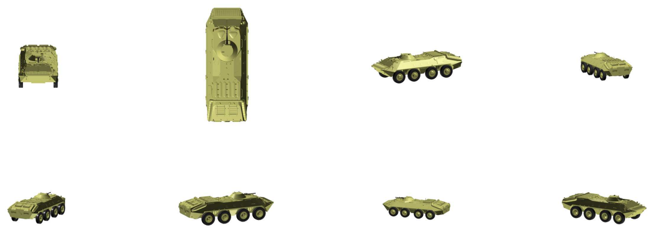
8.2 Future work

8.2.1 Drawbacks of the existing generative appearance models

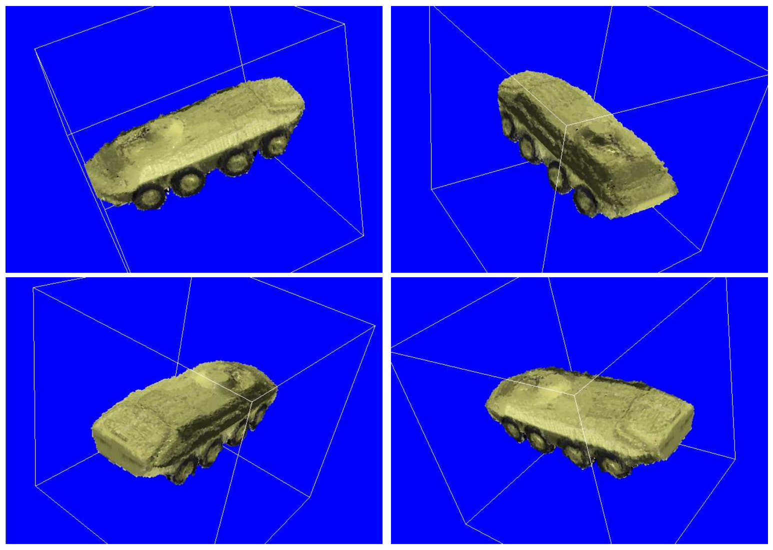
In the preceding chapters we had developed a non-linear generative model for target appearance synthesis that is dependent on the view angle and identity. This model was learnt based on synthetic data and was found to work efficiently. However, to learn the components of this model there are two main requirements 1) clean target template chips and 2) information of the pose associated with each template. In real videos, segmenting out a clean template from the image and assessing the pose information are both difficult tasks. We had also discussed issues with appropriate selection of the conceptual manifold shape in the previous section. Another important issue with infrared imagery is that the target appearance is a function of temperature profile and the environmental conditions. As an example, a target would appear bright if its engine were to be running over a period of time or if its surface was warmed up by the sun. Therefore a single target can exhibit multiple intensity profiles depending on internal and external factors. The generative model discussed previously concerns mainly with the target shape, rather than the intensity profile, and assumes a constant intensity profile for simplicity, which is hardly the case in the real world. Also if the appearance of the target were to change the generative model would have to be re-learnt. Therefore, it is necessary to develop an appearance model that can accommodate time varying intensity profiles.

8.2.2 Development of a 3D thermal appearance model

To overcome the limitations discussed above, as future work, we propose to develop a novel 3D thermal appearance model that is learnt directly from the observed image sequence. We propose to use the technique of voxel coloring [138, 139] which is commonly used in computer graphics to reconstruct the 3D shape of an objects from 2D images. In this way, we will be able to reconstruct not only the 3D shape of the target, but also its thermal



(a)



(b)

Figure 8.1: Example illustrating the reconstruction of 3D shape from 2D images using voxel coloring. (a) The 2D input images used for voxel coloring. (b) Views from multiple angles of the reconstructed 3D model.

intensity texture. An example of the results obtained using voxel coloring is shown in Fig. 8.1. Here we observe that both the shape and texture information are accurately rendered in the reconstructed model. Once this 3D textured model is developed it can be used to learn a generative appearance model including the factors of identity, and viewing angle and used in a tracking and recognition framework. In addition to the voxel coloring, we hope to improve the reconstructed shape by augmenting information from a known 3D CAD model of the target.

BIBLIOGRAPHY

[1] B. Bhanu, D. Dudgeon, E. Zelnio, A. Rosenfeld, D. Casasent, and I. Reed, "Introduction to the special issue on automatic target detection and recognition," *IEEE Trans. Image Processing*, vol. 6, no. 1, pp. 1–6, 1997.

[2] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, p. 13, 2006.

[3] C. Harris and M. Stephens, "A combined corner and edge detection," in *Proc. of The Fourth Alvey Vision Conference*, pp. 147–151, 1988.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int'l Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[5] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[6] L. Zhang, B. Wu, and R. Nevatia, "Pedestrian detection in infrared images based on local shape features," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2007.

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

[8] J. F. Khan and M. S. Alam, "Target detection in cluttered forward-looking infrared imagery," *Optical Engineering*, vol. 44, no. 7, p. 076404, 2005.

[9] U. Braga-Neto, M. Choudhary, and J. Goutsias, “Automatic target detection and tracking in forward-looking infrared image sequences using morphological connected operators,” *Journal of Electronic Imaging*, vol. 13, no. 4, pp. 802–813, 2004.

[10] N. Oliver, B. Rosario, and A. Pentland, “A bayesian computer vision system for modeling human interactions,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 831–843, 2000.

[11] A. Monnet, A. Mittal, N. Paragios, and R. Visvanathan, “Background modeling and subtraction of dynamic scenes,” in *Proc. IEEE Int’l Conference on Computer Vision*, 2003.

[12] C. Papageorgiou and T. Poggio, “A trainable system for object detection,” *Int’l Journal of Computer Vision*, vol. 38, no. 1, pp. 15–33, 2000.

[13] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[14] S. Belongie, J. Malik, and J. Puzicha, “Shape matching and object recognition using shape contexts,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[15] O. Tuzel, F. Porikli, and P. Meer, “Pedestrian detection via classification on Riemannian manifolds,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1713–1727, 2008.

[16] P. Dollár, Z. Tu, H. Tao, and S. Belongie, “Feature mining for image classification,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[17] J. Yuan, J. Luo, and Y. Wu, “Mining compositional features for boosting,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[18] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[19] D. Comaniciu, V. Ramesh, and P. Meer, “Kernel-based object tracking,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

[20] C. J. Veenman, M. Reinders, and E. Backer, “Resolving motion correspondence for densely moving points,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 1, pp. 54–72, 2001.

[21] D. Beymer and K. Konolige, “Real-time tracking of multiple people using continuous detection,” in *Proc. IEEE Int’l Conference on Computer Vision*, 1999.

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

[23] D. Reid, “An algorithm for tracking multiple targets,” *IEEE Trans. Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[24] B. Li and R. Chellappa, “Simultaneous tracking and verification via sequential posterior estimation,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[25] L. Tang, V. Venkataraman, and G. Fan, “Multi-aspect target tracking in image sequences using particle filters,” in *Proc. Int’l Symposium on Visual Computing*, 2005.

[26] E. Maggio and A. Cavallaro, “Hybrid particle filter and mean shift tracker with adaptive transition model,” in *Proc. IEEE Int’l Conference on Acoustics, Speech, and Signal Processing*, 2005.

[27] A. Yilmaz, K. Shafique, and M. Shah, “Tracking in airborne forward looking infrared imagery,” *Image and Vision Computing*, vol. 21, no. 7, pp. 623–635, 2003.

[28] M. Isard and A. Blake, “Condensation-conditional density propagation for visual tracking,” *Int’l Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[29] A. Jepson, D. Fleet, and T. El-Maraghi, “Robust online appearance models for visual tracking,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1296–1311, 2003.

[30] B. Han, Y. Zhu, D. Comaniciu, and L. Davis, “Kernel-based bayesian filtering for object tracking,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[31] J. Shaik and K. Iftekharuddin, “Automated tracking and classification of infrared images,” in *Proc. Int’l Joint Conference on Neural Networks*, 2003.

[32] A. Dawoud, M. Alam, A. Bal, and C. Loo, “Target tracking in infrared imagery using weighted composite reference function-based decision fusion,” *IEEE Trans. Image Processing*, vol. 15, no. 2, pp. 404–410, 2006.

[33] Z. Wang, Y. Wu, J. Wang, and H. Lu, “Target tracking in infrared image sequences using diverse AdaBoostSVM,” in *Proc. Int’l Conference on Innovative Computing, Information and Control*, 2006.

[34] L. Matthews, T. Ishikawa, and S. Baker, “The template update problem,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 810–815, 2004.

[35] S. Lankton, J. Malcolm, A. Nakhmani, and A. Tannenbaum, “Tracking through changes in scale,” in *Proc. IEEE Int’l Conference on Image Processing*, 2008.

[36] H. Nguyen and A. Smeulders, “Fast occluded object tracking by a robust appearance filter,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 8, pp. 1099–1104, 2004.

[37] N. Peng, J. Yang, and Z. Liu, “Mean shift blob tracking with kernel histogram filtering and hypothesis testing,” *Pattern Recognition Letters*, vol. 26, no. 5, pp. 605–614, 2005.

[38] C. Zhang and Y. Rui, “Robust visual tracking via pixel classification and integration,” in *Proc. Int’l Conference on Pattern Recognition*, 2006.

[39] I. Leichter, M. Lindenbaum, and E. Rivlin, “Tracking by affine kernel transformations using color and boundary cues,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 31, no. 1, pp. 164–171, 2009.

[40] H. Nguyen, M. Worring, and R. van der Boomagaard, “Occlusion robust adaptive template tracking,” in *Proc. IEEE Int’l Conference on Computer Vision*, 2001.

[41] J. Pan and B. Hu, “Robust object tracking against template drift,” in *IEEE Int’l Conference on Image Processing*, 2007.

[42] R. Mehra, “Approaches to adaptive filtering,” *IEEE Trans. Automatic Control*, vol. 17, no. 5, pp. 693–698, 1972.

[43] L. Latecki and R. Miezianko, “Object tracking with dynamic template update and occlusion detection,” in *Proc. Int’l. Conference on Pattern Recognition*, 2006.

[44] Y. Wu, T. Yu, and G. Hua, “Tracking appearances with occlusions,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[45] T. Poggio and S. Edelman, “A network that learns to recognize three-dimensional objects,” *Nature*, vol. 343, pp. 263–266, 1990.

[46] S. Ullman and R. Basri, “Recognition by linear combinations of models,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 10, pp. 992–1006, 1991.

[47] S. Ullman, “An approach to object recognition: Aligning pictorial descriptions,” *Cognition*, vol. 32, pp. 193–254, 1989.

[48] O. Ozcanli, A. Tamrakar, and B. Kimia, “Augmenting shape with appearance in vehicle category recognition,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[49] H. Su, M. Sun, L. Fei-Fei, and S. Savarese, “Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories,” in *Proc. IEEE Int’l Conference on Computer Vision*, 2009.

[50] R. Savarese S., Fergus and L. Fei-Fei, “Multi-view object categorization and pose estimation,” in *Computer Vision*, vol. 285 of *Studies in Computational Intelligence*, Springer, 2010.

[51] A. Toshev, A. Makadia, and D. K., “Shape-based object recognition in videos using 3D synthetic object models.,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[52] A. Kushal, C. Schmid, and J. Ponce, “Flexible object models for category-level 3D object recognition.,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[53] J. Lou, T. Tan, W. Hu, H. Yang, and S. Maybank, “3D model-based vehicle tracking,” *IEEE Trans. Image Processing*, vol. 14, no. 10, pp. 1561–1569, 2005.

[54] M. Leotta and J. Mundy, “Predicting high resolution image edges with a generic, adaptive, 3D vehicle model.,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[55] R. Sandhu, S. Dambreville, A. Yezzi, and T. A., “Non-rigid 2D-3D pose estimation and 2D image segmentation,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[56] Y. Tsin, Y. Gene, and V. Ramesh, “Explicit 3D modeling for vehicle monitoring in non-overlapping cameras,” in *Proc. IEEE Int’l Conference on Advanced Video and Signal-Based Surveillance*, 2009.

[57] J. Liebelt and C. Schmid, “Multi-view object class detection with a 3D geometric model,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[58] H. Bülthoff and S. Edelman, “Psychophysical support for a 2D view interpolation theory of object recognition,” in *Proc. of the National Academy of Science*, vol. 89, pp. 60–64, 1992.

[59] Z. Zhang, W. Dong, K. Huang, and T. Tan, “EDA approach for model based localization and recognition of vehicles,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[60] S. Zhou, R. Chellappa, and B. Moghaddam, “Adaptive visual tracking and recognition using appearance-adaptive models in particle filters,” *IEEE Trans. Image Processing*, vol. 13, no. 11, pp. 1491–1505, 2004.

[61] H. T. Nguyen, Q. Ji, and A. W. Smeulders, “Spatio-temporal context for robust multitarget tracking,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 52–64, 2007.

[62] K.-C. Lee, J. Ho, M.-H. Yang, and D. Kriegman, "Visual tracking and recognition using probabilistic appearance manifolds," *Computer Vision and Image Understanding*, vol. 99, no. 3, pp. 303–331, 2005.

[63] H. Murase and S. Nayar, "Visual learning and recognition of 3D objects from appearance.," *Int'l Journal of Computer Vision*, vol. 14, no. 1, pp. 5–24, 1995.

[64] J. B. Tenenbaum and W. T. Freeman, "Separating style and content with bilinear models," *Neural Computation*, vol. 12, no. 6, pp. 1247–1283, 2000.

[65] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces," in *Proc. European Conference on Computer Vision*, 2002.

[66] C. Gosch, K. Fundana, A. Heyden, and C. Schnörr, "View point tracking of rigid objects based on shape sub-manifolds," in *Proc. European Conference on Computer Vision*, 2008.

[67] C. S. Lee and A. Elgammal, "Modeling view and posture manifold for tracking," in *Proc. IEEE Int'l Conference on Computer Vision*, 2007.

[68] X. Li and V. Jilkov, "Survey of maneuvering target tracking. Part V: Multiple-model methods," *IEEE Trans. Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1255–1321, 2005.

[69] S. Maskell, "Joint tracking of maneuvering targets and classification of their maneuverability," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 1, pp. 2339–2350, 2004.

[70] D. Angelova and L. Mihaylova, "Joint target tracking and classification with particle filtering and mixture kalman filtering using kinematic radar information," *Digital Signal Processing*, vol. 16, no. 2, pp. 180–204, 2006.

[71] X. Fan, G. Fan, and J. Havlicek, "Generative model for maneuvering target tracking," *IEEE Trans. Aerospace and Electronic Systems*, vol. 46, no. 2, pp. 635–655, 2010.

[72] R. Rosales and S. Sclaroff, "3D trajectory recovery for tracking multiple objects and trajectory guided recognition of actions," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1999.

[73] F. Bashir, A. Khokhar, and D. Schonfeld, "Object trajectory-based activity classification and recognition using hidden markov models," *IEEE Trans. Image Processing*, vol. 16, no. 7, pp. 1912–1919, 2007.

[74] Z. Hu, X. Fan, Y. Song, and D. Liang, "Joint trajectory tracking and recognition based on bi-directional nonlinear learning," *Image and Vision Computing*, vol. 27, no. 9, pp. 1302–1312, 2009.

[75] C. Hu, Y. Chang, R. Feris, and M. Turk, "Manifold based analysis of facial expression," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2004.

[76] D. Fadi and D. Franck, "Simultaneous facial action tracking and expression recognition in the presence of head motion," *Int'l Journal of Computer Vision*, vol. 76, no. 3, pp. 257–281, 2008.

[77] Z. Shaohua, V. Krueger, and R. Chellappa, "Probabilistic recognition of human faces from video," *Computer Vision and Image Understanding*, vol. 91, no. 1-2, pp. 214–245, 2003.

[78] K. Minyoung, S. Kumar, V. Pavlovic, and H. Rowley, "Face tracking and recognition with visual constraints in real-world videos," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[79] C. Huang, H. Ai, Y. Li, and S. Lao, “Learning sparse features in granular space for multi-view face detection,” in *Proc. IEEE Int’l Conference on Automatic Face and Gesture Recognition*, 2006.

[80] G. Duan, C. Huang, H. Ai, and S. Lao, “Boosting associated pairing comparison features for pedestrian detection,” in *Proc. IEEE Int’l Workshop on Visual Surveillance*, 2009.

[81] Y. Freund and R. E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” *Journal of Computer System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[82] R. Collins, Y. Liu, and M. Leordeanu, “Online selection of discriminative tracking features,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631–1643, 2005.

[83] B. Han and L. Davis, “Robust observations for object tracking,” in *Proc. IEEE Int’l Conference on Image Processing*, 2005.

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

[85] D. Comaniciu, V. Ramesh, and P. Meer, “Real-time tracking of non-rigid objects using mean shift,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[86] M. Swain and D. Ballard, “Indexing via color histograms,” in *Proc. Int’l Conference on Computer Vision*, 1990.

[87] T. Han, M. Liu, and T. Huang, “A drifting-proof framework for tracking and online appearance learning,” in *Proc. IEEE Workshop on Applications of Computer Vision*, 2007.

[88] G. Harger and P. Belhumeur, “Real-time tracking of image regions with changes in geometry and illumination,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1996.

[89] Z. Peng, Q. Zhang, and A. Guan, “Extended target tracking using projection curves and matching pel count,” *Optical Engineering*, vol. 46, no. 6, pp. 066401–1–066401–6, 2007.

[90] C. Johnston, N. Mould, J. Havlicek, and G. Fan, “Dual domain auxiliary particle filter with integrated target signature update,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2009.

[91] J. Khan and M. Alam, “Efficient target detection in cluttered FLIR imagery,” in *Optical Pattern Recognition XVI*, vol. 5816 of *Proc. SPIE*, pp. 39–53, 2005.

[92] S. Yi and L. Zhang, “A novel multiple tracking system for UAV platforms,” in *Airborne Intelligence, Surveillance, Reconnaissance (ISR) Systems and Applications III*, vol. 6209 of *Proc. SPIE*, p. 620901, 2006.

[93] A. Dawoud, M. Alam, A. Bal, and C. Loo, “Decision fusion algorithm for target tracking in infrared imagery,” *Optical Engineering*, vol. 44, no. 2, p. 026401, 2005.

[94] C. del Blanco, F. Jaureguizar, N.Garcìa, and L. Salgado, “Robust automatic target tracking based on a Bayesian ego-motion compensation framework for airborne FLIR imagery,” in *Automatic Target Recognition XIX*, vol. 7335 of *Proc. SPIE*, p. 733514, 2009.

[95] N. Mould, C. Nguyen, C. Johnston, and J. Havlicek, "Online consistency checking for AM-FM target tracks," in *Proc. SPIE/IS&T Conference on Computational Imaging VI*, vol. 6814 of *Proc. SPIE*, 2008.

[96] V. Venkataraman, G. Fan, and X. Fan, "Target tracking with online feature selection in FLIR imagery," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2007.

[97] Y. Wu and T. S. Huang, "Robust visual tracking by integrating multiple cues based on co-inference learning," *International Journal of Computer Vision*, vol. 58, no. 1, pp. 55–71, 2004.

[98] B. Odelson, R. Rajamani, and B. Rawlings, "A new autocovariance least-squares method for estimating noise covariances," *Automatica*, vol. 42, no. 2, pp. 303–308, 2006.

[99] B. Odelson, A. Lutz, and J. Rawlings, "The autocovariance least-squares method for estimating covariances: application to model-based control of chemical reactors," *IEEE Trans. Control Systems Technology*, vol. 14, no. 3, pp. 532–540, 2006.

[100] G. Noriega and S. Pasupathy, "Adaptive estimation of noise covariance matrices in real-time preprocessing of geophysical data," *IEEE Trans. Geoscience and Remote Sensing*, vol. 35, no. 5, pp. 1146–1159, 1997.

[101] X. Li and Y. Bar-Shalom, "A recursive multiple model approach to noise identification," *IEEE Trans. Aerospace and Electronic Systems*, vol. 30, no. 3, pp. 671–684, 1994.

[102] A. H. Mohamed and K. P. Schwarz, "Adaptive Kalman filtering for INS/GPS," *Journal of Geodesy*, vol. 73, no. 4, pp. 193–203, 1999.

[103] H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer, 2 ed., 1994.

[104] R. Mehra, “On the identification of variances and adaptive kalman filtering,” *IEEE Trans. Automatic Control*, vol. 15, no. 2, pp. 175–184, 1970.

[105] B. Carew and P. Belanger, “Identification of optimum filter steady-state gain for systems with unknown noise covariances,” *IEEE Trans. Automatic Control*, vol. 18, no. 6, pp. 582–587, 1973.

[106] C. Neethling and P. Young, “Comments on ”identification of optimum filter steady-state gain for systems with unknown noise covariances”,” *IEEE Trans. Automatic Control*, vol. 19, no. 5, pp. 623–625, 1974.

[107] B. Odelson, *Estimating Disturbance Covariances from Data for Improved Control Performance*. PhD thesis, University of Wisconsin, Madision, WI, 2003.

[108] M. Rajamani, J. Rawlings, and T. Soderstrom, “Application of a new data-based covariance estimation technique to a nonlinear industrial blending drum,” Tech. Report 2007-03, Texas-Winsconsin Modeling and Control Consortium, 2007.

[109] D. L. Donoho, S. Mallat, R. von Sachs, and Y. Samuelides, “Locally stationary covariance and signal estimation with macrotiles,” *IEEE Trans. Signal Processing.*, vol. 51, no. 3, pp. 614–627, 2003.

[110] I. Tabus and A. Vasiache, “Low bit rate vector quantization of outlier contaminated data based on shells of Golay codecs,” in *Proc. Data Compresion Conference.*, 2009.

[111] T. V. Ramabadran and D. Sinha, “Speech data compression through sparse coding of innovations,” *IEEE Trans. Speech, Audio Processing.*, vol. 2, no. 2, pp. 274–284, 1994.

[112] S. Boucheron and M. R. Salamatian, “About priority encoding transmission,” *IEEE Trans. Information Theory*, vol. 46, no. 2, pp. 699–705, 2000.

[113] M. Raginsky, “Joint fixed-rate universal lossy coding and identification of continuous-alphabet memoryless sources,” *IEEE Trans. Information Theory*, vol. 54, no. 7, pp. 3059–3077, 2008.

[114] A. Cohen, I. Daubechies, O. G. Guleryuz, and M. T. Orchard, “On the importance of combining wavelet-based nonlinear approximation with coding strategies,” *IEEE Trans. Info. Theory*, vol. 48, no. 7, pp. 1895–1921, 2002.

[115] G. I. Shamir and D. J. Costello, “Asymptotically optimal low-complexity sequential lossless coding for piecewise-stationary memoryless sources – Part I: the regular case,” *IEEE Trans. Info. Theory*, vol. 46, no. 7, pp. 2444–2467, 2000.

[116] S. Avidan, “Ensemble tracking,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[117] H. Chen, T. Liu, and C. Fuh, “Probabilistic tracking with adaptive feature selection,” in *Proc. Int’l Conference on Pattern Recognition*, 2004.

[118] R. Powers and L. Pao, “Using Kolmogorov-Smirnov tests to detect track-loss in the absence of truth data,” in *Proc. IEEE Conference on Decision and Control*, 2005.

[119] K. She, G. Bebis, H. Gu, and R. Miller, “Vehicle tracking using on-line fusion of color and shape features,” in *Proc. IEEE Int’l Conference on Intelligent Transportation Systems*, 2004.

[120] F. Porikli, O. Tuzel, and P. Meer, “Covariance tracking using model update based on lie algebra,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[121] X. Li, W. Hu, Z. Zhang, X. Zhang, M. Zhu, and J. Cheng, "Visual tracking via incremental Log-Euclidean Riemannian subspace learning," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[122] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: A fast descriptor for detection and classification," in *Proc. European Conference on Computer Vision*, 2006.

[123] X. Mei, S. Zhou, and H. Wu, "Integrated detection, tracking and recognition for IR video-based vehicle classification," in *Proc. IEEE Int'l Conference on Acoustics, Speech and Signal Processing.*, 2006.

[124] M. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of American Statistical Association*, vol. 94, no. 446, pp. 590–599, 1999.

[125] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[126] L. D. Lathauwer, B. D. Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.

[127] R. Chellappa, C. Wilson, and S. Sirohey, "Human and machine recognition of faces: a survey," *Proceedings of the IEEE*, vol. 83, no. 5, pp. 705–741, 1995.

[128] A. Elgammal and C. S. Lee, "Separating style and content on a nonlinear manifold," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[129] J. Chang, Y. Zheng, and Z. Wang, "Facial expression analysis and synthesis : a bilinear approach," in *Proc. Int'l Conference on Information Acquisition*, 2007.

[130] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1481–1497, 1990.

[131] A. Doucet, N. Gordon, and V. Krishnamurthy, “Particle filters for state estimation of jump Markov linear systems,” *IEEE Trans. Signal Processing*, vol. 49, no. 3, pp. 613–624, 2001.

[132] N. de Freitas, R. Dearden, F. Hutter, R. Morales-Menendez, J. Mutch, and D. Poole, “Diagnosis by a waiter and a mars explorer,” *Proceedings of the IEEE*, vol. 92, no. 3, pp. 455–468, 2004.

[133] K. Mikolajczyk, “Multiple object class detection with a generative model,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[134] Y. Li, L. Gu, and T. Kanade, “A robust shape model for multi-view car alignment,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[135] R. Fergus, P. Perona, and A. Zisserman, “Object class recognition by unsupervised scale-invariant learning,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[136] D. Lowe, “Local feature view clustering for 3D object recognition.,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[137] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce, “3D object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints.,” *Int’l Journal of Computer Vision*, vol. 66, no. 3, pp. 231–259, 2006.

[138] S. M. Seitz and C. R. Dyer, “Photorealistic scene reconstruction by voxel coloring,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[139] W. B. Culbertson, T. Malzbender, and G. G. Slabaugh, “Generalized voxel coloring,” in *Proc. Int’l Workshop on Vision Algorithms: Theory and Practice*, 2000.

VITA

Vijay Venkataraman

Candidate for the Degree of

Doctor of Philosophy

Dissertation: ADVANCED MACHINE LEARNING APPROACHES FOR TARGET DETECTION, TRACKING AND RECOGNITION

Major Field: Electrical Engineering

Biographical:

Personal Data: Born in Mannargudi, Tamil Nadu, India on November 21, 1981.

Education:

Received the B.Tech degree from Indian Institute of Technology Madras (IITM), Chennai, Tamil Nadu, India, 2003, in Electrical Engineering
Received the M.S. degree from Oklahoma State University, Stillwater, Oklahoma, USA, 2005, in Electrical Engineering
Completed the requirements for the degree of Doctor of Philosophy with a major in Electrical Engineering, Oklahoma State University in December, 2010.

Research Experience:

- Visual Computation and Image Processing Lab (VCIPL), OSU Stillwater, OK
  Research Assistant, Sep 04 - Present
- Mitsubishi Electric Research Labs (MERL), Cambridge, MA
  Research Intern, Sep 09 - Dec 09

Patents and Awards:

- Patent pending: Object detection using combinations of relational features in images.
- Best Paper Award: IEEE OTCBVS Workshop, held in conjunction with CVPR 2010.
- Academic Distinction Award: Dr.Ing Dieter Kind prize for the best project in Electrical Engineering Department at the undergraduate degree level in IIT-M.

Name: Vijay Venkataraman Date of Degree: December, 2010

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: ADVANCED MACHINE LEARNING APPROACHES FOR TARGET DETECTION, TRACKING AND RECOGNITION

Pages in Study: 168 Candidate for the Degree of Doctor of Philosophy

Major Field: Electrical Engineering

This dissertation addresses the key technical components of an Automatic Target Recognition (ATR) system namely: target detection, tracking, learning and recognition. Novel solutions are proposed for each component of the ATR system based on several new advances in the field of computer vision and machine learning. Firstly, we introduce a simple and elegant feature, RelCom, and a boosted feature selection method to achieve a very low computational complexity target detector. Secondly, we present a particle filter based target tracking algorithm that uses a quad histogram based appearance model along with online feature selection. Further, we improve the tracking performance by means of online appearance learning where appearance learning is cast as an Adaptive Kalman filtering (AKF) problem which we formulate using both covariance matching and, for the first time in a visual tracking application, the recent autocovariance least-squares (ALS) method. Then, we introduce an integrated tracking and recognition system that uses two generative models to accommodate the pose variations and maneuverability of different ground targets. Specifically, a tensor-based generative model is used for multi-view target representation that can synthesize unseen poses, and can be trained from a small set of signatures. In addition, a target-dependent kinematic model is invoked to characterize the target dynamics. Both generative models are integrated in a graphical framework for joint estimation of the target's kinematics, pose, and discrete valued identity. Finally, for target recognition we advocate the concept of a continuous identity manifold that captures both inter-class and intra-class shape variability among training targets. A hemispherical view manifold is used for modeling the view-dependent appearance. In addition to being able to deal with arbitrary view variations, this model can determine the target identity at both class and sub-class levels, for targets not present in the training data. The proposed components of the ATR system enables us to perform low computational complexity target detection with low false alarm rates, robust tracking of targets under challenging circumstances and recognition of target identities at both class and sub-class levels. Experiments on real and simulated data confirm the performance of the proposed components with promising results.

ADVISOR'S APPROVAL: Dr. Guoliang Fan