ABSTRACT

Title of Thesis: Integration and Evaluation of a Video Surveillance System

Mohamed F. Abdelkader, Master of Science, 2005

Thesis directed by: Professor Rama Chellappa
Electrical and Computer Engineering Department

Visual surveillance systems are getting a lot of attention over the last few years, due to a growing need for surveillance applications. In this thesis, we present a visual surveillance system that integrates modules for motion detection, tracking, and trajectory characterization to achieve robust monitoring of moving objects in scenes under surveillance. The system operates on video sequences acquired by stationary color and infra-red surveillance cameras.

Motion detection is implemented using an algorithm that combines thresholding of temporal variance and background modeling. The tracking algorithm combines motion and appearance information into an appearance model and uses a particle filter framework for object tracking. The trajectory analysis module builds a model for a given normal activity using a factorization approach, and uses this model for the detection of any abnormal motion pattern.

The system was tested on a large ground-truthed data set containing hundreds of color and FLIR image sequences. Results of performance evaluation using these sequences are reported in this thesis.

INTEGRATION AND EVALUATION OF A VIDEO SURVEILLANCE SYSTEM

by

Mohamed F. Abdelkader

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2005

Advisory Committee:

Professor Rama Chellappa, Chair/Advisor
Professor Steve Marcus
Professor Min Wu

DEDICATION

This thesis is dedicated to my beloved parents, sisters, and brother.

ACKNOWLEDGMENTS

I would like to thank many people who helped me to bring this work together. First, to my advisor, Prof Rama Chellappa, whose guidance and support over the last two years have been of great help. To Dr Qinfen Zheng, for his valuable support and discussions during the course of this work. I would like to also thank the rest of my committee, Prof Steve Marcus and Prof Min Wu, for the valuable assistance they gave me either in classes or in my thesis work.

I would like to thank my group mates, Aswin Sankaranarayanan and Seong-wook Joo for helping me to start this work with there software and for their valuable discussions.

I would also like to express me deep gratitude to Ahmed Sadek and Wael Abd-Almageed for their valuable comments and help in the final stages of this work.

Finally, above all, I thank God for giving me the power to pass this stage of my life.

TABLE OF CONTENTS

List of Tables vi

List of Figures vii

1 Introduction 1

2 Moving-Object Detection 4
2.1 Overview . . . 4
2.2 Motion Detection: A Review . . . 4
2.3 Temporal Variance-Based Motion Detection . . . 5
2.4 Background Modeling . . . 7
2.5 Motion segmentation . . . 8

3 Object Tracking 11
3.1 Overview . . . 11
3.2 Object tracking: A review . . . 12
3.3 Particle Filter . . . 13
3.4 State transition model . . . 14
3.4.1 Adaptive velocity . . . 14
3.4.2 Adaptive Noise and Number of Particles . . . 15
3.5 Appearance-Motion Adaptive Observation Model . . . 16
3.5.1 Appearance Modeling . . . 16
3.5.2 Motion Modeling . . . 18
3.6 Tracking Results . . . 18

4 Abnormal Trajectory Detection 23
4.1 Overview . . . 23
4.2 Previous work . . . 24
4.3 Ground-Plane Calibration . . . 25
4.3.1 Perspective Imaging of Planes . . . 26
4.3.2 Projective Transformation and Hierarchy of transformations . 27
4.3.3 Ground Plane Recovery . . . 31
4.4 Learning the Shape of Trajectories . . . 34
4.5 Testing Trajectories . . . 37
4.6 Experiment and Results . . . 38
4.6.1 Ground Plane Calibration Results . . . 39
4.6.2 Trajectories Learning and Testing Results . . . 39

5 Performance evaluation 51
5.1 Overview . . . 51
5.2 Performance evaluation: A review . . . 51
5.3 Performance metrics . . . 52
5.4 Evaluation Results . . . 53
5.4.1 Comments on the Results . . . 58

6 Conclusions and Future Work 59

Bibliography 60

LIST OF TABLES

4.1 The different trajectory sequences generated from a three segments trajectory. 43

5.1 The Evaluation results on FPSS data set (1 of 4). 54

5.2 The Evaluation results on FPSS data set (2 of 4). 55

5.3 The Evaluation results on FPSS data set (3 of 4). 56

5.4 The Evaluation results on FPSS data set (4 of 4). 57

LIST OF FIGURES

1.1 The block diagram of our surveillance system showing the main modules. 2

2.1 The amplitude and variance for two ideal pulses of different duration. 6

2.2 The motion detection module .(a) The square root of the variance image showing the trails left by the moving objects, (b) The foreground confidence map with brighter area denoting a foreground object, and (c) The final motion detection map after multiplying (a) and (b) and applying the threshold. 9

2.3 The motion segmentation module.(a) The motion detection binary map with the white pixels indicating moving pixels,(b) The objects label image with each gray level indicating a differecnt object, and (c) The original frame with the moving objects bounded by boxes to be sent to the tracker. 10

3.1 A color tracking sequence. 19

3.2 A FLIR tracking sequence showing the difference between appearance-encoded tracking and motion-appearance-encoded tracking. 21

3.3 Contd. A FLIR tracking sequence showing the difference between appearance-encoded tracking and motion-appearance-encoded tracking. 22

4.1 Pinhole camera geometry model [30]. C is the camera center and p is the camera central point. The camera center is placed at the world coordinates origin. 26

4.2 Perspective images of points in a plane[30]. The world coordinate system is moved in order to be aligned with the plane π. 27

4.3 The different planar transformation under central projection. Images of a tiled floor. (From [30]) (a) **Similarity**: The circles are imaged as circles and all the angles and ratio of lengths are preserved. (b)**Affinity**: The circle is imaged as an ellipse and the right angles are not preserved any more while the parallel lines are still parallel. (c)**Projectivity**: Parallel lines intersect on the image and objects closer to the camera seem larger than far objects. 29

4.4 Vanishing line of the plane computed using the intersection of two sets of parallel lines. 32

4.5 The original frame used in the affine recovery process, showing the two pairs of parallel lines used to locate the vanishing line. 40

4.6 The Affine rectified image. 40

4.7 The original frame used in the affine to metric recovery process. . . . 41

4.8 The Metric recovered image. 42

4.9 The trajectories corresponding to normal activities used in the learning process, each red point representing the location of the object at a certain frame. 44

4.10 The learning trajectory set projected back to the ground plane. . . . 45

4.11 The NMSE values for the segments of the training sequence. 46

4.12 The First testing scenario. 47

4.13 The NMSE values for the segments of the normal trajectory sequences in blue, and for the second abnormal scenario in red. 48

4.14 The Second testing scenario. 49

4.15 Contd. The Second testing scenario. 50

Chapter 1

Introduction

In the last few years, visual surveillance has become one of the most active research areas in computer vision, especially due to the growing importance of visual surveillance for meeting security needs. A lot of promising applications are based on successful visual surveillance systems, such as access control in special areas, human identification at a distance, detection of abnormal activity, and threat evaluation.

Visual surveillance is a general framework that groups a number of different computer vision tasks such as detection, tracking and classification of objects of interest from image sequences, and understanding and describing the activities involving the objects. The ultimate goal in designing smart visual surveillance systems is to reduce the need for a human observer to monitor and analyze the visual data.

For these reasons, there have been a number of well-known visual surveillance systems. The real-time visual surveillance system W^4 [4] detects and tracks multiple people and monitors their activities in an outdoor scene. The system operates on single monocular gray-scale or IR camera. It uses a combination of shape analysis and tracking to locate people and their parts and to create models of their appearance so that they can be tracked through interactions, such as occlusions. The system also detects simple events, such as a person carrying an object. This is done by detecting the change in the symmetric shape of humans and their periodic motion.

The Pfinder system [3], developed by Wren *et al.*, recovers the 3D description of a person in a large room. The system operates on video sequences acquired by a single fixed camera, and tracks a single unoccluded person. The system builds a model of the scene using the color distribution and uses this model to detect any moving object. It uses a multi-class statistical model of color and shape to obtain a 2D representation of head and hands in a wide range of viewing conditions. The Pfinder system has been used in many applications including wireless interfaces, video databases, and low-bandwidth coding.

Another surveillance system was developed by CMU under the VSAM project [2]. This system allows a single human operator to monitor activities over a complex scene using a distributed network of active video sensors. The system detects objects and tracks them using a combination of temporal differencing and template tracking. The moving objects are classified and some of their geolocation and activity information are determined for subsequent processing. Results are displayed to the user in real-time on a graphical user interface (GUI).

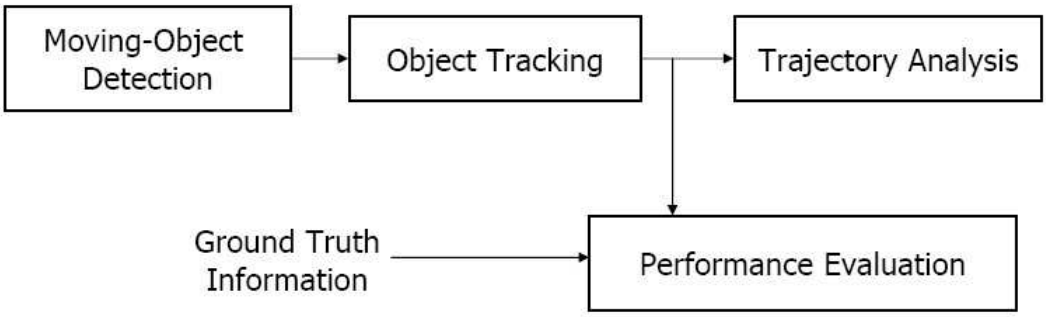


Figure 1.1: The block diagram of our surveillance system showing the main modules.

The MIT system [6] and [7] also uses a distributed set of sensors to observe moving objects in a site. The observed motion information is used to learn pattern of activities in the site. The system classifies these activities by using their joint co-occurrences to build a hierarchical binary-tree.

In this work, we present an integrated visual surveillance system to monitor an outdoor scene using a stationary forward-looking infra-red (FLIR) or a color camera. Figure 1.1 depicts the system architecture of our surveillance system. Our system begins with a motion detection module, which is responsible for detection and segmentation of the moving objects from the stationary background. This module is also responsible for initializing the tracking process for the detected objects. The tracking module estimates the location of the object at each new frame and tags the object in order to be used for high level processing modules. The trajectory characterization module is used to test the moving object trajectory in order to detect any unusual motion pattern. The performance evaluation module is used to evaluate the performance of the system using the available ground truth data. The main features of our system are as follows:

- A motion detection algorithm [11] that integrates both temporal variance and background modeling to allow for robust detection of moving objects. The temporal variance increase is used as an indication of motion, and the background model is used to remove any trails effect left by the moving object.
- An adaptive visual tracking algorithm [13] that uses both visual appearance and motion information in a statistical framework using particle filters. The tracking algorithm adaptively updates motion velocity, appearance, noise, and number of particles in each new frame. This adaptive nature increases the robustness of the tracking process.
- Integration of motion detection and tracking stages of the system by using the motion detection module for initializing the appearance model for the visual tracker, and for obtaining the motion observation.

- A new approach for learning motion trajectories. The algorithm projects all the motion trajectories back to the ground plane, which is calibrated in advance. A factorization like algorithm is used to learn the basis shape of the normal trajectory. This method can be used to detect any abnormal motion patterns in the scene under surveillance.

- Performance evaluation module responsible for measuring and evaluating the performance of the system using a ground-truthed detection data set.

This thesis is organized as follows: In Chapter 2, we describe our motion detection algorithm. We give more detailed review of existing motion detection algorithms. We describe how to update and use the temporal variance of pixel intensities to detect moving objects in the scene. A background model is combined with temporal variance detection to localize the moving object. The motion segmentation is performed using a connected component labeling technique.

In Chapter 3, we describe the visual tracker, which is based on combining both appearance and motion information into a single observation model. A statistical framework using particle filter is employed to estimate the motion state. Initialization techniques and adaptive update of the model are also addressed.

In Chapter 4, we propose an algorithm for learning the motion trajectories of normal activities in a scene. Our algorithm is based on projecting the motion trajectories into the scene ground plane. Then, a factorization like approach is used to estimate the 2D shape that corresponds to all the realizations of a single activity.

Chapter 5 addresses the issue of performance evaluation for surveillance system. We use an available ground-truthed data set to test the performance of our system, and we report these results.

Finally, conclusions and future work are provided in Chapter 6.

Chapter 2

Moving-Object Detection

2.1 Overview

The Detection of moving objects forms the first stage of a typical surveillance system. Motion detection aims at segmenting regions corresponding to moving objects from the rest of the image. This problem can also be described as differentiating between pixels that correspond to foreground objects and those corresponding to background based on the motion information. Subsequent processes such as tracking and behavior analysis are greatly dependent on it.

Our system adopts a method based on combining both the temporal variance of the pixels' intensities with background modeling to achieve robust and accurate motion detection and to reduce false alarms. This approach is suitable for both color and infra-red sequences and based on the assumption of a stationary surveillance camera, which is the case in many surveillance applications.

2.2 Motion Detection: A Review

We focus our attention on the stationary camera case. In this situation, the relative background area is nearly fixed with respect to the moving object so the changes between frames are mainly due to object motion.

A lot of algorithms have been introduced to solve the problem of motion detection but most of these algorithms can be categorized into three main approaches: temporal differencing, background subtraction, and optical flow. A brief review of the relevant work in each approach is presented here.

A. Temporal Differencing

Temporal differencing approaches use the fact that, for stationary camera case, the only changing areas between frames correspond to moving objects. Thus, these approaches are based on taking the pixel-wise difference between the frames to extract the moving regions. This method is very adaptive to dynamic environments, but does a poor job of extracting all the relevant pixels. Also using the pixel intensity difference can be sensitive to noise from texture motion (e.g. moving branches). Lipton *et al.* [5] detect moving targets in real video streams using temporal differ-

encing, where a threshold function is applied to the difference between the current and previous frames to determine changes. A connected component analysis is subsequently used for clustering the moving regions. An improved version by using three-frame instead of two-frame differencing was used in [2] along with background subtraction to reduce false alarms.

B. Background subtraction

Background subtraction is the most popular method for motion segmentation, especially in relatively static background. It detects moving regions by comparing the current image with a reference background image either by taking the difference between them or in a probabilistic framework . The effectiveness of this technique depends on the algorithm used to construct and update the background model. In [3] the pixel intensity was modeled using a single Gaussian distribution that is recursively updated to adapt to slow changes. This method could not handle the small motion of background objects such as vegetation. In this case, more than one process may be observed at each pixel. In [6] and [7] a mixture of Gaussians is used to model the intensity value at each pixel and online estimation is used to update the parameters of the model to adapt for illumination variations. Several other techniques were used like the Wallflower algorithm [8], in which background maintenance and subtraction are carried out at pixel, region, and frame levels. In W^4 system [4], the scene was modeled statistically by representing each pixel with three values: its minimum, maximum intensities, and the maximum intensity difference between consecutive frames. These values were learnt during the training period and updated periodically. The background subtraction method is simple but very sensitive to changes in dynamic scenes.

C. Optical Flow

This class of approaches uses characteristics of flow vectors of moving objects over time to detect moving regions in image sequences; a good discussion of these techniques can be found in [9]. Flow-based-methods can detect independently moving objects even in the presence of camera motion. However, these methods are usually computationally expensive and very hard to apply for real time applications.

2.3 Temporal Variance-Based Motion Detection

In our system we use the temporal variance as a parameter to detect moving areas in stationary scenes [10] and [11]. The idea is to calculate the mean and variance of the intensity value at each pixel over a window of few past frames and recursively update these values for each new frame. This value of the variance is used directly afterward for the detection of moving area. The use of temporal variance as a measure for motion has the following nice properties:

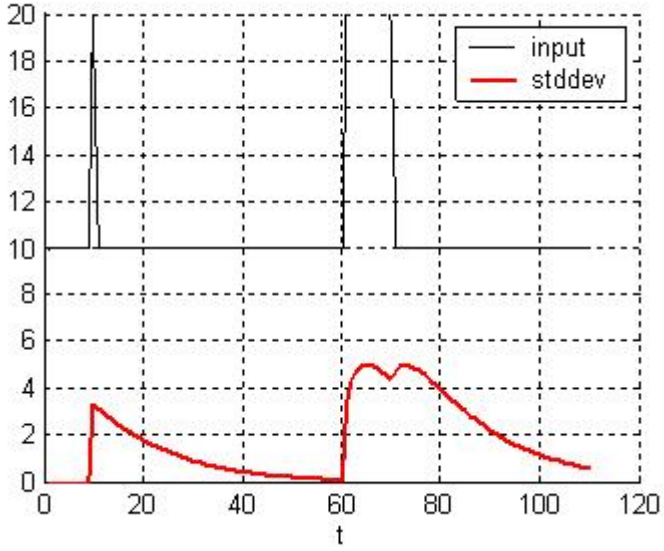


Figure 2.1: The amplitude and variance for two ideal pulses of different duration.

1. The variance of intensity at a certain pixel depends on both the amplitude of changes and the duration of this change as shown in figure 2.1, which makes it more robust to noise coming from moving texture that usually lasts only for a short duration.

2. There is no need for background training period as this method can build the model even when moving objects are present in the scene.

The mean and variance for the intensity at each pixel (i, j) are recursively computed using a simple exponentially decaying adaptive filter as follows

$$\begin{aligned} m(i,j,t) &= \alpha m(i,j,t-1) + (1-\alpha)x(i,j,t) \\ m_2(i,j,t) &= \alpha m_2(i,j,t-1) + (1-\alpha)x^2(i,j,t) \\ \sigma^2(i,j,t) &= m_2(i,j,t) - m^2(i,j,t) \end{aligned} \tag{2.1}$$

where

$x(i,j,t)$	is the intensity at pixel (i,j) at time t,
$m(i,j,t)$	is the first moment (mean) at pixel (i,j) at time t,
$m_2(i,j,t)$	is the second moment at pixel (i,j) at time t,
$\sigma^2(i,j,t)$	is the variance at pixel (i,j) at time t, and
α	is the decay rate.

The decay rate can be rewritten with respect to the filter window size N as:

$$\alpha = \frac{N-1}{N}, N = \frac{1}{1-\alpha} \tag{2.2}$$

The main problem with using the variance is that it takes a while for the variance to decay back to its original value after the change has ended, as shown in figure 2.1. This causes the moving object to leave a trail behind it, consisting of pixels that were in motion in the proceeding frames. The variance decay rate can be controlled by changing the window size N; however, reducing this size will make the model too adaptive to any changes in the scene. To overcome this problem, we propose using a simple background model, which is adaptively updated, to remove this *trail effect*.

2.4 Background Modeling

In order to remove the variance *trail effect*, we use the fact that although the variance remains at a higher value after an object passes the pixel, the pixel intensity quickly returns to its original background value. So, by building a background model with a large window size and comparing the pixels to this model, we can easily remove the trail effect. The background model used is simply a single Gaussian with the mean and the variance are computed as follow:

$$\begin{aligned} m_{bg}(i,j,t) &= \alpha_{bg} m_{bg}(i,j,t-1) + (1-\alpha_{bg}) x(i,j,t) \\ m_{2bg}(i,j,t) &= \alpha_{bg} m_{2bg}(i,j,t-1) + (1-\alpha_{bg}) x^2(i,j,t) \\ \sigma^2_{bg}(i,j,t) &= m_{2bg}(i,j,t) - m^2_{bg}(i,j,t) \end{aligned} \tag{2.3}$$

where:

$m_{bg}(i,j,t)$	is the first moment (mean) for the background at pixel (i,j) at time t,
$m_2(i,j,t)$	is the second moment for the background at pixel (i,j) at time t,
$\sigma^2_{bg}(i,j,t)$	is the variance for the background at pixel (i,j) at time t,
α_{bg}	is the decay rate for the background.

The decay rate can also be written with respect to the background filter window size used N_{bg} as:

$$\alpha_{bg} = \frac{N_{bg}-1}{N_{bg}}, N_{bg} = \frac{1}{1-\alpha_{bg}} \tag{2.4}$$

The difference between the two models is:

- The window size used for the background model N_{bg} is much larger than that used for the variance update, so that the background model is slowly varying and covers a larger history of the frames.
- More importantly, the update process for the background model is selective in the sense that only pixels that have not been identified as possible foreground pixels, using the variance, are updated at each new frame so that we assure that this background will not include any foreground object.

The background model is used to obtain a confidence weight representing the certainty of this pixel being a part of the foreground. This confidence weight is obtained as a function of the distance between the pixel intensity and the background model

$$C(i,j,t) = f\left(\frac{|x(i,j,t) - m_{bg}(i,j,t)|}{\sqrt{\sigma_{bg}(i,j,t)}}\right) \tag{2.5}$$

where $C(i,j,t)$ is the confidence weight that the pixel (i,j) is part of the foreground, f is a nonlinear-mapping function to map the distance to the range of [0,1] and also to emphasize the large distance points. The final binary detection map $L(i,j,t)$ is obtained as follows:

$$L(i,j,t) = \begin{cases} 1 & if \quad C(i,j,t)\sigma(i,j,t) \geq threshold \\ 0 & if \quad C(i,j,t)\sigma(i,j,t) < threshold \end{cases} \tag{2.6}$$

where the value of threshold can be obtained either empirically or by multiplying the average background variance by a factor.

In order to obtain the final detection result, we multiply the variance image, shown in figure 2.2(a), by the confidence weights, shown in figure 2.2(b), to remove the trail effect discussed in the last section. A thresholding step is then applied to the resulting image to produce the final detection map, shown in figure 2.2(c).

2.5 Motion segmentation

The motion segmentation step could be considered as the interface between motion detection and tracking stages of the surveillance system. It includes the segmentation of the moving areas from the binary detection map, figure 2.3(a), into disjoint objects, the removal of any small or isolated noise, and the initialization of the bounding boxes that are passed to the subsequent tracker.
In order to perform these tasks, we use the connected-component labeling algorithm presented in [12]. This algorithm performs several raster passes on the binary image $L(t)$ and uses sequential local operations with a one-dimensional table, which produces a fast connected-component result, that is shown in figure 2.3(b). After the labeling, a bounding box containing all the pixels with the same label is drawn as shown in figure 2.3(c). These bounding boxes are sent to the subsequent tracker in order to initiate the tracking process.

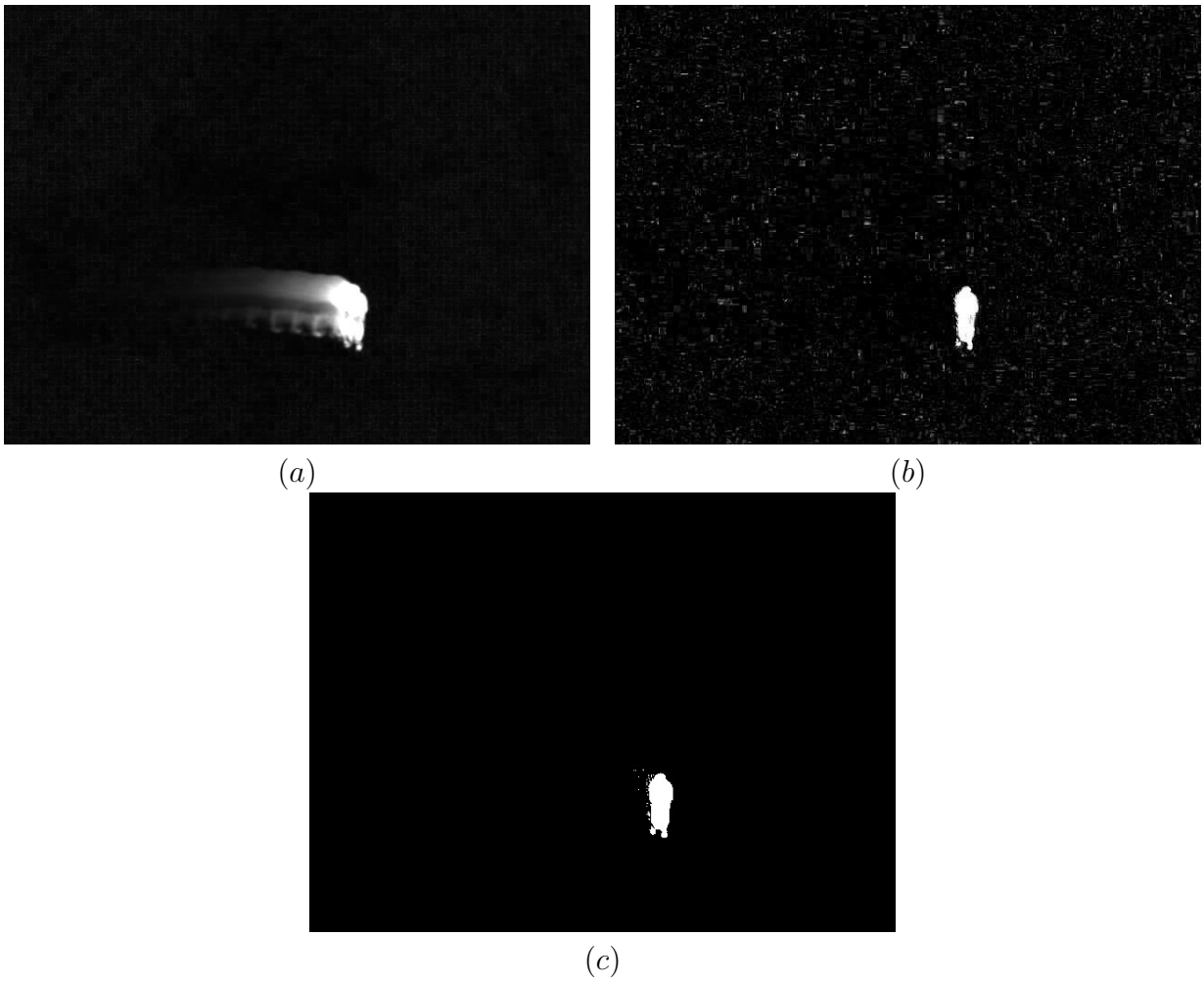


Figure 2.2: The motion detection module .(a) The square root of the variance image showing the trails left by the moving objects, (b) The foreground confidence map with brighter area denoting a foreground object, and (c) The final motion detection map after multiplying (a) and (b) and applying the threshold.

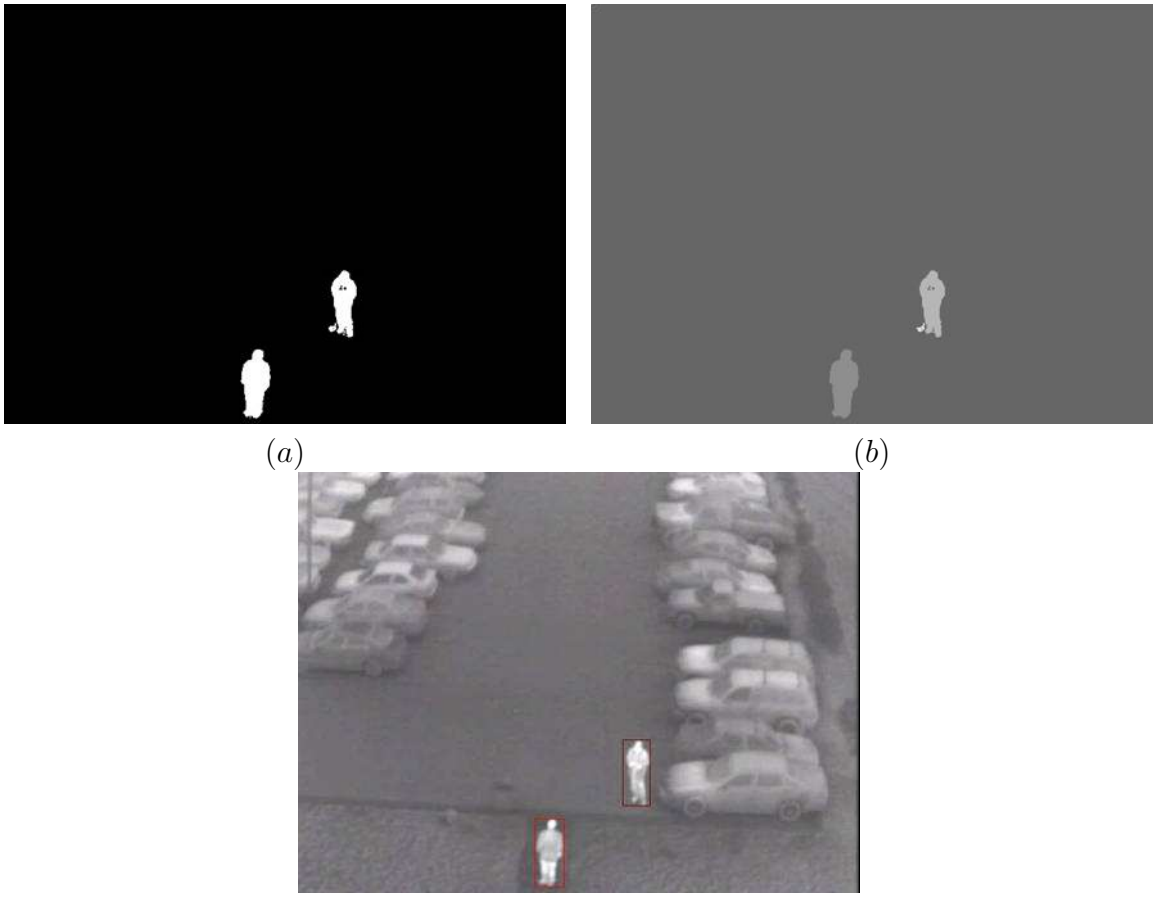


Figure 2.3: The motion segmentation module.(a) The motion detection binary map with the white pixels indicating moving pixels,(b) The objects label image with each gray level indicating a differecnt object, and (c) The original frame with the moving objects bounded by boxes to be sent to the tracker.

Chapter 3

Object Tracking

3.1 Overview

In order to build a temporal model of activity, individual objects generated by the motion detection module are tracked over time. The tracking process also allows us to tag each object for high level processing modules like trajectory analysis and activity classification. In our surveillance system, we use the tracking algorithm presented in [13]. We design a new observation model that incorporates both appearance and motion information of the object. The new location of the objects is estimated in a statistical framework using Particle Filters (PF). This algorithm appears to be very effective and robust even in challenging tracking conditions like static occlusion and cluttered background.

The tracking problem can be formulated as an estimation process, where the goal is to estimate the unknown motion state θ_t from a noisy collection of observations, $Y_{1:t} = \{Y_1, \ldots Y_t\}$ arriving in a sequential fashion. For each observed frame Y_t, different image patches Z_t correspond to different motion states θ_t by the relation $Z_t = T\{Y_t; \theta_t\}$, where T is the motion transformation used (Affine in our case). The system transition is usually modeled using a state space model with two important components, the state transition and observation models, which are generally expressed as:

$$\textit{State transition model}: \quad \theta_t = F_t(\theta_{t-1}, U_t), \tag{3.1}$$

$$\textit{Observation model}: \quad Y_t = G_t(\theta_t, V_t), \tag{3.2}$$

where U_t is the system noise, $F_t(\ldots)$ characterizes the kinematics of the object motion, V_t the observation noise, and $G_t(\ldots)$ models the observer. Due to the nonlinear and non-Gaussian nature of our system model, the particle filter [22] is used as a powerful technique to approximate the posterior distribution $p(\theta_t|Y_{1:t})$ using a set of weighted particles $\{\theta_t^{(j)}, w_t^{(j)}\}_{j=1}^{J}$. Then the state estimate $\hat{\theta}_t$can either be the minimum mean square error (MMSE) estimate.

$$\hat{\theta}_t = \theta_t^{mmse} = E[\theta_t|Y_{1:t}] \approx J^{-1} \sum_{j=1}^{J} w_t^{(j)} \theta_t^{(j)} \tag{3.3}$$

or the maximum a posteriori (MAP) estimate.

$$\hat{\theta}_t = \theta_t^{map} = arg\max_{\theta_t} p(\theta_t|Y_{1:t}) \approx arg\max_{\theta_t} w_t^{(j)} \tag{3.4}$$

or any other estimators based on $p(\theta_t|Y_{1:t})$ as shown in [22].

This chapter is organized as follow: In Section 3.2 we give a brief review of existing tracking algorithms, Section 3.3 presents an overview of particle filters, Section 3.4 describes the state transition model, the combined appearance and motion observation model is described in Section 3.5, and the last section presents results of experiments.

3.2 Object tracking: A review

The problem of visual tracking has applications in many fields, and many algorithms have been introduced to approach this problem.

The visual tracking research can be categorized into four main groups [1]:

- Region-based tracking

  In these algorithms, moving objects are tracked according to changes in the image regions corresponding to these objects. This can be performed by maintaining a background model and detecting motion regions by subtracting the background from the current image. An example of these methods is the Pfinder system [3], where the tracking of a single human object is achieved by dividing the body into small blobs. These blobs correspond to different body parts such as head, torso and the four limbs. Each moving pixel is assigned to one of the blobs using a log-likelihood measure.

  The region-based tracking algorithms perform well in scenes containing only a few objects, but they cannot reliably handle occlusion between objects. The other problem with these algorithms is their large dependence on the motion detection results. Accordingly, these algorithms cannot satisfy the requirements for surveillance in cluttered background or when multiple moving objects are present.

- Active-contour-based tracking

  These algorithms track objects by representing their outlines as bounding contours and updating these contours [14]. These algorithms provide more effective description of objects than region-based algorithms, and can handle some disturbance and partial occlusion. However, these algorithms are very sensitive to the initialization of the object which may make them ineffective in operational situations.

- Feature-based tracking

  In this class of algorithms, such as [15], some features of the object are extracted, clustered into higher level features and then matched between frames.

Some of the features used in these algorithms are centroid, areas, and corners. These methods can handle partial occlusion by using information on object motion. They also can operate easily in real time because of low complexity. However, their object recognition rate is low due to the effects of projection and view point variation on the features; also these algorithms are unable to recover the 3D pose of objects.

- Model-based tracking

  In these algorithms, we build a model for the tracked object and search for the best match for this model in subsequent frames. Different models have been used, one based on the use of appearance models was introduced in [20], where the appearance of the tracked region is modeled using a mixture of Gaussians and the parameters of these Gaussians are updated at each new frame.

  Different search strategies have been employed to search for the best match. Deterministic approaches that minimize a cost function as in [16], or probabilistic methods that estimate the motion state for a state space model using an estimation tool like Kalman Filter [17] or Particle Filter [18].

3.3 Particle Filter

General algorithm: Given the transition model in equation (3.1) characterized by the state transition probability $p(\theta_t|\theta_{t-1})$ and the observation model in equation (3.2) characterized by the likelihood function $p(Y_t|\theta_t)$, the problem is reduced to computing the posterior probability $p(\theta_t|Y_{1:t})$. Due to nonlinear and non-Gaussian models in Equations (3.1) and (3.2), the particle filter is used for approximating the posterior distribution $p(\theta_t|Y_{1:t})$ using a set of weighted particles $S_t = \{\theta_t^{(j)}, w_t^{(j)}\}_{j=1}^J$ with $\sum_{j=1}^J w_t^{(j)} = 1$. It was shown in [22] that S_t is *properly weighted* with respect to $p(\theta_t|Y_{1:t})$ in the sense that, for any bounded function $h(.)$,

$$\lim_{J\to\infty} \sum_{j=1}^{J} w_t^{(j)} h(\theta_t^{(j)}) = E_p[h(\theta_t)] \tag{3.5}$$

As shown in [13], given $S_{t-1} = \{\theta_{t-1}^{(j)}, w_{t-1}^{(j)}\}_{j=1}^J$ which is properly weighted with respect to $p(\theta_{t-1}|Y_{1:t-1})$, we first resample S_{t-1} to reach a new set of samples with equal weights $\{\acute{\theta}_{t-1}^{(j)}, 1\}_{j=1}^J$. We then draw samples $\{U_t^{(j)}\}_{j=1}^J$ for the system noise U_t and propagate $\acute{\theta}_{t-1}^{(j)}$ to $\theta_t^{(j)}$ using equation (3.1). The new weights are updated by

$$w_t \propto p(Y_t|\theta_t) \tag{3.6}$$

3.4 State transition model

The state in our model θ_t represents the motion parameters used in the system. We use an affine motion model, which consists of six variables representing the affine motion parameter. Other motion models, such as projective model, could be applied for representing more complex tracking conditions. The state transition model used is a simple adaptive Markov model of the form:

$$\theta_t = \hat{\theta}_{t-1} + \nu_t + U_t \tag{3.7}$$

where U_t is the system noise term and ν_t is the adaptive velocity term. The velocity term captures most of the shift in the state and system noise accounts for the remaining change.

3.4.1 Adaptive velocity

With the availability of the sample set $\Theta_{t-1} = \{\theta_{t-1}^{(j)}\}_{j=1}^{J}$ and the image patches of interest $Z_{t-1} = \{Z_{t-1}^{(j)}\}_{j=1}^{J}$ corresponding to different states and the previous observation Y_{t-1}, for a new observation Y_t, we can predict the shift in the motion vector (adaptive velocity) $\nu_t = \theta_t - \hat{\theta_{t-1}}$ by applying the constant brightness constraint, which states that there exist a θ_t such that

$$T\{Y_t; \theta_t\} \simeq \hat{Z}_{t-1} \tag{3.8}$$

where $T\{Y_t; \theta_t\}$ represents the transformation of the observation frame Y_t with the given state θ_t. This transformation can be approximated using a first order Taylor series expansion around $\tilde{\theta}_t$ (we set $\tilde{\theta}_t = \hat{\theta}_{t-1}$)

$$T\{Y_t; \theta_t\} \simeq T\{Y_t; \tilde{\theta}_t\} + C_t(\theta_t - \tilde{\theta}_t) = T\{Y_t; \tilde{\theta}_t\} + C_t\nu_t \tag{3.9}$$

where C_t is the Jacobian matrix.
Combining Equations (3.8) and (3.9) gives

$$\hat{Z}_{t-1} \simeq T\{Y_t; \tilde{\theta}_t\} + C_t\nu_t, \tag{3.10}$$

i.e.

$$\nu_t = \theta_t - \tilde{\theta}_t \simeq -B_t(T\{Y_t; \tilde{\theta}_t\} - \hat{Z}_{t-1}) \tag{3.11}$$

where B_t is the pseudo-inverse of the C_t matrix and can be efficiently estimated from the available data Θ_{t-1} and Z_{t-1}.

In order to estimate B_t, we stack into matrices the differences in motion vectors and image patches, using $\hat{\theta}_{t-1}$ and $\hat{Z}_{t-1}$ as pivotal points

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

$$Z_{t-1}^{\delta} = [Z_{t-1}^{(1)} - \hat{Z}_{t-1}, \ldots\ Z_{t-1}^{(J)} - \hat{Z}_{t-1}] \tag{3.13}$$

then B_t is estimated using a least squares (LS) solution of the form

$$B_t = (\Theta^{\delta}_{t-1} {Z^{\delta}_{t-1}}^T)(Z^{\delta}_{t-1} {Z^{\delta}_{t-1}}^T)^{-1} \tag{3.14}$$

where $(.)^T$ denotes matrix transposition. However, it turns out that the matrix $Z^{\delta}_{t-1} {Z^{\delta}_{t-1}}^T$ is very often rank-deficient due to the high dimensionality of the data (number of pixels in the template), unless the number of particles exceeds that. To overcome this, we use the singular value decomposition (SVD).

$$Z^{\delta}_{t-1} = USV^T \tag{3.15}$$

It can be easily shown that

$$B_t = \Theta^{\delta}_{t-1} V S^{-1} U^T \tag{3.16}$$

which can be further approximated, to gain some computational efficiency, by retaining only the top q components.

$$B_t = \Theta^{\delta}_{t-1} V_q S_q^{-1} U_q^T \tag{3.17}$$

In practice, several iterations may be needed till $\tilde{Z}_t = T\{Y_t; \tilde{\theta}_t + \nu_t\}$ stabilizes, i.e., the error ϵ_t defined to measure the distance between $\tilde{Z}_t$ and the updated appearance model A_t, described in the next section, is small enough.

3.4.2 Adaptive Noise and Number of Particles

After calculating the adaptive velocity ν_t, the value of the error ϵ_t determines the quality of prediction. Therefore, if ϵ_t is small, implying a good prediction, we only need noise with a small variance to absorb the residual motion. On the other hand, if ϵ_t is large, implying a poor prediction, we then need noise with large variance to cover potentially large jumps in the motion state. We use noise U_t of the form $U_t = R_t * U_0$, where R_t is a function of ϵ_t on the form

$$R_t = \max(\min(R_0\sqrt{\epsilon_t}, R_{max}), R_{min}) \tag{3.18}$$

where R_{min} is a lower bound to maintain a reasonable sample coverage and R_{max} is an upper bound to constrain the computational load. These bounds are are empirically determined depending on the tracking sequence.

As a large noise variance needs more particles while fewer particles are needed for small variance, the number of particles in our system is chosen to be proportional to the noise variance R_t as

$$J_t = J_0 R_t / R_0 \tag{3.19}$$

3.5 Appearance-Motion Adaptive Observation Model

We use an observation model that combines the motion information (background modeling) and appearance information (foreground modeling) to achieve more robustness, especially in cases where one of the two cues fails. We use the model presented in [21] to combine both cues and assume that the observation Y_t is segmented by the motion state θ_t into two mutually exclusive and collectively exhaustive sets: $\beta(\theta_t)$ and $F(\theta_t)$, which are the regions identified by the state θ_t as background and foreground (target), respectively. With this assumption, the observation equation $p(Y_t|\theta_t)$ can be written as follows:

$$\begin{aligned} p(Y_t|\theta_t) &= p(Y_t(\beta(\theta_t)), Y_t(F(\theta_t))|\theta_t) \\ &= p(Y_t(\beta(\theta_t))|\theta_t)p(Y_t(F(\theta_t))|\theta_t) \\ &= p(Y_t(\beta(\theta_t))|\beta_t)p(Y_t(F(\theta_t))|A_t) \end{aligned} \tag{3.20}$$

Where A_t is the appearance model and β_t is the background model. Under the assumption that the background is composed of independent pixels.

$$p(Y_t(\beta(\theta_t))|\beta_t) = \frac{p(Y_t|\beta_t)}{p(Y_t(F(\theta_t))|\beta_t)} \tag{3.21}$$

So the likelihood function $p(Y_t|\theta_t)$ can be written in the following way

$$p(Y_t|\theta_t) = p(Y_t|\beta_t)\frac{p(Y_t(F(\theta_t))|A_t)}{p(Y_t(F(\theta_t))|\beta_t)} \tag{3.22}$$

As the first term in the LHS is independent of θ_t, the observation probability $p(Y_t|\theta_t)$ is directly proportional to the ratio $\frac{p(Y_t(F(\theta_t))|A_t)}{p(Y_t(F(\theta_t))|\beta_t)}$. This term is the likelihood ratio associated with a two class hypothesis testing problem on the set $Y_t(F(\theta_t)$, where the *alternate* hypothesis suggests that $Y_t(F(\theta_t)$ is from the foreground model and the *null* hypothesis suggests that $Y_t(F(\theta_t)$ is from the background model. This means that θ_t that maximizes this likelihood will also minimizes the Bayesian risk associated with this hypothesis test.

3.5.1 Appearance Modeling

The appearance-based tracking technique relies on building an adaptive appearance model for the object and estimating the new location of this object in subsequent frames. In conventional tracking algorithms, the appearance model is either fixed or rapidly changing which cause the visual tracker to be unstable. We use the online adaptive appearance model A_t presented in [13], which is a modified version from the online appearance model (OAM) developed in [20].

Mixture appearance model

Assuming that the observation appearance change between frames can be explained by different causes, a mixture density model was suggested in [20]. In the same context, we use a time varying OAM that consists of a mixture density of three components $A_t = \{W_t, S_t, F_t\}$. The W-component models a short time-course, such as in a two-frame tracker, so it is modeled as a Gaussian component that is conditioned on the estimated image patch from the last frame $\hat{Z}_{t-1}$. The fixed template component F is modeled as a Gaussian with fixed mean and variance. Finally, the S-component depicts a slowly varying Gaussian component whose parameters are updated with every new frame to account for the slow changes in the appearance over time. The observation likelihood is written as

$$p(Y_t(F(\theta_t))|A_t) = p(Z_t|\theta_t) = \prod_{j=1}^{d}\{\sum_{i=w,s,f} \alpha_{i,t}(j)N(Z_t(j); m_{i,t}(j), \sigma^2_{i,t}(j))\}, \quad (3.23)$$

where: d denotes the number of pixels in the bounding box, $\{m_{i,t}; i = w, s, f\}$ the mixture means of the three components, $\{\sigma^2_{i,t}; i = w, s, f\}$ are the variances, $\{\alpha_{i,t}; i = w, s, f\}$ are the densities mixing probabilities, and $N(x; m, \sigma^2)$ is a normal density of mean m and variance σ^2 of the form

$$N(x; m, \sigma^2) = (2\pi\sigma^2)^{\frac{-1}{2}} exp\left\{-\left(\frac{(x-m)^2}{2\sigma^2}\right)\right\} \quad (3.24)$$

Each of the model parameters $\{m_{i,t}, \sigma^2_{i,t}, \alpha_{i,t}; i = w, s, f\}$ is an image consisting of d independent pixels. These parameters are initialized at the beginning and updated at each frame to maintain an adaptive appearance.

Model Initialization and Update

The initialization of the appearance model is done by the tracking module. After the detection of moving object, the bounding box corresponding to this object is defined. The image patch corresponding to this object Z_0 is forwarded to the tracker. To initialize the appearance model A_1, we set $W_1 = S_1 = F_1 = Z_0$

In order to maintain the appearance model we will show how to update the current appearance model A_t to A_{t+1} after the best estimate $\acute{\theta}_t$ and the corresponding image patch $\acute{Z}_t$ become available, i.e., how to compute the new mixing probabilities, mixture means and variances for time $t+1$,$\{a_{i,t+1}, m_{i,t+1}, \sigma^2_{i,t+1}; i = w, s, f\}$.
It is assumed the past observations are exponentially 'forgotten' with respect to their contribution to the current appearance model. Denote the exponential envelope by $\varepsilon_t(k) = ae^{-(t-k)/\tau}$ for $k \leq t$, where $\tau = n_h/\log 2$, n_h is the half-life of the envelope in frames, and $a = 1 - e^{-1/\tau}$ to guarantee that the weights sum to 1. The Expectation Maximization (EM) algorithm [19] is used to update the model parameters for each

pixel, as we assume that the pixels are independent of each other. We refer to [13] and [20] for complete mathematical descriptions of the updating process.

3.5.2 Motion Modeling

The motion-based modeling in our observation model is formulated by defining a density function $p(Y_t(F(\theta_t))|\beta_t)$ and inserting it into the likelihood equation (equation (3.22)). For this purpose, we use the parameters of the background model introduced in the motion detection algorithm for removing the trail effect. Assuming that individual pixels are independent

$$p(Y_t(F(\theta_t))|\beta_t) = p(Z_t|\beta_t) = \prod_{j=1}^{d} p(Z_t^i|\beta_t) \tag{3.25}$$

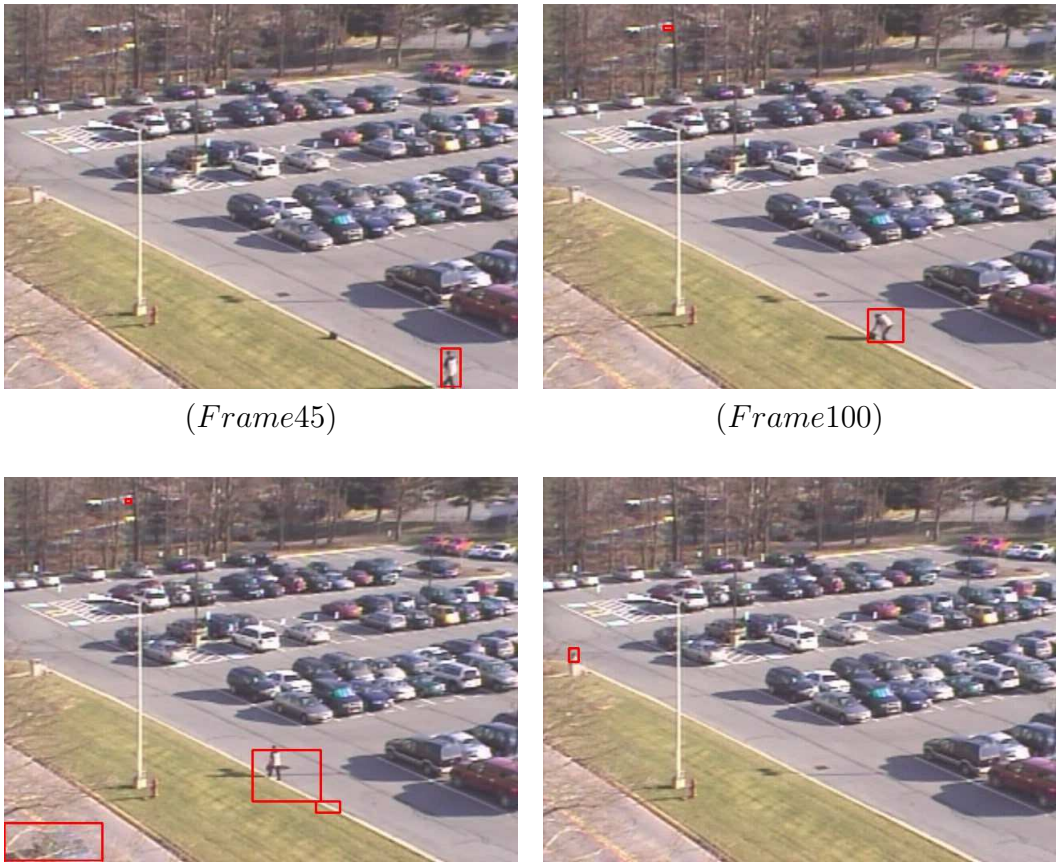
where $p(Z_t^i|\beta_t)$ is the probability distribution of the i^{th} pixel of the patch under the background model. This distribution is a Gaussian $N(Z_t^i; m_{bg}^i, \sigma_{bg})$ where σ_{bg}^2 is a fixed variance, the use of a fixed variance instead of the recursively updated variance of the model allows for more robustness. Meanwhile, we use this variance as an indication of the level of confidence in the motion information, which determines to what degree this information should contribute to the whole likelihood term.
As the same model is used in motion detection and for modeling the background in the tracker, efficient integration of the modules is feasible in our surveillance system which in turn increases the robustness of the system and enhances its performance.

3.6 Tracking Results

In this subsection, we show some of the significant tracking results. These results are not meant as an evaluation of the whole system performance, as this will be the subject of the last chapter of this thesis. Thus, these results only present a demonstration of the tracking performance in some situations.

All of the presented results have been obtained by testing our surveillance system on the force protection surveillance system (FPSS) data set provided by the U. S. Army Research Laboratory (ARL). The data set consists of a large collection of color and forward looking infra red (FLIR) sequences of a parking lot. These sequences represent a wide variety of views, complexity, times of the day, and weather conditions.

In the color tracking sequence shown in figure 3.1, we see a human object appears in the scene, picks something from the ground and then leaves the parking lot. The human object was tracked successfully over these frames, even when partial



($Frame45$) ($Frame100$)

($Frame160$) ($Frame390$)

Figure 3.1: A color tracking sequence.

occlusion is present, as in frame 390, where the motion observation plays a large role. We also notice that the motion detection algorithm was able to detect the absence of the object that was on ground. This is due to the deviation of this area from the background model. This result is extremely helpful in activity analysis, where detecting an object left behind is a very important activity to be monitored.

In figures 3.2 and 3.3, we show an example of the advantages of combining motion and appearance observations in the observation model. The sequence shows the results of tracking a video sequence using an appearance observation model as in [13] (in the left column), and a combined motion and appearance observation model (in the right column). The location of the object is shown by the bounding red box, and the object ID is shown above the box to track the reinitialization of the object.

We start with $Frame(30)$ where both the systems initialize the object in the same way, as they both use the same detection module. The human object of interest is given the $ID = 1$ in the appearance model, $ID = 3$ in the appearance-motion model. At $Frame(56)$ the object goes through a static occlusion where most of the

object is occluded by a parking car. We see that the appearance model was unable to continue tracking the object because of the large change in the appearance so this tracker got stuck on the parked car; meanwhile, the availability of the motion information helped the second tracker to continue the tracking of the object. By the end of this occlusion, $Frame(65)$, we see the appearance tracker already lost the track while the other one is still tracking the object. Using the motion information also helped the tracker to update the model to include the whole body instead of just the head.

In figure 3.3, The object goes through another occlusion, in $Frame(87)$ after it was reinitialized by the appearance tracker and given $ID = 7$. By the end of this occlusion $Frame(93)$, the appearance tracker got stuck at part of the background while the other tracker is still tracking the object. By the last tracked frame, $Frame170$, we see that the appearance tracked object has his third ID which is 9, while the motion-appearance tracker still has the same initialized ID of three.

($Frame30$)

($Frame56$)

($Frame65$)

Figure 3.2: A FLIR tracking sequence showing the difference between appearance-encoded tracking and motion-appearance-encoded tracking.

($Frame87$)

($Frame93$)

($Frame170$)

Figure 3.3: Contd. A FLIR tracking sequence showing the difference between appearance-encoded tracking and motion-appearance-encoded tracking.

Chapter 4

Abnormal Trajectory Detection

4.1 Overview

Subsequent to tracking the moving objects over time from one frame to another, it is of great interest to analyze the behavior of these objects. The analysis of object behavior can be achieved in a variety of ways, depending on the scene under surveillance and the type of object's behavior.

Understanding object behavior is important in many applications, such as in video indexing systems. However, in visual surveillance, it is of more interest to monitor any abnormal behavior that differs from normal actions. For example, in a bank surveillance system, the ability to detect robbery or abnormal transaction could be valuable. The same holds in our video surveillance system as we focus on monitoring the trajectories of moving objects for any abnormality, rather than understanding the objects' behavior.

In our technique, the typical outdoor surveillance scene conditions are fully utilized. In this context, we use the fact that our system operates on a stationary stand-off mode camera in two ways. The first is that we assume that each object can be modeled as a single point in each frame. This point can be chosen as the centroid of the object or the center of the bounding box. The second is that we assume that the objects are moving on a common ground plane, represented by the ground of the parking lot in our experiment.

Using these two assumptions, we build a model for learning the shape of the motion trajectory for the normal activities in the scene. An observed trajectory is compared to each of these shapes. If a match is found then the trajectory is classified as part of this activity, otherwise it is declared as an abnormal motion trajectory.

In order to explain our approach, we return to Kendall's definition of the shape of a group of discrete points. Kendall defines this shape [28] as all the geometric information that remains when location, scale and rotational effects are filtered out. In other words, the shape of a group of discrete points is their relative geometry that remains after removing all the geometric transformation effects.

In our system, there are two different sources of geometric effects that could affect the shape of the trajectory. First, the projection effect resulting from the perspective imaging device. Second, the geometric effect resulting from the way the

activity is being performed.

In order to remove these effects, we propose a two step approach. The first step is to remove the imaging projection effect, which is done using the assumption that all of the trajectories are taking place on the ground plane of the scene. A calibration method, based on some features from the scene, is used to recover the ground plane up to a similarity transformation (translation, rotation and scaling). The second step is to remove the geometric changes between the trajectories, and we assume that these changes correspond to a similarity transformation from the basis shape. A factorization like approach, based on the rank of the measurement matrix, is used to recover the basis shape and find these transformations.

This Chapter is organized as follows: we first present a literature review of previous work in activity modeling, focusing on trajectory modeling algorithms. Section 4.2 starts by describing the perspective projection geometric effects and the various kinds of 2D transformations, then the method used for recovering the ground plane is introduced along with some calibration results from the literature. Section 4.4 describes the trajectory learning problem, the proposed algorithm for learning the basis shape, and discusses the testing procedure. The last section presents experimental results.

4.2 Previous work

A lot of work has been done in the last few years to deal with the problem of activity modeling. In [7], the authors use their surveillance system to monitor a site for a long period of time. The motion trajectories over this period are stored and clustered using the joint co-occurrence statistics of each representation. Classification is then performed using a hierarchical binary-tree classification map.

Johnson *et al.* [24] describe the motion trajectory of an object in terms of a sequence of flow vectors, which represent the position and velocity of the object in the image plane. The probability density function of these sequences is calculated using a neural network implementing vector quantization. Owens *et al.* [25] applied the same method, but performed smoothing by a moving average window. They used a self-organising feature map to learn the distributions.

Dimitrios Makris and Tim Ellis [26] develop a spatial model to represent the routes in an image. Routes are learned and a trajectory is matched with these routes using a simple distance measure. They suggested using the ground plane coordinates to remove perspective effect and merge more than one camera. Junejo *et al.* use a similar approach but take into consideration the velocity and curvature information.

In many applications, the main goal is to detect abnormal shape trajecto-

ries without taking into effect the spatial location of these trajectories. For these application the idea of using shape based analysis for the trajectories is appropriate.

In [29], the authors used wavelet transform to decompose the raw trajectories into components of different scales. The coarsest scale components is used to model the global motion information, and the finer scale ones to partition the trajectory into subtrajectories. These subtrajectories are matched against the data base to recognize the trajectory.

Vaswani *et al.* [27] model the shape of the configuration of interacting objects using Kendall's statistical shape theory [28]. They model the deformation of this shape in the tangent space using a hidden Markov model (HMM), and track the state of the model using particle filters. Any abnormal activity is detected when the model fail to follow the deformations of the shape.

The most relevant work to our approach is [23], where the factorization approach is used to model the activity performed by a group of objects. The deformable polygonal shape formed by joining the locations of moving entities is modeled as a non-rigid object with the assumption that the basis shapes corresponding to this object refer to the different activities.

The main differences between our approach and [23] is that we use the shape of each object trajectory to infer about this object behavior instead of using the polygon connecting many moving objects. Also, we use the ground plane coordinates to remove the effects of the perspective projection. Another difference is that instead of using the learnt rotation matrices to test the test sequence, we try to find the best LLSE rotation matrix that would match the test sequence to the learnt basis shape.

4.3 Ground-Plane Calibration

Most of the outdoor surveillance systems monitor a ground plane of an area of interest. This area could be the floor of a parking lot, the ground plane of an airport, or any other monitored area. Most of the objects being tracked and monitored are moving on this dominant plane. We use this fact to remove the camera projection effect by recovering the ground plane and projecting all the motion trajectories back into this ground plane. In other words, we try to map the motion trajectories measured at the image plane into the real plane coordinates to remove the projection effects.

The recovery of the ground plane is achieved using some of the common features that show up in a man-made environment, such as parallel lines, right angles and objects with equal length. Most of the notations used here are from Hartley and Zisserman book [30].

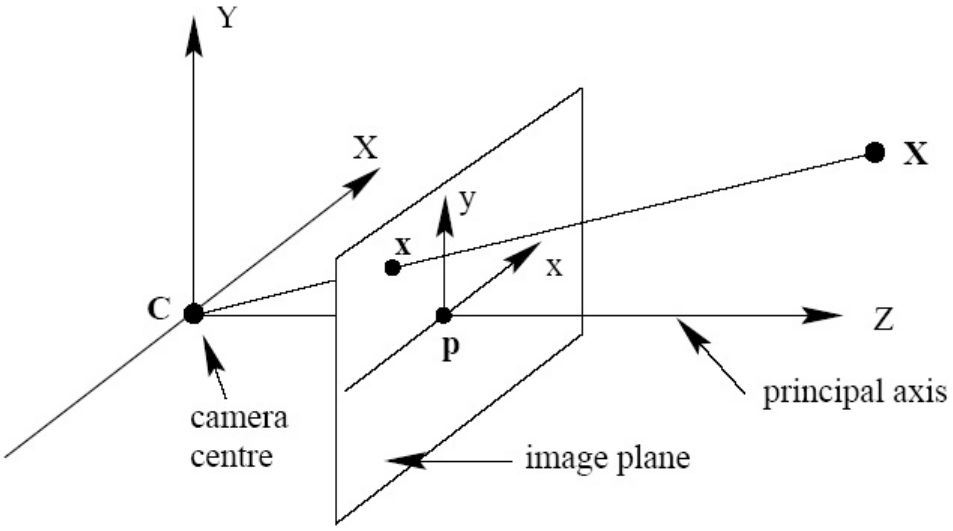


Figure 4.1: Pinhole camera geometry model [30]. C is the camera center and p is the camera central point. The camera center is placed at the world coordinates origin.

4.3.1 Perspective Imaging of Planes

The process of image formation can be defined as the formation of two-dimensional (2D) representation of three-dimensional (3D) world. The usual way of modeling this process is by *central projection* in which a ray from a point in space is drown from a 3D world point through a fixed point in space, the *center of projection*. This ray intersects a specific plane in space chosen as the *image plane*. This intersection represents the image of the 3D point.

This model of projection is the model used for describing the pinhole camera model, where we assume that the camera is just a point that is chosen as the *center of projection*. Under such a model, points in the world coordinate system are mapped into points in the image coordinates according to a perspective projection relation as shown in figure 4.1, and the mapping between a 3D world point $\mathbf{X}$ and it's 2D image $\mathbf{x}$ is given by the equation

$$\mathbf{x} = P\mathbf{X} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{4.1}$$

where $\mathbf{x}$ is the homogenous 3-vector representing the image point in the image coordinate system, $\mathbf{X}$ is the homogenous 4-vector representing the world point point in the world coordinate, and P is the 3×4 camera projection matrix, with columns denoted by $\mathbf{p}_i; i = 1 : 4$.

Considering the special case of points lying on a scene plane π, as shown in

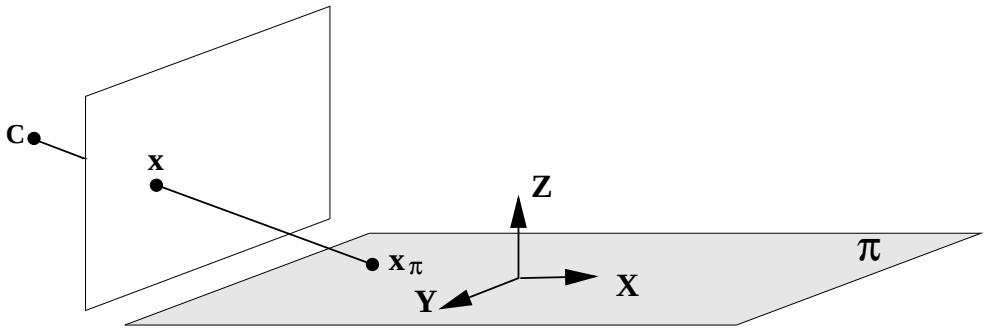


Figure 4.2: Perspective images of points in a plane[30]. The world coordinate system is moved in order to be aligned with the plane π.

figure 4.2, and since we have the freedom to move the world coordinate system, it will be placed such that the XY-plane corresponds to the plane π in the scene. This means that points on the scene plane have zero Z-coordinate. Then, the mapping in equation(4.1) will be given by

$$\mathbf{x} = P\mathbf{X} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix} \begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_4 \end{bmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{4.2}$$

This indicates that the mapping between points $\mathbf{X}_\pi = (X, Y, 1)^T$ on the world plane π and their image $\mathbf{x}$ is a general planar homography- a plane to plane projective transformation- on the form $\mathbf{x} = \mathrm{H}\mathbf{X}_\pi$, with H is a 3×3 matrix of rank 3. This fact is stated by [30] as follows:

- *The most general transformation that can occur between a scene plane and an image plane under perspective imaging is a plane projective transformation.*

4.3.2 Projective Transformation and Hierarchy of transformations

In this section, we discuss the plane projective transformation and its characteristics.

A projective transformation (projectivity) [30] is an invertible mapping from $\mathbb{P}^2$ (that is the space of homogenous 3-vectors) to points in $\mathbb{P}^2$ that maps lines to lines. An algebraic definition of projectivity is possible based on the following theorem.

Theorem 1 *A mapping* $h : \mathbb{P}^2 \rightarrow \mathbb{P}^2$ *is a projectivity if and only if there exists a non-singular* 3×3 *matrix* H *such that for any point in* $\mathbb{P}^2$ *represented by a vector* $\mathbf{x}$ *it is true that* $h(\mathbf{x}) = \mathrm{H}\mathbf{x}$.

As a result of this theorem, an alternative definition of a projective transformation may be given as follows

Definition A planar projective transformation is a linear transformation on homogenous 3-vectors represented by a non-singular 3×3 matrix:

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \tag{4.3}$$

or more briefly, $\mathbf{x}' = \mathrm{H}\mathbf{x}$

As a result from using the homogenous coordinates, the matrix H can be multiplied by a an arbitrary non-zero scale factor without altering the projective transformation. For this reason, the matrix H is called a homogenous transformation matrix and has only eight degree of freedom.

Hierarchy of transformations In order to remove the projective transformation effect on ground plane, we will explore this transformation and describe its important specializations and their geometric effects. We will introduce these transformations starting with the most specialized and progressively generalizing until the general form of projective transformation. Figure 4.3 shows an example of these transformations of the image of a tiled floor plane.

Definition. Invariants of a transformations are defined to be the set of elements or quantities that are preserved under this transformation.

Euclidean transformation

Euclidean transformations are a class of transformations that preserve Euclidean distances. It is presented as

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.4}$$

where θ is the rotation angle, t_x is the translation in the x-coordinate direction, and t_y is the translation in the y-coordinate direction.

Euclidean transformations model the motion of a rigid objet as they consist of a composition of translation and rotation. They can be written more concisely in

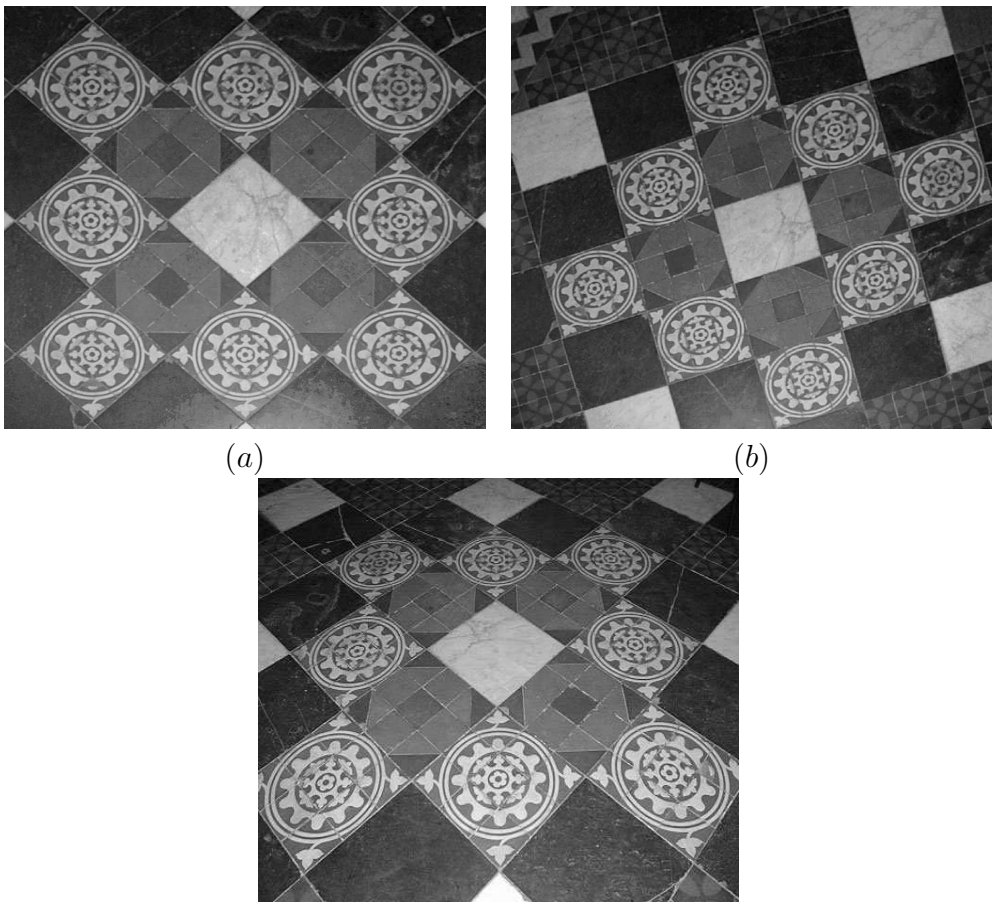


Figure 4.3: The different planar transformation under central projection. Images of a tiled floor. (From [30]) (a) **Similarity**: The circles are imaged as circles and all the angles and ratio of lengths are preserved. (b)**Affinity**: The circle is imaged as an ellipse and the right angles are not preserved any more while the parallel lines are still parallel. (c)**Projectivity**: Parallel lines intersect on the image and objects closer to the camera seem larger than far objects.

the following form

$$\mathbf{x}' = \mathrm{H_E}\mathbf{x} = \begin{bmatrix} \mathrm{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \mathbf{x} \tag{4.5}$$

where, R is a 2×2 rotation matrix (an orthonormal matrix such that $\mathrm{RR}^T = \mathrm{R}^T\mathrm{R} = \mathrm{I}$) and $\mathbf{t}$ is a translation 2-vector. The planar Euclidean transformation has three degrees of freedom. One for the rotation and two for the translation. As for the invariants, the Euclidean transformations preserve the metric properties of the objects, for instance: length, angle and area.

Similarity transformation

A similarity transformation is an Euclidean transformation with an isotropic scaling. It is presented as

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.6}$$

It can be written more concisely in the following form

$$\mathbf{x}' = \mathrm{H_S}\mathbf{x} = \begin{bmatrix} s\mathrm{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \mathbf{x} \tag{4.7}$$

where the scalar s represents the isotropic scaling.

The planar Euclidean transformation has four degrees of freedom. As for the invariants, the similarity transformation like the Euclidean ones preserve the angle between the lines, in particular parallel lines are mapped to parallel ones. The length between two points is not a similarity invariant, but the ratio of two lengths is invariant.

Affine transformation

An Affine transformation (affinity) is a non-singular linear transformation followed by a translation. It is presented by

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{4.8}$$

or in a block form

$$\mathbf{x}' = \mathrm{H_A}\mathbf{x} = \begin{bmatrix} \mathrm{A} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \mathbf{x} \tag{4.9}$$

where A is a 2×2 non-singular matrix.

The planar affine transformation has six degrees of freedom. As for the invariants, the most important invariant property is that an affine transformation maps parallel lines to parallel lines. This results from the fact that parallel lines intersect at what are called as points-at-infinity $(x, y, 0)^T$. These points are mapped by affine transformation to points at infinity also. Other invariants are ratio of length of parallel line segments and ratio of areas.

Projective transformation

A projective transformation is the most general case of this class of linear transformations of homogenous coordinates. The matrix form of this transformation is shown in equation (4.3). In a block form

$$\mathbf{x}' = \mathrm{H_P}\mathbf{x} = \begin{bmatrix} \mathrm{A} & \mathbf{t} \\ \mathbf{v}^T & v \end{bmatrix} \mathbf{x} \tag{4.10}$$

This transformation, as shown before, has eight degrees of freedom. Thus correspondences of four pairs of points can be used to recover the projective transformation between two planes. The most fundamental projective invariant is the cross ratio of four collinear points. An important feature of projective transformation is the mapping of points at infinity into finite points, which is called the vanishing points. This results in intersection of the images of parallel lines, as shown in figure 4.4. A projective transformation can be decomposed into a chain of transformations, as

$$\mathrm{H} = \mathrm{H_S H_A H_P} = \begin{bmatrix} s\mathrm{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathrm{K} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathrm{I} & \mathbf{0} \\ \mathbf{v}^T & v \end{bmatrix} = \begin{bmatrix} \mathrm{A} & \mathbf{t} \\ \mathbf{v}^T & v \end{bmatrix} \tag{4.11}$$

where A is a non-singular matrix given by $\mathrm{A} = s\mathrm{RK}+\mathbf{t}\mathbf{v}^T$, and K an upper-triangular matrix normalized as $\det \mathrm{K} = 1$. This decomposition is valid provided $v \neq 0$, and is unique if s is chosen positive.

4.3.3 Ground Plane Recovery

We need to have an automatic or semi-automatic method to calibrate for the ground plane in video sequences. By calibration we mean the removal of the projective and affine transformation effects, and recovery of the planar scene up to a similarity transformation. We use the method presented in [31] for metric recovery. In this method the recovery of the metric properties is achieved by first recovering the affine and then the metric properties. This is done by estimating the $\mathrm{H_P}$ and $\mathrm{H_A}$ matrices in equation (4.11) using some of the features often found in man made scenes, such as, parallel lines and right angles.

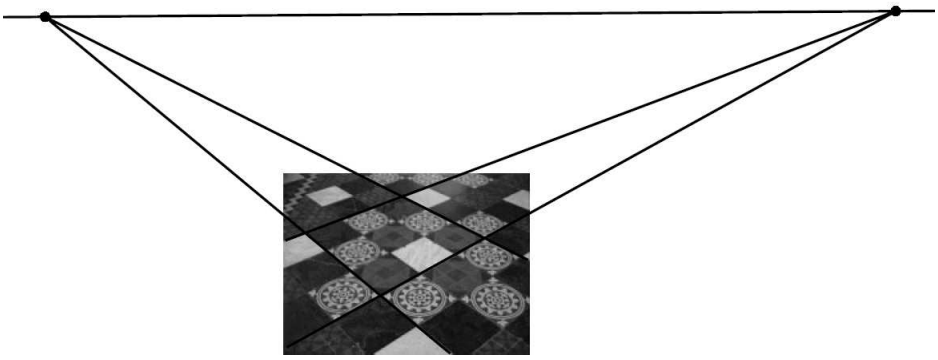


Figure 4.4: Vanishing line of the plane computed using the intersection of two sets of parallel lines.

From projective to affine

In order to recover the affine transformed image, we need to estimate the pure projective transformation matrix $\mathrm{H_P}$. We note that the inverse of this projective transformation is also a projective transformation $\hat{\mathrm{H}}_\mathrm{P}$, which can be written in the following form

$$\hat{\mathrm{H}}_\mathrm{P} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{bmatrix} \tag{4.12}$$

where $\mathbf{l}_\infty = (l_1, l_2, l_3)^T$ is the vanishing line of the plane, defined as the line connecting all the vanishing points for lines lying on the plane as shown in figure 4.4. The vector $\mathbf{l}_\infty$ is homogenous, so it has two degrees of freedom only, where the homogenous vector $(a, b, c)^T$ and $k(a, b, c)^T$ represents the same line, for any nonzero k.

From equation (4.12), it is evident that identifying the vanishing line is enough to remove the pure-projective part of the projection. In order to identify the vanishing line, two sets of parallel lines should be identified. Parallel lines are easy to find in man made environments (e.g. parking spaces, curbs, and road lanes).

From affine to metric

The second stage of the rectification is the removal of the affine projection. In the same way as in the first stage, The inverse affine transformation matrix $\hat{\mathrm{H}}_\mathrm{A}$ can be written in the following form

$$\hat{\mathrm{H}}_\mathrm{A} = \begin{bmatrix} \frac{1}{\beta} & -\frac{\alpha}{\beta} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.13}$$

Also, this matrix has two degree of freedoms represented by α and β. These two parameters have a geometric interpretation as representing the circular points, which are a pair of points-at-infinity that are invariant to Euclidean transformation. Once these points are identified, metric properties of the plane are available.

Different methods can provide constraints on the values of α and β. Each of these constraint is of the form of a circle equation. Therefore, providing two of these constraints can be enough to determine the affine transformation. Some of these methods are as follows.

1. A known angle between lines
   Suppose θ is the angle on the world plane between the lines imaged as $\mathbf{l}_a$ and $\mathbf{l}_b$. These lines are homogenous 3-vectors. Then it can be shown that α and β lie on the circle with center
   $$(c_\alpha, c_\beta) = \left(\frac{(a+b)}{2}, \frac{(a-b)}{2} \cot\theta \right) \tag{4.14}$$
   and radius
   $$r = \left| \frac{(a-b)}{2\sin\theta} \right| \tag{4.15}$$
   where $a = -l_{a2}/l_{a1}$ and $b = -l_{b2}/l_{b1}$ are the line directions.

2. Equality of two unknown angles
   Suppose the angle on the world plane between two lines imaged with directions a_1, b_1 equals that between two line imaged with directions a_2, b_2. Then α and β lie on a circle with center on the α axis
   $$(c_\alpha, c_\beta) = \left(\frac{a_1 b_2 - b_1 a_2}{a_1 - b_1 - a_2 + b_2}, 0 \right) \tag{4.16}$$
   and squared radius
   $$r^2 = \left(\frac{a_1 b_2 - b_1 a_2}{a_1 - b_1 - a_2 + b_2} \right)^2 + \frac{(a_1 - b_1)(a_1 b_1 - a_2 b_2)}{a_1 - b_1 - a_2 + b_2} - a_1 b_1 \tag{4.17}$$

3. A known length ratio
   Suppose that we have s, the length ratio between two non-parallel line segments on the world plane. The two segments are imaged between $((x_{11}, y_{11}), (x_{12}, y_{12}))$ and $((x_{21}, y_{21}), (x_{22}, y_{22}))$. Let $\Delta x_n = x_{n1} - x_{n2}$ and the same for y. Then, α and β lie on a circle with center
   $$(c_\alpha, c_\beta) = \left(\frac{\Delta x_1 \Delta y_1 - s^2 \Delta x_2 \Delta y_2}{{\Delta y_1}^2 - s^2 {\Delta y_2}^2}, 0 \right) \tag{4.18}$$
   and radius
   $$r = \left| \frac{s(\Delta x_2 \Delta y_1 - \Delta x_1 \Delta y_2}{{\Delta y_1}^2 - s^2 {\Delta y_2}^2} \right| \tag{4.19}$$

Two of the above constraints are required to determine α and β, and thus $\mathrm{H_A}$, as the intersection of the two circles. The most common used constraint is the known right angles, which is very common in a man-made environments.

4.4 Learning the Shape of Trajectories

In this section we present an algorithm that aims at finding a common representation for all the motion trajectories that correspond to the same activity. After performing the ground plane recovery (i.e. finding the projective $\hat{\mathrm{H}}_\mathrm{P}$ and affine $\hat{\mathrm{H}}_\mathrm{A}$ inverse transformations) the motion trajectories of the objects are reprojected to their ground plane coordinates. Having m different realizations of each activity trajectory, the goal is to obtain a common trajectory that represents all of these trajectories . We assume that all these realization correspond to the same 2D shaped trajectory but have undergone through similarity transformations. We use a factorization like algorithm to obtain this basis trajectory shape from all of the realizations.

Let T_j be the j^{th} realization of some activity trajectory that we want to learn. This trajectory was obtained by tracking an object performing some activity in the image plane over n frames. The homogenous representation of this trajectory is given by:

$$\mathrm{T}_j = \begin{bmatrix} u_{j1} & u_{j2} & \cdots & u_{jn} \\ v_{j1} & v_{j2} & \cdots & v_{jn} \\ 1 & 1 & \cdots & 1 \end{bmatrix} \tag{4.20}$$

where u, v are the 2D image plane coordinates.
Since the ground plane has been recovered as shown in the previous section, the ground plane homogenous presentation of this trajectory can be of the form

$$\tilde{\mathrm{T}}_j = \hat{\mathrm{H}}_\mathrm{A}\hat{\mathrm{H}}_\mathrm{P}\mathrm{T}_j = \begin{bmatrix} x_{j1} & x_{j2} & \cdots & x_{jn} \\ y_{j1} & y_{j2} & \cdots & y_{jn} \\ 1 & 1 & \cdots & 1 \end{bmatrix} \tag{4.21}$$

where x, y are the ground plane coordinates, and $\hat{\mathrm{H}}_\mathrm{P}$ and $\hat{\mathrm{H}}_\mathrm{A}$ are the pure-projective and affine transformations from image to ground planes respectively.
Assuming that all the different realizations correspond to the same 2D trajectory S but gone through a 2D similarity transformations (scale, rotation and translation), then

$$\begin{aligned} \tilde{\mathrm{T}}_j &= \mathrm{H}_{\mathrm{S}j}\mathrm{S} \\ &= \begin{bmatrix} s_j\cos\theta_j & -s_j\sin\theta_j & t_{xj} \\ s_j\sin\theta_j & s_j\cos\theta_j & t_{yj} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{x}_1 & \tilde{x}_2 & \cdots & \tilde{x}_n \\ \tilde{y}_1 & \tilde{y}_2 & \cdots & \tilde{y}_n \\ 1 & 1 & \cdots & 1 \end{bmatrix} \end{aligned} \tag{4.22}$$

where $\mathrm{H_{S}}_j$ is the similarity transformation between the j^{th} realization and S. This relation can be rewritten in inhomogeneous coordinates in the form

$$\begin{aligned} \hat{\mathrm{T}}_j &= \begin{bmatrix} s_j\cos\theta_j & -s_j\sin\theta_j \\ s_j\sin\theta_j & s_j\cos\theta_j \end{bmatrix} \begin{bmatrix} \tilde{x}_1 & \tilde{x}_2 & \cdots & \tilde{x}_n \\ \tilde{y}_1 & \tilde{y}_2 & \cdots & \tilde{y}_n \end{bmatrix} + \begin{bmatrix} t_{xj} \\ t_{yj} \end{bmatrix} \\ &= s_j\mathrm{R}_j\mathrm{S} + \mathbf{t_j} \end{aligned} \tag{4.23}$$

where s_j, R_j and $\mathbf{t}_j$ are the scale, rotation matrix and translation vector, respectively, between the j^{th} realization and the basis trajectory S.

Given m different realizations of this trajectory, each with n points length, we can construct a measurement matrix on the form

$$\mathrm{W} = \begin{bmatrix} \hat{\mathrm{T}}_1 \\ \hat{\mathrm{T}}_2 \\ \vdots \\ \hat{\mathrm{T}}_m \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ y_{11} & y_{12} & \cdots & y_{1n} \\ \vdots & \vdots & & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \\ y_{m1} & y_{m2} & \cdots & y_{mn} \end{bmatrix}_{2m\times n} \tag{4.24}$$

By substituting from equation (4.23), the measurement matrix can be written on the form:

$$\mathrm{W} = \begin{bmatrix} s_1\mathrm{R}_1 \\ s_2\mathrm{R}_2 \\ \vdots \\ s_m\mathrm{R}_m \end{bmatrix} \mathrm{S} + \begin{bmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \\ \vdots \\ \mathbf{t}_m \end{bmatrix} \tag{4.25}$$

The translation effect can be removed by subtracting out the mean of all the 2D points, as in [32]. So we will have a new measurement matrix W, as in equation (4.24) with the mean subtracted from each row.

$$\begin{aligned} \mathrm{W} &= \begin{bmatrix} s_1\mathrm{R}_1 \\ s_2\mathrm{R}_2 \\ \vdots \\ s_m\mathrm{R}_m \end{bmatrix} \mathrm{S} \\ &= \mathrm{P}_{2m\times 2}\mathrm{S}_{2\times n} \end{aligned} \tag{4.26}$$

Thus the measurement matrix has a maximum rank of two. The matrix P contains the pose or orientation for each realization. The matrix S contains the trajectory basis shape of the trajectory that captures the structure of the trajectory.
Using a rank constraint on the measurement matrix as in the famous factorization approach [32], the measurement matrix can be factorized into two matrices $\tilde{\mathrm{P}}$ and $\tilde{\mathrm{S}}$ by using singular value decomposition (SVD) and retaining the top two singular values, as shown in equation (4.27).

$$\mathrm{W} = \mathrm{UDV}^T \tag{4.27}$$

and taking $\tilde{\mathrm{P}} = \mathrm{U}'\mathrm{D}'^{\frac{1}{2}}$ and $\tilde{\mathrm{S}} = \mathrm{D}'^{\frac{1}{2}}\mathrm{V}'^{T}$, where $\mathrm{U}', \mathrm{D}', \mathrm{V}'$ are the truncated versions of $\mathrm{U}, \mathrm{D}, \mathrm{V}$ by retaining only the top two singular values. However, this factorization is not unique, as for any non-singular 2×2 matrix Q

$$\mathrm{W} = \tilde{\mathrm{P}}\tilde{\mathrm{S}} = (\tilde{\mathrm{P}}\mathrm{Q})(\mathrm{Q}^{-1}\tilde{\mathrm{S}}) \tag{4.28}$$

So, we want to remove this ambiguity by finding the matrix Q that would transform $\tilde{\mathrm{P}}$ and $\tilde{\mathrm{S}}$ into the pose and shape matrices $\mathrm{P} = \tilde{\mathrm{P}}\mathrm{Q}$ and $\mathrm{S} = \mathrm{Q}^{-1}\tilde{\mathrm{S}}$ as in equation (4.26). To find Q we use the metric constraint on the rows of P, that was used in [32].

By multiplying P by its transpose P^T

$$\begin{aligned} \mathrm{P}\mathrm{P}^T &= \begin{bmatrix} s_1\mathrm{R}_1 \\ s_2\mathrm{R}_2 \\ \vdots \\ s_m\mathrm{R}_m \end{bmatrix} \begin{bmatrix} s_1\mathrm{R}_1 & s_2\mathrm{R}_2 & \cdots & s_m\mathrm{R}_m \end{bmatrix} \\ &= \begin{bmatrix} s_1^2\mathbf{I}_2 & & \\ & \ddots & \\ & & s_m^2\mathbf{I}_2 \end{bmatrix} \end{aligned} \tag{4.29}$$

where $\mathbf{I}_2$ is a 2×2 identity matrix. This follows from the orthonormality of the rotation matrices R_j. Substituting for $\mathrm{P} = \tilde{\mathrm{P}}\mathrm{Q}$

$$\begin{aligned} \mathrm{P}\mathrm{P}^T &= \tilde{\mathrm{P}}\mathrm{Q}\mathrm{Q}^T\tilde{\mathrm{P}}^T \\ &= \begin{bmatrix} \mathbf{a_1} \\ \mathbf{b_1} \\ \vdots \\ \mathbf{a_m} \\ \mathbf{b_m} \end{bmatrix} \mathrm{Q}\mathrm{Q}^T \begin{bmatrix} \mathbf{a_1}^T & \mathbf{b_1}^T & \cdots & \mathbf{a_m}^T & \mathbf{b_m}^T \end{bmatrix} \end{aligned} \tag{4.30}$$

where $\mathbf{a}_i$ and $\mathbf{b}_i$, $i = 1 : m$, are the odd and even rows of $\tilde{\mathrm{P}}$, respectively. From Equations (4.29) and (4.30), we can obtain the following constraints on the matrix $\mathrm{Q}\mathrm{Q}^T$, $\forall i = 1, \ldots, m$, such that

$$\begin{aligned} \mathbf{a}_i\mathrm{Q}\mathrm{Q}^T\mathbf{a}_i^T = \mathbf{b}_i\mathrm{Q}\mathrm{Q}^T\mathbf{b}_i^T &= s_i^2 \\ \mathbf{a}_i\mathrm{Q}\mathrm{Q}^T\mathbf{b}_i^T &= 0 \end{aligned} \tag{4.31}$$

Using these $2m$ constraints on the elements of $\mathrm{Q}\mathrm{Q}^T$, we can find the linear least square estimate (LLSE) solution for $\mathrm{Q}\mathrm{Q}^T$. Then Q can be estimated through SVD, and it is unique up to a 2×2 rotation matrix. This ambiguity comes from the selection of the reference coordinate system and can be eliminated by selecting the first realization as a reference, i.e selecting $\mathrm{R}_1 = \mathrm{I}_{2\times 2}$.

4.5 Testing Trajectories

In order to test an observed trajectory T_x for belonging to a certain learnt activity, two steps are needed:

1. Compute the optimal rotation and scaling matrix $s_x\mathrm{R}_x$ in the LLSE sense, such that,

$$\mathrm{T}_x \simeq s_x\mathrm{R}_x\mathrm{S} \tag{4.32}$$

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \end{bmatrix} \simeq s_x\mathrm{R}_x \begin{bmatrix} \tilde{x}_1 & \tilde{x}_2 & \cdots & \tilde{x}_n \\ \tilde{y}_1 & \tilde{y}_2 & \cdots & \tilde{y}_n \end{bmatrix} \tag{4.33}$$

The matrix $s_x\mathrm{R}_x$ has only two degrees of freedom, which correspond to the scale s and rotation angle θ, we can write the matrix $s_x\mathrm{R}_x$ as

$$s_x\mathrm{R}_x = \begin{bmatrix} s_x\cos\theta_x & -s_x\sin\theta_x \\ s_x\sin\theta_x & s_x\cos\theta_x \end{bmatrix} = \begin{bmatrix} q_1 & -q_2 \\ q_2 & q_1 \end{bmatrix} \tag{4.34}$$

where $q_1 = s_x\cos\theta_x$ and $q_2 = s_x\sin\theta_x$. For each of the columns of the trajectory matrix T_x, we have the following relation

$$\begin{bmatrix} x_j \\ y_j \end{bmatrix} = \begin{bmatrix} q_1 & -q_2 \\ q_2 & q_1 \end{bmatrix} \begin{bmatrix} \tilde{x}_j \\ \tilde{y}_j \end{bmatrix} \tag{4.35}$$

which can be written in the following form

$$\begin{bmatrix} x_j \\ y_j \end{bmatrix} = \begin{bmatrix} \tilde{x}_j & -\tilde{y}_j \\ \tilde{y}_j & \tilde{x}_j \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \tag{4.36}$$

Repeating this procedure for each column of T_x, we get $2n$ equations in the two variables q_1 and q_2 that can be stacked as

$$\begin{bmatrix} x_1 \\ y_1 \\ \vdots \\ x_m \\ y_m \end{bmatrix} = \begin{bmatrix} \tilde{x}_1 & -\tilde{y}_1 \\ \tilde{y}_1 & \tilde{x}_1 \\ \vdots & \vdots \\ \tilde{x}_m & -\tilde{y}_m \\ \tilde{y}_m & \tilde{x}_m \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \tag{4.37}$$

$$\mathbf{x} = \mathrm{M}\mathbf{q} \tag{4.38}$$

A LLSE technique is used to fine the optimal $s_x\mathrm{R}_x$ parameters that minimizes the mean square error between the tested trajectory and the rotated basis shape for this activity by finding the LLSE solution for $\mathbf{q}_{LLSE}$.

$$\mathbf{q}_{LLSE} = \arg\min_{\mathbf{q}\in\mathbb{R}^2} ||\mathbf{x} - \mathrm{M}\mathbf{q}||^2 \tag{4.39}$$

2. After the optimal transformation matrix is calculated, the correlation between the trajectory and the transformed basis shape is calculated and used for making a decision. The Frobenius norm of the error matrix is used as an indication of the level of correlation, which represents the mean square error (MSE) between the two matrices. The error matrix is calculated as the difference between the tested trajectory matrix T_x and the rotated activity shape, as follows

$$\Delta_x = \mathrm{T}_x - s_x \mathrm{R}_x \mathrm{S} \tag{4.40}$$

The Frobenius norm of a matrix $\mathbf{A}$ is defined as the square root of the sum of the absolute squares of its elements,

$$||A||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}^2|} \tag{4.41}$$

The value of the error is normalized with the signal energy to give the final normalized mean square error (NMSE) on the following form

$$NMSE = \frac{||\Delta_x||_F}{||\mathrm{T}_x||_F + ||s_x \mathrm{R}_x \mathrm{S}||_F} \tag{4.42}$$

Comparing the value of this NMSE to this normal activity learnt NMSE values, a decision can be made as to weather the observed trajectory belongs to this activity or not.

4.6 Experiment and Results

We tested the proposed algorithm using a set of real trajectories, which resulted from applying our motion detection and tracking over the FPSS data set mentioned before. These data set represent the monitoring of humans and vehicles moving around a large parking lot. We focused on learning the trajectories corresponding to human activities; however the same experiment can be applied to vehicle trajectories. The normal activity in these sequence correspond to a person moving into the parking lot and approaching his car, or stepping out of his car and moving out of the parking lot. We manually picked the trajectories corresponding to normal activities from the tracking results to assure stable tracking results in the training set.

The ground plane calibration is done using the semi automatic algorithm presented earlier in this chapter. No ground truth information is used in the calibration procedure.

After learning of the shape of the basis trajectory, the NMSE of some of the normal activity trajectories is computed. We use both the learning probe and other

normal trajectories that were not involved in the learning process. Some abnormal trajectories are tested representing people wondering around in the parking lot area, standing in the same place for a long time, or doing suspicious motion around parked cars. The results of the NMSE for both sets are presented.

4.6.1 Ground Plane Calibration Results

As described earlier, the first step in our algorithm is the recovery of the ground plane. This process need to be done once for each camera, and the transformation matrix can then be used for all the subsequent sequences. The availability of some ground truth information, such as the exact distance between two points or the measured angle between two line can, facilitate the process and increase the accuracy. However, in this experiment we perform the calibration without using any of these information.

The first step is the recovery of the affine parameters. As described earlier the recovery of the pure-projective transformations is achieved by finding the vanishing line of the ground plane. Figure 4.5 shows the original frame from the uncalibrated sequence. Four points p_1, p_2, p_3, p_4 are identified on the image such that $(p_1p_2//p_4p_3)$ and $(p_1p_4//p_2p_3)$, where $(x//y)$ denote the parallel relation between the line segments x and y. These lines represent the horizontal and vertical borders of a parking space. Using the intersection of each pair of blue lines as a vanishing point, the vanishing line can be identified and the affine transformation recovery matrix is found as in equation (4.12). The affine recovered image is shown in figure 4.6.

The second step in the recovery process is the recovery of the metric properties. For this step we use two different constraints shown in figure 4.7. The first constraint is the right angle between the horizontal and vertical borders of the parking space (R_2R_3) and (R_1R_2), respectively. The other constraint is found by tracking the span between the tires of a truck between two far sequences, denoted by the weight points (S_1, S_2) and (S_3, S_4). Another constraint can be found using the length ratio between the width and length of a parking spot, which is usually a known ratio. Using these constraints the values of α and β can be found, and hence the metric recovery matrix can be found as in equation (4.13). The metric recovered image of the plane is shown in figure 4.8.

4.6.2 Trajectories Learning and Testing Results

As described earlier, the process of testing an observed trajectory is performed in two phases. First, a learning set of trajectories for each normal activity is used to learn the basis shape of this activity. The number of activities depends on the nature of the scene under surveillance. In our experiment, we had only one activity

Figure 4.5: The original frame used in the affine recovery process, showing the two pairs of parallel lines used to locate the vanishing line.

Figure 4.6: The Affine rectified image.

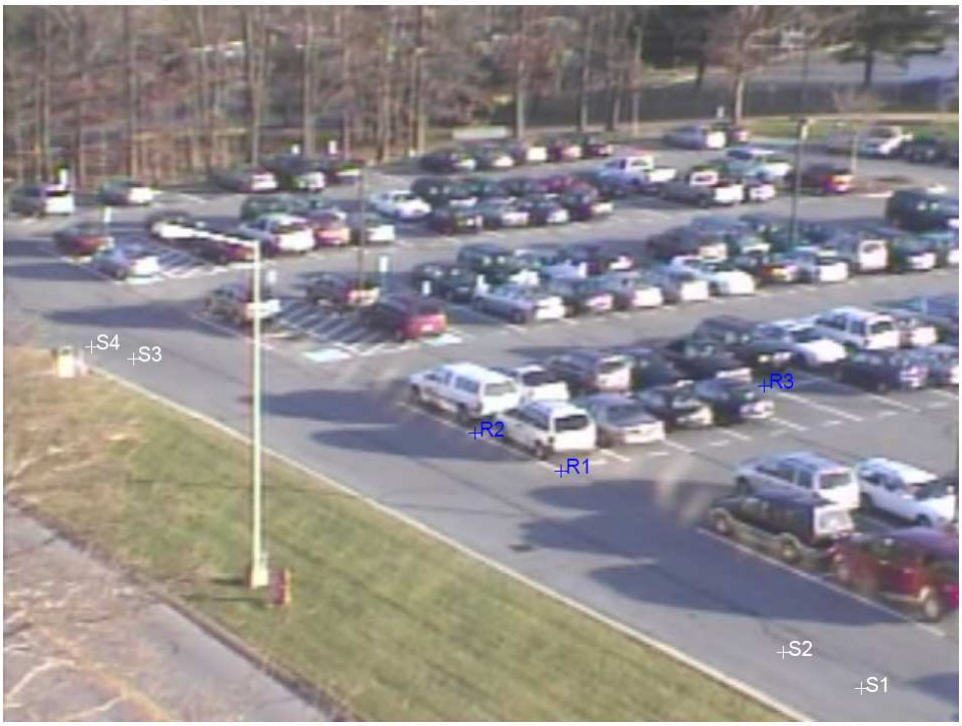


Figure 4.7: The original frame used in the affine to metric recovery process.

corresponding to the normal human motion approaching or leaving a vehicle in the parking lot. The second phase is the testing phase, where we compute the NMSE values for comparing the observed trajectory to each of the normal activities, these values are used for classifying the trajectory to one of the activities or as an abnormal trajectory.

Time scaling

Our formulation is based on comparing equal length trajectories (same number os samples n), so in order to deal with the problem of different length trajectories we adopt the following procedures:

- Each trajectory is divided into segments of a common length n. We pick $n = 50$ frames in our experiment.
- A multi-scale technique is used by performing the testing on different combination of segments, ranging from the finest scale (the line segments) to the course scale (the whole trajectory). This technique gives the ability to evaluate each section of the trajectory along with the overall trajectory. An example of the different training sequences that can be obtained from a $3n$ trajectory

Figure 4.8: The Metric recovered image.

x_1	$\cdots$	x_n	x_{n+1}	$\cdots$	x_{2n}	x_{2n+1}	$\cdots$	x_{3n}
y_1	$\cdots$	y_n	y_{n+1}	$\cdots$	y_{2n}	y_{2n+1}	$\cdots$	y_{3n}

Scale	Segment Representation	Trajectory points	Processing type
1	(1,1)	$x_1 : x_n$	*No Processing*
		$y_1 : y_n$	*No Processing*
	(1,2)	$x_{n+1} : x_{2n}$	*No Processing*
		$y_{n+1} : y_{2n}$	*No Processing*
	(1,3)	$x_{2n+1} : x_{3n}$	*No Processing*
		$y_{2n+1} : y_{3n}$	*No Processing*
2	(2,1)	$x_1 : x_{2n}$	$Downsample_2$
		$y_1 : y_{2n}$	$Downsample_2$
	(2,2)	$x_{n+1} : x_{3n}$	$Downsample_2$
		$y_{n+1} : y_{3n}$	$Downsample_2$
3	(3,1)	$x_1 : x_{3n}$	$Downsample_3$
		$y_1 : y_{3n}$	$Downsample_3$

Table 4.1: The different trajectory sequences generated from a three segments trajectory.

is given in Table 4.1, where $Downsample_m$ denotes the process of keeping every m^{th} sample and discarding the rest. We provide a representation of the segments in the form of an ordered pair (i, j) where i represent the scale of the segment and j represent the order of this segment within the scale i. If different trajectories are to be addressed, another variable k will be added to represent the trajectory number in the form (k, i, j).

The Training Phase

For learning the normal activity trajectory, we used a training data set containing the tracking results for 17 objects of different track length. The trajectories were first smoothed using a five points moving average to remove the instability resulting from the tracking and then they were used to generate track segments of 50 points length as described earlier, resulting in 60 learning segments. All the trajectories represent the same activity of a person entering the parking lot and moving towards his/her car or a person moving out of the parking lot. Figure 4.9 shows the image plane trajectories used in the learning process, each of the red crosses representing the center of the bounding box of an object at a certain frame. Figure 4.10 shows the projection of these trajectories on the recovered ground plane.

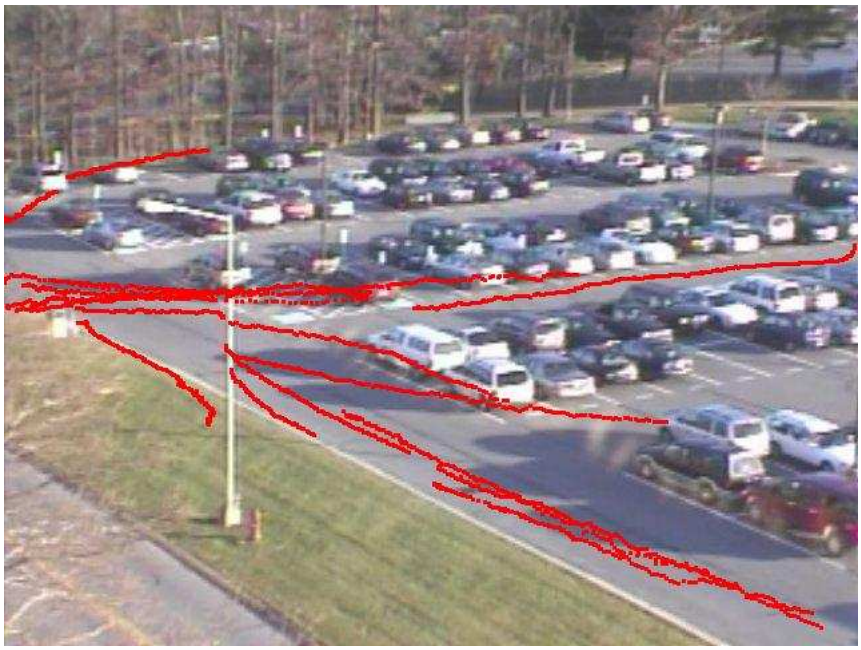


Figure 4.9: The trajectories corresponding to normal activities used in the learning process, each red point representing the location of the object at a certain frame.

The testing phase

In this stage the NMSE between the tested segment and the rotated basis shape is calculated and compared with a threshold.

First, we used a set of normal trajectories to determine the range of the NMSE in case of normal trajectory. We used both the training set and some other normal trajectories. Figure 4.6.2 shows the NMSE values for the segments of the training set sequence.

Abnormal Scenario Number one This testing sequence represents a human moving in the parking lot and then stopping in the same location for some time. The first part of the trajectory, which lasts for 100 frames (two segments), is a normal activity trajectory, but the third segment represent an abnormal act. This could be a situation of interest in surveillance scenario. Figure 4.12 shows the different segments of the object trajectory, along with the NMSE associated with each new segment. We see that as the object stops moving in the third segment, the NMSE values raise to indicate a possible drift of the object trajectory from the normal trajectory.

Abnormal Scenario Number two In this abnormal scenario, a group of tracked humans drift from their path into the grass area surrounding the parking lot, stop there for a while lifting a large box and then move the box out. Figures

Figure 4.10: The learning trajectory set projected back to the ground plane.

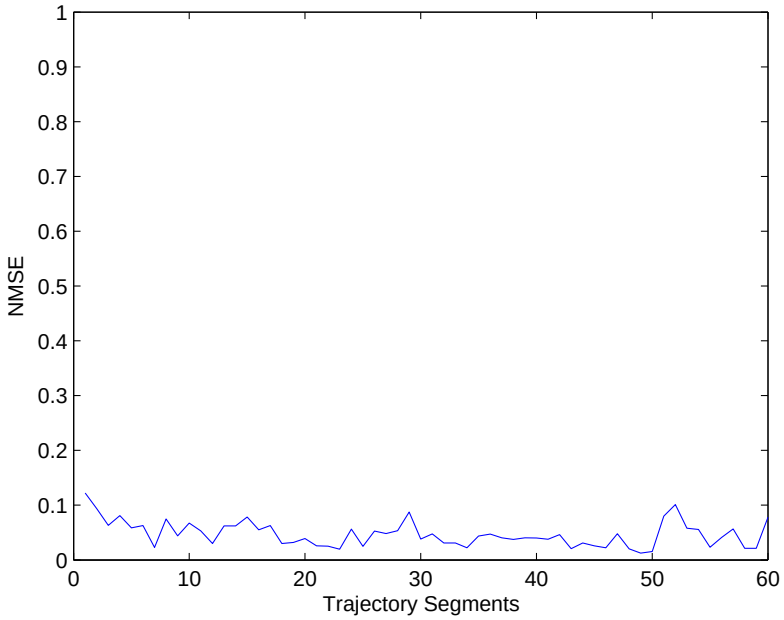


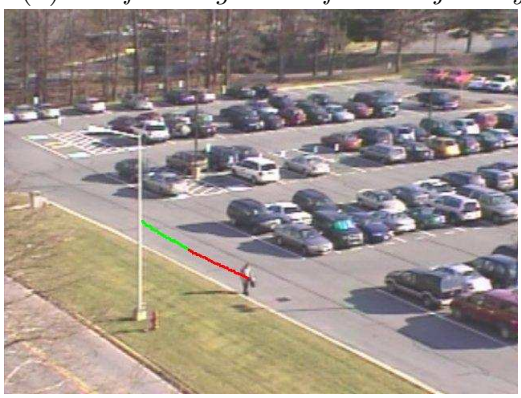
Figure 4.11: The NMSE values for the segments of the training sequence.

4.14 and 4.15 show the different segments of the object trajectory, along with the NMSE associated with each new segment. Figure 4.13 shows a plot of the NMSE of all the segments, in red, with relative to the normal trajectory NMSE, in blue. It can be verified from the figure that the trajectory was changing from normal to abnormal one in the last three or four segments, which caused the global trajectory NMSE to raise respectively.

An important property that can be noticed in our algorithm is that it captures the change in motion pattern between segments. This is because of the use of time scaling segmentation of the trajectory and grouping all possible combinations of adjacent segments. This can be helpful as the abrupt change in human motion pattern, like sudden running, is a change that is worthy of being signaled out from surveillance point of view.

$NMSE(1,1) = 0.0225$

(a) *The first segment of the trajectory with its associated NMSE*



$NMSE(1,2) = 0.0296$
$NMSE(2,1) = 0.0146$

(b) *The second segment of the trajectory with its associated NMSE*

$NMSE(1,3) = 0.3357$
$NMSE(2,2) = 0.1809$
$NMSE(3,1) = 0.1123$

(c) *The third segment of the trajectory with its associated NMSE*

Figure 4.12: The First testing scenario.

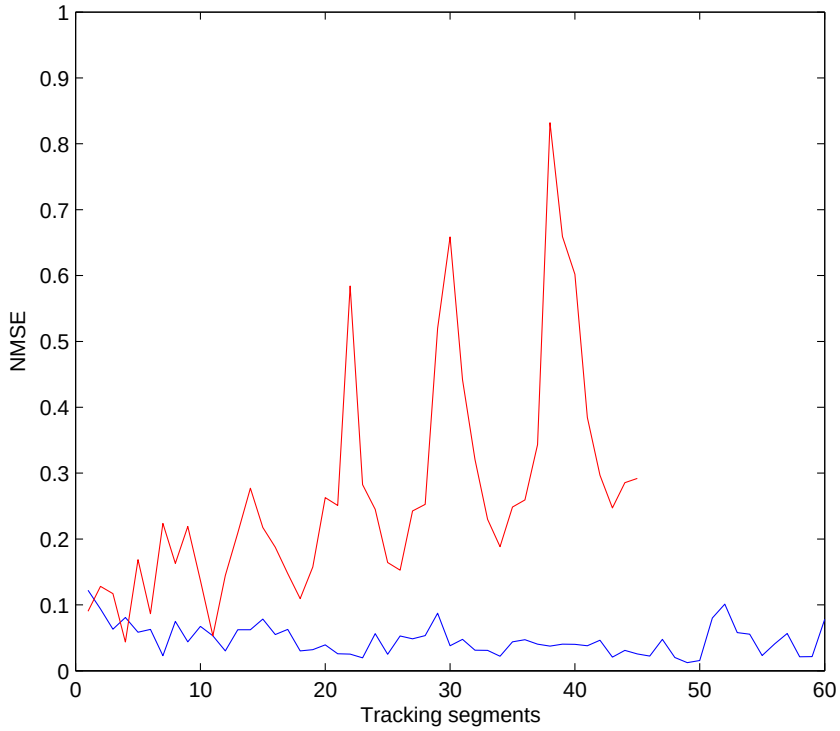
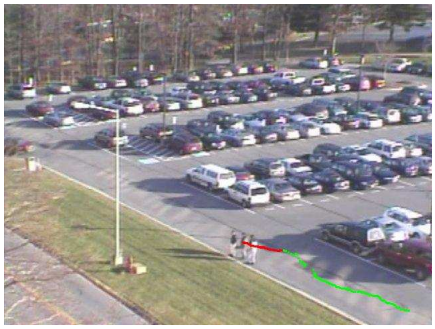


Figure 4.13: The NMSE values for the segments of the normal trajectory sequences in blue, and for the second abnormal scenario in red.

$NMSE(1,1) = 0.0903$

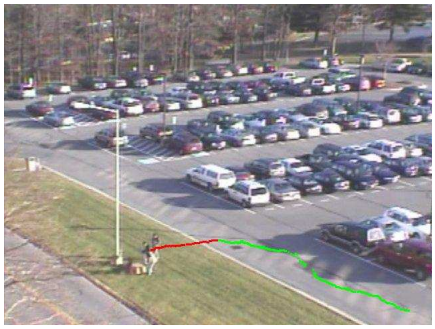
$NMSE(1,2) = 0.1280$
$NMSE(2,1) = 0.1169$



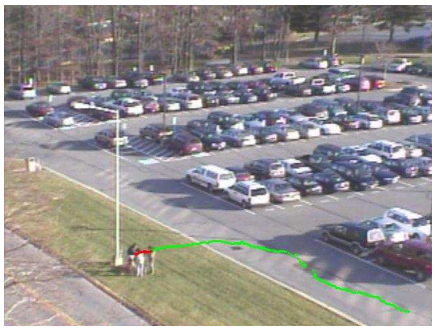
$NMSE(1,3) = 0.04347$
$NMSE(2,2) = 0.1686$
$NMSE(3,1) = 0.0866$

$NMSE(1,4) = 0.2235$
$NMSE(2,3) = 0.1629$
$NMSE(3,2) = 0.2190$
$NMSE(4,1) = 0.1369$

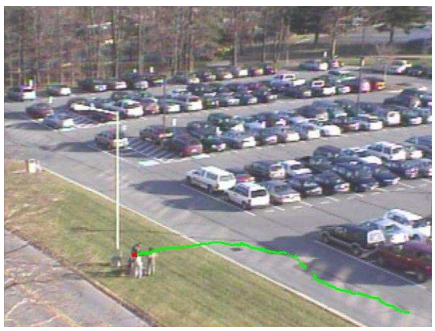
Figure 4.14: The Second testing scenario.



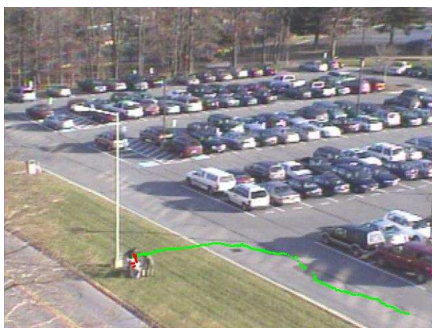
$NMSE(1,5) = 0.0531$
$NMSE(2,4) = 0.1444$
$NMSE(3,3) = 0.2092$
$NMSE(4,2) = 0.2772$
$NMSE(5,1) = 0.2176$



$NMSE(1,6) = 0.1872$
$NMSE(2,5) = 0.1474$
$NMSE(3,4) = 0.1095$
$NMSE(4,3) = 0.1574$
$NMSE(5,2) = 0.2629$
$NMSE(6,1) = 0.2508$



$NMSE(1,7) = 0.5842$
$NMSE(2,6) = 0.2826$
$NMSE(3,5) = 0.2449$
$NMSE(4,4) = 0.1642$
$NMSE(5,3) = 0.1527$
$NMSE(6,2) = 0.2425$
$NMSE(7,1) = 0.2524$



$NMSE(1,8) = 0.5197$
$NMSE(2,7) = 0.6585$
$NMSE(3,6) = 0.4417$
$NMSE(4,5) = 0.3207$
$NMSE(5,4) = 0.2297$
$NMSE(6,3) = 0.1884$
$NMSE(7,2) = 0.2484$
$NMSE(8,1) = 0.2594$

Figure 4.15: Contd. The Second testing scenario.

Chapter 5

Performance evaluation

5.1 Overview

Performance evaluation of video surveillance systems has become a very essential requirement, specially with the increased interest in this area in the last few years. Many surveillance systems were introduced, which initiated the need for an objective evaluation of the effectiveness of a surveillance system. performance evaluation studies facilitate comparisons of these systems and measure improvements of existing systems.

We will present a survey of the past work in the performance evaluation area and then present evaluation results obtained by testing our system on a set of manually ground-truthed dataset.

5.2 Performance evaluation: A review

Ellis [36] provides a discussion on different approaches to performance evaluation. He covers many areas, such as how algorithm cope in different physical conditions, i.e. weather, illumination and clutter motion, and the need to compare the tracked data to marked up data.

A number of semi-automatic tools are currently available for generating ground truth from pre-recorded video. The Open development environment for evaluation of video systems (ODViS) [33] allows the user to generate ground truth, incorporate any tracking algorithm into the environment and define any error metric through a nice graphical user interface. The Video performance evaluation Resource (ViPER) [34] is directed more towards performance evaluation for video analysis systems. It also provides an interface to generate ground truth, metrics for evaluation and visualization of video analysis results.

The use of synthetic video for evaluation purpose has gained some interest as a mean of avoiding the manual ground truth generation process. In [38], an automatic evaluation system on object surveillance (AESOS) is introduced. This system is used to evaluate the operational range of video surveillance systems in terms of robustness and reliability. It uses semi and full-synthesis video sequences under controlled variation of selected parameters like number of objects, occlusion

probability and foreground-background contrast.

In [37] pseudo-synthesis videos are used to evaluate performance of surveillance systems, where tracks are automatically selected from a surveillance database and then used to generate ground truthed video sequence with a controlled level of perceptual complexity. A nice set of surveillance metrics are used to characterize the tracking performance. Needham and Boyle in [35] focus on the evaluation of how well a tracker is able to determine the position of a target object by comparing the resulting trajectory with a ground truth trajectory of the moving object. They compare the trajectories using mean, median, standard deviation, max and min of the displacement error between the spatially and temporally aligned trajectories.

5.3 Performance metrics

In order to evaluate the performance of our surveillance system we adopted some of the performance metric used in [37]. These metrics and their significance are described below

$$\begin{aligned} \textit{Tracker Detection Rate (TRDR)} &= \frac{\textit{Total True Positives}}{\textit{Total Number of Ground Truth Points}} \\ \textit{False Alarm Rate (FAR)} &= \frac{\textit{Total False Positives}}{\textit{Total True Positives+Total False Positives}} \end{aligned} \tag{5.1}$$

Where a ***True Positive*** is defined as a ground truth point that is located within the bounding box of an object detected and tracked by the tracking algorithm. A ***False negative*** is a ground truth point that is not located within the bounding box of any object tracked by the tracking algorithm. A ***False positive*** is an object that is tracked by the system that does not have a matching ground truth point. These two metrics: tracker detection rate ***(TRDR)*** and false alarm rate ***(FAR)*** characterize the tracking performance of the motion detection and object tracking algorithms. They do not involve the ability to maintain the same object identity through the object tracks.

$$\textit{Track Detection Rate (TDR)} = \frac{\textit{Number of true positives for tracked object}}{\textit{Total number of ground truth points for object}}$$

$$\textit{Track Fragmentation(TF)} = \textit{Number of tracks matched to a ground truth track}$$

$$Tracking\ Success\ Rate\ (TSR) = \frac{Number\ of\ non\text{-}fragmented\ tracks}{Number\ of\ ground\ truth\ tracks} \tag{5.2}$$

The track detection rate ***TDR*** indicates the tracking completeness of a specific ground truth track also without taking into consideration maintaining the same identity for the object which is essential for subsequent activity analysis. The object identity is considered in the last two metrics. Track fragmentation ***(TF)*** indicates how often the track label changes, which is ideally should be equal to one and larger values indicates poor trajectory maintenance, and the tracking success rate ***(TSR)*** is used to summarize the performance of the tracking algorithm with respect to track fragmentation over all ground truth tracks.

5.4 Evaluation Results

We present in this section the results obtained by testing our system on the force protection surveillance system (FPSS) dataset provided by U.S. Army Research Laboratory (ARL). The fist set of this data consists of 85 FLIR and color sequences, all of which were manually ground-truthed with respect to the location and type of moving targets on each frame. These sequences represent a wide variety of views, complexity, day times and weather conditions (shine, rain and snow) of a large parking lot. Tables (5.1, 5.2, 5.3 and 5.4) represent the evaluation results on most of this data set.

Sequence	Overall						Humans				Vehicles			
	NP	TP	FP	FN	TRDR	FAR	NP	TP	FN	TRDR	NP	TP	FN	TRDR
rf040326_0650fi	1486	1210	511	276	81%	29%	908	826	82	90%	138	98	40	71%
rf040326_0654fi	1029	645	558	384	62%	46%	527	411	116	77%	323	226	97	69%
rf040326_0657fi	2864	2225	1871	639	77%	45%	1737	1582	155	91%	866	504	362	58%
rf040326_0706fi	984	678	445	306	68%	39%	506	467	39	92%	345	159	186	46%
rf040326_0708fi	1271	1159	78	112	91%	6%	1163	1141	22	98%	0	0	0	-%
rf040326_0712fi	1913	1583	951	330	82%	37%	1339	1138	201	84%	216	134	82	62%
rf040326_0714fi	666	372	177	294	55%	32%	473	323	150	68%	47	12	35	25%
rf040326_0719fi	1480	1105	165	375	74%	12%	1082	953	129	88%	220	128	92	58%
rf040326_0729fi	1320	1097	286	223	83%	20%	1069	1050	19	98%	90	35	55	38%
rf040326_0708fi	906	741	313	165	81%	29%	680	671	9	98%	124	39	85	31%
rf040326_0733fi	597	438	353	159	73%	44%	250	245	5	98%	302	182	120	60%
rf040326_0739fi	838	744	134	94	88%	15%	692	681	11	98%	81	33	48	40%

Table 5.1: The Evaluation results on FPSS data set (1 of 4).

Sequence	Overall						Humans				Vehicles			
	NP	TP	FP	FN	TRDR	FAR	NP	TP	FN	TRDR	NP	TP	FN	TRDR
rf040617_1642fi	294	262	290	32	89%	52%	224	192	32	85%	70	70	0	100%
rf040617_1640fc	397	319	469	78	80%	59%	298	230	68	77%	99	89	10	89%
rf040617_1637fc	1294	937	2231	357	72%	70%	836	600	236	71%	305	268	37	87%
rf040510_1254fi	377	343	185	34	90%	35%	377	343	34	90%	0	0	0	-%
rf040617_1458fi	1051	975	362	76	92%	27%	703	651	52	92%	348	324	24	93%
rf040617_1537fi	2160	1974	755	186	91%	27%	1857	1689	168	90%	303	285	18	94%
rf040617_1530fi	1605	1339	261	266	83%	16%	1587	1339	248	84%	0	0	0	-%
rf040617_1531fi	1378	1131	944	247	82%	45%	771	717	54	92%	509	335	174	65%
rf040617_1533fi	2142	2039	1991	103	95%	49%	1369	1333	36	97%	766	706	60	92%
rf040617_1538fi	2809	2661	2047	148	94%	43%	2259	2193	66	97%	550	468	82	85%
rf040617_2135fi	0	0	0	0	-%	-%	0	0	0	-%	0	0	0	-%
rf040618_0636fi	470	328	166	142	69%	33%	470	328	142	69%	0	0	0	-%

Table 5.2: The Evaluation results on FPSS data set (2 of 4).

Sequence	Overall						Humans				Vehicles			
	NP	TP	FP	FN	TRDR	FAR	NP	TP	FN	TRDR	NP	TP	FN	TRDR
rf040618_0937fi	599	394	383	205	65%	49%	375	197	178	52%	224	197	27	87%
rf040618_0636fi	600	393	44	207	65%	10%	590	393	197	66%	0	0	0	-%
rf040618_0637fi	1109	775	389	334	69%	33%	984	673	311	68%	118	97	21	82%
rf040618_0633fi	1647	1555	382	92	94%	19%	1199	1154	45	96%	443	401	42	90%
rf040618_0634fi	3164	2400	1176	764	75%	32%	2509	1794	715	71%	655	606	49	92%
rf040618_0631fi	651	566	278	85	86%	32%	423	391	32	92%	228	175	53	76%
rf040618_0638fi	923	908	318	15	98%	25%	445	433	12	97%	478	475	3	99%
rf040618_0930fi	593	460	411	133	77%	47%	350	331	19	94%	243	129	114	53%
rf040618_0931fi	1312	707	581	605	53%	45%	1281	686	595	53%	0	0	0	-%
rf040618_1932fi	859	714	696	145	83%	49%	722	618	104	85%	88	68	20	77%
rf040618_0934fi	603	426	423	177	70%	49%	280	127	153	45%	323	299	24	92%
rf040618_0935fi	800	675	91	125	84%	11%	799	675	124	84%	0	0	0	-%

Table 5.3: The Evaluation results on FPSS data set (3 of 4).

Sequence	Overall						Humans				Vehicles			
	NP	TP	FP	FN	TRDR	FAR	NP	TP	FN	TRDR	NP	TP	FN	TRDR
rf040618_0030fi	0	0	0	0	-%	-%	0	0	0	-%	0	0	0	-%
rf040618_0630fi	2549	2364	679	185	92%	22%	1906	1820	86	95%	594	542	52	91%
rf040326_0736fi	1075	899	381	176	83%	29%	839	802	37	95%	82	27	55	32%
rf040618_1230fi	943	653	575	290	69%	46%	850	568	282	66%	93	85	8	91%
rf040618_1237fi	1607	1547	232	60	96%	13%	1400	1359	41	97%	207	188	19	90%
rf040618_1238fi	767	691	227	76	90%	24%	0	0	-	%	767	691	76	90%
rf040618_1233fi	1511	1205	701	306	79%	36%	901	669	232	74%	589	518	71	87%
rf040618_1410fi	284	252	80	32	88%	24%	0	0	0	-%	284	252	32	88%
rf040618_1415fi	1738	1549	67	189	89%	4%	1731	1549	182	89%	0	0	0	-%
rf040618_1419fi	757	547	259	210	72%	32%	670	471	199	70%	87	76	11	87%
rf040618_1420fi	394	277	409	117	70%	59%	347	277	70	79%	0	0	0	-%

Table 5.4: The Evaluation results on FPSS data set (4 of 4).

5.4.1 Comments on the Results

The evaluation results shown above were affected by some very important factors:

1. The nature of the used sensor: Most of the ground truthed data sequence that is available for evaluation is an FLIR sequences, except for three color sequences in a very rainy environment; however the quality of the FLIR sensor used varies widely between different sensor types. This caused the evaluation result to vary also and increased the FAR for the poor quality sequences.

2. The time of the day and the weather conditions have a large effect on the performance in FLIR sequences, due to the fact that these factors affect the relative temperature between the moving object and the still background.

3. For some of the evaluated sequences there was the problem of sudden camera jitter, which can be caused by a flying object or strong wind. This sudden motion cause the motion detector to fail and identify the whole scene as a moving object, although the system readjusts itself after a while but this cause a large drop in the performance.

4. The number of objects in the sequence alters the performance significantly where the optimum performance occurs in the case of a few independent objects in the scene. Increasing the number of objects increase the dynamic occlusion and loss of track

Chapter 6

Conclusions and Future Work

This thesis presented a video surveillance system for monitoring outdoor activities. The system operates on video sequences captured by a single stationary FLIR or a color camera. It integrates algorithms for motion detection, tracking and abnormal trajectory detection.

Temporal variance of pixel intensities is used for motion detection accompanied by background modeling to remove trail effect. Tracking is using a visual tracking algorithm that combines motion and appearance information into an observation model and uses an adaptive state transition model with adaptive velocity and noise. The particle filter is used as a tool for estimating the unknown state vector. The shape of normal activity motion trajectories is learnt using a factorization approach on the recovered ground plane trajectories. The learnt shape is used to detect any abnormal motion trajectory. The system was tested on a large data set of FLIR and color sequences collected by ARL FPSS program and the performance was evaluated with respect to manually ground-truthed data.

The system can be enhanced by including an object classification module, which classifies the moving object into categories, such as a human or a vehicle. Classification can be of great help to the trajectory characterization module where a different set of activities can be included with each type of objects. Another possible extension is to explore the combination of more than one sensor at the same time. For example, one can combines FLIR and color images for the same scene; the motion information is more reliable in the FLIR sequences while the color images give more details for the appearance information. Fusing these two sources will increase the robustness of the system and provide more reliable tracking for the moving objects.

BIBLIOGRAPHY

[1] W. Hu,T. Tan,L.Wang and S. Maybank , *A survey on visual surveillance of object motion and behaviors* ,IEEE Trans. on systems, man, and cybernetics part C: Applications and Reviews, Vol.34, NO.3, AUGUST 2004.

[2] R. T. Collins, A. J. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, O. Hasegawa, P. Burt, and L.Wixson, *A system for video surveillance and monitoring*, Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep., CMU-RI-TR-00-12, 2000.

[3] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, *Pfinder: real-time tracking of the human body,* IEEE Trans. Pattern Anal. Machine Intell., vol. 19, pp. 780-785, July 1997.

[4] I. Haritaoglu, D. Harwood, and L. S. Davis, W^4 *: Real-time surveillance of people and their activities*, IEEE Trans. Pattern Anal. Machine Intell., vol. 22, pp. 809-830, Aug. 2000.

[5] A. J. Lipton,H. Fujiyoshi and R. S. Patil , *Moving target classification and tracking from real-time video* , in Proc. of Fourth IEEE Workshop on Applications of Computer Vision, 1998. *WACV '98*

[6] C. Stauffer and W. Grimson, *Adaptive background mixture models for real-time tracking,* in Proc. IEEE Conf. Computer Vision and Pattern Recognition, vol. 2, 1999, pp. 246-252.

[7] W. E. L. Grimson, C. Stauffer, R. Romano, and L. Lee, *Using adaptive tracking to classify and monitor activities in a site*, in Proc. IEEE Conf. Computure Vision and Pattern Recognition, Santa Barbara, CA, 1998, pp. 22-31.

[8] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, *Wallflower: principles and practice of background maintenance*, in Proc. Int. Conf. Computer Vision, 1999, pp. 255-261.

[9] J. Barron, D. Fleet, and S. Beauchemin. *Performance of optical flow techniques*, International Journal of Computer Vision, 12(1):42-77, 1994.

[10] Q. Zheng and S. Der,*Moving target indication in LRAS3 sequences*, in 5th Annual Fedlab Symposium College Park MD, 2001

[11] S. Joo and Q. Zheng,*A Temporal Variance-Based Moving Target Detector*, in Proceedings of the IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS), Jan. 2005

[12] K. Suzuki, I. Horiba and N. Sugie,*Fast Connected-Component Labeling Based on Sequential Local Operations in the Course of Forward Raster Scan Followed by Backward Raster Scan* , IEEE International Conference on Pattern Recognition (ICPR'00)- Volume 2, p. 24-34

[13] S. Zhou, R. Chellappa, and B. Moghaddam, *Visual tracking and recognition using appearance-adaptive models in particle filters*. IEEE Transactions on Image Processing (IP), Vol. 11, pp. 1434-1456, November 2004.

[14] A. Baumberg and D. C. Hogg, *Learning deformable models for tracking the human body*, in Motion-Based Recognition, M. Shah and R. Jain, Eds. Norwell, MA: Kluwer, 1996, pp. 39-60.

[15] R. Polana and R. Nelson,*Low level recognition of human motion*, in Proc. IEEE Workshop Motion of Non-Rigid and Articulated Objects, Austin, TX, 1994, pp. 77-82.

[16] G. D. Hager and P. N. Belhumeur,*Efficient region tracking with parametric models of geometry and illumination*, IEEE Trans. PAMI, vol. 20, pp. 1025-1039, Oct. 1998.

[17] T. J. Broida, S. Chandra, and R. Chellappa, *Recursive techniques for estimation of 3-d translation and rotation parameters from noisy image sequences*, IEEE Trans. Aerosp. Electron. Syst., vol. 26, pp. 639-656, Apr. 1990.

[18] M. Isard and A. Blake, *Contour tracking by stochatic propagation of conditional density*, in Proc. Eur. Conf. Computer Vision, 1996.

[19] A. P. Dempster, N. M. Laird, and D. B. Rubin, *Maximum likelihood from incomplete data via the EM algorithm*, J. Roy. Stat. Soc. B, 1977.

[20] A. D. Jepson, D. J. Fleet, and T. El-Maraghi, *Robust online appearance model for visual tracking*, in Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition, vol. 1, 2001, pp. 415422.

[21] Aswin C. Sankaranarayanan, Rama Chellappa and Qinfen Zheng, *Tracking Objects in Video Using Motion and Appearance Models*, To appear in IEEE International Conference on Image Processing (ICIP 2005).

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

[23] A. Roy Chowdhury and R. Chellappa, *A factorization approach for event recognition*, in CVPR Event Mining Workshop, Madison, WI, June 2003.

[24] N. Johnson and D. Hogg,*Learning the distribution of object trajectories for event recognition*, Image Vis. Comput., vol. 14, no. 8, pp. 609-615, 1996.

[25] J. Owens and A. Hunter,*Application of the self-organizing map to trajectory classification*, in Proc. IEEE Int.Workshop Visual Surveillance, 2000, pp. 77-83.

[26] D. Makris, T.J. Ellis,*Path Detection in Video Surveillance*, in 'Image and Vision Computing', 20(12) pp. 895-803. October 2002.

[27] N. Vaswani, A. RoyChowdhury, R. Chellappa, *Shape Activities": Stochastic Models for Moving/ Deforming Shapes with Application to Abnormal Activity Detection*, To Appear in IEEE Trans. on Image Processing, 2005

[28] D. Kendall, D. Barden, T. Carne, and H. Le, *Shape and Shape Theory*. John Wiley and Sons, 1999.

[29] W. Chen, S. F. Chang,*Motion trajectory matching of video objects*, Proceedings of SPIE, Storage and Retrieval for Media Databases 2000, 3972, pp. 544-553, 2000.

[30] Hartley, R., and Zisserman, A. ,*Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000

[31] Liebowitz, D. and Zisserman, A. ,*Metric rectification for perspective images of planes*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 482488, Santa Barbara, California, June 1998.

[32] C. Tomasi and T. Kanade,*Shape and motion from image streams under orthography: A factorization method*, International Journal of Computer Vision, vol. 9, pp. 137-154, November 1992.

[33] C. Jaynes, S. Webb, R. Matt Steele, and Q. Xiong,*An Open Development Environment for Evaluation of Video Surveillance Systems*, Proceedings of the Third International Workshop on Performance Evaluation of Tracking and Surveillance (PETS2002), Copenhagen, June 2002.

[34] D. Doermann, and D. Mihalcik, *Tools and Techniques for Video Performance Evaluation*, Proceedings of the International Conference on Pattern Recognition (ICPR00), Barcelona, September 2000, pp 4167-4170.

[35] C.J. Needham and R.D. Boyle. *Performance Evaluation Metrics and Statistics for Positional Tracker Evaluation*. International Conference on Computer Vision Systems (ICVS'03), Graz, Austria, April 2003, pp 278-289.

[36] T.J. Ellis, *Performance Metrics and Methods for Tracking in Surveillance*, Proceedings of the Third International Workshop on Performance Evaluation of Tracking and Surveillance (PETS2002), Copenhagen, June 2002.

[37] J. Black, T. J. Ellis, and P. Rosin *A novel method for video tracking performance evaluation*. In Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS), pages 125132, October 2003.

[38] T. Schlögl, C. Beleznai, M. Winter and H. Bischof *Perofrmance evaluation metrics for mtion detection and tracking*, IEEE International Conference on Pattern Recognition (ICPR'04)- Volume 4, pp. 519-522