Learning an Activity-Based Semantic Scene Model

by

Dimitrios Makris

City University, London
School of Engineering and Mathematical Sciences
Information Engineering Centre

This thesis is submitted to the School of Engineering and Mathematical Sciences, City University, London, for the degree of Doctor of Philosophy. This thesis is entirely my own work and except where otherwise indicated, describes my own research

London, UK, July 2004

To my parents,
Andreas and Sofia

Contents

Acknowledgements

First of all, I would like to thank my parents for their support during all my studies. I am very grateful to my supervisor Prof. Tim Ellis for his continuous guidance, support and patience during my PhD. I would like to thank Dr. Ming Xu and James Black for our collaboration as members of the Machine Vision Group, Dr Paul Rosin for his comments, especially during those endless project meetings, Prof. Nick Karchanias, Dr. George Halikias and Dr. Efstathios Milonidis for the interesting scientific discussions that I enjoyed having with them. Many thanks to all my friends that supported me and accompanied me all these years: Eleni, Daniel, Stavros, Elena, Ermina, Giorgos, Margarita, Elli, Georgalena, Apostolis, Katerina, Galini, Dimitris and so many others. Last but not least, special thanks to my partner Georgia for her moral support and her assistance in English.

Declarations

The author grounds powers of discretion to the university librarian to allow this thesis to be copied in whole or part without further reference to him. This permission covers only single copies made for study purposes, subject to normal conditions of acknowledgement.

Some parts of the work presented in this thesis have been published or submitted for publication in the following articles:

Dimitrios Makris, Tim Ellis, “Finding Paths in Video Sequences”, *British Machine Computer Vision, BMVC2001,* pp.263-272, Manchester, UK, September 2001.

Dimitrios Makris, Tim Ellis. “Spatial and Probabilistic Modelling of Pedestrian Behaviour”, *British Machine Vision Conference, BMVC2002,* pp.557-566, Cardiff, UK, September 2002.

Dimitrios Makris, Tim Ellis, “Path Detection in Video Surveillance”, *Image and Vision Computing,* vol.20(12), pp.895-903, October 2002.

Dimitrios Makris, Tim Ellis, “Automatic Learning of an Activity-Based Semantic Scene Model”, *IEEE Int. Conf. On Advanced Video and Signal Vased Surveillance, AVSS2003,* pp.183-188, Miami, FL, July 2003.

Tim Ellis, Dimitrios Makris, James Black, “Learning a Multicamera Topology”, *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, VS-PETS2003,* pp. 165-171, Nice, France, October 2003.

J. Black, T.J. Ellis, D. Makris, “Wide Area Surveillance With a Multi-Camera Network”, *IEE Intelligent Distributed Surveillance Systems IDSS'04,* London, February 2004.

Dimitrios Makris, Tim Ellis, James Black, “Learning Scene Semantics”, *Early Cognitive Vision Workshop*, *ECOVISION 2004*, Isle of Skye, May 2004.

Dimitrios Makris, Tim Ellis, James Black, "Bridging the Gaps between Cameras", *IEEE Conference on Computer Vision and Pattern Recognition, CVPR2004,* Washington DC, USA, June 2004.

James Black, Tim Ellis, Dimitrios Makris, "A Hierarchical Database for Visual Surveillance Applications", *IEEE International Conference on Multimedia and Expo, ICME2004,* Taipei, Taiwan, June 2004.

Dimitrios Makris, Tim Ellis, “Learning Semantic Scene Models from Observing Activity in Visual Surveillance”, *IEEE Transactions on Systems, Man and Cybernetics, Part B,* submitted.

James Black, Dimitrios Makris, Tim Ellis, "Hierarchical Database for a Multi-Camera Surveillance System", *Pattern Analysis and Applications* (PAA), submitted.

Tim Ellis, James Black, Ming Xu and Dimitrios Makris, “Integration and learning of Information from multiple camera views”, *In Visual Surveillance: A Computational Approach, J. Ferryman (ed.),* Springer, to be published.

Tim Ellis, James Black, Ming Xu, Dimitrios Makris, “A Distributed Multicamera Surveillance System”, *In Ambient Intelligence, A Novel Paradigm, P. Remagnino, G.L. Foresti, T. Ellis (ed.),* Kluwer Academic Publishers, to be published.

Dimitrios Makris, Tim Ellis, James Black, “Mapping an Ambient Environment”, *In Ambient Intelligence, A Novel Paradigm, P. Remagnino, G.L. Foresti, T. Ellis (ed.),* Kluwer Academic Publishers, to be published.

Abstract

This thesis investigates how scene activity, which is observed by fixed surveillance cameras, can be modelled and learnt. Modelling of activity is performed through a spatio-probabilistic scene model that contains semantics like entry/exit zones, paths, junctions, routes and stop zones. The spatial nature of the model allows physical and semantic representation of the scene features, which can be useful in applications like video annotation and contextual databases. The probabilistic nature of the model encodes the variance and the related uncertainty of the usage of the scene features, which is useful for activity analysis applications, such as motion prediction and atypical motion detection.

A variety of models and learning methods are used to represent and automatically derive particular activity-based semantic scene elements. Expectation-Maximisation is used for learning Gaussian Mixture Models and accumulative statistics in image maps are integrated in the methods presented. Also, a novel route model and an appropriate learning algorithm are introduced. Additionally, a Hidden Markov Model superimposed on the scene model is used for enabling activity analysis.

The application of the methods is investigated for single cameras and collectively across multiple cameras. Additionally, a novel automatic cross-correlation method is introduced that reveals the topology of a network of activities, as observed by a network of uncalibrated cameras. The method is important not only because it provides an integrated activity model for all the cameras, but also because it provides a mechanism to automatically estimate the topology of the camera network, modelling the activity across the “blind” areas of the surveillance system.

All the proposed learning algorithms are unsupervised to allow automatic learning of the scene model. Their input is a set of noisy trajectories derived automatically by motion tracking modules, attached to each of the cameras.

List of Abbreviations

AHMM:	Abstract Hidden Markov Model
BBN:	Bayesian Belief Network
blob:	Binary Large OBject
CCTV:	Closed Circuit TeleVision
CHMM:	Coupled Hidden Markov Model
CUES:	City University Experimental Surveillance
DBN:	Dynamic Belief Network
DDN:	Dynamic Decision Network
DPN:	Dynamic Probabilistic Network
EM:	Expectation Maximisation
FLTC:	Fuzzy Logic Trajectory Classifier
FOV:	Field Of View
GM	Gaussian Model
GMM:	Gaussian Mixture Model
GP:	Ground Plane
GPC:	Ground Plane Constraint
HHMM:	Hierarchical Hidden Markov Model
HMM:	Hidden Markov Model
ICA:	Independent Component Analysis
LMS:	Least Median of Squares
MCAN:	Multiple Camera Activity Network
MDL:	Minimum Description Length
ML:	Maximum Likelihood
MLTC:	Maximum Likelihood Trajectory Classifier
NIHC:	Numeric Iterative Hierarchical Cluster
NN:	Neural Network
PCA:	Principal Component Analysis
pdf:	probability distribution function
PDM:	Point Distribution Model
RBHMM	Route-Based Hidden Markov Model
SVM:	Support Vector Machine
TCM:	Tracking Correspondence Model
VLHMM	Variable Length Hidden Markov Model
VMD:	Video Motion Detection
VQ:	Vector Quantisation

Chapter 1

Introduction

The main application area of this thesis is automatic visual surveillance. Nowadays, surveillance cameras are common to many public areas in the UK, from small off-licence stores to train stations, large buildings, motorways and park areas.

A traditional security surveillance system can be described as a set of CCTV cameras that send their video signals to display monitors and perhaps at the same time to analogue recording devices. Human personnel are required to monitor the display devices in real time, or to check the recorded videos off-line.

The main purposes of surveillance systems are to prevent, confront, record and identify criminal actions. Potential criminals are more cautious when they know that they are in an “under surveillance” environment. A criminal action can be viewed by security personnel instantly and may be confronted immediately. After a crime, recorded video data can be viewed and searched for evidence of the crime and its perpetrators.

The human factor in surveillance systems is very important, as the security personnel have to use their cognitive knowledge about the observed scene and about what a suspicious action may be, and to identify those events that may need further attention. However, this is a very tedious work, as the personnel are usually located in a room full of monitors that mainly display trivial and boring events for the majority of time. Fatigue, distractions and interruptions are unavoidable and it is almost certain that a significant percentage of interesting events are overlooked.

Visual surveillance systems have significantly benefited from the progress of digital technology. It is not only the replacement of the analogue devices (cameras, monitors, recording devices) with digital ones, but also the fact that digitised video data can be processed and analysed using Digital Image Processing and Computer Vision methods. Indeed, a significant amount of research has been performed in the last 10-15

years that aimed to assist visual surveillance systems. Algorithms have been developed for automatic motion detection, motion tracking, system/camera calibration, event logging, video annotation, activity and behaviour analysis, face detection and recognition and object recognition [78].

Current commercial systems are limited to only detect and track moving objects within the field of view of the cameras. For example, Video Motion Detection (VMD) systems can detect motion within user-defined windows and alert operators. However, many false alarms can be triggered due to illumination changes or non-interesting apparent motion. Consequently, operators learn to ignore them.

The goals of current research in the area of surveillance systems are the development of methods that will allow integration of information from multiple sensors and coverage of wide-area scenes, high-level of understanding and system reconfigurability. A high-level of understanding of the scene and the observed activity and behaviour helps to minimise the role of human personnel to just responding to proper alarms raised by the system, or to setting questions to the system for retrieving specific events. The ability of the system to be automatically reconfigured does not only mean that its installation will be much easier ("plug 'n' play"), but also that it will be able to adapt to any environmental changes and work efficiently for extended periods of time under varying conditions.

This thesis investigates the application of unsupervised learning methods to single and multiple camera surveillance systems. More specifically, it aims to provide surveillance systems with a semantic scene model and proposes analysis of the activity based on the semantic model. Unsupervised learning is used to automate the whole procedure and minimise the human effort.

The proposed approach is inspired by the way that human operators develop and use their vision system. Initially, a scene may be completely unknown for new security guards. Gradually, they identify interesting scene elements (using their cognitive vision system that has been developed since their birth) as well as which activities and behaviours are typical and which are not. Activity within the scene is usually understood with respect to the context of the scene.

In the proposed framework, a surveillance system builds up its own "cognitive" model of the scene, by just observing the scene. Then, it is able to analyse the activity, based on the derived model. The cognitive model consists of semantic nouns like "entry zone", "exit zone", "path", "route", "junction" and "stop zone". These semantic nouns refer to specific regions of the scene that are related to specific activity events described by verbs like "enter", "exit", "move along", "go through", "stop". The subjects of the verbs are the targets, usually pedestrians or vehicles that the surveillance system aims to capture their activity.

The work presented in this thesis can provide multiple benefits to surveillance systems: The installation and maintenance of a surveillance system may be much easier, thanks to the automatic methods that are provided. High-level extracted information can be fed back to low level tasks and enhance the overall performance of the system. A means of automatic sophisticated activity analysis is provided that can be used to alert operators in their own human language. Finally, operators can set queries to the surveillance system that can be handled by a contextual database.

1.1 Overview of the Thesis – Aims and objectives

The principal aim of this thesis is to provide surveillance systems with an automatically derived, semantic, activity-based model of the scene that is observed by single or multiple surveillance cameras. Semantic elements of the model are entry/exit zones, paths, junctions, routes and stop zones. Figure 1.1 visualises some of these features in a specific scene. In addition to the semantic character of the model, it is required to successfully describe the variance of the usage of the scene features.

The proposed models have two main required characteristics: spatial and probabilistic. A spatial description of the scene is considered essential, because the human interpretation of the semantic scene features is closely related to their spatial extent. Additionally, a probabilistic description is able to capture the variance of the usage of these semantics features and the related uncertainty.

The semantic description is useful because it allows better interaction of human operators with the surveillance system. For example, applications like video annotation, where the surveillance system "interprets" the observed activity to the human language,

and contextual databases, where the system "understands" and processes queries expressed in the human language, are both based on semantic descriptors.

The probabilistic description of the model is desirable for many reasons. A probabilistic representation allows the application of "soft" logic, compared to the "hard" logic of a Boolean representation. Soft logic can deal with uncertainties and allows adaptation of the model. Additionally, it allows the scene model to be used as the basis of a probabilistic analysis of the activity and in applications like long-term prediction, atypical activity detection and tracking enhancement.

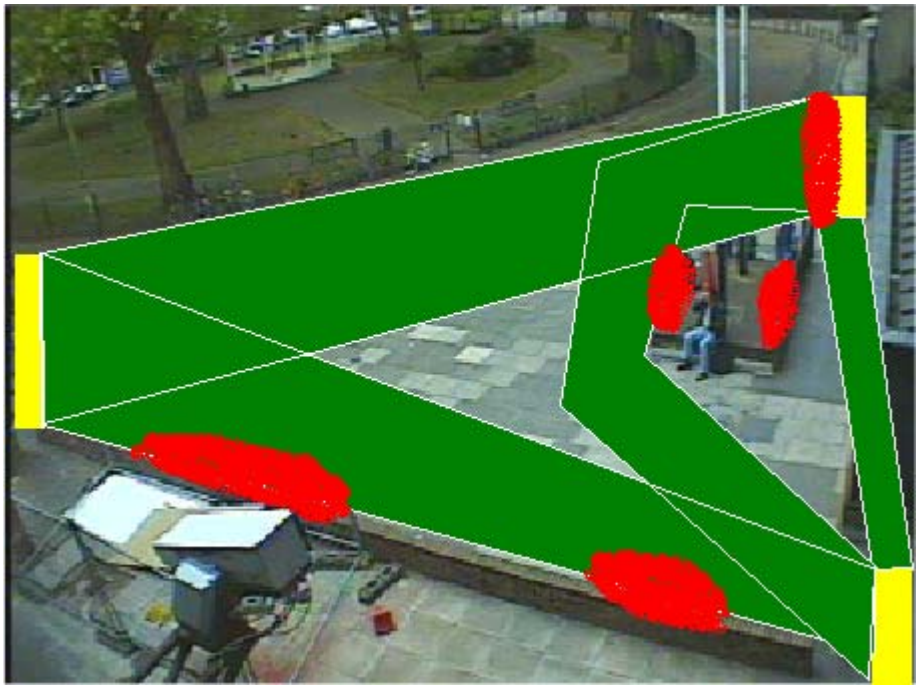


Figure 1.1: A semantic, manually derived, description of the observed scene. Yellow areas correspond to the entry and exit areas of the scene, green areas to commonly used paths and red areas to areas where pedestrians normally stop for a while.

Although scene models can be defined manually (and this is the case for a large variety of surveillance systems), this thesis suggests that they could be learnt automatically from observations, with minimum human intervention. Such an approach not only minimises the human effort to install a surveillance system, but also allows it to detect and adaptively respond to possible changes of the scene environment.

Learning scene models from observations is actually a reverse engineering technique. The observed target activity is directly affected by the structure of the scene, therefore observations of the activity can uncover this structure. More specifically, the

input of all the methods that are presented here is a set of trajectories, derived from a motion tracking system.

A variety of models and learning methods are used to represent and derive the individual types of scene elements. The core idea of the learning methods presented in this thesis is to exploit the large number of observations that can be derived by the surveillance system and group similar data to automatically derived activity-related models. Well-known techniques like Expectation Maximisation (EM) are used for learning Gaussian Mixture Models (GMMs) and accumulative statistics in image maps are integrated in the presented methods. In addition, a novel route model and an appropriate learning algorithm are introduced. A Hidden Markov Model (HMM) is superimposed on the scene model and used for activity analysis. Finally, a cross-correlation technique is used to reveal the topology of the cameras and the activities of a multiple camera surveillance system.

Chapter 2 presents useful background information for the context of this thesis. Main modules of the surveillance systems, like motion detection, motion tracking and calibration are discussed. Also, it summarises learning methods used in Computer Vision and discusses their suitability in surveillance applications.

The semantic scene model is introduced in chapter 3. Models and learning methods for scene elements like entry/exit zones, stop zones, motion noise sources and occlusion areas are also proposed in chapter 3. An EM-based algorithm is used that aims to deal with noisy data.

Chapter 4 discusses models and learning methods for paths, junctions and routes. The route model is introduced and details are given for a proposed algorithm that allows learning of route models from trajectories.

In chapter 5, the scene model is used to analyse the activity. The chapter demonstrates the concept of how probabilistic networks such as HMM may be overlaid on the scene model.

Chapter 6 illustrates how the presented methods are used in a multiple camera surveillance system. Results are given for the City University Experimental Surveillance (CUES) system. In addition, a cross-correlation algorithm is introduced that “bridges the

gaps" of a multi-camera surveillance system and allows automatic integration of the learnt models in a multiple camera activity network (MCAN).

Finally, conclusions and suggestions for further work are presented in chapter 7.

Chapter 2

Background

This chapter provides background information that is useful for the rest of the thesis. More specifically, it summarises how tracking data can be derived by a surveillance system, which is used as input to the algorithms proposed in this thesis. The chapter also discusses the role and the application of the learning processes in computer vision in general and in surveillance systems in particular.

An overview of the architecture of visual surveillance systems is given in §2.1. Basic tasks of automatic surveillance systems are described in §2.2 (motion detection), §2.3 (motion tracking) and §2.4 (camera calibration). §2.5 discusses the term “activity” as it possesses a key role in the context of this thesis. The importance and the application of learning in computer vision is discussed in §2.6 and special discussion is made for learning in surveillance applications, in §2.6.1.

2.1 Visual Surveillance

Traditional visual surveillance systems, which firstly appeared in the ‘60s, consist of a set of CCTV cameras, connected to display monitors and possibly to recording devices. Later, in the ‘90s, computer units were added that allowed the deployment of computer vision modules. The current trend is to integrate processing power and computer algorithms in “smart” cameras.

In Figure 2.1, a software-level description of the City University Experimental Surveillance (CUES) system is given. Although other systems may have a different architecture, in general, all visual surveillance systems have modules like motion detection and motion tracking.

The focus of this thesis will be the “Learning module”. Because its input data consists of tracks, an overview of the procedure of extracting tracks from video is

considered essential. In the next few sections, overviews of motion detection, motion tracking and camera calibration methods are presented. Their description is kept general, as the presented learning methods are assumed independent of the track extraction method. However, brief details are given of the motion detection and motion tracking methods that were used for extracting the results that are illustrated in this thesis.

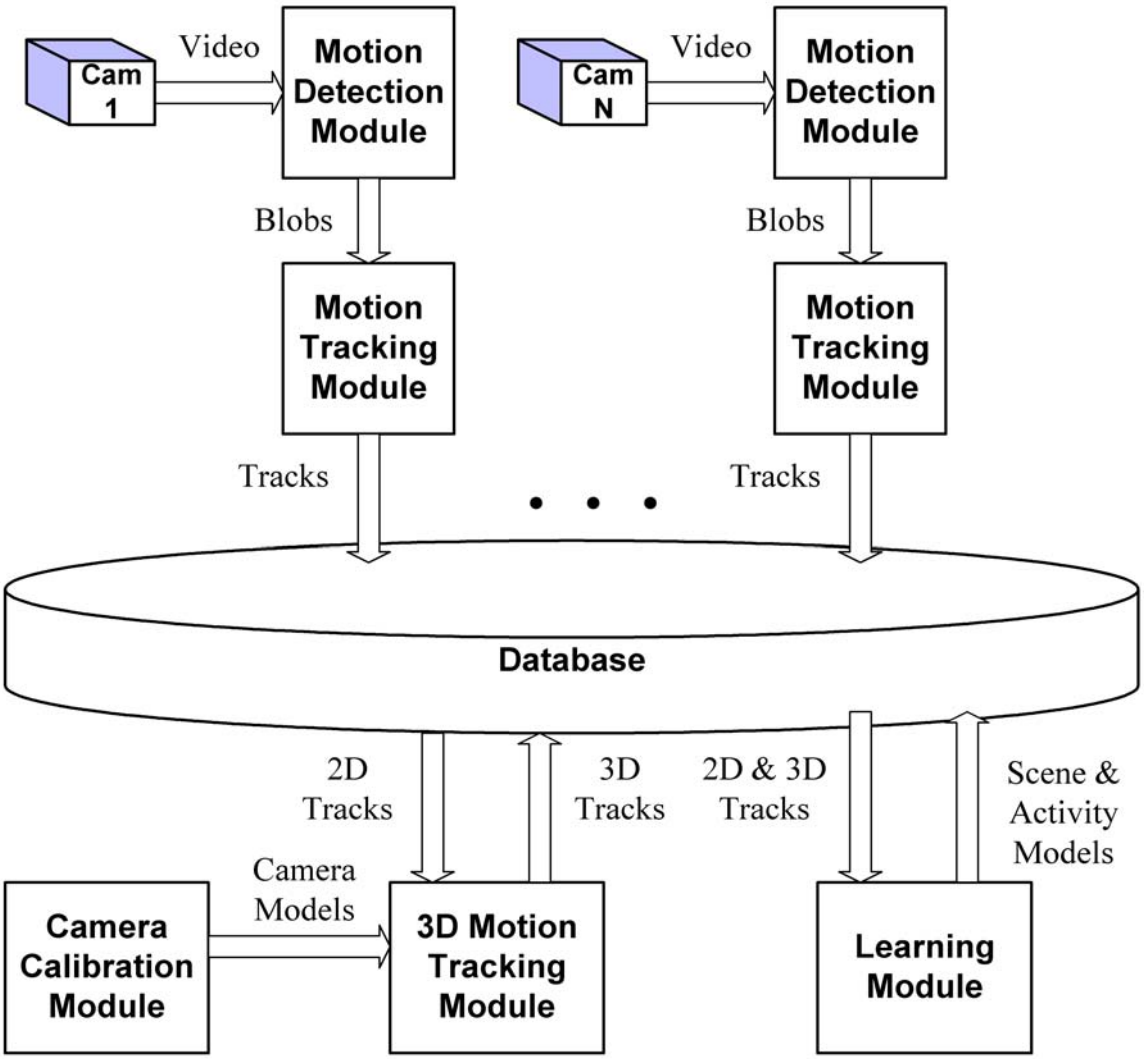


Figure 2.1: Software Architecture of the City University Experimental Visual Surveillance System.

2.2 Motion Detection

Motion detection algorithms aim to identify the regions of the image where object motion is present. All motion detection methods attempt to exploit the fact that observed motion causes changes of pixels, over time and space. However, they have to deal with

the aperture problem, where pixel values are unchanged by motion, or noise that causes change of pixel values independently of the motion.

Optical flow [42] was one of the first motion detection and estimation methods. If a video sequence is assumed as a 3D intensity signal (one temporal and two spatial dimensions), then motion can be detected using the optical flow constraint equation, which is a differential equation of the video signal.

Although optical flow is quite general, the solution of the differential equation is computationally very expensive and inapplicable in real time systems. A simplified version of the optical flow is frame differencing [50], which actually considers the variation of the video signal only along the temporal dimension, in two consecutive frames. Frame differencing is considerably faster than optical flow, however it is sensitive to noise and to the aperture problem.

Background subtraction, which is actually an evolution of the frame differencing method, uses a background image as reference to identify foreground moving objects in the video sequence. The background image is continuously updated to adapt to the environment changes.

Background subtraction is described by the flowchart of Figure 2.2. Each new frame of the video sequence is subtracted from the background image. At the same time, the new frame (Figure 2.3a) is used to update the existing background image (Figure 2.3b). A threshold is applied [84][85] to the difference image between the current frame and the background image, and a binary image is produced (Figure 2.3c) which indicates the areas of change. Finally, connectivity algorithms are applied on the binary image and localise “blobs” (Binary Large OBjects), that provide cues for detecting moving objects (Figure 2.3d).

Common problems in the application of the background subtraction methods are:

i. Change of the illumination conditions of the scene can result in difference of the pixel values and false motion detection. Illumination changes can be sudden or gradual, global or local.
ii. Temporarily stationary objects can be confused with the background.
iii. Shadows of moving objects are detected as part of the moving objects.
iv. Reflections, computer screens, trees and curtains seem as moving objects, because

of their apparent motion.

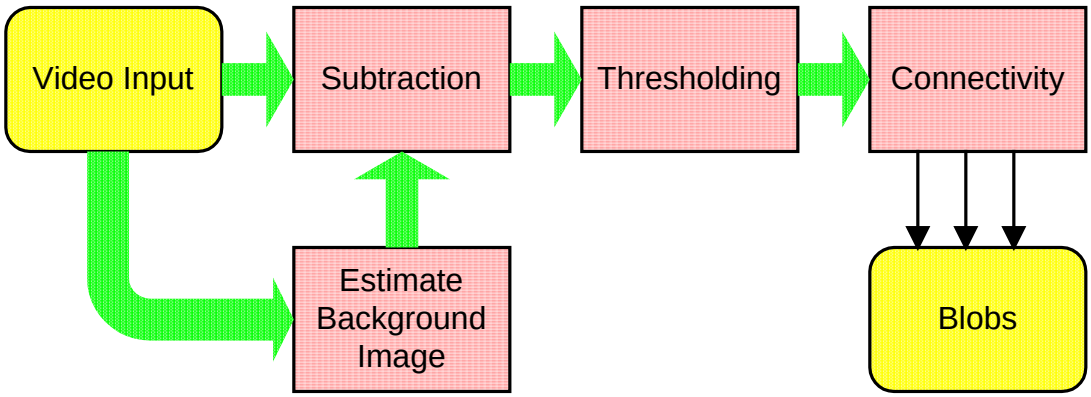


Figure 2.2: Flowchart of motion detection algorithms based on background subtraction.

Figure 2.3: (a) Original frame, (b) background image,(c) The thresholded difference image, (d) Detected objects (blobs).

Alternative to simple background subtraction methods are the pixel classification methods (see Figure 2.4), which are generally more sophisticated and promising. The Gaussian mixture model method, proposed by Stauffer and Grimson [8] classifies each pixel to pixel-based Gaussian models for the foreground and the background. A version of this algorithm in the chromaticity colour space, proposed by Xu and Ellis [105] allows

the background model to adapt to illumination changes very fast. The Wallflower algorithm, proposed by Toyama et al [97], uses pixel-level, region level and frame-level background models.

Pixel classification methods still suffer from the same problems as the background subtraction algorithms do. However, they provide better results, in general, because they model the noise process.

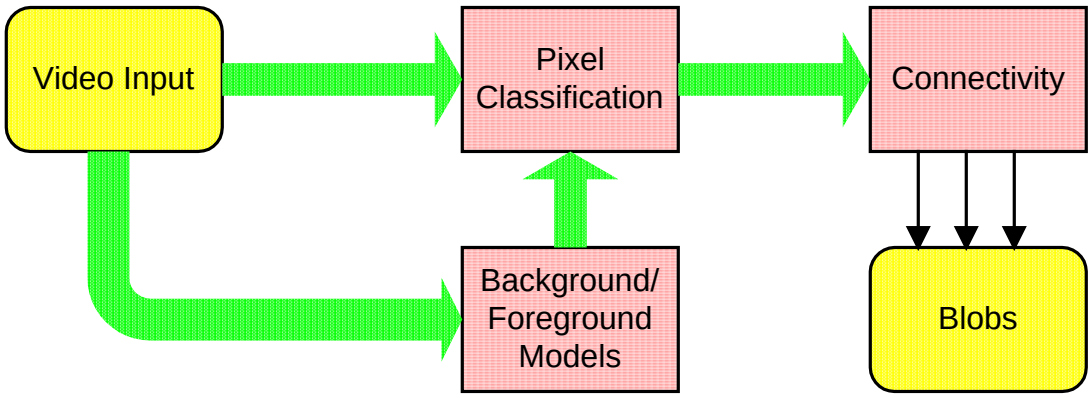


Figure 2.4: Flowchart of motion detection algorithms based on pixel classification.

2.3 Motion Tracking

Motion tracking aims to extract the motion history of targets from video sequences. A simple representation of the motion history is a sequence of spatial locations in time that is visualised by a trajectory (Figure 2.5). Although motion detection algorithms provide the cues for detecting moving objects in the scene, they do not track moving objects. Motion tracking algorithms are used to correspond detected objects in consecutive frames and ideally provide one track for each observed target.

Object correspondence can be based on various appearance and dynamic characteristics of a target, such as position, velocity, size, shape and colour. Tracking data is generally expressed using the 2D coordinates of the image plane. However, it can be converted to the 3D scene coordinates using a ground plane model and/or camera models and/or multiple views of the scene.

When the observed scene contains only one target, the motion tracking problem seems to be easy. However, a motion tracking algorithm must be able to identify the target among possible falsely detected blobs. In addition, it has to cope with static

occlusions, which occur when the target moves behind a stationary object and cannot be viewed by the camera.

Figure 2.5: A set of trajectories derived by a motion tracking algorithm. Each trajectory visualises the sequence of the locations of an individual target.

In a real surveillance system, simultaneous tracking of multiple objects in cluttered environment is required. This case is much more complicated, because the interaction of the targets can cause ambiguities about the number and the identities of the targets. For example, a target may be occluded by another, merged in a group of targets or split from the group. All these interactions are generally described as dynamic occlusions.

Each target is characterised by a set of attributes that allows the target to be distinguished from others. These attributes are usually related to the position, the velocity and the appearance of the target. The appearance of the target has been described by its size, shape descriptors (height-to-weight ratio, Point Distribution Models (PDMs) [7]), and colour descriptors [18].

Many strategies have been proposed to correspond blobs at consecutive frames. Because of the fact that the values of the target attributes may be changed over time, predictive schemes are used to assist the correspondence task. Some examples of predictive schemes that have been used are linear prediction, the Kalman filter [56] and the particle filter [48] [49].

The results that will be presented in this thesis are based on trajectory data that has been derived using the CUES system. Motion detection is performed by adaptive

Gaussian mixture models for background in intensity and chromaticity space [27] [105] [106]. Motion tracking on the image plane establishes correspondences among detected blobs in consecutive frames using position, velocity, size and colour information. A special Kalman filter was developed to cope with the static and dynamic occlusion problem [107]. Image plane tracks from different cameras can be combined using the homography alignment method and camera calibration models and the ground plane constraints are used to localise the combined trajectories on a common world coordinate system [9] [10].

For each tracked object, the derived trajectory data contains information about its position, velocity, bounding box, and average colour, for all frames from the time the object is initially detected, until it exits the scene. A status field indicates whether the object is initialised, matched successfully or only predicted, or finally terminated at specific frames. Additionally, each frame is time-stamped using a synchronised clock.

The methods that will be presented in this thesis assume that an object trajectory is described as $\{x_i, y_i, u_i, v_i, t_i, s_i\}$, where i indicates the frame, $\{x_i, y_i\}$ is the position of the object centroid, $\{u_i, v_i\}$, is the velocity of the centroid, t_i is the frame timestamp and s_i is the status of the object which can be assigned to one of the following values: {New, Matched, Predicted, Terminated}. Position and velocity can be expressed in either image plane coordinates or in ground plane coordinates. In the latter case, instead of the object centroid, its projection on the ground plane is considered.

2.3.1 Motion Noise

Unfortunately, motion detection and tracking in a cluttered environment can experience many problems due to a variety of reasons: illumination changes (local-global, slow-fast), static occlusions, stationary targets absorbed in the background, “semi-stationary” motion that may result in either falsely detected blobs, or un-detected objects. Additionally, the presence of multiple targets may cause errors to the blob matching process. We can identify different types of system noise that are manifested as incomplete trajectories, false trajectories and trajectories corresponding to apparent motion (e.g., associated with moving vegetation, curtains, computer screens, reflections on windows and other surfaces, background motion) that are of no direct interest. Many

techniques have been developed to improve the reliability of motion detection and tracking. However, because of the requirement that a surveillance system has to operate continuously for extended periods of time, under a variety of conditions, it seems that there is no guarantee for perfect tracking.

Two types of noise are recognised in the trajectory datasets that have been used for the experimental work described in the chapters 3-6.

i. Tracking failure noise: Tracking failure noise is due to the failure of the motion-tracking algorithm to track a target successfully for its whole activity in the scene. It may appear in the form of false trajectories (trajectories where the motion history of more than one targets have been mixed), or split trajectories (trajectories that represent only a portion of the motion history of a target).

ii. Semi-stationary motion noise: If sources of semi-stationary motion noise are present (such as trees, curtains, window reflections), then a high level of apparent activity is detected in the vicinity of the source of the semi-stationary motion noise. The characteristic of this apparent activity is that it is restricted in the area of and around its source. Therefore, all the points of a semi-stationary trajectory are within the same local region.

In chapter 3, an automatic method of identifying these types of noise is introduced. The benefit of the method is that it allows automatic cleaning of the data from the noise.

2.4 Camera Calibration

2.4.1 Single Camera Calibration

A camera provides a projection of the real 3D scene on the 2D plane of the image. This geometric transformation is expressed by the following mathematical equation:

$$\lambda \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.1}$$

where x, y are the 2D coordinates on the image plane, X, Y, Z are the 3D coordinates of the real scene, λ is an arbitrary scale factor and **P** is the projection matrix of the camera

[32]

The projection matrix represents the geometrical model of the camera. The camera model can also be described by the intrinsic and extrinsic camera parameters. The intrinsic parameters are related to the internal structure of the camera (focal length, aspect ratio, image centre coordinates and radial distortion), while the extrinsic parameters are related to the position and the orientation of the camera with respect to a defined 3D world coordinate system of the scene.

If the majority of the observed motion is restricted on a plane, the world coordinate system can be defined so that the motion plane is described by the equation Z=const. (usually Z=0) and is named ground plane. The assumption that all the motion is restricted on a plane is named ground plane constraint (GPC) [96]. The GPC is widely used in Visual Surveillance, because it simplifies the representation of the spatial data from 3D to 2D.

Camera calibration is the task of the estimation of the camera model, i.e. the estimation of the projection matrix **P** or equivalently the estimation of the intrinsic and extrinsic parameters of the camera. Traditionally, stationary camera calibration is achieved by using a calibration pattern and the 8-point algorithm [98].

In addition to the geometrical camera calibration, colour camera calibration can be defined [2]. Colour camera calibration estimates the colour response of the camera, using standardized colour surfaces, like standard grey cards or the Macbeth Colour Checker. Colour camera calibration is considered useful, because it allows homogenous perception of colours from different cameras, as it is explained in §2.4.2.

2.4.2 Multiple Camera Calibration

A multiple camera system has to be calibrated in three different senses. In addition to the geometric and the colour sense that have already been introduced in the single camera calibration, network camera calibration is also required.

The views of a pair of cameras may be overlapped, adjacent or distant. Generally, information from different cameras is possible when they have overlapped or adjacent views. Therefore a multi-camera surveillance system should be aware of the relationships between the different camera views and this logical structure can be represented by an

NxN binary matrix, where N is the total number of the cameras and the entries of the matrix are {1} for overlapped or adjacent views and {0} for distant views. In this thesis, the procedure of estimating the required logical structure of the network of cameras is defined as network camera calibration. In chapter 5, an automatic method for network camera calibration is introduced.

When two cameras have overlapped field of views, corresponding geometric calibration is also required. The geometric calibration of a pair of two cameras is defined as the estimation of the intrinsic parameters of each camera and the estimation of the relative position and orientation of the two cameras (the extrinsic parameters of the cameras). The geometric calibration is performed using again a calibration pattern that can be viewed by both cameras at the same time.

For the CUES system, all camera models were estimated with respect to a world coordinate system, where the plane Z=0 represents the ground plane. Landmarks that were clearly visible in views were used for camera calibration, using the Tsai algorithm [98].

Alternatively, to associate cameras with substantial overlapped field of views, homography calibration may be used. In this case, a homography transformation, defined by a 3x3 homogenous matrix, associates the projective coordinates of the two cameras.

The same object can generate different apparent colours in the sensors of different cameras because of possible different camera colour responses and the variation of illumination conditions (in the case that the cameras are placed in locations where the spectral distributions of light are different). The different apparent colours of the same object mean that the colour component of the identity tag of the object may not be reliable information that can be passed from one camera to the other, except in the case when the ratio of the colour responses of the cameras is estimated. Colour calibration estimates the relationship of the responses of the two cameras, using similar methods and techniques to the single camera colour calibration.

2.5 Activity

Activity:
1 a) the condition of being active or moving about. b) the exertion of energy; vigorous action.
2(often in plural) a particular occupation or pursuit (outdoor activities).

In the surveillance research domain, the terms activity and behaviour are used for similar and some times confusing meanings. Because this thesis investigates activity modelling and analysis, it is considered essential to define the term “activity”, as used in the context of this thesis, and to distinguish it from the term “behaviour”.

The activity of a target, within an observed scene, is defined as the complete motion history of the target that can be described by the sequence of target positions over time (one position per frame). Therefore, the activity of a target is closely related to its trajectory. The centroid of the detected blob will be used to indicate the position of the target in image-plane coordinates, because it is considered more stable to disturbances than for example the highest or the lowest point of the blob. In the case that ground plane coordinates are used, the position of the target is indicated by the projection of the object centroid on the ground plane.

The behaviour of a target is a more complex term that takes account not only of the history of the target’s centroid positions but also of the particular motion of the target’s subparts (articulated motion). Therefore, intuitive descriptions of activity can be verbal phrases such as “goes from area A to area B”, “is stopped in area C”, “moves fast”, “meets”, while similar descriptions for behaviour can include phrases such as “walks”, “runs”, “jumps”, “picks up”, “sits”, leans”, “fights” etc.

The set of activities of all the targets defines the activity in the scene. However, describing the activity in the scene by all the observed trajectories is not a compact representation. This thesis proposes scene activity-based models and suggests appropriate learning methods.

The activity in the scene is closely related to the structure of the scene. This dependency is obvious in highly restricted environments, such as in roads, where the

vehicle motion must be consistent with any traffic restrictions. In pedestrian environments, direct restrictions may also exist or pathways can be considered as indirect restrictions that influence the scene activity. But even in environments where neither direct nor indirect restrictions are present, the general structure of the scene, consisting of entrances, exits and areas of interest, can influence the activity.

Activity can be described not only in image-based coordinates but also in ground plane coordinates. To convert image-based coordinates to ground plane coordinates, an average target height of 170cm is taken into account for pedestrian environments and consequently the centroid of the target is assumed to be 85cm above the ground plane. A similar assumption can be made for vehicle environments. Although this is a rough approximation, it seems to be acceptable in most of the cases. The approximation may fail when the optical axis of the camera is almost parallel to the ground, (in the extreme case that it is parallel, no ground plane coordinates can be derived at all), or when the targets' heights vary significantly from the proposed average value (for example, in vehicle environments, double decker London buses are much higher than the rest of the vehicles). In the extreme case that the camera optical axis is perpendicular to the ground plane, image plane coordinates can be directly converted to ground plane coordinates, without any assumption about the target height.

2.6 Learning in Computer Vision

Computer vision is the research area that aims to endow computers and machines with a visual perception capability similar to the human visual system. Although researchers have put significant effort in the area of computer vision during the last 20-30 years, computer vision is far from its goal. The effectiveness of the human vision system is based not only on the complex and unknown way that the human brain perceives the images, but also on humans' ability to learn and adapt to their environments. E.g., the visual system of a newly born baby is not fully developed. Eventually the visual system of the infant evolves and the infant learns to perceive its environment and keep an abstract image of it [47]. Similarly, if computer vision systems can learn models of their environment, their performance may be improved and their functionality be extended.

Many learning techniques have been developed and applied in the area of Computer Vision. Learning is performed through a training dataset and the outcome is usually a model that "explains" the training dataset and aims to explain any similar dataset. Actually, learning methods are closely related to the models that are used to represent the data.

Learning methods are categorised into supervised, unsupervised and reinforcement methods. Supervised learning methods require manual labelling of the training dataset, while unsupervised methods attempt to extract labelling information from the dataset itself. Reinforcement learning methods are different in the way that they are applied to active systems (e.g., robots) and learning is performed through the actions of the system and the perceived feedback from the environment.

Methods like the K-means algorithm, the hierarchical clustering algorithm and the Fuzzy K-Means algorithm assume that data can be represented by clusters of points. The K-means algorithm [64] and its fuzzy version [8][26] assume a constant number K of clusters and attempt to identify the K centres of the clusters, iteratively. Solution is guaranteed in a finite number of steps. However, because the K-means algorithm is based on a Euclidean distance measure, it actually fails to identify underlying models of the data.

Hierarchical clustering methods produce a hierarchical tree of clusters, where the number of clusters at consecutive levels differs by one. The hierarchical tree can be learnt either top-down, starting with a single cluster and using a splitting strategy, or bottom-up, starting with each dataset sample as a cluster and using a merging strategy. A classic example of the hierarchical method is the hierarchical agglomerative clustering methods [54] that use a set of distance criteria. However, results depend heavily on the specific type of distance criterion that is used. They also suffer from an inability to identify the underlying models of the data for the same reasons as the K-means algorithm.

Expectation-Maximisation (EM) [25] models the clusters using probability density function (pdf) and searches for the pdf parameters that maximise the likelihood of the dataset. A common use of EM is to assume that clusters are modelled by Gaussian distributions, in which case the set of clusters is represented by a Gaussian Mixture

Model (GMM). EM benefits from its strong theoretical base and it can handle cases where models have overlapped distributions.

K-means, Fuzzy K-means and EM assume that the model order (number of the clusters) is known. Unfortunately, this is not always the case and model order is desired to be set automatically. Many criteria have been proposed that define cost functions of the model order and allow its automatic estimation. Most popular is the Minimum Description Length (MDL) criterion [83], inspired by information theory.

Support Vector Machines (SVM) [20] are widely used in high dimensionality problems, where the samples of the training dataset may be fewer than the dimensions of the data. The SVM method is actually a supervised method that attempts to learn an optimal separating hyperplane in order to classify new unseen examples. However, because the SVM algorithm complexity depends on the amount of the training data, they are inappropriate for learning from large datasets.

Probabilistic network modelling is based on a set of states and conditional probabilities that are used to represent state transitions. Markovian chains and Hidden Markov Models (HMM) [80] are well-known special cases of probabilistic networks and they are used to model the progress of events over time. There is no known analytical method that allows optimal learning of HMM from observations. Instead, iterative algorithms like the Baum-Welch method [5] (an implementation of the Expectation-Maximisation [25]) and gradient methods [60] are used. However, these iterative algorithms may be very slow, especially when the number of the states is large.

Neural Networks (NN) are inspired by a model of the human brain that consists of neurons organised in layers. A neuron's task is to process information, normally by applying some transformation to the input variable in order to produce an output. Whilst NN may produce acceptable results, it is often not clear how and when their performance is reliable. NN learning can be performed by either supervised (back-propagation algorithm) or unsupervised (hebbian learning, competitive learning) methods [94]. However, in comparison with other methods, they tend to be rather expensive computationally and not always appropriate for real-time online learning.

Genetic learning is another biologically inspired method that attempts to simulate the evolution process that is observed in nature. In genetic learning, a large number of

possible solutions is assumed. Solutions are evaluated and the best ones are combined to produce a new "generation" of solutions that tend to be better. Because genetic learning is based on the mechanisms of Darwinian evolution, it is a type of stochastic directed search. However, it may be appropriate for investigating solutions in large-scale problems, with little or no background information [102].

2.6.1 Learning in Surveillance Systems

Visual surveillance systems are required to operate over extended periods, under varying conditions and in different environments. Their performance depends on the configuration of a set of parameters that affect the effectiveness of the various modules. The current practice is to manually set these parameters when the system is installed. The disadvantage of the manual installation and configuration is not only that it requires human effort (which in the case multi-camera systems can be significant), but also that it does not guarantee the optimal performance of the system, over extended periods and under varying conditions.

A surveillance system can benefit significantly if it can learn properties related to its environment and adapt to changes. Such capability not only allows an effortless installation of a surveillance system ("plug 'n' play"), but also promises some stability of the performance, under varying conditions.

The advantage of surveillance systems over other computer vision systems is that they can easily acquire a large number of observations, which can be used for learning and adaptation of the system parameters. The ideal visual surveillance system should be similar to a child that observes the scene from a window, learns the scene and understands its changes.

A list of characteristics related to the observed scene that can potentially be learnt are:

– Properties related to the motion detection and motion tracking methods. A well-known example is the estimation of a background model, used for motion detection and the adaptive GMMs [89] is a widely used method. Also, models for moving targets can be learnt and used for assisting the motion tracking module.

For example, [6] and [7] discuss learning silhouette models of pedestrians, while [36] discusses learning of 3D models for vehicles.

– The camera models (network, geometric, colour) that are traditionally obtained using calibration patterns. For example, in [95], single camera calibration models are learnt using the gravity constraint, and in [82], the scene ground plane is learnt for single cameras. In [3], multiple camera calibration is attempted using blob information and in [9], [59] and [93] spatial-temporal calibration of multiple cameras is learnt from observations and the homography matrix of pairs of cameras with overlapped views is estimated.

– Features of the 3D scene, like occluding objects, paths, junctions, entry/exit areas. For example, in [88], a depth map of the scene is constructed, in [33], [34] and [35] the geometry of scene paths is learnt by accumulation of blob objects and in [92], entry/exit zones are estimated.

– Activity and/or behaviour models. For instance, in [52] activity and behaviour models are learnt using vector quantisation. A review of learning activity models is given in [22].

In this thesis, models and methods are presented for learning features of the 3D scene (chapter 3-4), activity models (chapter 5-6) and network camera models (chapter 6).

The selection of the appropriate models aimed to fulfil two requirements: The semantic description of the scene and modelling of uncertainty. Semantic description of the scene is fulfilled by the spatial representation of scene features, related to specific activities (entries, exits, stops, motions), while uncertainty is represented within a probabilistic framework.

The above requirements were determined according to possible applications in visual surveillance. For example, the semantic character of the model is essential for applications like video annotation and context-based databases, while the uncertainty modelling can benefit applications like motion prediction and atypical activity detection.

Algorithms were selected according to the special characteristics of the surveillance systems: the need to operate for extended periods under a wide variety of operating conditions and the ability to acquire a large number of observations. Therefore,

the learning algorithms must be able to operate automatically with ideally no human intervention and exploit the large number of observations. Additionally, algorithms must be fast enough to operate in real time and ideally allow online adaptation of the models to the new data.

Supervised learning requires manual labelling of the data, which can be a tedious work, especially for large datasets. Additionally real time adaptation of the system is not possible, due to the required human intervention.

Reinforcement learning requires external feedback from the environment. However, in the context of this thesis, cameras are considered fixed and passive and no other information is available, therefore reinforcement learning is inappropriate.

Therefore, only unsupervised methods seem to be appropriate for learning in visual surveillance. From the wide variety of unsupervised methods and models, computationally expensive methods like NN methods, genetic learning and the Baum-Welch method were avoided, because they are not suitable for real time, online learning

The next chapter illustrates how an EM-based algorithm is used to learn some semantic features. In chapter 4, a novel route model and an appropriate learning algorithm are introduced. In chapter 5, the structure of the semantic scene models, learnt in the previous chapters, is used as the basis to estimate a HMM model using accumulative statistics. Chapter 6 uses a cross-correlation technique to reveal the structure of activities within a network of cameras as well the topology of the network

Chapter 3

Learning Activity-Based Semantic Regions

3.1 Introduction

An activity-based semantic scene model is introduced in this chapter. The model consists of elements like entry/exit zones, paths, junctions, routes and stop zones. Modelling and learning issues of entry/exit zones and stop zones are detailed here, while the discussion on paths, junctions and routes is left until the next chapter. Additionally, scene features like semi-stationary motion noise sources and occlusion regions are discussed here. All the elements that are discussed in this chapter are learnt from single point datasets, while the next chapter discusses learning from sequences of points, i.e. trajectories.

The scene model and its semantics are introduced in §3.2. The scene model fulfils the requirement of the semantic description of the scene in spatial terms and the requirement for a probabilistic description of activity.

Modelling and learning of entry/exit zones, stop zones and semi-stationary motion noise sources are discussed in §3.3-§3.5. These scene elements are generally modelled by GMMs and are learnt by a novel EM-based algorithm. The EM-based algorithm aims to deal with the unavoidable presence of noisy data in the trajectory dataset.

Occlusion regions are discussed in §3.6. Occlusion regions are represented by a probabilistic image map, which is estimated using histograms of matched and non-matched target positions.

3.1.1 Previous work

Extracting semantics from images has attracted the attention of computer vision researchers since the early stages. For example, the VISIONS system by Hanson and Riseman [41] aimed to segment static images into semantically consistent regions, according to their pixel values.

Howarth and Buxton [43] identified the lack of an explicit semantic model and they proposed a spatial (polygon based) model of the scene, consisting of "leaf" and "composite" regions. Leaf regions aim to model the primitive elements of the scene and they are non-overlapped polygons. Composite regions aim to model semantic areas and they consist of leaf regions. However, those areas have to be defined manually and are not learnt from the video data [44]. Additionally, these polygon-based models lack a probabilistic representation.

Manually defined polygon-based scene models have been used in many surveillance applications. For example, the VIGILANT surveillance system [39] uses manually segmented scene areas to support contextual database queries. Cuppilard et al [24] suggests a scenario-based analysis of behaviour, whose spatial reasoning is based on a manually defined 3D scene model. Ellis and Xu [27] use a polygon-based scene model to define short-term and long-term occlusions and support the motion tracking process.

Fernyhough et al [33] and Fernyhough [34] used the spatial model presented in [43] as a basis for a learning algorithm that can automatically learn object paths by accumulating the traces of targets. The trace of a target consists of all the pixels that have been assigned to the specific target by the motion detection and motion tracking modules, in all the frames that the target was visible. Paths are formed by clustering similar traces, i.e. traces with sufficient overlap areas. The benefits of his method are that it is unsupervised and auto-initialised. However it is based on "hard" Boolean logic, therefore it cannot adapt to environmental changes and it may be sensitive to noise.

Grimson et al [40] describe activity in a 6-dimensional space (position, velocity, size, aspect ratio). They use two different methods to classify activities: a) Clustering of observations in the 6D space with Gaussian models by using the Numeric Iterative Hierarchical Cluster (NIHC) and b) accumulating co-occurrence statistics in a quantised 6D space. Using the former method, activity-related areas, e.g., queues, can be retrieved

by backprojecting Gaussian clusters with specific characteristics onto the spatial (x, y) plane. However, similar results can be derived working in lower dimensionality spaces, e.g., in the 2D (x, y) space. The latter method aims to cluster similar trajectories, which is a problem discussed in the next chapter.

In [92], Stauffer presented a method of learning "sources" and "sinks", which are actually entry/exit zones. He proposed a HMM where all sequences are two-state long. His algorithm deals with tracking failure noise using a "track stitching" method. The model-order is selected by a MDL-like heuristic that penalises high-order models, according to an exponentially decreasing function. The knowledge of entry/exit zones is used to correct and stitch tracks in a closed-loop manner. His method and the method that is introduced in this thesis have considerable similarities and further discussion is made at the end of the chapter.

Stauffer [88] accumulated target observations to reconstruct a rough depth map of the scene, using the line-of-sight constraint. A target detected by a camera is an indication that the space between the target and the camera is free of obstacles. The relative depth of a target, which in this case is a human, is estimated by the top position of the detected blob and assuming that all the tops of the heads always lie near a plane which is parallel to the floor. The accumulation of observations provides a model of the open space of the scene that is used by the targets and indirectly a rough approximation of the static objects in the scene. However, the construction of the map is based on hard Boolean logic, therefore it is sensitive to noise and it cannot be adapted to scene changes.

3.2 Semantic Scene Model

A semantic description of activity is usually performed in relation to semantic elements of the scene, e.g., "John entered the house from the front door, walked along the corridor, sat at the desk and then left from the back door". However, to allow automatic description of such activities (video annotation) from visual surveillance, semantics, like "front door", "corridor", "desk" must be defined. Ideally, these features must be automatically recognised by the visual surveillance system.

In the context of this thesis, semantics are defined in relation to the activity of targets. For instance, doors are characterised by the fact that are associated with entry/exit

events, a desk is an element of interest that targets may stop nearby and a corridor is a common path that targets use.

A semantic model of the scene is introduced, before any semantic activity areas can be learnt. The model must have both spatial and probabilistic elements in order to enable characterisation of the target activity in terms of spatial features of the scene. The model includes regions associated with a particular semantic interpretation, such as entry/exit zones, paths, routes, junctions and stop zones.

Figure 3.1 shows a simplified depicture of an outdoor scene, consisting of a number of interconnected pathways. The seat icon (I, J) represents regions where targets normally stop, while other labels are associated with entry/exit regions (A, C, E, G, H) and junctions (B, D, F) where targets moving down the pathways may change their routes. The segments between entry/exit zones or/and junctions (AB, CB, BD, DF, FG, **...**) represent paths, while routes are represented by the sequence of paths between an entry zone and an exit zone (ABDFH, CBDFG, EDFG, **...**).

Entry zones are regions where targets enter the scene. Similarly, exit zones are regions where targets leave the scene. An entry zone and an exit zone may be coincident, e.g., as in most pedestrian environments, or not e.g., in road traffic environments, where traffic is constrained by road traffic regulations. Typically, entry/exit zones are either scene-based, e.g., doors and gates, or view-based, e.g., located parts on the boundaries of the camera view. The distinction between scene-based and view-based entry/exit zones can be useful when information from multiple cameras is integrated.

Junctions are the areas where two or more pedestrian pathways or roads meet. At junctions, there is an uncertainty about the future motion of a target, as it can follow more than one path.

Whilst in common English, paths and routes have similar meanings, they are distinguished in the context of this thesis in the following manner. Paths are segments of either pedestrian pathways or roads in between entry/exit zones and junctions. A target route is the complete history of a target activity around the scene, from its entry zone to its exit zone, through various paths and junctions. More precisely, paths should be referred to as path segments, but the above given definitions are kept for the sake of simplicity.

Stop zones are defined as the regions where targets are stationary or almost stationary, for some minimum period of time. For example, pedestrians are stationary when they stop in order to sit, rest, queue, wait to access a resource, merely observe the scene or just wander around. Stop zones are included in the scene model for two reasons: Firstly, a stop zone is usually related to a physical scene feature, such as a bus stop, an ATM machine, a park seat, a shop window, a cashier, a computer, a printer, etc. Secondly, although the majority of research in video surveillance has focused on detecting and tracking motion, it is actually when targets stop and interact with each other or with these fixed elements of the scene that the system is more likely to be interested in them.

Two different presentations of the scene model are suggested: topographical and topological. The scene model is naturally represented by a topographical map (Figure 3.1) based on either an image plane(s) or a ground map representation. Ground map representations have an advantage over image-based representations not only because they can represent the physical features of the scene in proper proportion, but also because they allow integration of information from multiple cameras (see §6.4).

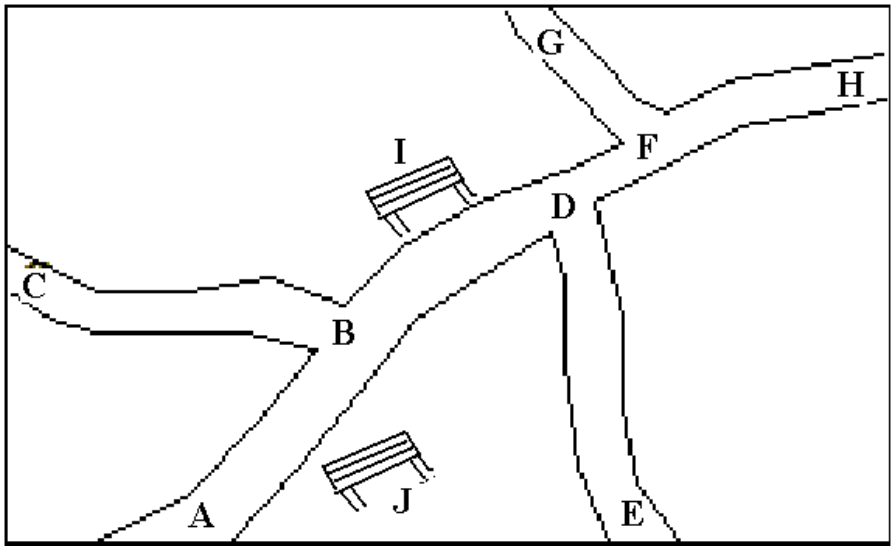


Figure 3.1: Topographical map of the scene.

The scene model can be represented by a topological map, i.e. an abstract network of nodes and connections, as shown in Figure 3.2 and Figure 3.3. In Figure 3.2 entry/exit zones and junctions consist the nodes of the network connected by paths. In Figure 3.3,

paths and stop zones are also represented as nodes of the network and the connections are actually region boundaries.

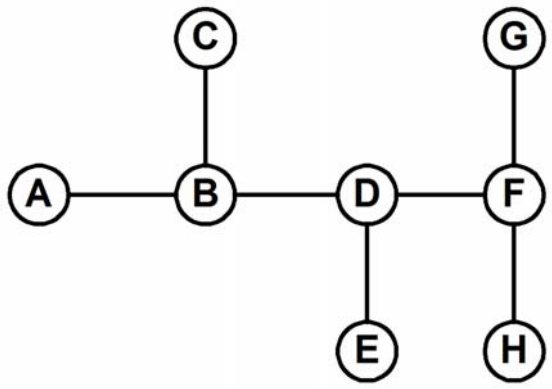


Figure 3.2: Topological map of the scene. Entry/exit zones and junctions are the nodes of the network.

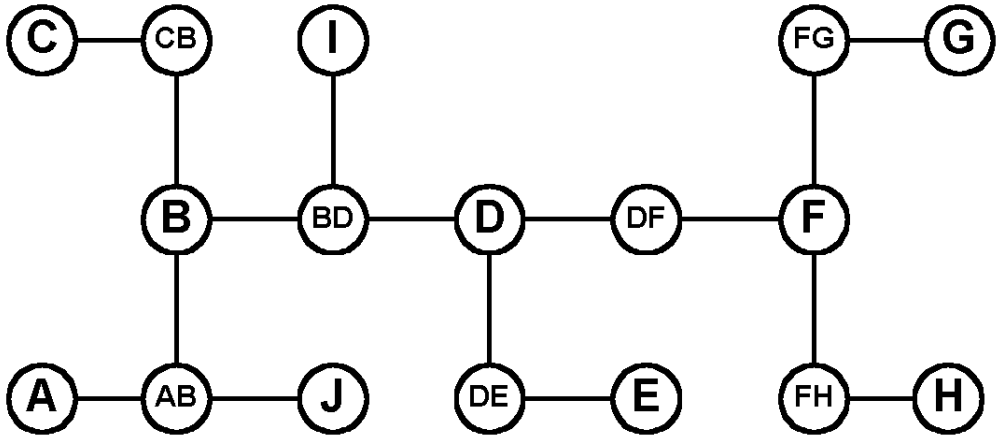


Figure 3.3: Extended topological map of the scene where stop zones and paths are included as nodes of the network.

The two different representations are used to illustrate two different aspects of the model. More specifically, the topographical map visualises the spatial characteristics of the scene elements, as interpreted by a human, whereas the topological map can be the basis of a Dynamic Probability Network (DPN) that can be used for a probabilistic analysis of the activity. In chapter 6, a HMM overlaid onto the scene model is used to support activity analysis.

In contrast to the features of the scene model that were described, other types of scene features are not directly related to the activity of the scene. However, because of the fact that they affect the way that the activity is viewed and interpreted by the surveillance system, their detection may support the motion tracking process. Such scene features are the semi-stationary motion noise regions and occluding regions.

Semi-stationary motion noise was described in §2.3.1. It may be related to real scene features, such as trees, windows, computer screens or curtains that are sources of apparent motion, constantly detected by the motion tracking algorithm, Semi-stationary motion is of no interest, therefore, a mechanism to discard it is required.

Targets may be occluded as they move around the scene. Occlusions can be either static, due to a stationary object such as a wall, or dynamic because of the presence of other targets. Occluding regions are the regions where targets seem to disappeared from the camera view. Occluding regions are usually related to real scene features, i.e. real objects that cause static occlusions. In the case of dynamic occlusions, they may be related to a densely occupied region in the scene.

3.2.1 Spatio-probabilistic modelling

The elements of the semantic scene model must have both a spatial and probabilistic character in order to label the target activity in terms of spatial features of the scene, using a probabilistic framework. Considering the spatial and probabilistic requirements, many options seem to be available in order to model the individual scene features:

i) Accumulative maps that explicitly represent the frequency of an event for each location. Accumulative maps were used in [73] and [74] to indicate the activity areas. They provide accurate, low level and rather expensive encoding of the spatial and the probabilistic information.

ii) Polygons (or degenerate polygons such as line segments and points) that can explicitly define the spatial area of each feature. Such an approach is proposed in [43] and is consistent with a manual estimation of the scene features. However, a pure polygon representation lacks probabilistic information. A solution to this can be the enrichment of polygon representations with overlaid probabilistic distributions in order to fulfil the modelling requirements.

iii) Specific types of probability density functions (pdfs). A common choice is the Gaussian (normal) function. Pdfs describe the probabilistic character of the feature with specific restrictions and accuracy, whereas the spatial area is defined

implicitly. Their representations are, in general, compact and there are many learning algorithms that can produce representations based on specific types of pdfs, especially on Gaussian distributions. Mixture models, e.g., GMMs (see Appendix I) can be used to represent features, where a single pdf is not sufficient.

iv) Hidden Markov Models (see Appendix III), which can be assumed as probabilistic networks of pdfs. They can be assumed as a dynamic extension of the mixture models, according to Roweis and Ghahramani [86]. For this reason, they are associated not with instantaneous events (e.g., entrance), but with continuous events (e.g., motion).

In this thesis, all the prescribed models are used. Accumulation maps are mainly used to illustrate the distribution of events on the image plane. GMMs are used to model regions related to instantaneous events (entrance, exit, stop) in this chapter, while route models, which can be assumed as polygons enriched with probabilistic distributions, are used to model regions related to motion in chapter 4. Finally, the application of HMM on route models is investigated in chapter 5.

3.3 Entry/exit zones

3.3.1 Modelling

Targets enter and exit the scene from either the borders of the image (view-based features), or doors and gates (scene-based features). The first and the last successfully tracked positions of an object (in other words the first and the last points of its trajectory) are used to indicate the entry and the exit event respectively. Entry/exit zones could be modelled by polygons for scene-features or line segments (degenerate polygons) for view-based features, according to the human interpretation. However, GMMs are used instead, because they represent compactly the variability and the uncertainty of the entry/exit observed events.

As seen in Figures 3.4-3.17, which depict entry/exit point datasets and fitted GMMs, GMMs can approximate successfully the shape of the majority of the entry/exit zones. For instance, a narrow Gaussian ellipse can closely approximate a line segment and a general Gaussian ellipse can approximate a convex-hull polygon. If the shape of the entry/exit zone is not a convex-hull, then a set of Gaussian models is used instead.

When the entry/exit speeds of the targets are high and the frame-rate of the system is relatively slow, the distribution of the entry/exit points is wider, because the position of first/last detected point of each target varies significantly around the real entry/exit zone. In these cases, the distribution of points can be well represented by a Gaussian model (GM), while line segments and polygon representations do not seem appropriate, because they cannot represent the variation of the entry/exit points.

A multi-step learning method is introduced in this thesis that is based on the Expectation-Maximization (EM) algorithm [25] (see Appendix II). An entry-point dataset consists of the start points of each trajectory, as derived by the motion-tracking algorithm. Similarly, the end points of a trajectory form the exit-point dataset.

One of the advantages of EM is that it can successfully distinguish overlapped distributions. Therefore, if noise is overlaid onto the signal, then it is possible to generate separate clusters for the signal and the noise, subject to different statistics. The two types of noise that are present in the trajectory data are also found in the entry/exit-point datasets and they have specific statistical characteristics.

In the entry-point/exit-point datasets, tracking failure noise (defined in §2.3.1) appears as false positive points, distributed over all the activity areas. The greater the activity in an area, the more false positive points are generated. This can be interpreted for the following reasons: a) If the probability of a tracking failure p_{tf} for every target is assumed constant and K/t is the rate of detected objects in the area per second, then the rate of false positive points in the area is $p_{tf} \cdot K/t$. b) Actually, the probability p_{tf} is not constant for every target; it is greater where the activity is higher, due to higher rate of dynamic occlusion.

In the entry-point/exit-point datasets, semi-stationary motion noise appears as a dense distribution of false positive points on and around the noise source, e.g., at the top-left corner of Figure 3.4.

Both types of noise are modelled, as they appear in the entry-point/exit-point datasets, using Gaussian distributions. The tracking failure noise is identified by wide Gaussian distributions over the activity areas, whereas the semi-stationary motion noise is usually interpreted by narrow Gaussian distributions at the noise sources. Because of the different nature of the two types of noise, different methods are used to filter them out.

3.3.2 Learning

A multi-step learning algorithm, based on the Expectation Maximisation (EM) algorithm (see Appendix II) has been developed. To justify the selection of the EM algorithm, its performance is compared with the performance of the K-means algorithm, at the end of this section.

The multi-step algorithm first discards the semi-stationary noise and then the tracking failure noise. The actual number of entry/exit zones in the scene is undefined. In order to overcome this problem, the number of entry/exit zones in the scene is overestimated and the number of the signal clusters is estimated by eliminating the noisy ones. The algorithm is described and results for each step are illustrated for an entry-point dataset taken from a pedestrian scene (Figure 3.4).

i. The EM algorithm with model order N is applied to the entry-point dataset **E** and a GMM is derived (Figure 3.5). To initialise the EM algorithm, the outcome of a K-means algorithm starting from N random points is used.

ii. If all the points of a trajectory belong to a single GM, as derived in the previous step, the trajectory is considered semi-stationary. A new cleaned entry-point dataset **E'** (Figure 3.6) is formed that does not contain the entry points of the semi-stationary trajectories.

iii. The EM algorithm with model order N' is applied to the clean entry-point data-set **E'** (Figure 3.7).

iv. Gaussian clusters are classified as either signal clusters or noise clusters, according to a density criterion (Figure 3.8). More specifically, if w_i is the prior probability of a cluster i and $\mathbf{\Sigma_i}$ is its covariance matrix, where i=1.. N', then a measure of the density d_i is given by:

$$d_i = \frac{w_i}{\pi \cdot \sqrt{|\mathbf{\Sigma}_i|}} \tag{3.1}$$

A threshold value T is defined by the clean entry-point dataset **E'**:

$$T = \frac{\alpha}{\pi \cdot \sqrt{|\mathbf{\Sigma}|}} \tag{3.2}$$

where α is a user-defined weight (typical values are between 1.5 and 2) and $\mathbf{\Sigma}$ is the covariance matrix of the dataset $\mathbf{E'}$.

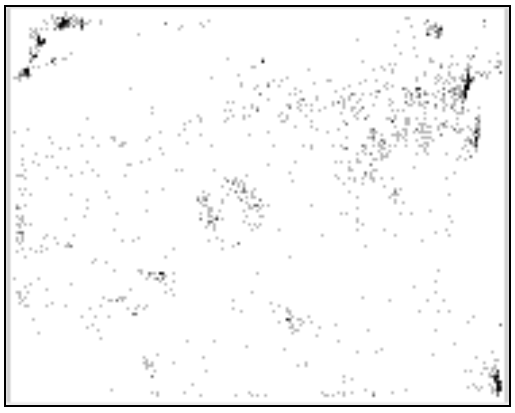


Figure 3.4: Entry-point dataset (4250 samples) with both types of noise present.

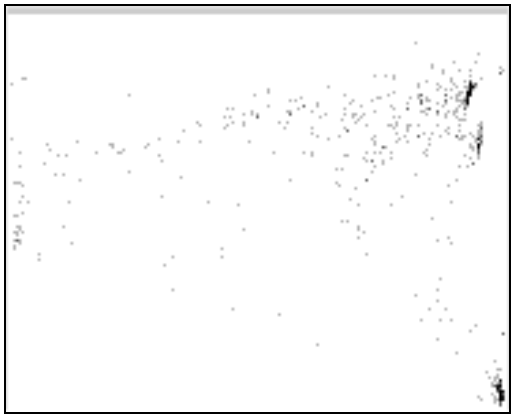


Figure 3.6: Clean entry-point dataset (1767 samples) after discarding stationary motion noise.

Figure 3.8: Three Entry zones derived by the multi-step algorithm.

Figure 3.5: GMM after first run of EM (N=6). The upper left cluster is caused by the semi-stationary movement of the tree branches.

Figure 3.7: GMM after second run of EM (N'=8).

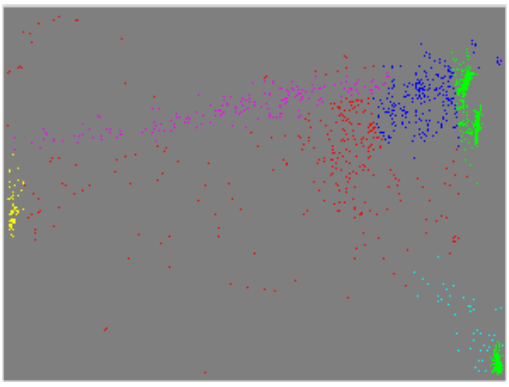


Figure 3.9: Classification of entry points to clusters (colours taken from Figure 3.7).

The clusters derived in steps (i) and (iii) are characterised according to their density. High-density Gaussian clusters correspond to either entry zones or semi-stationary motion noise, while low-density clusters correspond to tracking failure noise. The algorithm eliminates semi-stationary motion noise at step (ii) and tracking failure noise at step (iv).

The density criterion is justified by the Table 3.1, which lists the densities d_i of the clusters of Figure 3.7 and their ratios to the threshold T. The table also shows that the popularity of the clusters, determined by the prior probabilities w_i, cannot be used as a criterion to distinguish signals from noise clusters. For instance, although cluster 4 has very low popularity, it is dense enough to be considered a signal entry zone and indeed, it corresponds to a real entry zone, as it can be seen in Figure 3.8. Another example is cluster 3, the second most popular cluster. However, due to its wide distribution, it is correctly classified as a tracking failure noise area.

Cluster id	w_i (%)	d_i (10^{-6} pixels^{-2})	d_i/T	Signal?
1 (red)	10.2	2	0.03	No
2 (green)	13.2	1622	21.41	Yes
3 (blue)	19.0	61	0.80	No
4 (yellow)	4.3	134	1.77	Yes
5 (magenta)	11.7	23	0.31	No
6 (cyan)	3.3	13	0.17	No
7 (red)	11.2	24	0.32	No
8 (green)	27.0	417	5.51	Yes

Table 3.1: Signal clusters of the Figure 3.7 are separated from noise. Decision is based on threshold T=78.7x10^{-6}pixels^{-2}, as estimated using Eq.3.2.

Further results on entry/exit point datasets taken from a pedestrian scene and a traffic road scene are shown in Figures 3.10-3.17. The pedestrian scene is the part of Northampton Square that is in front of the main entrance of the university. Entry/exit zones are coincident, as in most of the pedestrian scenes. A person that knows the area can easily identify that the three entry/exit zones (see Figure 3.11 and Figure 3.13) correspond to the main entrance of City University, the way from and to Goswell Road (lower left) and the way from and to City University Student Union (bottom right). There is considerable tracking failure noise around the main entrance of the university and this is due to the fact that students normally gather at this area and therefore the rate of dynamic occlusions is higher.

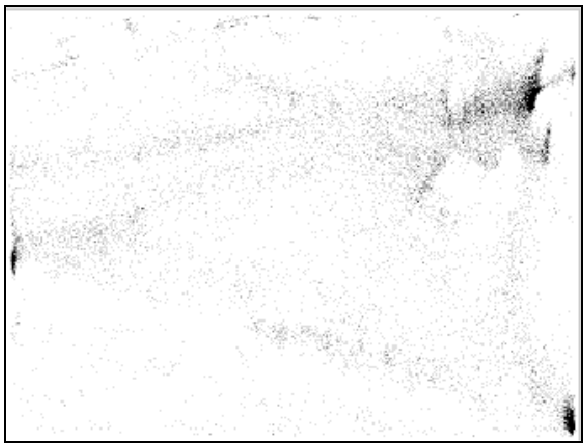


Figure 3.10: Entry-point dataset (13223 samples).

Figure 3.11: Three detected entry zones.

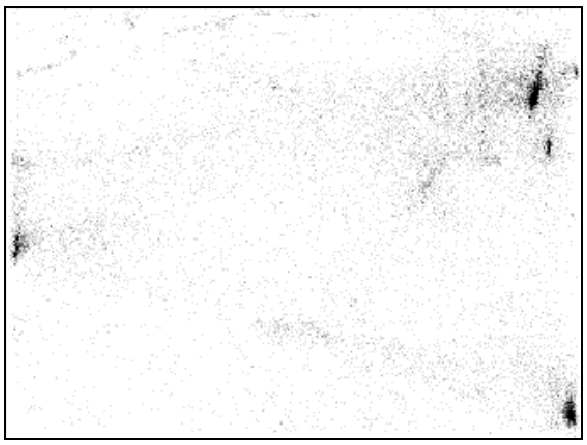


Figure 3.12: Exit-point dataset (13223 samples).

Figure 3.13: Three detected exit zones.

The road traffic environment is the part of Goswell Road that is next to the main university site. Two entry zones and two exit zones were identified (see Figure 3.15 and Figure 3.17) that are not coincident, because they correspond to two unidirectional road lanes. Although in this scene pedestrian traffic is normally present, the dataset was derived over a weekend, when pedestrian traffic was very light.

In Figures 3.14-3.17, the clusters at the left side of the image are less dense than the ones at the right side. The phenomenon can be explained by the higher image-based speed of the vehicles at the left side (closer to the camera), in combination with a low frame rate that results in a wider distribution of the first/last tracked positions for the vehicles. This difference of the distributions of the entry/exit zones is actually desirable,

as the cluster explicitly determines where the targets should be initialised/terminated, according to their entry/exit speed and the system frame rate.

Figure 3.14: Entry-point (12746 samples) dataset in a road traffic environment.

Figure 3.15: Two detected entry zones in a road traffic environment.

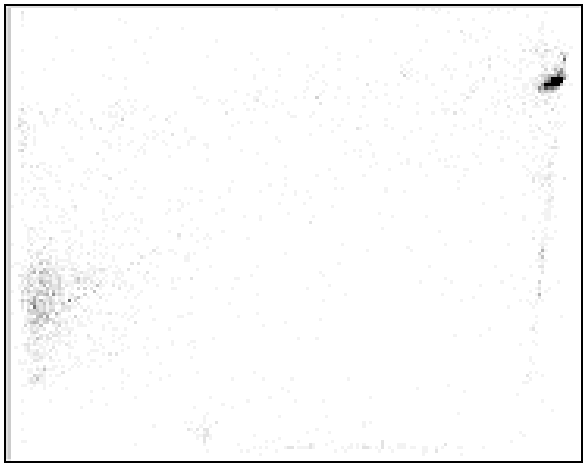


Figure 3.16: Exit-point dataset (12746 samples) in a road traffic environment.

Figure 3.17: Two detected exit zones in a traffic environment road

The selection of the threshold T in Eq.3.2 is justified as follows: $\pi \cdot \sqrt{|\mathbf{\Sigma}|}$ is a measure of the area over which dataset **E'** is distributed (see Appendix I). Therefore, $\pi \cdot \sqrt{|\mathbf{\Sigma}|} / N'$ is a measure of the area of an "average" cluster. If α/N' represents an acceptable prior probability for a signal "average" cluster, then T indicates the density of an "average" signal cluster.

A typical value of α was estimated using experimental results on the entry-point dataset of the Figure 3.10. The EM algorithm was applied for values of $N'=3,...,16$. For

each model order, the algorithm was applied 5 times with different randomly selected initialisation points.

For all 70 experiments, clusters were manually labelled as signal or noise, according to a combined interpretation of the dataset histogram and the real scene that the data came from. For each experiment, the signal cluster with the minimum density and the noise cluster with the maximum density were taken into account and a suggested threshold density was estimated as the mean average of the two densities. Results for all three densities are shown in Figure 3.18.

Figure 3.18, shows that signal and noise clusters are separable by a threshold density value, in most of the cases. The variation of the estimated threshold density indicates that it should not depend on the model order and this fact is reflected in Eq.3.2.

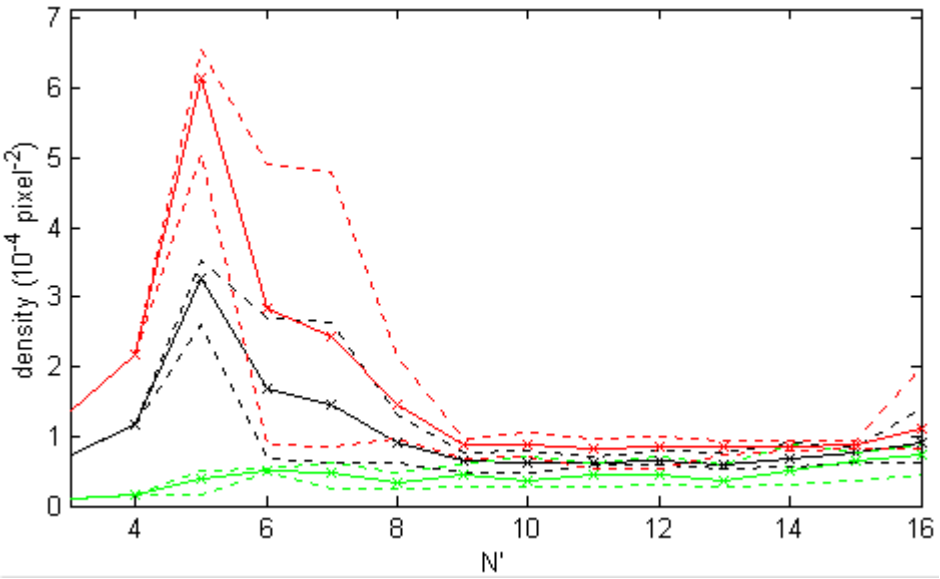


Figure 3.18: Diagram of the minimum density of signal clusters (red), the maximum density of noise clusters (green) and the estimated threshold value (black). Solid lines show the mean values, while dashed lines indicates the range of values around the averages.

To estimate an optimal threshold value for the given dataset, clusters were separated into signal and noise according to threshold densities T(α), for $\alpha \in [0.5, 2.5]$. Separation results were compared to the manual separation and the success rates of classifications are shown in Figure 3.19: The success rate is determined as the percentage of experiments for which automatic and manual separation coincides. The maximum success rate (85.71%) is reached for values $\alpha \in [1.7, 1.85]$. Detailed results for $\alpha=1.8$ are shown in Table 3.2.

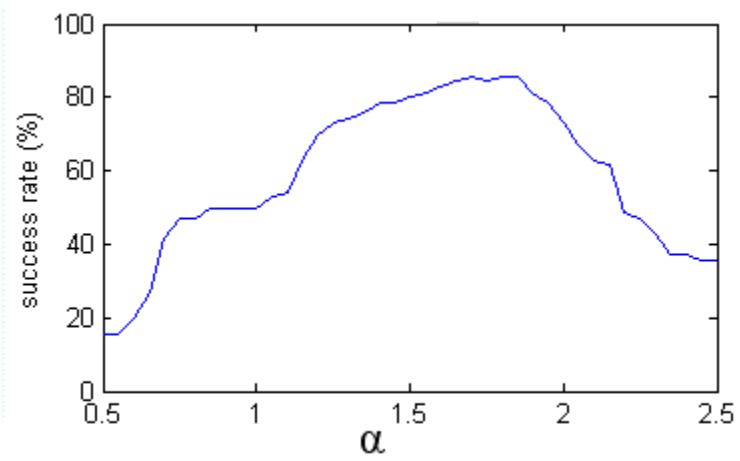


Figure 3.19: Success rate of signal-cluster separation for various values of α.

N'	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**Overall**
%	100	100	100	100	100	100	80	60	80	80	80	60	100	40	85.71

Table 3.2: Separation success rate for N'=3...16 and α=1.8.

The multi-step algorithm is relatively insensitive to the model order selection at steps (i) and (ii). *N* is manually selected to ensure that any possible motion source noise is not multi-modelled. Similarly, *N'* must be large enough to ensure that all the zones can be modelled, whilst not over-fitting the data. Therefore, if N_e is the number of the real entry zones, N_{sm} is the number of the semi-stationary motion sources and N_{tf} (usually 1-5) clusters are allowed to model the tracking failure noise, then $N=N_e+N_{sm}+N_{tf}$ and $N'=N_e+N_{tf}$. The values of N, N' are quite flexible thanks to the fact that in most cases, extra clusters tend to model tracking failure noise.

The relative insensitivity of the algorithm is supported by results that were extracted from the previous experiments, using the optimal α. Figure 3.20 illustrates results for N'=5, 7, 9, 11, 13, 15 and the values of N', N_e, and N_{tf} are shown in Table 3.3. If the results for N'=5 and N'=15 are compared, it can be seen that out of the 10 extra clusters, only 3 were allocated to signal clusters and the remaining 7 were allocated to noise clusters.

N'	**5**	**7**	**9**	**11**	**13**	**15**
N_e	3	4	4	4	5	6
N_{tf}	2	3	5	7	8	9

Table 3.3: Number of signal and noise clusters for the results of Figure 3.20.

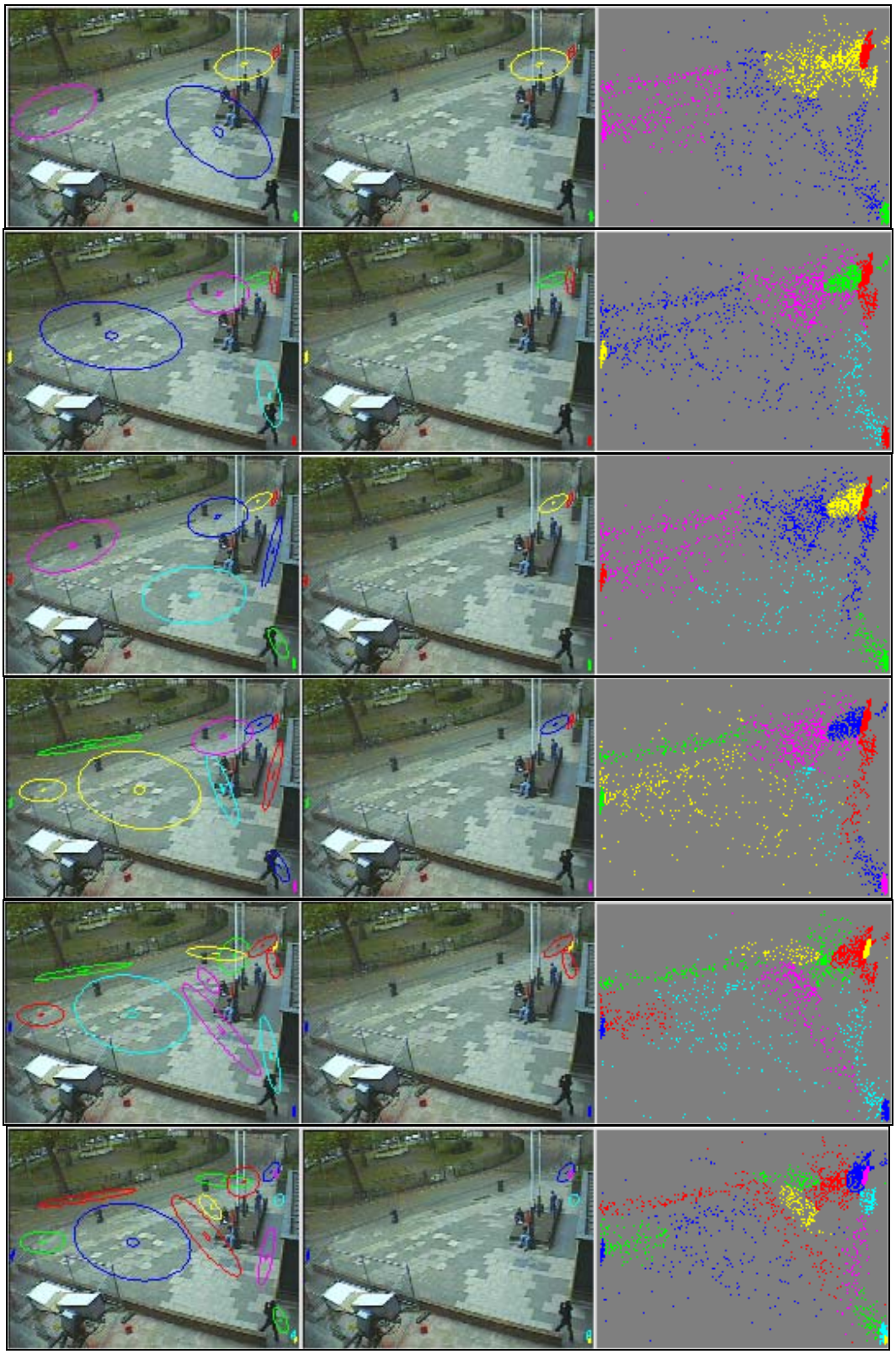


Figure 3.20: Results of the proposed method, on the dataset of the previous set for N'=5,7,9,11,13,15. First column shows all extracted clusters, second column shows only the signal clusters and third column shows classified points.

EM was selected as the basis of the proposed algorithm for three main reasons: Firstly, it can successfully localise signal clusters and therefore it can extract the spatial extent of the entry/exit zones. Secondly, it can learn overlapped distributions and therefore it can model any tracking failure noise that is overlapped with entry/exit zones. Finally, the separation of the signal clusters from noise clusters is possible, because of a detectable difference of their densities.

In order to illustrate those advantages, the EM algorithm was compared against a common clustering method: the K-means algorithm [64]. Entry/exit-point datasets with considerable amount of noise were used from two different scenes. Scene A refers to Northampton Square area and scene B refers to Goswell Road area. In all the cases, the model order was set to N=10 and the extracted clusters were manually labelled as signal or noise, according to the interpretation of the histogram dataset and the scene.

As can be seen in Figures 3.21-3.32, the K-means algorithm experiences problems accurately localising the signal clusters. In most of the cases, no separate clusters for noise are extracted. On the other hand, the EM algorithm's performance is clearly more satisfactory; it can successfully determine the position and the spatial extent of the signal clusters and separate clusters are derived for the noise.

To estimate the separability of the signal clusters from the noise ones, a separation criterion D_{ij} is defined:

$$D_{ij} = d_i - d_j \tag{3.3}$$

where i is the signal cluster with the minimum density and j the noise cluster with the maximum density. The higher the value of D_{ij}, the more separable the clusters. If $D_{ij}<0$, then the clusters are not linearly separable, i.e. there is no threshold density T to successfully separate signal clusters from noise clusters. As seen in Table 3.4, the separation criterion for the K-means clusters is always negative, while for the EM clusters is always positive. Therefore EM clusters are separable as opposed to the K-means clusters, which cannot be separated according to their densities. In addition, the EM clusters in all the experiments can be separated using the threshold density, as calculated by Eq.3.2.

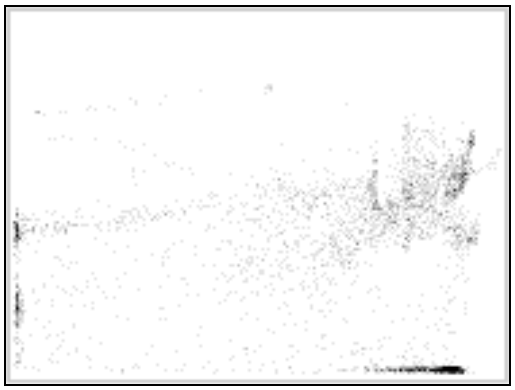


Figure 3.21: Histogram of 9424 entry points of scene A.

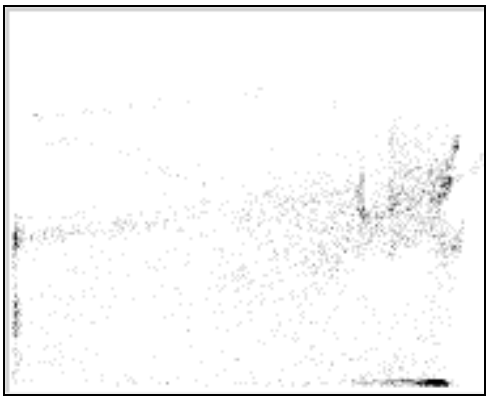


Figure 3.22: Histogram of 9424 exit points of scene A.

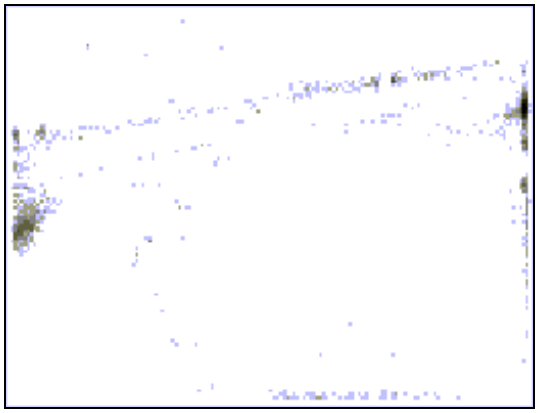


Figure 3.23: Histogram of 4643 entry points of scene B.

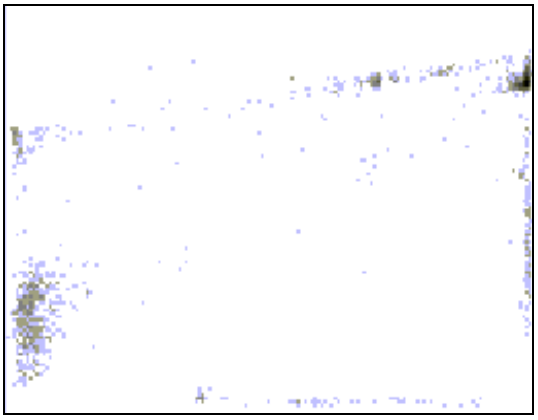


Figure 3.24: Histogram of 4643 exit points of scene B.

	Scene A		Scene B	
Method	**Entry zones**	**Exit zones**	**Entry zones**	**Exit zones**
K Means	$-54 \cdot 10^{-6}$ pix^{-2}	$-44 \cdot 10^{-6}$ pix^{-2}	$-6 \cdot 10^{-6}$ pix^{-2}	$-3 \cdot 10^{-6}$ pix^{-2}
EM	$+26 \cdot 10^{-6}$ pix^{-2}	$+46 \cdot 10^{-6}$ pix^{-2}	$+32 \cdot 10^{-6}$ pix^{-2}	$+33 \cdot 10^{-6}$ pix^{-2}

Table 3.4: Summary of the values of the separation threshold for different scenes, datasets and algorithms.

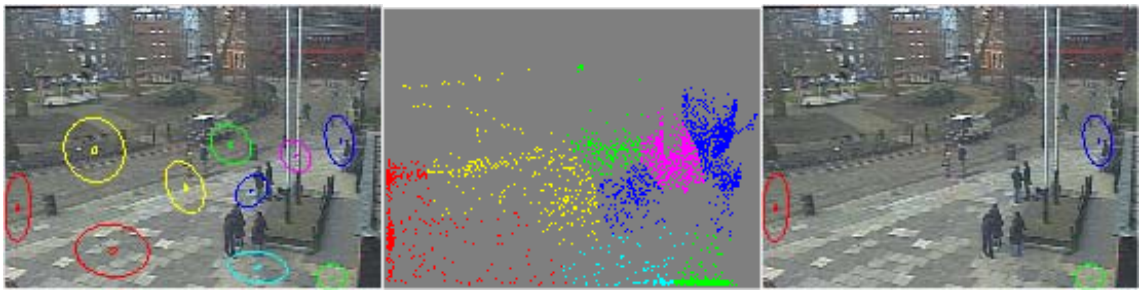


Figure 3.25: Results of K-means algorithm for the dataset of the entry points of the dataset. Clusters extracted (left), classified points (middle) and manually selected entry zones (right).

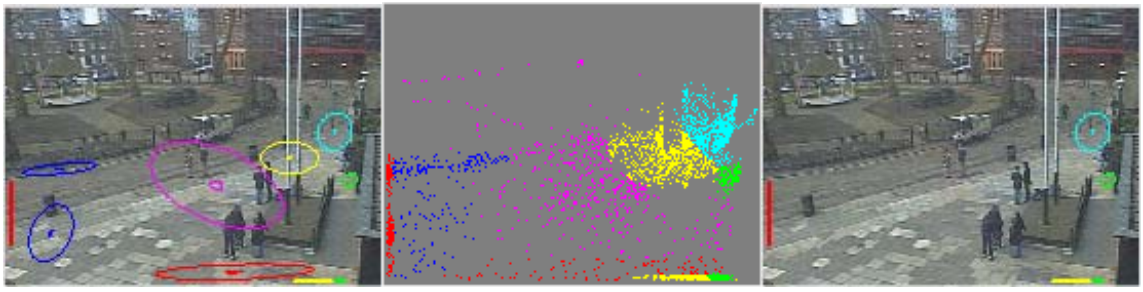


Figure 3.26: Results of EM algorithm for the dataset of the entry points of scene A. Clusters extracted (left), classified points (middle) and manually selected entry zones (right).

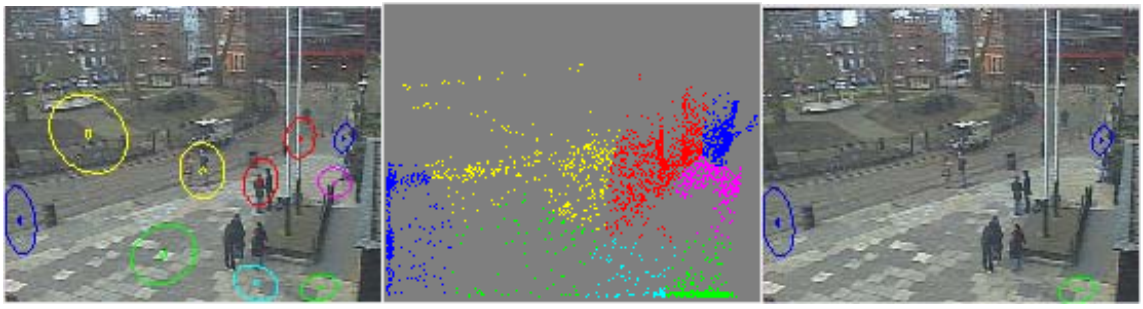


Figure 3.27: Results of K-means algorithm for the dataset of the exit points of scene A. Clusters extracted (left), classified points (middle) and manually selected exit zones (right).

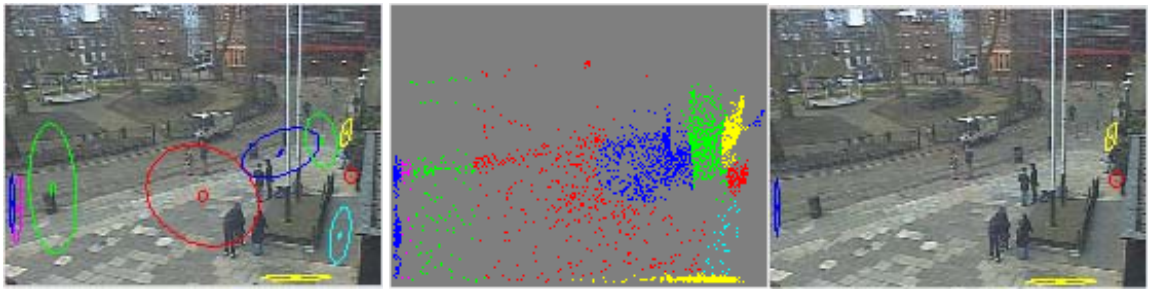


Figure 3.28:Results of EM algorithm for the dataset of the exit points of scene A. Clusters extracted (left), classified points (middle) and manually selected exit zones (right).

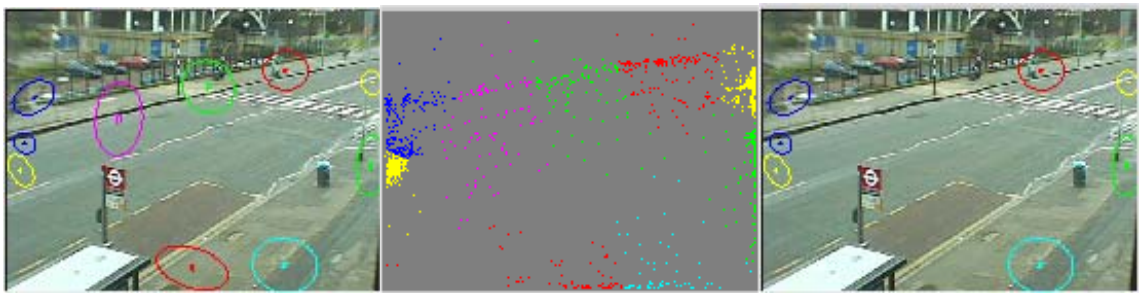


Figure 3.29: Results of K-means algorithm for the dataset of the entry points of scene B. Clusters extracted (left), classified points (middle) and manually selected entry zones (right).

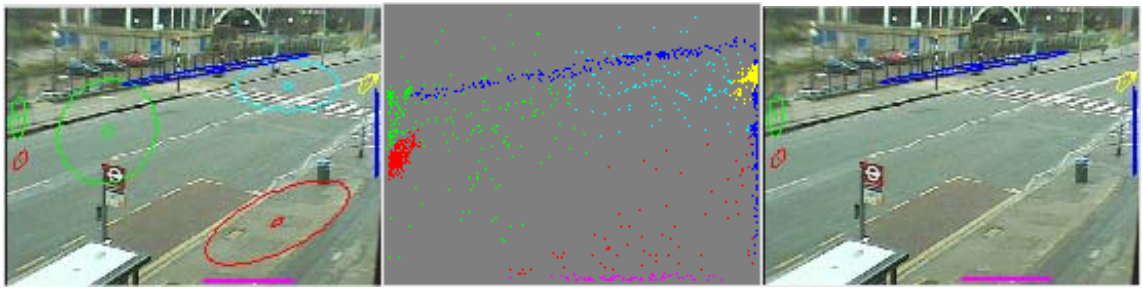


Figure 3.30: Results of EM algorithm for the dataset of the entry points of scene B. Clusters extracted (left), classified points (middle) and manually selected entry zones (right).

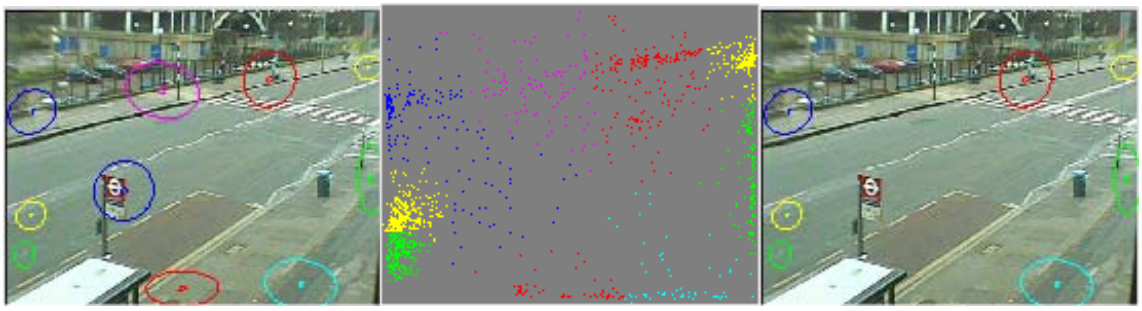


Figure 3.31: Results of EM algorithm for the dataset of the exit points of scene B. Clusters extracted (left), classified points (middle) and manually selected exit zones (right).

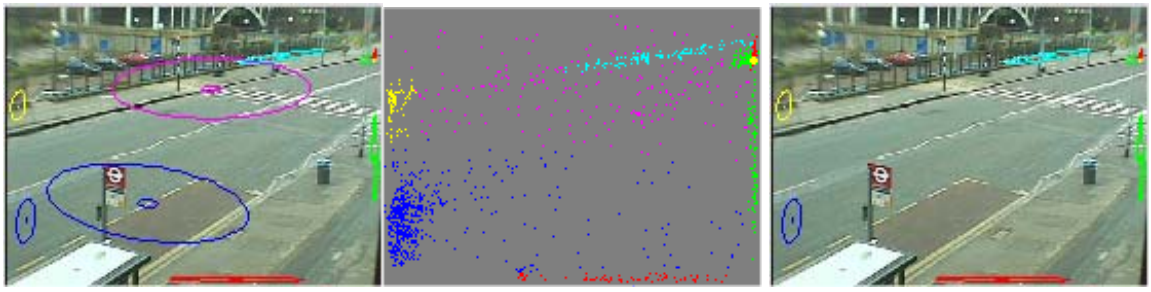


Figure 3.32: Results of K-means algorithm for the dataset of the exit points of scene B. Clusters extracted (left), classified points (middle) and manually selected exit zones (right).

Cluster id	w_i (%)	d_i (pix^{-2})	Signal
1 (green)	17.23	$109 \cdot 10^{-6}$	Yes
2 (magenta)	17.94	$84 \cdot 10^{-6}$	No
3 (blue)	25.50	$80 \cdot 10^{-6}$	Yes
4 (red)	12.02	$30 \cdot 10^{-6}$	Yes
5 (blue)	6.59	$30 \cdot 10^{-6}$	No
6 (green)	5.38	$15 \cdot 10^{-6}$	No
7 (yellow)	4.96	$11 \cdot 10^{-6}$	No
8 (cyan)	3.77	$9 \cdot 10^{-6}$	No
9 (yellow)	4.72	$5 \cdot 10^{-6}$	No
10 (red)	1.89	$2 \cdot 10^{-6}$	No

Table 3.5: K-means clustering results of entry points of scene A. D_{ij}=-54x10^{-6}pix^{-2}.

Cluster id	w_i (%)	d_i (pix^{-2})	Signal
1 (green)	9.59	$1640 \cdot 10^{-6}$	Yes
2 (yellow)	6.49	$308 \cdot 10^{-6}$	Yes
3 (red)	8.22	$190 \cdot 10^{-6}$	Yes
4 (green)	3.76	$88 \cdot 10^{-6}$	Yes
5 (cyan)	20.77	$70 \cdot 10^{-6}$	Yes
6 (yellow)	22.55	$54 \cdot 10^{-6}$	Yes
7 (blue)	4.76	$29 \cdot 10^{-6}$	No
8 (magenta)	18.92	$8 \cdot 10^{-6}$	No
9 (red)	2.80	$5 \cdot 10^{-6}$	No
10 (blue)	2.14	$4 \cdot 10^{-6}$	No

Table 3.6: EM clustering of entry points of scene A. D_{ij}=+26x10^{-6}pix^{-2}.

Cluster id	w_i (%)	d_i (pix^{-2})	Signal
1 (blue)	25.66	$243 \cdot 10^{-6}$	Yes
2 (magenta)	13.96	$67 \cdot 10^{-6}$	No
3 (red)	12.82	$49 \cdot 10^{-6}$	No
4 (green)	7.95	$39 \cdot 10^{-6}$	Yes
5 (red)	10.28	$31 \cdot 10^{-6}$	No
6 (blue)	10.55	$23 \cdot 10^{-6}$	Yes
7 (cyan)	5.72	$17 \cdot 10^{-6}$	No
8 (yellow)	6.70	$12 \cdot 10^{-6}$	No
9 (green)	53.79	$5 \cdot 10^{-6}$	No
10 (yellow)	2.58	$2 \cdot 10^{-6}$	No

Table 3.7: K-means clustering of exit points of scene A. D_{ij}=-44x10^{-6}pix^{-2}

Cluster id	w_i (%)	d_i (pix^{-2})	Signal
1 (yellow)	19.44	$344 \cdot 10^{-6}$	Yes
2 (yellow)	10.35	$184 \cdot 10^{-6}$	Yes
3 (blue)	6.49	$144 \cdot 10^{-6}$	Yes
4 (red)	4.12	$100 \cdot 10^{-6}$	Yes
5 (green)	17.34	$46 \cdot 10^{-6}$	No
6 (blue)	21.02	$26 \cdot 10^{-6}$	No
7 (magenta)	2.05	$18 \cdot 10^{-6}$	No
8 (red)	14.50	$6 \cdot 10^{-6}$	No
9 (cyan)	1.49	$6 \cdot 10^{-6}$	No
10 (green)	3.21	$2 \cdot 10^{-6}$	No

Table 3.8: EM clustering of exit points of scene A. D_{ij}=+54x10^{-6}pix^{-2}.

Cluster id	w_i (%)	d_i (pix^{-2})	Signal
1 (yellow)	31.34	$292 \cdot 10^{-6}$	Yes
2 (yellow)	13.74	$81 \cdot 10^{-6}$	Yes
3 (blue)	7.90	$76 \cdot 10^{-6}$	Yes
4 (red)	9.43	$24 \cdot 10^{-6}$	Yes
5 (blue)	6.83	$22 \cdot 10^{-6}$	Yes
6 (green)	8.59	$21 \cdot 10^{-6}$	Yes
7 (green)	6.66	$11 \cdot 10^{-6}$	No
8 (magenta)	7.28	$9 \cdot 10^{-6}$	No
9 (red)	4.03	$6 \cdot 10^{-6}$	No
10 (cyan)	4.20	$5 \cdot 10^{-6}$	Yes

Table 3.9: K-means clustering of entry points of scene B. D_{ij}=-6x10^{-6}pix^{-2}.

Cluster id	w_i (%)	d_i (pix^{-2})	Signal
1 (yellow)	19.02	$4049 \cdot 10^{-6}$	Yes
2 (red)	16.07	$356 \cdot 10^{-6}$	Yes
3 (blue)	11.52	$190 \cdot 10^{-6}$	Yes
4 (yellow)	4.35	$89 \cdot 10^{-6}$	Yes
5 (magenta)	4.25	$56 \cdot 10^{-6}$	Yes
6 (green)	7.42	$54 \cdot 10^{-6}$	Yes
7 (blue)	12.52	$50 \cdot 10^{-6}$	Yes
8 (cyan)	9.09	$8 \cdot 10^{-6}$	No
9 (green)	10.67	$5 \cdot 10^{-6}$	No
10 (red)	5.10	$2 \cdot 10^{-6}$	No

Table 3.10: EM clustering of entry points of scene B. D_{ij} =+42x10^{-6}pix^{-2}.

Cluster id	w_i (%)	d_i (pix^{-2})	Signal
1 (yellow)	26.86	$195 \cdot 10^{-6}$	Yes
2 (green)	13.89	$115 \cdot 10^{-6}$	Yes
3 (yellow)	15.25	$86 \cdot 10^{-6}$	Yes
4 (green)	7.88	$21 \cdot 10^{-6}$	Yes
5 (red)	12.23	$20 \cdot 10^{-6}$	Yes
6 (blue)	7.75	$17 \cdot 10^{-6}$	Yes
7 (red)	4.11	$8 \cdot 10^{-6}$	No
8 (magenta)	5.51	$6 \cdot 10^{-6}$	No
9 (cyan)	3.83	$5 \cdot 10^{-6}$	Yes
10 (blue)	2.67	$3 \cdot 10^{-6}$	No

Table 3.11: K-means clustering of exit points of scene B. D_{ij} =-3x10^{-6}pix^{-2}.

Cluster id	w_i (%)	d_i (pix^{-2})	Signal
1 (yellow)	10.96	$5369 \cdot 10^{-6}$	Yes
2 (red)	5.03	$800 \cdot 10^{-6}$	Yes
3 (green)	5.68	$223 \cdot 10^{-6}$	Yes
4 (blue)	26.60	$146 \cdot 10^{-6}$	Yes
5 (green)	8.00	$91 \cdot 10^{-6}$	Yes
6 (cyan)	8.91	$77 \cdot 10^{-6}$	Yes
7 (yellow)	4.77	$51 \cdot 10^{-6}$	Yes
8 (red)	5.58	$39 \cdot 10^{-6}$	Yes
9 (magenta)	16.18	$6 \cdot 10^{-6}$	No
10 (blue)	8.29	$2 \cdot 10^{-6}$	No

Table 3.12: EM clustering of exit points of scene B. D_{ij} =+33x10^{-6}pix^{-2}.

3.4 Stop zones

Stop zones are defined as regions where the targets are stationary or almost stationary. A variety of different areas can be characterised as stop zones, like areas where people rest, wait for the opportunity to continue their journey (e.g., at a pedestrian crossing or a road traffic junction), wait to access a particular resource (e.g., an automatic teller machine or at a bus stop), or merely observe the scene. Targets may also become stationary when they meet and interact with other targets, for example two people meeting in a park and sitting on a bench to chat or a vehicle waiting for a pedestrian to walk across a pedestrian crossing.

A stop event is detected when a target's speed becomes lower than a predefined threshold. A stop-event dataset is formed by checking the target trajectories for stop events. Speed is estimated in ground plane coordinates because the apparent speed on the image plane may be strongly affected by the perspective view of the camera. Therefore the stop-event dataset is formed using ground plane coordinates (see Figure 3.33).

As with the entry/exit zones, a GMM is used to model the spatio-probabilistic characteristics of the stop zones and EM is applied to learn them (Figure 3.35). However, it is required to model an additional property of the stop zones: the duration of the stop events. Data have shown (see Figure 3.37) that the pdf of the stop event duration t_{se} can be adequately approximated by an exponential function.

$$p(t_{se}) = A \cdot e^{-bt_{se}} \tag{2.4}$$

The parameter A is associated with the popularity of the stop zone, while the parameter b is associated with the probability of terminating the stop-event (i.e. the target starts moving). More specifically, the probability of terminating a stop-event within the next second is constant and equal to -ln(b). The zone popularity is already represented by the GMM, therefore the stop event duration of each stop zone can be cheaply modelled by only one additional parameter.

Table 3.13 summarises the detected stop zones in Figure 3.34. To distinguish signal from noise stop zones a density threshold T=12.492 10^{-3} m^{-2} is used. (T was estimated from Eq.3.2, with α=0.5). For the accepted stop zones (i.e. $d_i/T>1$), the decay parameter b was estimated by non-linear least-square optimisation (the matlab function *curvefit* was used). The stop event duration average $E(t_{se})$ is equal to 1/b, according to the definition of the exponential function.

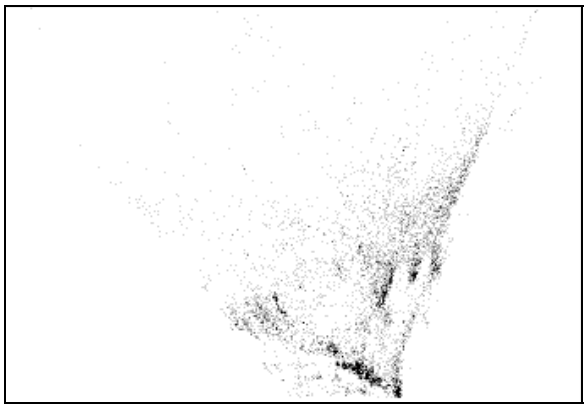


Figure 3.33: Stop events (9455) on the ground plane. Stop events were detected when targets' speed became lower than 0.25m/sec.

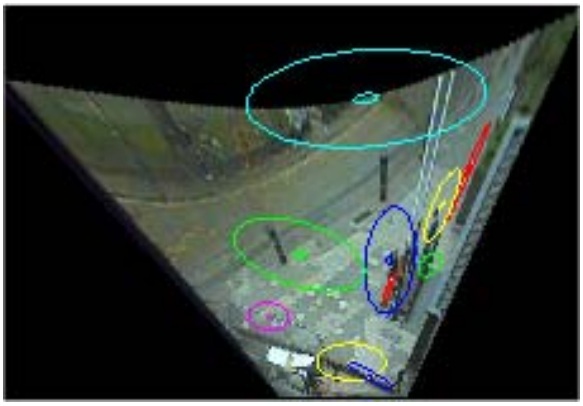


Figure 3.34: Gaussian clusters derived by EM algorithm with model order 10.

Figure 3.35: Five signal stop zones as derived by the EM algorithm.

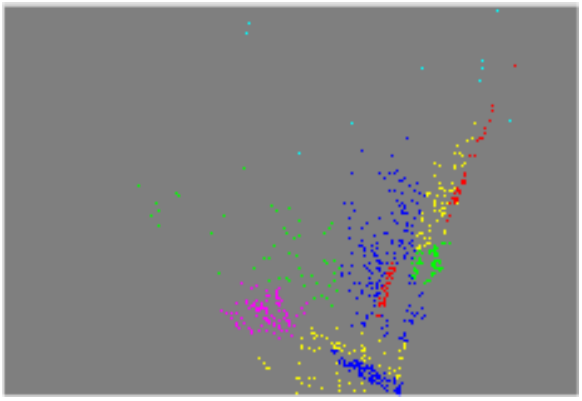


Figure 3.36: Classified stop events to stop zones.

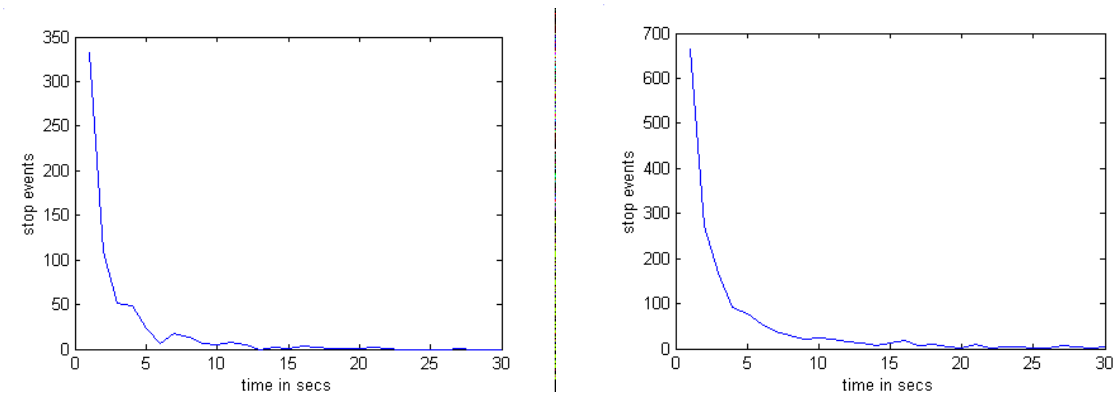


Figure 3.37: Stop event duration diagrams. The x-axis indicates the duration of a stop event and the y-axis the number of detected stop events with the specific duration. Results are given for the two most popular stop zones.

Cluster id	w_i (%)	d_i (10^{-3} m^{-2})	d_i /T	b (sec^{-1})	$E(t_{se})$ (sec)
1 (red)	4.94	56.18	4.50	0.35	2.88
2 (green)	5.17	17.07	1.37	0.36	2.74
3 (blue)	26.25	7.98	0.64	-	-
4 (yellow)	13.17	8.24	0.66	-	-
5 (magenta)	10.85	16.01	1.28	0.33	3.09
6 (cyan)	2.05	0.14	0.14	-	-
7 (red)	4.51	14.71	14.71	0.97	1.03
8 (green)	8.43	1.57	1.57	-	-
9 (blue)	15.27	61.86	61.86	0.21	4.71
10 (yellow)	9.37	10.28	10.28	-	-

Table 3.13: Stop zones derived by the EM algorithm.

Summarising, the GMM of the stop zones provides a means to localise these semantic features of the scene that are related to stop-events, while the stop-event duration function provides a probabilistic estimate of the time that an individual target may be stopped in the specific region.

3.5 Semi-stationary motion noise sources

Trees or curtains may be detected as moving objects, because of their possible slight and recurring motion, caused by blowing air. Similarly, apparent motion is detected

in reflective surfaces (e.g., windows, mirrors, static water surface) as a result of reflections of real moving objects. The falsely detected motion in all these cases is defined as semi-stationary motion, because of its characteristic to be almost stationary and restricted in a specific area. It is characterised as noise because not only it is of no interest, but also it obstructs the motion-tracking algorithm. An active background (e.g., a busy traffic road at the back of the scene) may also lead to detection of motion that is spatially restricted and of little interest, therefore, background motion may be considered as semi-stationary motion noise.

Knowledge of semi-stationary motion noise sources can be beneficial for visual surveillance systems, as semi-stationary trajectories can be localised and ignored. Semi-stationary motion noise sources were implicitly detected as a by-product of the entry/exit zones estimation. Their explicit estimation can be derived from the set of trajectories that are rejected as semi-stationary, according to sets of entry/exit zones. A trajectory is characterised as semi-stationary if and only if all its samples belong to the same entry/exit cluster.

The EM algorithm is applied to a dataset of points, constructed from the first points of the semi-stationary trajectories that were rejected from the dataset, as described in §3.3.2. Clusters are separated into “signal” and “noise”, according to their densities. Dense “signal” clusters represent semi-stationary motion noise sources, whereas low-density “noise” clusters usually represent tracking failure noise.

Results are presented from the same scene that was used for the entry/exit zones. The proposed method successfully detects a tree as a motion noise source (Figures 3.38-3.41).

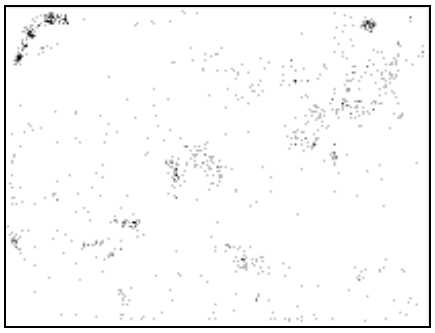


Figure 3.38: Histogram of the first points of 5099 semi-stationary trajectories.

Figure 3.39: Gaussian clusters estimated by the EM algorithm.

Figure 3.40: One semi-stationary motion noise detected which corresponds to the branches of a tree.

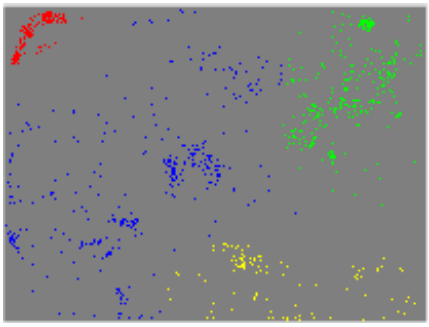


Figure 3.41: Classification of points.

3.6 Occlusion regions

As stated above, many tracking failures are due to occlusion. Occlusion is one of the major challenges for motion tracking algorithms, not only because it is a main source of problems, but also because dealing with occlusion is difficult due to the variety of occlusion types.

Occlusions are mainly classified as static and dynamic ones. Static occlusions are caused when a target goes behind a static object of the scene, so it cannot be viewed and detected by the system. Static occlusions can be further characterised as short-term or long-term [27], depending on the duration of the occlusion.

Dynamic occlusions are caused due to other targets of the scene, e.g., when a target moves behind another target. Some motion tracking algorithms not only fail to detect the occluded target, but they also fail to match the occluding target. This failure occurs because the two targets are seen as merged on the motion detection image,

therefore, instead of two separate blobs, only one blob is detected and its characteristics may not correspond to any of the two blobs.

Targets may also be assumed occluded because of lack of motion. If a target becomes apparently stationary, then it gradually becomes part of the background and the motion detection algorithm fails to detect it.

Furthermore, occlusions can be total or partial, depending on the visible proportion of the occluded target and the solidity of the occluding object (e.g., solid wall, non-solid fence or bush).

Faulty occlusions, i.e. apparent occlusions where the target is not really hidden by another object, are also possible. In addition to the target absorption in the background and the dynamic occlusion (for the occluding target), a target may appear as occluded (i.e. not detected) if its colour appearance is similar to the background. Also, if the target moves in front of an active semi-stationary motion noise source (e.g., an active background or a waving tree), then its blob may be merged with the noise source blob and its separation from the noise can be difficult.

One solution to the static occlusion problem is to model [27] and learn the location of occlusion regions. Some areas that are related to specific types of occlusions are already modelled. For instance, semi-stationary motion noise sources can be detected as shown in §3.5. Because dynamic occlusions or occlusions that occur due to similarity to background usually cause a considerable proportion of tracking failures, tracking failure noise clusters are a spatio-probabilistic indication of those occlusions. Finally, in stop zones, targets are absorbed in the background, due to lack of motion.

An occlusion event can implicitly be detected by the motion tracking algorithm. If a blob has been successfully detected previously but it fails to be matched at the current frame, then its position is predicted, according to the tracking filter. If the predicted position is within the scene, then the target is possibly occluded. However, no indication is given for the type of the occlusion (static, dynamic, etc).

A histogram $\mathbf{H_p}$ of occlusion events is given in Figure 3.42. A high rate of occlusion is evident in a heavily used area in front of the City University main entrance where many students normally gather. Although this histogram $\mathbf{H_p}$ does provide an estimate of the rate of occlusions, similar estimation can also be derived from the

tracking failure noise clusters. Additionally, the interesting question is not "how many objects are occluded" but "how likely it is for an object to be occluded".

Therefore, it is necessary to take into account the activity in the area. Figure 3.43 shows a histogram $\mathbf{H_m}$ of successfully matched blob positions. Occlusions are represented by a probabilistic image map $\mathbf{H_o}$, defined by the following formula:

$$\mathbf{H}_{\mathbf{o}(i,j)} = \frac{\mathbf{H}_{\mathbf{p}(i,j)}}{\mathbf{H}_{\mathbf{m}(i,j)}} \tag{3.5}$$

where i,j are indices to the pixels of the image. $\mathbf{H_o}$ is shown on the Figure 3.44

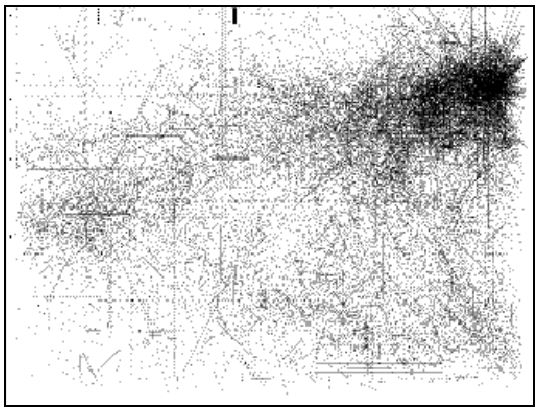


Figure 3.42: Log histogram of 19772 predicted positions (occlusion events)

Figure 3.43: Log histogram of 221356 matched positions

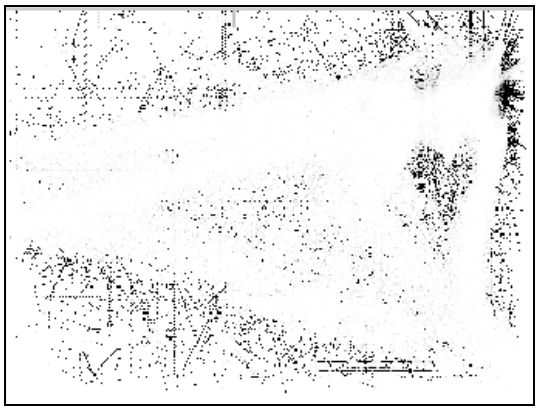


Figure 3.44: Representation of H_o. The dark areas indicate high probability of occlusions. The university building can be detected as occlusion regions. Also, part of the flag pole and its base are detected.

To further investigate the performance of the occlusion detection algorithm, a synthetic occlusion was produced (Figure 3.45) and occlusions were detected in the probabilistic image map $\mathbf{H_o}$.

Figure 3.45: A synthetic occlusion was produced by sticking some black tape on the window in front of the camera.

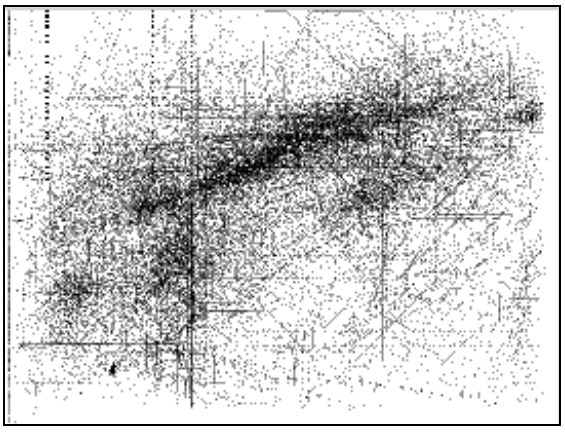


Figure 3.46: Predicted positions (occlusion events)

Figure 3.47: Log histogram of 309380 matched positions

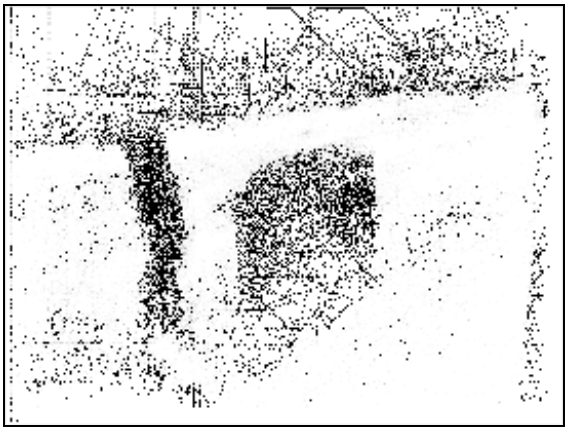


Figure 3.48: Representation of Ho. The black tape areas and the bus stop sign are associated to large values of H_o.

Occlusion regions in Figure 3.44 and Figure 3.48 are represented by probabilistic image maps. Their representation with a more compact model is an open issue. Gaussian models are inappropriate because $\mathbf{H_o}$ cannot be assumed to result from a Gaussian process. Whilst polygons could be used to represent the spatial extent of the occlusion, such a representation assumes that occlusions are static, fixed, solid and no targets can move in front of the occlusion area. Scene depths maps were used in [88] to provide better occlusion reasoning, considering the 3D positions of the occlusion object and the targets. However, such a model is still "hard", in the sense that is Boolean and therefore it cannot model non-solid occlusions or adapt to scene changes.

3.7 Discussion

In this chapter, a semantic activity-based scene model was introduced. Modelling and learning entry/exit and stop zones location and usage from point-datasets has been investigated. Modelling and learning of scene features, such as occlusion regions and semi-stationary motion noise sources, were also discussed, although these scene features are not directly related to the proposed scene model.

The multi-step algorithm that was introduced appears to cope with noisy data. The method is not very sensitive to the initial model order selection, as extra models tend to be used for the noise data. However, as the model order is increased, some signal clusters may be multi-modal. The MDL criterion seems to fail [92], as it favours high-order GMMs that are unrepresentative. Stauffer [92] tried to deal with this problem, using a heuristic similar to MDL that penalises high-order GMMs by using an exponentially decreasing function. However, no information is given on how the decay parameter is estimated.

Because EM searches for a local maximum of the likelihood, its results depend on the initialisation. In the proposed method, random points are selected by the dataset, K-mean algorithm is applied and its result is used for the initialisation of the EM. However, the dependency on the selected points remains. A hierarchical EM [23] that starts from a large number of clusters and gradually merges them may be a solution to this problem, as dependency on the initialisation is decreased, due to the large number of initial points.

The data that is used for occlusion region detection is, as in all the other methods of this thesis, a set of centroid points, derived from the motion tracking algorithm. However, especially for this particular problem, it seems that using information about the shape and the size of detected blobs is more beneficial than using only their centroids. For instance, when a blob is partially occluded, its apparent centroid may differ from the real one and still be detected, while part of the blob is occluded.

The scene features that have been discussed so far have a simple geometry; therefore, their spatial extent can be well approximated by GMs. However, routes may have a more complicated shape as it can be seen in an accumulative activity-histogram. Therefore, the single Gaussian model per feature model that was used cannot be applied.

In the next chapter an alternative solution of a polygon-like model, enriched with appropriate pdfs is suggested.

Junctions and paths are closely related to routes and if the routes of the scene are known, then junctions and paths can be extracted more easily. All these scene elements are discussed in the next chapter.

Chapter 4

Learning Routes

4.1 Introduction

In the previous chapter, a scene model was introduced and models and learning methods for semantics from single-point datasets, like entry/exit and stop zones, were presented. The scene model also represents paths, junctions and routes. These features require datasets of sequences of points rather than individual points. This chapter deals with modelling and learning of these features from observed trajectories. For this reason, a novel route model and an associated learning algorithm are introduced. Paths and junctions are derived from the set of scene route models.

In §4.2, the route model is introduced. This model is consistent with the requirements of a spatial semantic description, combined with probabilistic encoding of activity. Furthermore, it is discrete, allowing direct deployment of probabilistic networks. In §4.3, a novel algorithm is described that learns route models from trajectories. The algorithm has the advantage of being unsupervised, auto-initialised and capable of operating online. It can deal with incomplete trajectories and is easily configured. Trajectory classification to route models is discussed in §4.4. Trajectory classification is used during the route learning process and during the operation time of a surveillance system. Approaches based on Boolean logic, fuzzy logic and maximum likelihood (ML) are discussed. In §4.5, experimental results illustrate the performance of the algorithm. In §4.6, junctions and paths are extracted from the set of scene route models, using computational geometry.

4.1.1 Previous work

Fraile and Maybank [37] proposed a polynomial approximation of vehicle trajectory segments and a simple HMM to correspond trajectory segments to events like "turn left", "turn right", "ahead", "stop". The set of events is a rough approximation of the target dynamics, though no spatial modelling was attempted. Additionally, a spline representation seems to be a more appropriate approximation of a trajectory than a set of unrelated polynomial functions that do not guarantee continuity of the trajectory derivatives.

Another HMM-based approach is to model trajectories as transitions between states representing Gaussian distributions on the 2D image plane (e.g., Remagnino and Jones [81], Kaewtrakulpong [55]) or on the 4D space of image locations and velocities (e.g., Walter et al [101]). However, the "hidden" states associated with point-centred distributions do not necessarily correspond to real semantic features of the scene. Also, the representation of routes as sets of point-centred distributions is not consistent to a physical interpretation that is analogue and continuous.

Koller-Meier et al [58] used a node-based model to represent the average of trajectory clusters. The proposed learning algorithm is based on an average distance criterion. An extension mechanism allows learning from partial trajectories. A similar technique is suggested by Lou et al [61]. A K-means-like algorithm was used to cluster trajectories, based on the distance criterion of the maximum separation between trajectories. Although both methods successfully identify the mean-average of common trajectory patterns, no explicit information is derived regarding the distribution of trajectories around the mean average.

Fernyhough et al [33] and Fernyhough [34] proposed an automatic path-learning algorithm that automatically learns object paths by accumulating the trace of tracked blobs. The proposed model represents the spatial extent of the paths, though no probabilistic information can be derived. Although the algorithm is auto-initialised and some promising results have been demonstrated, it requires full trajectories, cannot handle occlusions and the results depend on the shape and size of the blobs, as they appear on the image plane.

Johnson and Hogg [52] and Johnson [53] proposed a vector quantisation NN method to learn typical routes taken by pedestrians from representative trajectories. The main advantage of this method over previous approaches is that it provides a probabilistic representation of activity. However, no high level semantic information is derived and no partially observed trajectories are handled.

Alternative Neural Network approaches have been proposed by Owens and Hunter [76], Hunter et al [46] and Hu et al [45]. Although results are promising in all the cases, the weakness of the NN learning methods is that they are computationally expensive, they lack adaptability, and therefore they are incompatible with online operation.

Grimson et al [40] and Stauffer and Grimson [90] proposed a hierarchical clustering of trajectories. Trajectories are represented as a sequence of states in a quantised 6D space and trajectory classification and the method is based on a co-occurrence matrix that assumes that all trajectory sequences are equally long. However, this assumption is usually not true and it is unclear how their method deals with it.

4.2 Route model

In road traffic environments, vehicles must follow specific predefined routes. In pedestrian environments, people normally walk on well-prescribed pathways. Even in cases where no predefined routes exist, the structure of the scene affects the behaviour of pedestrians and normal route-patterns of activity exist, which is verified by results presented in this chapter.

A route model that is both consistent to the human interpretation and fulfils the requirements for probabilistic analysis is required, according to the goals that were defined in §2.6.1. A route can be described intuitively by its start and end areas, its main axis between the start and the end and its boundaries along the main axis. Additionally, quantitative information about the usage along and across the route is required to describe the statistics of typical usage.

The scene is assumed to contain multiple routes that may have overlapped sections. A single route model must encode the following properties, so that it is consistent to both the spatial semantic and probabilistic description requirements:

- The main axis of the route.
- The terminators (start and end points) of the route.
- A description of the width along the route.
- Indication of the level of usage of the route, both along and perpendicular to the direction and in comparison to the other routes.

The route model that it is proposed in this thesis (Figure 4.1) consists of a central spline axis, defined by a sequence of equi-distant nodes that represents some average of the route. The constant distance between adjacent nodes is referred to as the resample factor R of the model. In addition, two bound splines around the central axis form an envelope and represent the width of the path. Alternatively, the width can be represented by the standard deviation of the observations across the route. A route has two terminator nodes (start and end) that typically correspond to entry/exit zones of the scene. Finally, a weight factor represents the usage frequency of the route.

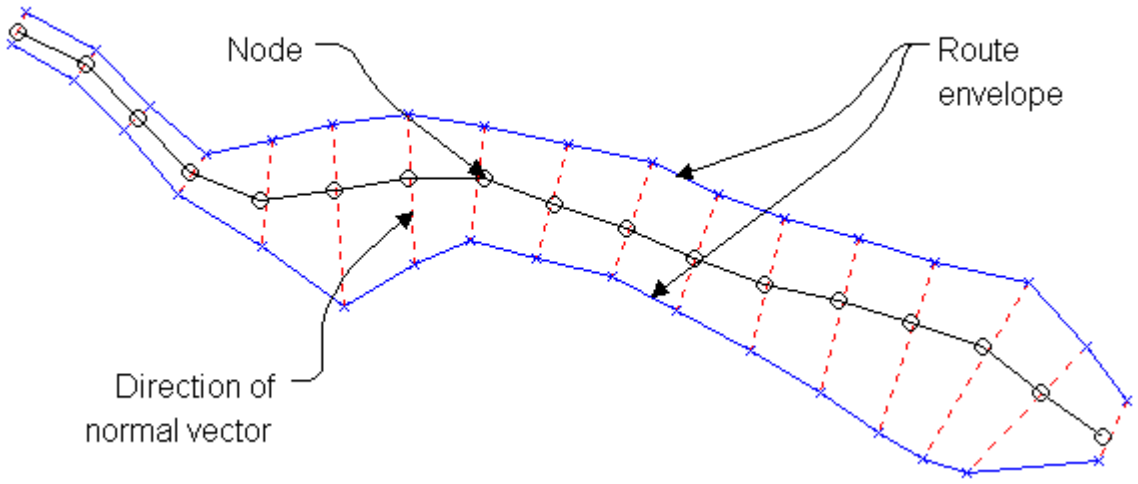


Figure 4.1: Spatial representation of the proposed route model.

Specifically, each node i, i=1..L where L is the number of the model nodes, is characterized by:

– a 2D position vector that represents the coordinates of the node: $\mathbf{x_i}=[x_i, y_i]$

– a weight factor w_i that reflects the strength of the node, based on the number of times that it has been updated, during learning.

– a normal vector $\mathbf{n_i}=[n_{xi}, n_{yi}]$, defined as the unit vector perpendicular to the local spline direction, defined by the sequence of vectors **x**.

– two bound 2D points along the normal vector line, the left boundary $\mathbf{l_i}=[l_{xi}, l_{yi}]$ and the right boundary $\mathbf{r_i}=[r_{xi}, r_{yi}]$. The two bound points encode the width of the route at the specific position. They can be set according to either the extremes of the matching trajectories, or a Gaussian distribution around the node position, based on the standard deviation σ_i of observations along the local normal vector.

– A probability density function $g_i(d)$ of the usage of the node, across the route, where d is the signed distance from the node position.

The direction of the normal vector $\mathbf{n_i}$ is assumed that satisfies the following condition:

$$\forall i, \quad \mathbf{n_i} \cdot (\mathbf{l_i} - \mathbf{x_i}) \geq 0 \quad \textit{and} \quad \mathbf{n_i} \cdot (\mathbf{r_i} - \mathbf{x_i})_i \leq 0 \tag{4.1}$$

The weight factor w of the whole route is given as the average of weights of its nodes:

$$w = \sum_{i=1}^{L} w_i \Big/ L \tag{4.2}$$

A set of splines can be defined for each route model, derived by the sequence of nodes. A main axis spline **Sx** is defined by the sequence of the node centres $\mathbf{x}=\{\mathbf{x_i}\}$. Similarly, the left boundary **Sl** and the right boundary **Sr** splines are defined by the sequence of the node bound points $\mathbf{l}=\{\mathbf{l_i}\}$ and $\mathbf{r}=\{\mathbf{r_i}\}$, respectively. Additionally, two splines $\mathbf{Sl_\sigma}$ and $\mathbf{Sr_\sigma}$ are defined, based on the sequence of points $\mathbf{l_\sigma}=\{\mathbf{x_i}+\sigma_i \cdot \mathbf{n_i}\}$ and $\mathbf{r_\sigma}=\{\mathbf{x_i}-\sigma_i \cdot \mathbf{n_i}\}$, respectively.

A region $\mathbf{E_i}$ is defined for the node i as the region enclosed by the two boundary splines **Sl** and **Sr** and two lines that are parallel to the normal $\mathbf{n_i}$ and in distances R/2 from it.

The above-described model allows explicit representation of the spatial extent of routes and therefore it is consistent to the semantic representation requirement. In addition, the probabilistic representation of the usage, both along and across the route, and the discrete nature of the model allow direct deployment of a probabilistic network (e.g., HMM).

4.3 Route learning

The input data of the algorithm is a set of trajectories, derived by a motion tracking algorithm that estimates the location of the centroid of the moving objects, from a single fixed camera. It is desirable to learn paths using representative trajectories unconstrained by tracking failure noise (see §2.3.1) . For this reason, short trajectories or trajectories with many sudden changes of direction are filtered. Further validation of the trajectory dataset is based on knowledge of the entry/exit zones. Specifically, trajectories are accepted only if they start from a valid entry zone and terminate at a valid exit zone.

Trajectories are resampled over the spatial distance, according to the resample factor R, to normalize the trajectories of high and low speed objects and to counter the effects of perspective. A resample trajectory **p** is represented as a sequence of points $\mathbf{p_i}=\{p_{xi}, p_{yi}\}$, i=1..K, where K is the number of the samples.

The model order of the scene routes is not defined explicitly, but it is determined implicitly by the algorithm parameters and the dataset. The first trajectory of the dataset initialises the first route model. Other route models will also be initialised automatically by trajectories that do not match an existing model.

Theoretically, there is no restriction in the number of route models of the scene. In practice, route models with very low weight factors w_i are discarded through the learning process, for computational efficiency.

A summary of the learning algorithm is as follows:

- The first trajectory of the dataset initialises the first route model.
- Each new trajectory is compared with the existing route models.
    - If a trajectory matches a route model, then the route model is updated.
    - If a trajectory does not match to any route model, a new model is initialised.
- The updated route model is resampled, so inter-node distances are kept equal to *R.*
- Each updated route model is compared with the other route models.
    - If two route models are sufficiently overlapped, they are merged.

The algorithm requires mainly two parameters: a) the resample factor R, b) the distance threshold T. The resample factor R defines how detailed are the route models.

Very small values for the resample factor are not recommended, because this selection can make the algorithm computationally expensive without significant benefit. The distance threshold T defines the minimum allowed gap between different routes. Its recommended value is related with the quantity of the learning data and specifically the less data the larger the value of T. Comparison of results for different values of the parameters R and T is given at §4.5.

The estimation of the probability density functions $g_i(d)$ of usage across the route is performed after spatial clustering. This means that after determining the other route model properties, the trajectory dataset is re-used to explicitly estimate these distributions (see §4.4.2). However, if $g_i(d)$ is assumed Gaussian, it can be directly determined online by the standard deviation σ_i of the node, simultaneously with the other attributes of the model.

In the following paragraphs, the basic steps of the learning algorithm are described in more detail.

4.3.1 Route model initialisation

The first trajectory of the dataset and any trajectory that does not match any of the existing routes initialise a new route model. Each sample i of the trajectory **p** defines a node of the new route model, according to the following equations:

$$\mathbf{x_i} = \mathbf{p_i} \quad (4.3a)$$

$$\mathbf{l_i} = \mathbf{p_i} \quad (4.3b)$$

$$\mathbf{r_i} = \mathbf{p_i} \quad (4.3c)$$

$$w_i = 1 \quad (4.3d)$$

$$\sigma_i = 0 \quad (4.3e)$$

The normal vectors are determined to be perpendicular to the spline **Sx.**

4.3.2 Classification of trajectory to route model

Each trajectory is compared with each route model. Comparison is based on the distance between a route model and a trajectory. The route model l with the minimum distance from the trajectory k is a candidate match for the trajectory. If this distance D_{kl} is smaller than the threshold distance T ($D_{kl} \leq T$), the trajectory matches the candidate

route model and updates it. Otherwise, a new route model must be initialised by the trajectory.

The following distances are defined for trajectory clustering:

Distance d_{ij} of trajectory sample i from route node j: If $\mathbf{p_i}$ is within the region $\mathbf{E_j}$ of the node j (as defined in §4.2), then d_{ij} is equal to zero. Otherwise, d_{ij} is set to the minimum Euclidean distance of $\mathbf{p_i}$ from the borders of the node area $\mathbf{E_j}$.

$$d_{ij} = \begin{cases} 0 & if \quad p_i \in E_j \\ \min D(p_i, E_j) & if \quad p_i \notin E_j \end{cases} \tag{4.4}$$

Distance d'_{il} of trajectory sample i from route l: Defined as the minimum of the distances of the trajectory sample from the route nodes:

$$d'_{il} = \min_{j}^{L_l}(d_{ij}) \tag{4.5}$$

where L_l is the number of the nodes of the route model l.

Distance D_{kl} of trajectory k from route l: Defined as the maximum of the distances of the trajectory samples from the route model:

$$D_{kl} = \max_{i}^{K_k}(d'_{il}) \tag{4.6}$$

where K_k is the number of samples of the trajectory k.

The above classification scheme allows only one matched route per trajectory. Also, it assumes that a trajectory is not longer than the matched route. However, if the training dataset contains partial trajectories, then the algorithm may fail.

For this reason, partial matching of route models from trajectories is allowed. That means that a trajectory may match a route model, even if the route model is shorter than the trajectory. In this case, the samples of the trajectory k that are beyond the end of the route l are excluded from the estimation of the distance D_{kl}, as calculated by Eq.4.6.

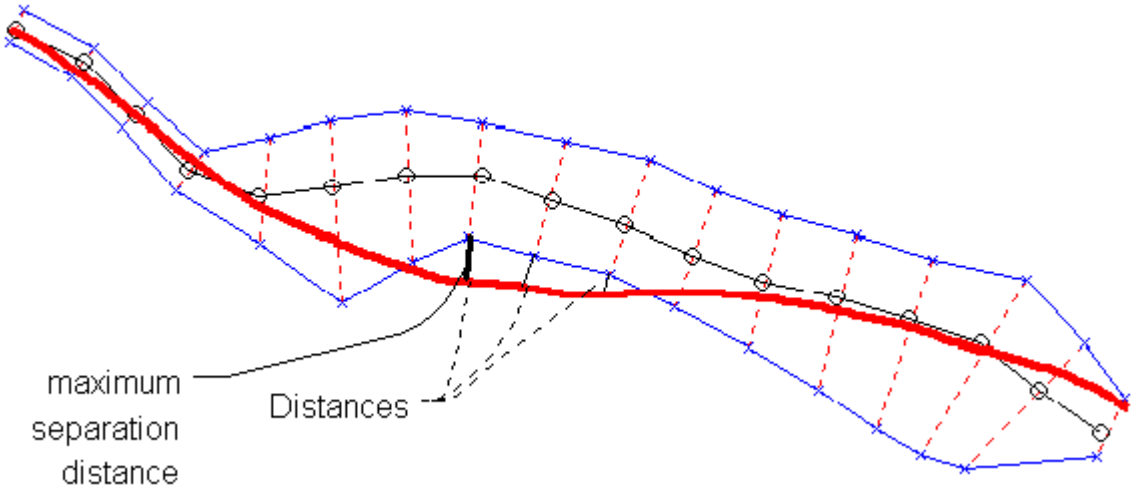


Figure 4.2: Distances of a trajectory points from a route. The maximum separation distance is smaller than the threshold, so that trajectory matches the route.

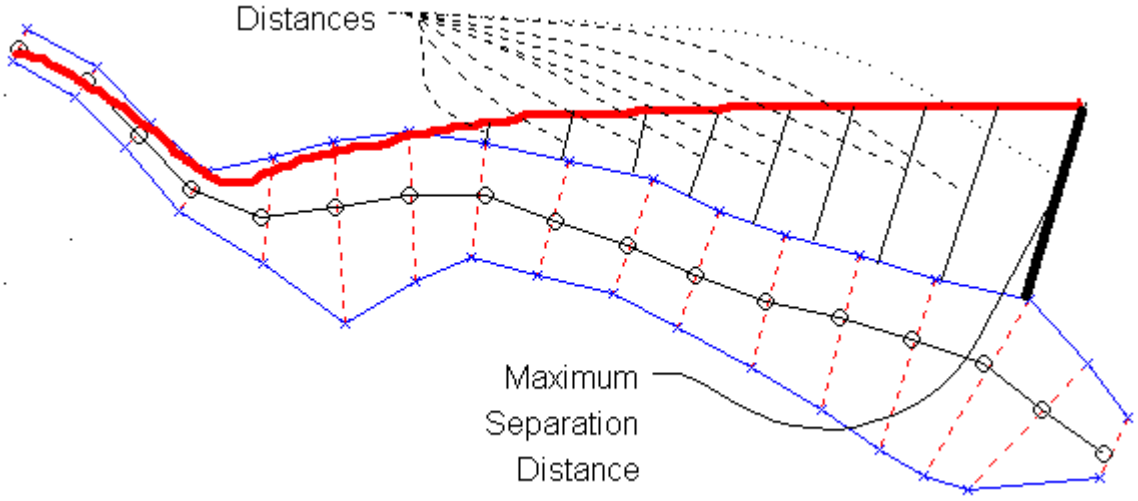


Figure 4.3: Distances of trajectory points from a route. The maximum separation distance is larger than the threshold, so that the trajectory is unmatched.

Because route models may have common paths, it is possible that a partial trajectory will match more than one route model. Although partial trajectories can be eliminated from the dataset using the knowledge of entry/exit zones, the ability to classify partial trajectories is still desirable. For example, an online trajectory classifier must be able to cope with partial trajectories and multiple matchings.

However, the classifier that was described above is based on Boolean logic that does not allow multiple classifications. Later in this chapter, two other classifiers that

allow multiple matchings are discussed: the Fuzzy Logic Trajectory Classifier and the Maximum Likelihood Trajectory Classifier.

4.3.3 Update route model with trajectory

When a trajectory matches a route model, during the learning process, the route model is updated by the trajectory. Route model updating is performed in three steps:

i) Node updating: The trajectory is resampled by the normals of the route model, so that each route model node i is associated with a trajectory sample $\mathbf{p_i}$ laying on the direction of the normal $\mathbf{n_i}$. The node properties are updated according to the following equations:

$$\mathbf{x}_i' = \frac{w_i \cdot \mathbf{x_i} + \mathbf{p_i}}{w_i + 1} \tag{4.7a}$$

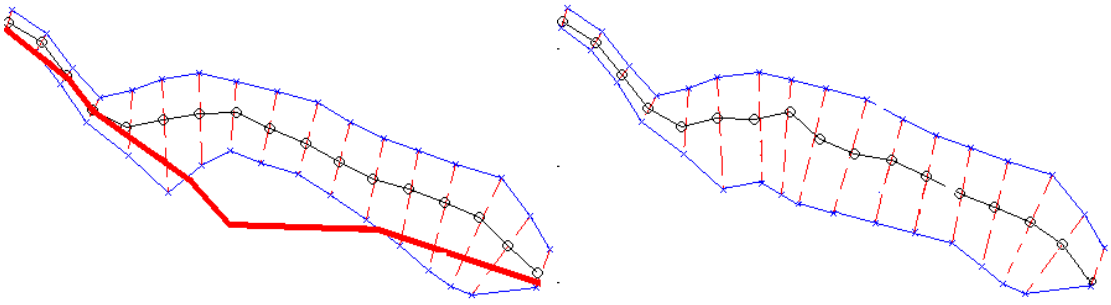
$$\mathbf{l_i'} = \begin{cases} \mathbf{l_i}, & \forall i : (\mathbf{p_i} - \mathbf{l_i}) \cdot \mathbf{n_i} \le 0 \\ \mathbf{p_i}, & \forall i : (\mathbf{p_i} - \mathbf{l_i}) \cdot \mathbf{n_i} > 0 \end{cases} \tag{4.7b}$$

$$\mathbf{r_i'} = \begin{cases} \mathbf{r_i}, & \forall i : (\mathbf{p_i} - \mathbf{r_i}) \cdot \mathbf{n_i} \ge 0 \\ \mathbf{p_i}, & \forall i : (\mathbf{p_i} - \mathbf{r_i}) \cdot \mathbf{n_i} < 0 \end{cases} \tag{4.7c}$$

$$w_i' = w_i + 1 \tag{4.7d}$$

$$\sigma_i'^2 = \frac{w_i}{w_i + 1} \cdot \sigma_i^{\,2} + \frac{w_i \cdot 1}{(w_i + 1)^2} \cdot \|\mathbf{x_i} - \mathbf{p_i}\|^2 \tag{4.7e}$$

The above equations allow accumulation of the information by the route model nodes.



(a) (b)

Figure 4.4: Route model matched by a trajectory (a) and then updated (b). The red thick line represents the trajectory.

ii) Route model extension: If partial matching is allowed, then it is possible that a route model is matched by a longer trajectory (see Figure 4.5). In this case, the part of the

trajectory that is beyond the terminators of the route model is used to initialise new nodes of the route model (using Eq.4.3) to extend the model. To avoid unrepresentative extensions, extension may not be allowed if the matched route model part is short or if the weight of route terminators is high. In this case, some extra parameters determine the restriction rules.

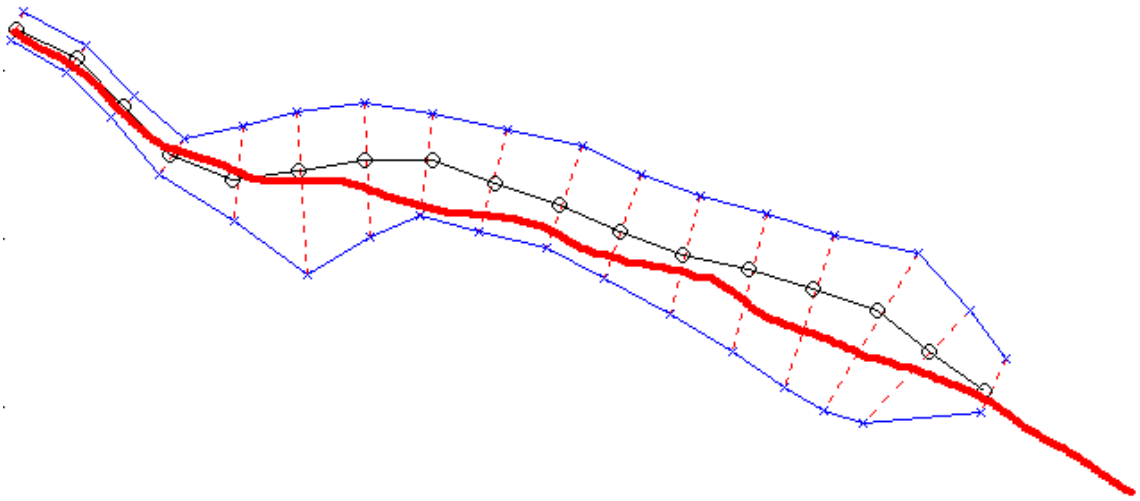


Figure 4.5: Example of a route that should be extended.

iii) Route model resampling: After updating, the route model nodes may have moved from their position and the distance between consecutive nodes may not be equal to R. To keep the nodes at equal distances, new sequences of position vectors **x'** and weights **w'** are calculated from the spline **Sx**. The sequence of normals **n** is updated appropriately, so that each new normal $\mathbf{n_i'}$ is perpendicular to the direction of the spline **Sx**, at the position $\mathbf{x_i'}$. The bound vectors $\mathbf{l_i'}$ and $\mathbf{r_i'}$ are defined as the points that the normal $\mathbf{n_i'}$ crosses the splines **Sl** and **Sr** respectively. Standard deviations σ_i are estimated considering the splines $\mathbf{l_\sigma}$ and $\mathbf{r_\sigma}$. The crossings of the two splines **Slσ** and **Srσ** with the normals $\mathbf{n_i'}$ define two new sequences of points $\mathbf{l'_{\sigma i}}$ and $\mathbf{r'_{\sigma i}}$. Then, the standard deviation is defined as:

$$\sigma_i' = \frac{1}{2} \cdot \left(\left\| \mathbf{l'_{\sigma i}} - \mathbf{x'_i} \right\| + \left\| \mathbf{r'_{\sigma i}} - \mathbf{x'_i} \right\| \right) \tag{4.8}$$

4.3.4 Route comparison

Route models are initialised according to the sequence of trajectories in the dataset. Therefore, during the learning process it is possible for a trajectory that corresponds to an existing but incomplete route model to initialise a second route model, due to the lack of trajectories that fill the "gap" between the two route models, at that time. Obviously, it is desirable that the two route models are merged, when evidence from the trajectory dataset can prove that they actually belong to the same scene route. Two route models are considered for merging, when their maximum separation falls below a threshold T.

Route comparison is based on the following distances:

i) Distance $d_{ik,jl}$ of node i of route model k from node j of route model l: A region for each node is defined, according to the definition given at §4.2. If the regions E_{ik} and E_{jl} of the two nodes are overlapped, then $d_{ik,jl}$ is set to zero, otherwise it is set to the minimum distance between the two areas.

$$d_{ik,jl} = \begin{cases} 0 & if \quad E_{ik} \cap E_{jl} \neq \{\} \\ \min D(E_{ik}, E_{jl}) & if \quad E_{ik} \cap E_{jl} = \{\} \end{cases} \tag{4.9}$$

ii) Distance $d'_{ik,l}$ of node i of route model k from route model l: Defined as the minimum of the distances of the node i from the nodes of route model l:

$$d'_{ik,l} = \min_{j}^{L_l}(d_{ik,jl}) \tag{4.10}$$

where L_l is the number of the nodes of the route model l.

iii) Distance D_{kl} of route model k from route model l: Defined as the maximum of the distances of nodes of route model k from the route model l:

$$D_{kl} = \max_{i}^{K_k}(d'_{ik,l}) \tag{4.11}$$

where K_k is the number of nodes for the route model k.

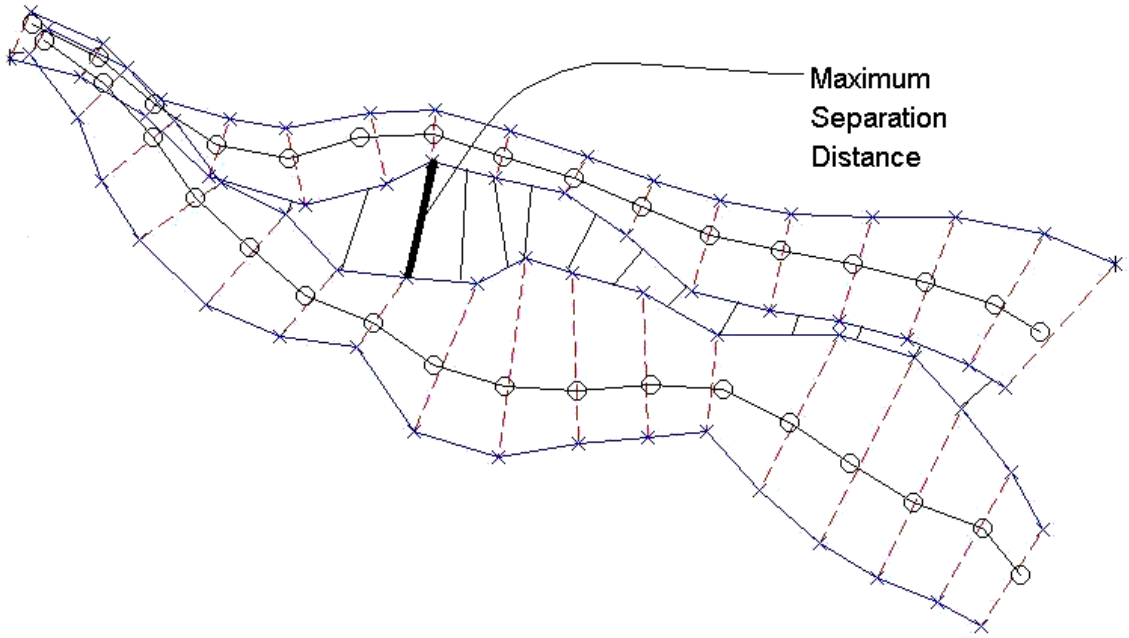


Figure 4.6: Maximum Separation Distance of two paths. In this example, the distance is smaller than the threshold, so the paths will be merged.

4.3.5 Route merging

If the maximum separation distance between two route models is smaller than the threshold T, the two route models can be merged. Route merging is similar to route updating. First, the route with the highest weight is selected as primary and this route is updated with the other, secondary, route. Each node i of the main route is updated, using the crossing $\mathbf{x}_{2,i}$, $\mathbf{l}_{2,i}$, $\mathbf{r}_{2,i}$, $\mathbf{l}_{\sigma 2,i}$, $\mathbf{r}_{\sigma 2,i}$ of the splines $\mathbf{Sx}_2$, $\mathbf{Sl}_2$, $\mathbf{Sr}_2$, $\mathbf{Sl}_{\sigma 2}$, $\mathbf{Sr}_{\sigma 2}$, of the secondary route, with the normal vector $\mathbf{n}_{1,i}$.

$$\mathbf{x}'_{1,i} = \frac{w_{1,i} \cdot \mathbf{x}_{1,i} + w_{2,i} \cdot \mathbf{x}_{2,i}}{w_{1,i} + w_{2,i}} \tag{4.12a}$$

$$\mathbf{l}'_{1,i} = \begin{cases} \mathbf{l}_{2,i}, & \text{if } (\mathbf{l}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} \geq (\mathbf{r}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} \text{ and } (\mathbf{l}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} > (\mathbf{l}_{1,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} \\ \mathbf{r}_{2,i}, & \text{if } (\mathbf{l}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} < (\mathbf{r}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} \text{ and } (\mathbf{r}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} > (\mathbf{l}_{1,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} \\ \mathbf{l}_{1,i}, & \text{otherwise} \end{cases} \tag{4.12b}$$

$$\mathbf{r}_{1,i} = \begin{cases} \mathbf{r}_{2,i}, & \text{if } (\mathbf{l}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} \geq (\mathbf{r}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} \text{ and } (\mathbf{r}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} < (\mathbf{r}_{1,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} \\ \mathbf{l}_{2,i}, & \text{if } (\mathbf{l}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} < (\mathbf{r}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} \text{ and } (\mathbf{l}_{2,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} < (\mathbf{r}_{1,i} - \mathbf{x}_{1,i}) \cdot \mathbf{n}_{1,i} \\ \mathbf{r}_{1,i}, & \text{otherwise} \end{cases} \tag{4.12c}$$

$$w'_{1,i} = w_{1,i} + w_{2,i} \tag{4.12d}$$

$$\sigma'^{\,2}_{1,i} = \frac{w_{1,i} \cdot \sigma_{1,i}^{\,2} + w_{2,i} \cdot \sigma_{2,i}^{\,2}}{w_{1,i} + w_{2,i}} + \frac{w_{1,i} \cdot w_{2,i}}{\left(w_{1,i} + w_{2,i}\right)^2} \cdot \left\| \mathbf{x_{1,i}} - \mathbf{x_{2,i}} \right\|^2 \tag{4.12e}$$

If the secondary route has nodes that extend beyond the terminators of the main route, then the main route is extended. Finally, the updated route is resampled, as described in the previous section.

4.4 Trajectory classification

Trajectories or parts of trajectories are classified to route models either during the learning process or during the normal operation of the surveillance system. In §4.3.2, a Boolean trajectory classifier, based on a distance criterion, has been described. This approach is appropriate when trajectories are in areas where routes are not overlapped and no ambiguity exists. However, when substantial overlapping occurs, an uncertainty arises which cannot be resolved by the Boolean classifier. An optimal approach (in a Maximum Likelihood sense) would require knowledge of the distribution of the observations across each route. Because in general this distribution is not known prior to the generation of the geometric model, a method is introduced that encodes the trajectory distribution across a route, and uses Fuzzy Logic in order to bootstrap the route construction process.

Once the route models have been learnt, a maximum likelihood classifier can be used to recognise and label new trajectories. If cross route distributions are assumed Gaussian, then cross route distributions are known during the online learning and it is possible to use the maximum likelihood classifier for learning.

4.4.1 Fuzzy Logic Trajectory Classifier (FLTC)

Using a fuzzy logic approach, certainty estimates are calculated for point-node matches and then an overall certainty is estimated for the complete trajectory-route match. An asymmetric membership function is used (Figure 4.7) which captures the typical characteristics of the distribution. The distribution is modelled by a function g of a point $\mathbf{p_i}$ lying in the normal direction of the node i and is defined as:

$$g_i(d)=\begin{cases} w_i\left(\cdot\dfrac{threshold-d}{\|\mathbf{b_i}-\mathbf{x_i}\|}+\dfrac{threshold\cdot d}{\|\mathbf{b_i}-\mathbf{x_i}\|^2}\right), & d\le 0. \\ w_i\cdot\dfrac{threshold-d}{\|\mathbf{b_i}-\mathbf{x_i}\|}, & 0<d<threshold. \\ 0, & d\ge threshold. \end{cases} \tag{4.13}$$

where $d=\|\mathbf{p_i}-\mathbf{x_i}\|-\|\mathbf{b_i}-\mathbf{x_i}\|$ and $\mathbf{b_i}=\mathbf{l_i}$ or $\mathbf{b_i}=\mathbf{r_i}$, under the condition $(\mathbf{p_i}-\mathbf{x_i})\cdot(\mathbf{b_i}-\mathbf{x_i})\ge 0$. If $\|\mathbf{b_i}-\mathbf{x_i}\|<threshold$, we set $\|\mathbf{b_i}-\mathbf{x_i}\|=threshold$ in Eq.4.13. The threshold distance is defined as the distance from either side of the route boundaries where matching is still possible.

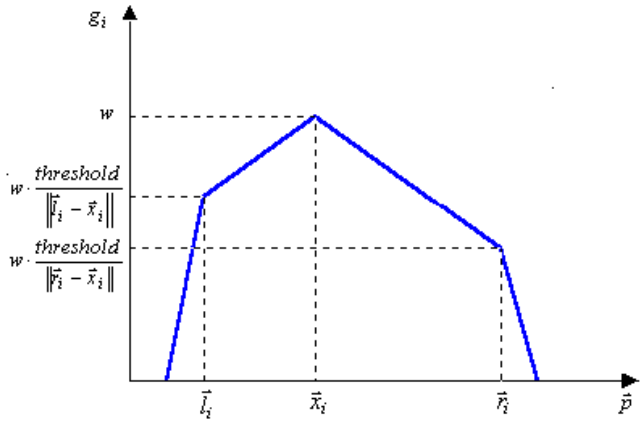


Figure 4.7: Proposed membership function that models the distribution of observations across a route node. $\mathbf{x_i}$ is the position of the node and $\mathbf{l_i}$, $\mathbf{r_i}$ are the two bound points. The point $\mathbf{p_i}$ is assumed to be lying on the normal direction of the route.

To classify a trajectory within a route model j using the FLTC, the trajectory is sampled by the node normals of the route. For each trajectory sampled-point $\mathbf{p_{i_j}}$ a weighted estimate [87] g_{i_j} is given according to Eq.4.13. Then, the certainty estimate of a trajectory-route match $h'_j(\{\mathbf{p_i}\})$ is given by the minimum of the certainty estimates of the point-node matches (4.14). Finally, the certainty estimates are normalized by dividing by the sum of the certainty estimates for all the possible trajectory-route matches (Eq.4.15). The trajectory $\{\mathbf{p}\}$ is classified to the route j that maximizes the certainty estimate.

$$h'_j(\{\mathbf{p}\})=\min_{i_j}\left(g_{i_j}(\mathbf{p}_{i_j})\right) \tag{4.14}$$

$$h_j(\{\mathbf{p}\}) = \frac{h'_j(\{\mathbf{p}\})}{\sum_k h'_k(\{\mathbf{p}\})} \tag{4.15}$$

The FLTC is used during the learning process to match trajectories with existing route models. In the case that multiple matching occurs, all the matched route models are updated according to the particular certainty estimations.

4.4.2 Maximum Likelihood Trajectory Classifier (MLTC)

Having acquired the spatial model, distributions of observations across the routes are learnt and incorporated into the model. Figure 4.8 and Figure 4.9 show the estimated pdfs of observations across selected nodes for two specific routes, from 465 and 6172 matched trajectories, respectively. An alternative, more compact, approach is to approximate the cross-route distributions using Gaussian functions. In this case, the standard deviations σ_i have already been estimated during the learning process.

The knowledge of the distributions across the route allows for the use of a Maximum Likelihood Trajectory Classifier (MLTC) as a more accurate alternative to the FLTC. If g_{i_j} represents the pdf of observations across node i_j of route j, the probability $h'_j(\{\mathbf{p}\})$ of a trajectory **p**, under the condition that it belongs to the route j, is given by Eq.4.16, assuming independent observations $\mathbf{p}_{i_j}$. According to the ML criterion, the trajectory is assigned to the route that maximizes the probability h_j as given by Eq.4.17.

$$h'_j(\{\mathbf{p}\}) = \prod_i g_{i_j}(\mathbf{p}_{i_j}) \tag{4.16}$$

$$h_j(\{\mathbf{p}\}) = \frac{h'_j(\{\mathbf{p}\})}{\sum_k h'_k(\{\mathbf{p}\})} \tag{4.17}$$

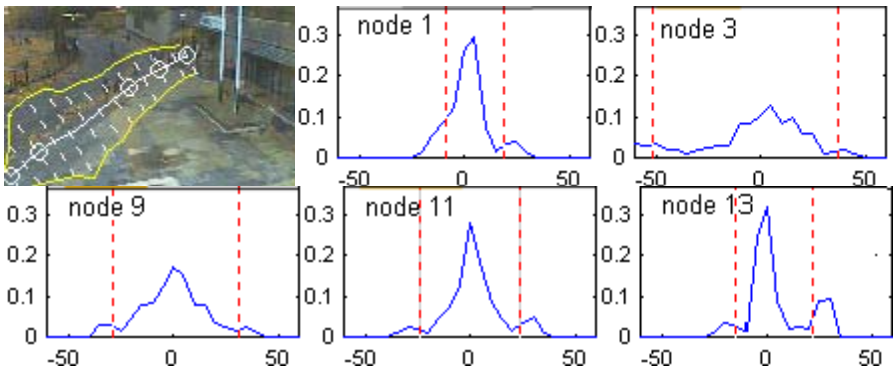


Figure 4.8: Route model and derived distributions of observations across five selected nodes. Node ordering is from bottom left to top right along the route. Red dashed lines indicate the boundaries (envelope) of the route at each node. The x-axis indicates the distance from the main axis of the route in pixels.

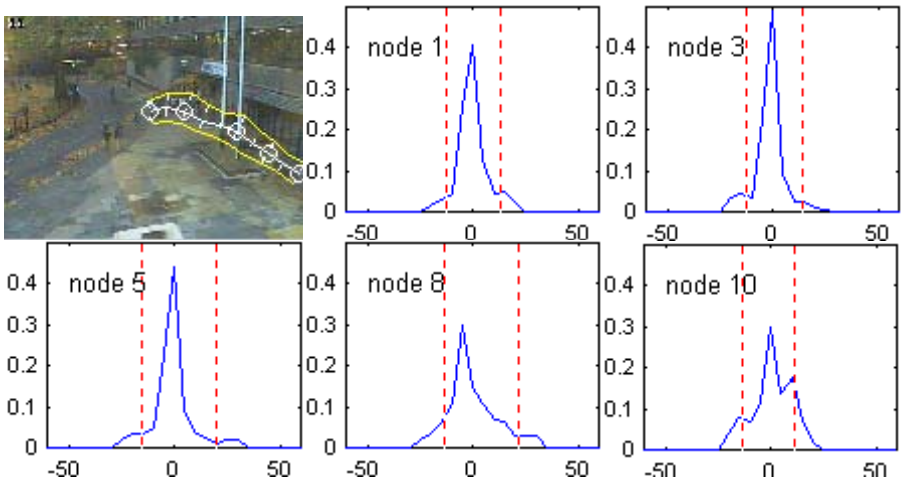


Figure 4.9: Route model and derived distributions of observations across 5 selected nodes. Node ordering is from bottom right to top left.

4.5 Route learning results

The performance of the proposed algorithm has been tested in different scenes and different datasets. Figure 4.10 depicts a dataset of 752 trajectories that are consistent with the entry/exit zones that were learnt in the previous chapter. Figure 4.11 shows the five route models that are automatically derived from the route learning algorithm, using a resample factor R=20 pixels and minimum separation distance threshold T=30 pixels, where frame size is 640x480. Figure 4.12 depicts the five route models individually.

The second and the fifth route model seem to be very similar. However, the distance between their terminators at the right side is larger than the minimum separation distance threshold (physically corresponding to university entrance and a cash machine),

therefore the route models are assumed different. For larger T, the two routes are merged (see Figure 4.15).

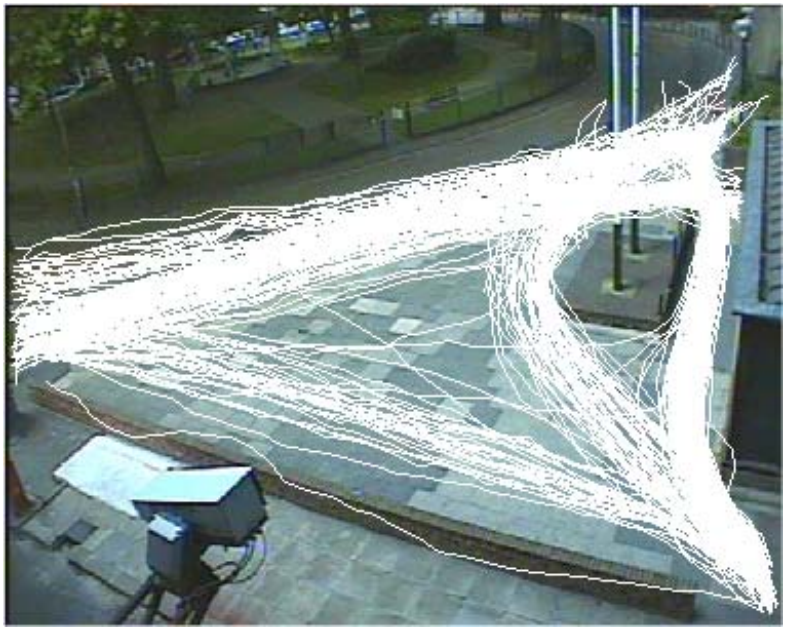


Figure 4.10: Dataset of 752 trajectories.

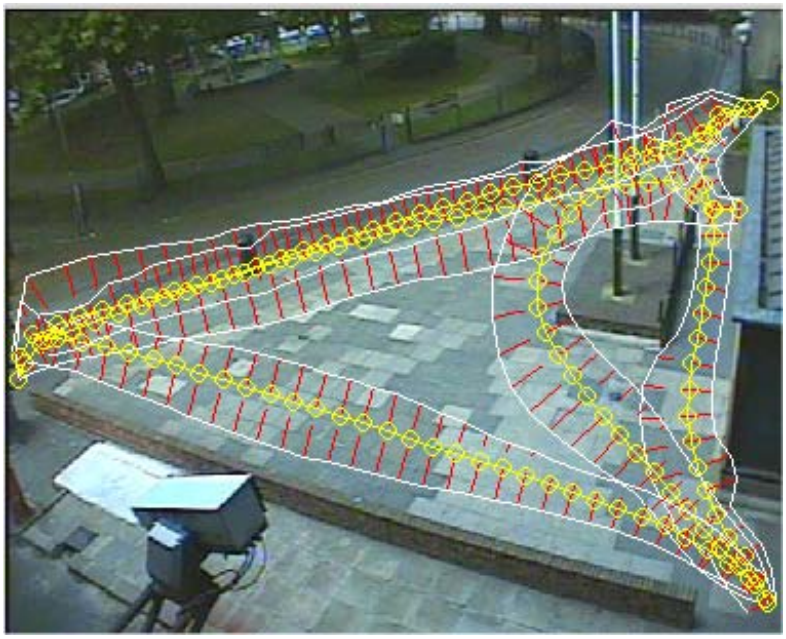


Figure 4.11: Set of five route models, as they derived by the route learning algorithm, for R=20 pix. and T=30 pix.

Figure 4.12: The five derived route models.

Route Model	Nodes	Usage
1	23	40.54%
2	33	31.30%
3	27	15.12%
4	34	8.96%
5	32	4.07%

Table 4.1: Usage of route models of Figure 4.12.

An alternative representation of the route width can be based on the standard deviation of the observations across the route. Figure 4.13 shows the spatial extent of the routes, if the boundaries of each node are set to a distance of $3 \cdot \sigma_i$ for each node i.

For most of the routes, results are identical for both representations. However, because sometimes the distribution of observations is not symmetrical around the node position $\mathbf{x_i}$, a Gaussian function is not always accurate. Therefore, some difference may exist on the two boundary representations, as it can be seen in two of the route models. Although width representation based on the standard deviation is not so accurate, it has the advantage of allowing adaptive learning of routes.

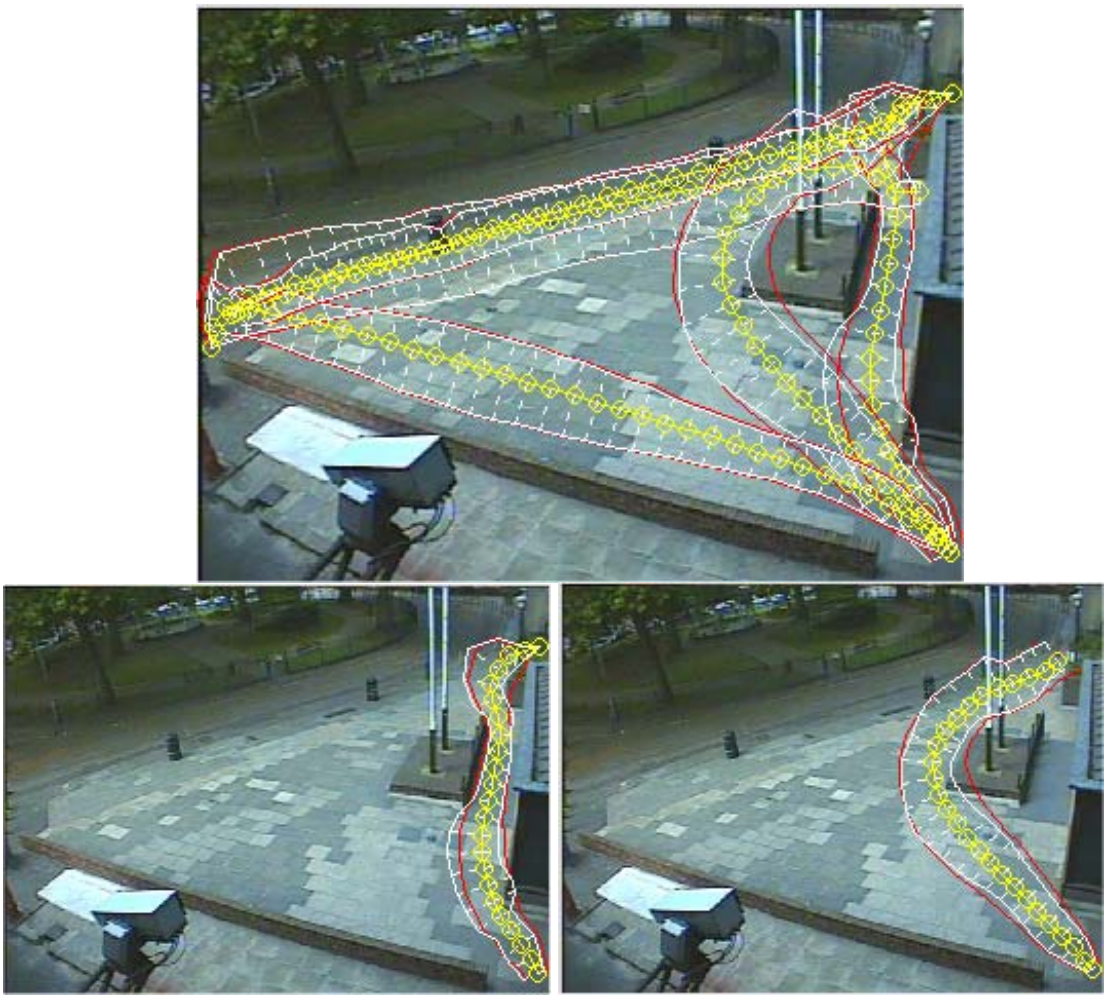


Figure 4.13: Route width modelling using the standard deviation of observations across the route. Red line indicates route width set to 3·σ, while white line indicates route width set according to the extremes of matched trajectories . Illustrations are given for the entire set of route models and for two specific route models.

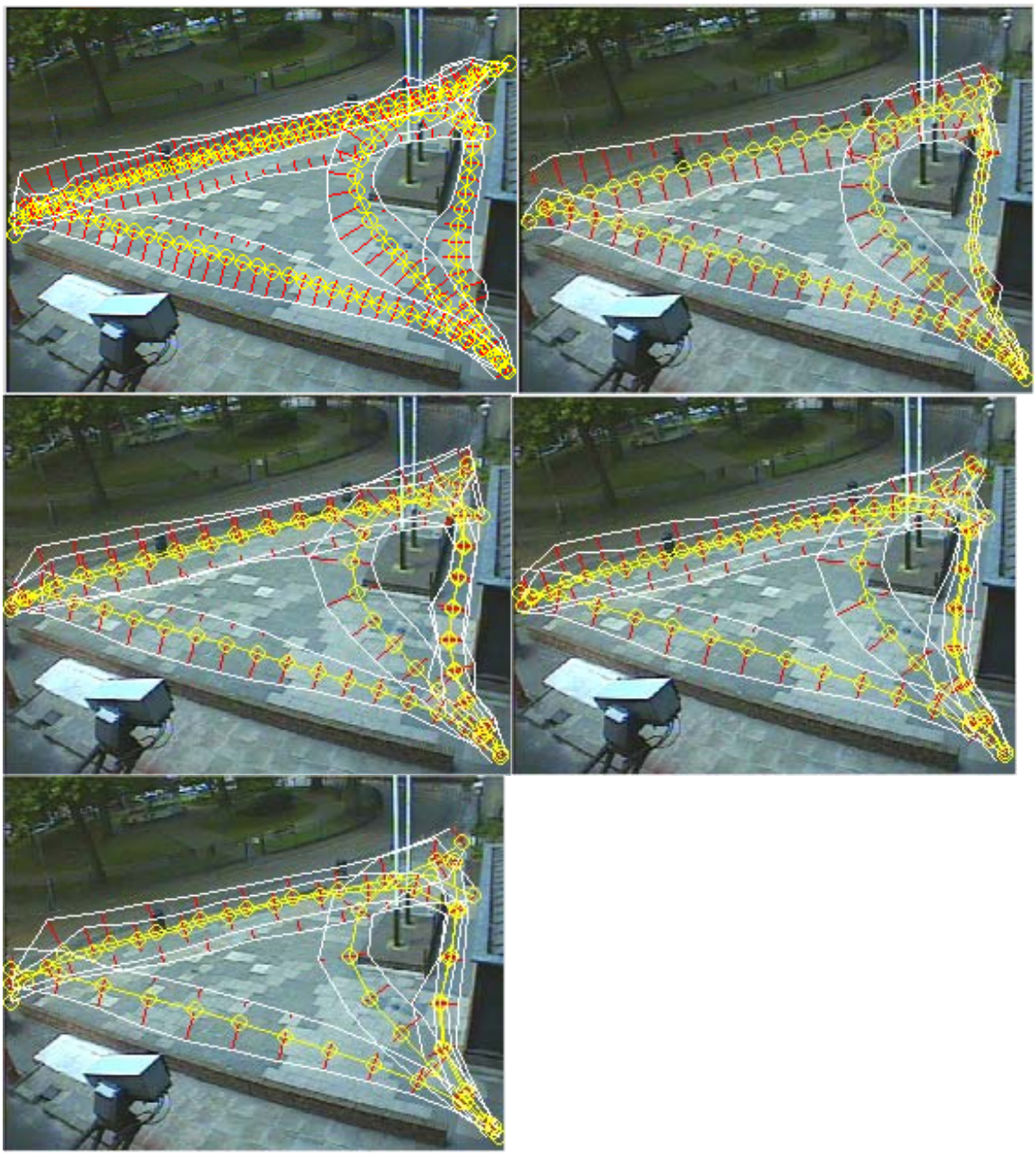


Figure 4.14: Route models derived for different value of the resample factor R. Results are given for R=20, 30, 40, 50, 60 and T=30.

The resample factor R defines the accuracy of the model. Results for different values of R (see Figure 4.14) are considered similar by a common visual inspection. Table 4.1 shows the average learning time per trajectory for different values of R. The route learning algorithm was implemented in Matlab and running on a Pentium3-550Mhz Linux system with 256MB RAM. The current implementation can serve real-time learning of routes with rates of 0.2-0.4 trajectories per second. Implementation of the

algorithm in C and deployment in faster processing units would allow real-time learning of routes, even for high-activity scenes, subject to satisfactory extraction of data from the motion tracking module.

R(pix)	*20*	*30*	*40*	*50*	*60*
Time(secs)	*8.455*	*4.849*	*3.851*	*2.833*	*2.548*

Table 4.2: Variation of learning time per trajectory.

The minimum separation distance threshold T parameter affects the number of derived route models and their widths. Smaller values of T lead to a large number of narrow route models that can be useful for motion prediction. For higher values of T, the learnt scene routes are fewer and wider and fit closer to a human interpretation of the scene. Figure 4.15 illustrates this variation of the scene route models.

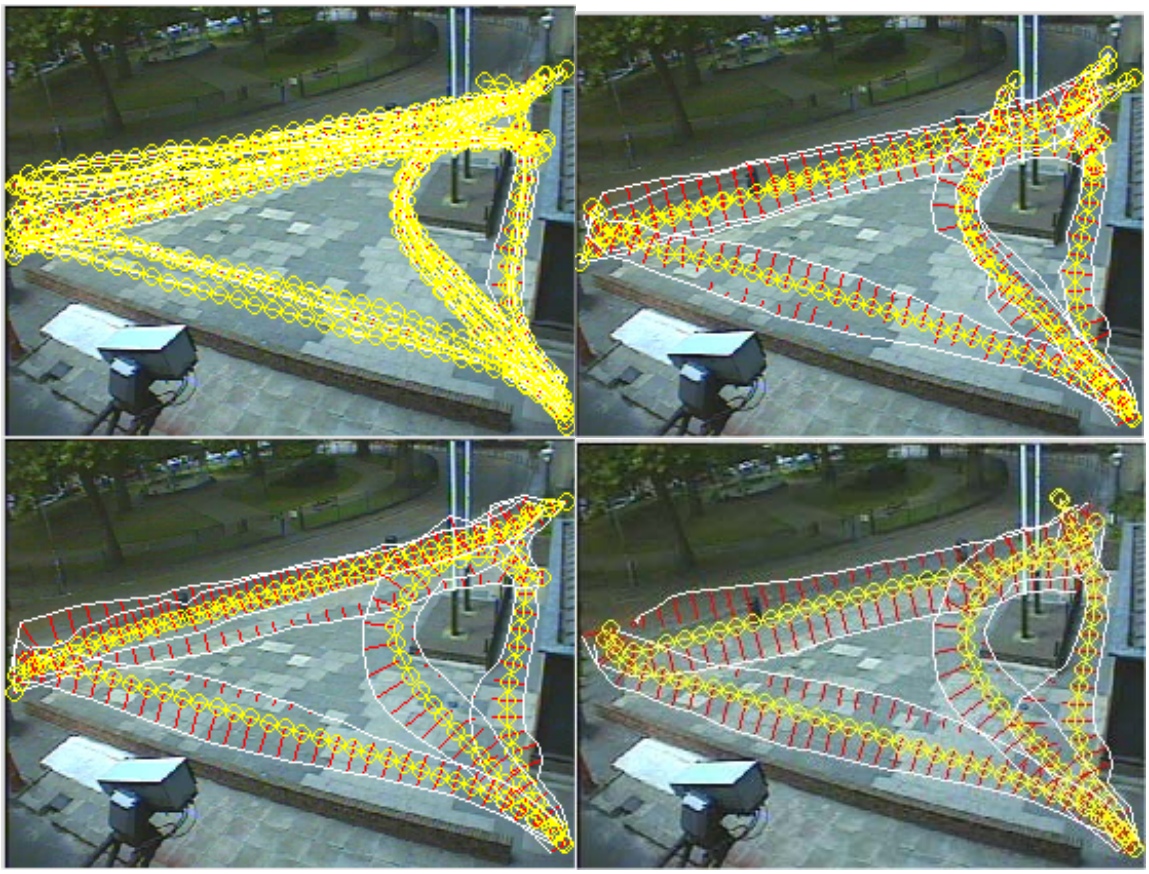


Figure 4.15: Route models derived for different values of minimum separation distance threshold T. Results are given for T=10, 20, 30, 40 and R=20. The number of accepted route models with usage above 1% is 19, 8, 5 and 4 respectively.

The route learning algorithm is auto-initialised and its initialisation depends on the order of the trajectories in the dataset. The sensitivity of the algorithm to the order of the trajectories in the training set was investigated and found to be low. Figure 4.16

illustrates different sets of route models, derived from the same training dataset with different (random) ordering of trajectories. Results differ slightly but their semantic interpretation (4-5 routes between 3-4 entry/exit zones) is equivalent.

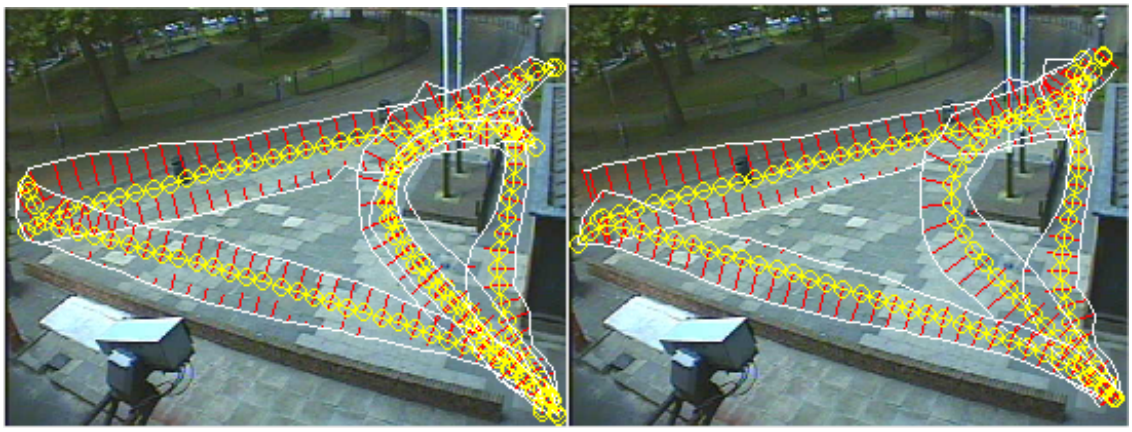


Figure 4.16: Results for different (random) order of the trajectory dataset. Results indicate that the algorithm is not sensitive to the order of the data (initialisation).

The route learning algorithm is demonstrated on two shorter video sequences that display pedestrian activity (Figure 4.17). From the first video sequence (Curtin video, resolution 768x576, 14 minutes, 2 frames/sec), 190 trajectories were extracted. The parameters of the algorithm were: the resample factor: R=40 pixels, distance threshold T=60 pixels. From the second video sequence (Northampton Square Video, resolution 384x288, 10 minutes, 2.5 frames/sec), 155 trajectories were extracted and the parameters of the algorithm were the following: R=10 pixels and T=20 pixels. The two scenes and the derived trajectory data are shown in Figure 4.17. No entry/exit zone information was used and the route extension was unrestricted.

Although the activity in the Curtin video is more complex, the performance of the algorithm is satisfactory, as indicated by the results (Figure 4.18). The Northampton Square video depicts the same scene with the Figure 4.10, from a different perspective and results are shown in Figure 4.19.

Figure 4.17: The video sequences that they have used for learning routes. The first video (top, Curtin) was captured in university of Curtin, in Australia. The second video (bottom, Northampton Square) images the main entrance of the City University. The trajectory dataset is depicted at the right.

Figure 4.18: The main axes of the extracted route models is shown on the top-left image. The most popular route models are shown on the rest. Notice at the bottom right image that a real scene feature (a bollard) separates the two detected routes.

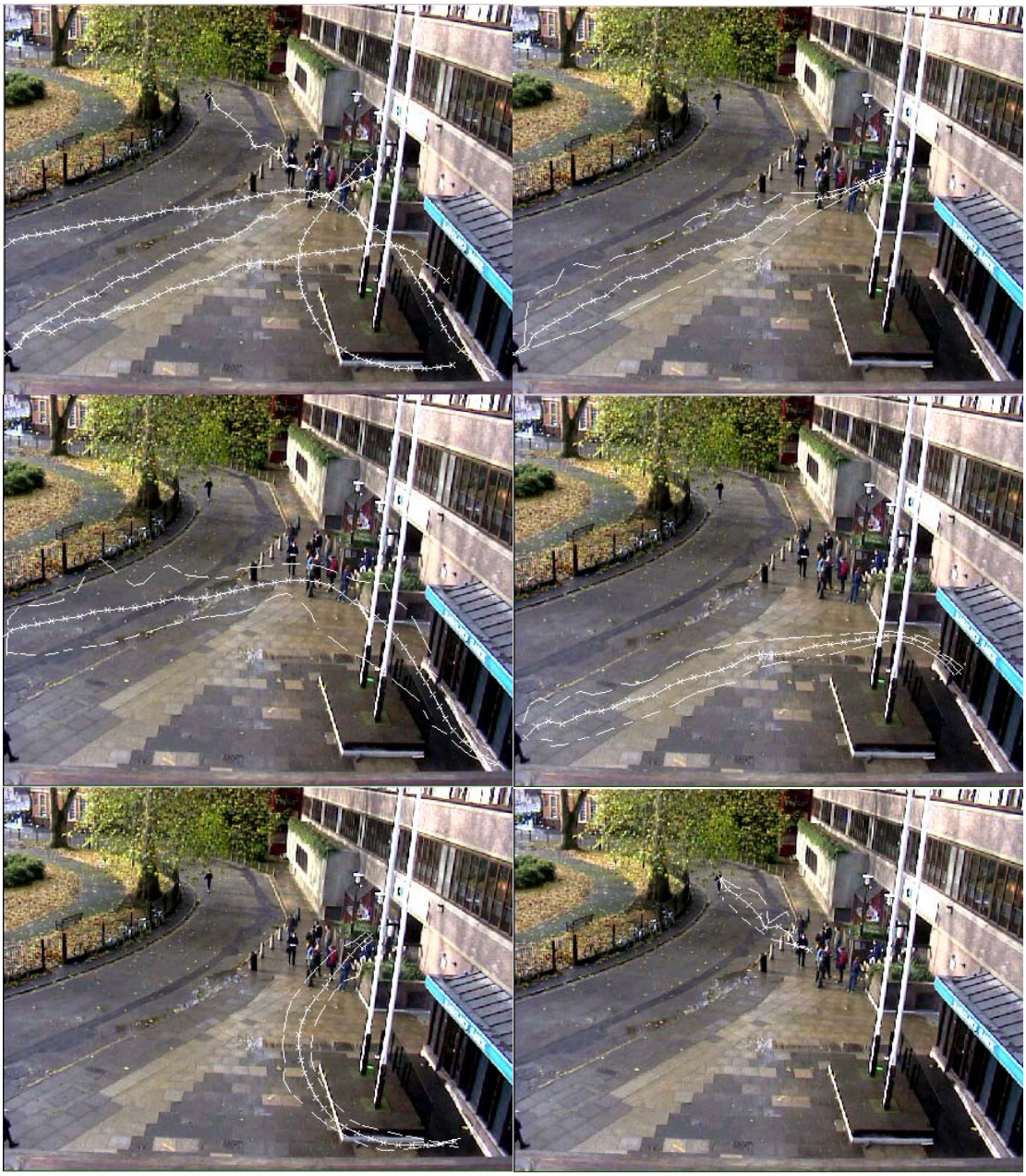


Figure 4.19: The main axes of the extracted route models is shown on the top-left image. The most popular route models are shown on the rest.

4.6 Junctions - Paths

Intuitively, a junction is the area where two routes cross. A more rigorous definition is adopted by this thesis: a junction is the region of intersection of two routes, where route directions differ by more than an angle ω. This definition reflects the fact that while a target is on a junction, some uncertainty is raised about its future direction.

Paths are considered as route parts in between junctions and/or entry/exit zones. Paths may also relate to route overlapping. If route parts are overlapped and the route directions are similar along the overlapped route parts, their union represents a path. For instance, the upper path of Figure 4.22 is formed by two route parts with similar direction.

Accumulative statistics could be used to identify the areas where target directions are similar (paths) or different (junctions). However, from the above definitions, it is concluded that junctions and paths are closely related to the geometry of the scene route models; therefore, they can be easily extracted from a set of route models.

Let $\mathbf{E_{ki}}$ be the area of the node i of the route model k, as defined in §4.2. All nodes of the scene route models are checked for whether they are part of junctions, according to the following criteria:

$$\mathbf{E_{ki}} \cap \mathbf{E_{lj}} \neq \{\} \tag{4.18}$$

$$abs(\mathbf{n_{ki}} \cdot \mathbf{n_{lj}}) < \cos\omega \tag{4.19}$$

where ω sets a decision threshold angle for the direction change. If both criteria are fulfilled for a pair of nodes i and j, belonging to route models k and l respectively (k≠l), then a junction area is $\mathbf{J_{ki,lj}}$ which is related to the two nodes, is set as the intersection of the two nodes:

$$\mathbf{J_{ki,lj}} = \mathbf{E_{ki}} \cap \mathbf{E_{lj}} \tag{4.20}$$

A junction $\mathbf{J_{kl}}$ of two routes k and l is defined as:

$$\mathbf{J_{kl}} = \bigcup_i \bigcup_j \mathbf{J_{ki,lj}} \tag{4.21}$$

The route models of Figure 4.11 are visualised as polygons in Figure 4.20. Junctions are detected for ω=5º and ω=30º and results are shown in Figure 4.21. As can be seen, larger values of ω provide a more conservative estimate of junctions. Junctions are used to split the route models to paths. Figure 4.22 visualises the segmentation of routes to paths and junctions.

If a junction $\mathbf{J_{kl}}$, as defined by eq.4.20, consists of unconnected components, each component defines an individual junction element. For example, the green and yellow routes of Figure 4.20 cross each other in two different areas, as it can be seen in Figure 4.22. Therefore, two individual junctions are defined.

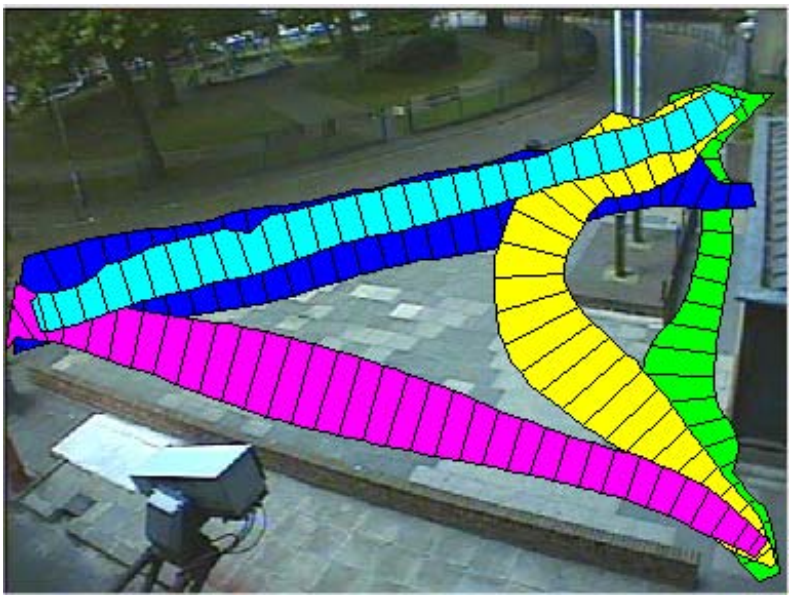


Figure 4.20: Five routes detected by the route learning algorithm.

Figure 4.21: Junction estimation, using ω=5º (left) and ω=30º (right).

The partitioning of routes to paths and junctions is performed not only to identify semantic features, but also to support activity analysis. For example, entry/exit zones, paths and junctions can be considered as primitives of routes and rare, complicated routes can be described as sequences of these primitives. Also, junctions are regions of interest for a long-term prediction module, because a target's motion within junctions reflects the target's intention to move towards specific regions.

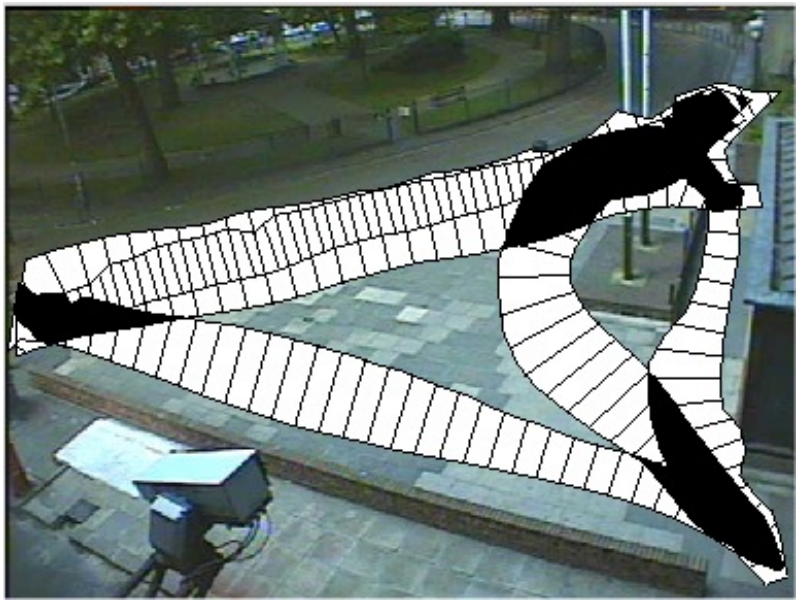


Figure 4.22: Segmentation of the scene route models to paths (white areas) and junctions (black areas).

4.7 Discussion

A route model is introduced that aims to represent both the spatial extent and the probabilistic distribution of similar trajectories. A route learning algorithm is proposed that is unsupervised and auto-initialised. Additionally, the algorithm has only two parameters; therefore, it can be easily configured.

One of the main dilemmas in developing the algorithm was whether to allow route extension or not. Route extension allows the algorithm to handle a trajectory dataset consisting of incomplete trajectories. On the other hand, the extension mechanism may extend routes in an unrepresentative way. To overcome the dilemma, trajectories can be validated using the entry/exit zones of the scene and the extension mechanism can be eliminated.

Sometimes it is desirable to provide an algorithm that not only learns the routes, but also adapts to possible variation of the routes, due to temporal or environmental changes. Some mechanisms of the route learning algorithm, like the extension and the definition of the bound points, do not allow the algorithm to operate adaptively. However, since the extension step can be skipped, if the training dataset is consistent with the scene entry/exit zones and bound points can be determined using the standard deviation of the

observations across the route, the algorithm will function without significant loss of accuracy (see Figure 4.13). Some minor modifications (for example introducing an adaptive learning rate parameter to the Eq.4.7a-e) should allow the algorithm to be used online and to adapt to the scene route models.

Further validation of the learning algorithm is given in the chapter 6. The next chapter discusses how activity can be analysed using the scene model. For this reason, a HMM is superimposed on the set of the route models.

Chapter 5

Activity Modelling and Analysis

5.1 Introduction

This chapter illustrates how the scene model is used for activity analysis. As stated previously, the observed activity in a scene is closely related to the scene. Actually, the structure of the scene explicitly or implicitly affects the activity. In the previous two chapters, a reverse engineering approach was used to determine a semantic scene model using patterns of observed activity. This chapter does not attempt to develop a new activity model or to go deep in details of activity analysis methods. It rather demonstrates the potential of using the scene model in activity modelling and analysis. In particular, it illustrates how the discrete and probabilistic character of the scene model can be exploited by overlaying a probabilistic network onto the scene model.

Activity is analysed in terms of the semantic scene model. For instance, activity can be annotated according to the scene semantics and high-level descriptors can be used for a context-based surveillance database. Individual targets activities can be characterised as typical or atypical and an automatic alarm system can relieve the operators. Also, long-term predictions can be generated to reveal the target intentions.

The motivation for the scene-based activity analysis is the fact that humans usually interpret the activity in relationship to scene features. This approach has been used in many surveillance applications, although in most cases the scene features are manually labelled.

The chapter is organised as follows: A summary of previous work on activity analysis is given in §5.1.1. The proposed scene-based activity model is presented in §5.2. The requirement for time-variant activity models is discussed in §5.3. In §5.4, it is

illustrated how the proposed models are used for applications like long-term prediction and atypical activity detection. The chapter concludes with the discussion section §5.5.

5.1.1 Previous Work

Many researchers have used a variety of probabilistic networks to model and analyse activities. For instance, Buxton and Gong [21] proposed that Bayesian Belief Networks (BBNs) can be used for activity analysis. The nodes of the network are related to both mid-level trajectory data (position, speed, size) and high-level scene descriptors. The scene descriptors are ground-plane geometric primitives, related to the semantics of the scene [43] and they are defined manually [44]. A similar approach is taken by Cupillard et al [24]. In their framework, specific types of abnormal activities, such as "Fighting", "Blocking", "Vandalism" and "Overcrowding" are described by "scenarios" that are defined either by AND/OR trees or BBNs. The nodes of the networks are associated to either "states" (sets of specific values of properties of single/multiple targets), or "events" (change of "state"), or sub-scenarios. However, in both approaches, the structure of the BBNs must be defined manually and the BBN parameters are determined through supervised learning.

The Hidden Markov Model (HMM), a different type of probabilistic network, has attracted much interest. Bobick [14] and Bobick and Ivanov [15] propose a HMM analysis of activity recognition of single targets. Analysis of interaction between targets is performed with Coupled HMM [17], proposed by Brand et al. Variable Length HMM (VLHMM) have been proposed by Galata et al [38] to deal with the cases where variable memory is required for optimal activity prediction and recognition, in contrast to the limitation of the constant memory-length that is imposed by traditional n-order HMMs. The states of the VLHMM are automatically learnt using a vector quantisation (VQ) method [53]. For wide-area surveillance systems, scalability of the activity models is required, which can be addressed using a hierarchical structure. For instance, the Abstract HMM (AHMM), proposed by Bui et al [19] and the Hierarchical HMM (HHMM), proposed by Luhr et al [63] aim to model activity observed from multiple camera surveillance systems that are distributed in structured scenes. Although a HMM and its

special variants have been proved quite efficient to analyse activity, HMM training is an issue. More specifically, the complexity of the HMM training methods that assume that the states are unknown is exponentially increased by the number of the states [80]. This chapter does not attempt to introduce another type of HMM; it rather exploits the scene models that were learnt in previous chapters and uses them as the basis of HMM. This approach allows fast learning of the HMM parameters and association of the HMM states with semantic features of the scene.

In all the above approaches, model learning and activity analysis is based on available trajectory data. Some researchers have proposed activity analysis methods that do not require the knowledge of complete trajectories. For instance, Boyd et al [16], proposed a method where scene activity is represented by a source-destination network traffic model. A statistical network model of the activity is constructed not by tracking individual targets, but by counting transitions between manually defined areas and applying network tomography [99]. The benefit of the method is that it is useful for systems that are not able to reliably derive full trajectories, but that have some limited capability to track targets as they transit between areas. However, in this case, the overall activity of the scene can be analysed, but not activities related to individual targets.

Pixel-based activity models have also been proposed. Pixel level event detection and semantic understanding, without the requirement of explicit target tracking is proposed by Jeffrey and Gong [75] and Xiang et al [104]. A GMM representation of the scene activity is suggested in [75], which is automatically learnt using EM and MDL [104]. However, pixel-based methods do not benefit from the history of target motion that may be crucial for activity analysis.

A review of generative models that have been used for learning and understanding activity was present by Buxton [22]. A unified view of these models, based on [86], is presented in Table 5.1.

Initial	Extension	Final
Gaussian	Mixture	VQ
Gaussian	reduce dimension	Principal Component Analysis (PCA)
VQ	Dynamic	HMM
PCA	Independence	Independent Component Analysis (ICA)
HMM	Coupling	CHMM
HMM	variable length	VLHMM
ICA	Hierarchy	BBN
HMM	Hierarchy	AHMM, HHMM
BBN	Dynamic	DBN
DBN	Utility	DDN

Table 5.1: Relationships between probabilistic models that have been used for activity learning and understanding (adapted from [22]).

5.2 Scene-based activity modelling

Target activity can be related to elements of the scene model. For example, a target will enter the scene by an entry zone, follow a path, then reach a junction and take another path, stop at a stop zone and finally exit the scene through an exit zone. If it is detected outside the scene model, its activity will be characterised as atypical.

The discrete character of the scene model, as illustrated in a topological representation, allows discrete-state models like Markov Chains and Hidden Markov Models (HMM) to be applied. Both markovian models can be used for activity modelling and analysis. The suggested approach is HMM-like for two reasons: Firstly, HMMs are considered more powerful, because they distinguish observations and states and model the uncertainty of correspondences of observations to states using membership functions. Secondly, the probabilistic nature of each of the scene elements allows the required membership functions to be easily determined for each of the states of the model.

Two network representations can be derived by the scene model. The first consists of scene elements like entry/exit zones, paths, junctions and stop zones. The second consists of all the nodes of all the scene route models. HMMs can be overlaid onto both types of network. In this chapter, the former approach of a route-based model (see Figure 4.1) is investigated, while the next chapter illustrates an application of an entry/exit zone-based model.

5.2.1 Route-based Hidden Markov Model (RBHMM)

The states of a route-based Hidden Markov Model (RBHMM) are defined to be the nodes of all the accepted route models, plus two extra states: an "out-of-any-node state", which indicates activity outside the modelled routes and an "end state", which indicates the end of the observation. It is sensible to derive the nodes from uni-directional routes, so that directionality information is incorporated at each node.

Let assume that a scene contains W route models and each route model w=1..W consists of L_w nodes. The number of the states N of the RBHMM is given by the following formula:

$$N = 2 + \sum_{w=1}^{W} L_w \tag{5.1}$$

The elements of the RBHMM are:

- **S**=$\{S_i\}$, i=1..N, the set of states.
- **Q**=$\{q_k\}$, k=1..M, the sequence of the states.
- **A**=$\{a_{ij}\}$, i,j=1..N, the transition probability distribution, where $a_{ij}=P(q_{t+1}=S_j \mid q_t=S_i)$.
- $\boldsymbol{\pi}=\{\pi_i,\}$ i=1..N, the initial state distribution, where $\pi_i=P(q_1=S_j)$.
- **O**=$\{O_k\}$, k=1..M, the sequence of the observations.
- **B**=$[b_i(v)]$, i=1..N, v a position vector and $b_i(v)$ is the membership function of the observation v to the state i.

The HMM parameters are generally recommended to be learnt using iterative algorithms [80]. However, because of the large number of states, these algorithms are very slow and often impractical, especially for online learning. Instead, we use the pdf distributions of observations across the routes (see §4.2) to encode the observation vector **B** (Eq.5.2). Then the RBHMM parameters are estimated cumulatively using Eq.5.3 and Eq.5.4.

$$b_i(O_{l.k}) = \frac{g_i(O_{l.k})}{\sum_{j}^{N} g_j(O_{l.k})} \tag{5.2}$$

$$\pi_i = \frac{\sum_{l=1}^{L} b_i(O_{l,1})}{L} \tag{5.3}$$

$$a_{ij} = \frac{\left(\sum_{l=1}^{L} \sum_{k=1}^{K_L} b_i(O_{l,k}) \cdot b_j(O_{l,k+1}) \right)}{\sum_{l=1}^{L} \sum_{k=1}^{K_L} b_i(O_{l,k})} \quad (5.4)$$

where $O_{l,k}$ is the *k*th observation from the *l*th trajectory *l*=1..L, k=1..K_L and $g_i(O_{l,k})$ is the estimate of the probability that the observation $O_{l,k}$ corresponds to the state i, according to the associated learnt cross-route distribution.

5.3 Long term variations

The HMM that we described in the previous section is built under the assumption that the trajectories are generated by a stationary stochastic process. In many cases, this assumption is invalid, as the target activity may vary depending on the time of day. For instance, for the scene of Figure 5.1 which is the entrance of the University, we expect that in general, between 8am and 10am, most pedestrians will be walking to the entrance of University; from 1pm to 2pm, they will be wandering around outside the entrance; at 5pm many will be leaving the University and at midnight, hardly anyone will be around.

However, we can assume that target behaviour remains the same for short periods of time (e.g., 15 min-1 hour) and this pattern is repeated every weekday at the same time. In addition, we assume that this period of time is long compared to the lifetime of a trajectory. Therefore, we can establish a static HMM model for each time slot. It is obvious that such models need learning data from more than one day, so sufficient observations are provided for estimating the models of each time slot.

In Eq.5.3 the initial probability π_i of the state i has been defined over the sum of the observed trajectories. However, a more reasonable approach is to multiply this relative probability by the rate of pedestrians' appearances (*M/T)* to derive an absolute measure of the probability of a new appearance over time: $\pi_i' = \pi_i \cdot \frac{M}{T}$. To illustrate the advantage of π_i' over π_i, let's consider the case of 100 out of a total of 1000 pedestrians starting their route from node i, between 10am-11am and 1 out of a total of 10 pedestrians starting from node i, between 10pm-11pm. In both cases, π_i takes the 0.1

value, but this fails to indicate the fact that a pedestrian appearing so late is anyway atypical, which is indicated by using π_i'.

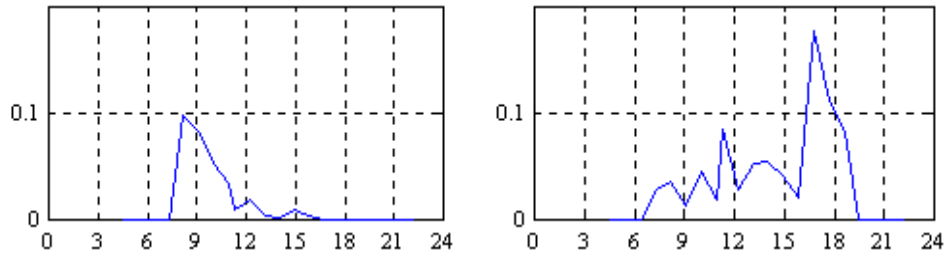


Figure 5.1: Initial probability π_i of the two terminator nodes of the route in Figure 4.8, during a 24-hour period. x-axis indicated the time of the day. At 8-9am, almost all trajectories occur towards the entrance of the University, whilst around 5pm, people tend to leave.

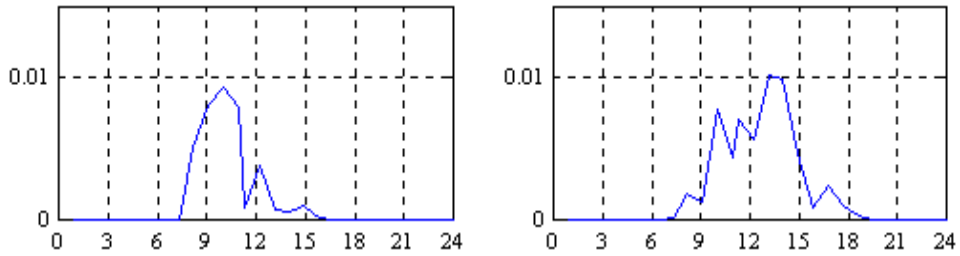


Figure 5.2: Initially probability π_i' over time for the two terminator nodes of the route in Fig.4. The peaks of the distributions have moved to 10am and 1-2pm, indicating that these are the most popular periods for people to come and leave the University respectively.

5.4 Activity Analysis

The activity of the scene is generally represented by a set of trajectories. However, this representation is mid-level and does not allow for analysis of any unseen activity. Therefore, it is desirable to provide a higher level of understanding of the activity. For instance, activity can be annotated in terms of the semantic features of the scene or it can be characterised as typical or atypical. It is also useful to provide a mechanism for activity prediction, based on previous observed activities. This section demonstrates how the proposed probabilistic models can be used for the prescribed applications.

5.4.1 Activity Labelling

Activity can be represented using a semantic, high-level representation, consistent to the scene model. For instance, the first/last points of a trajectory can be matched to an

entry/exit zone (see Appendix I) and the entire trajectory to one or more routes. Further characterisation can be based on the paths and the junctions that are related to the matched part of the route.

For instance, Figure 5.3 illustrates the labelling of 53 previous unseen trajectories, according to a route model shown in Figure 4.18. The colour coding uses solid colour lines (red, magenta, cyan, green and black) to identify trajectories that have been matched with known routes. Dashed deep blue lines (see bottom left) indicate trajectories that do not match any existing route model.

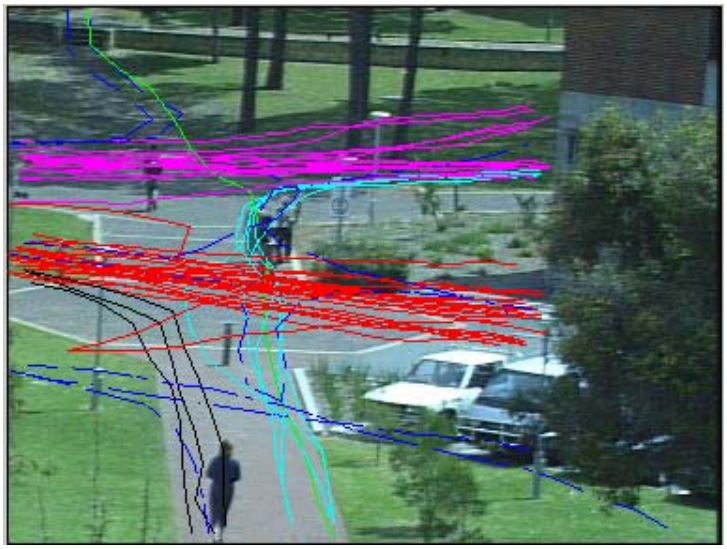


Figure 5.3: Trajectories labelling consistent to the route model of the scene.

5.4.2 Activity Prediction

Motion prediction is usually derived from the dynamics of a target, using linear prediction, Kalman filter [56] or particle filter [48] [49]. However, these approaches do not consider the general behaviour of the targets in a specific scene and they fail to provide reliable predictions about the position of the target after extended periods or the target's final destination.

Long-term predictions can be derived from the RBHMM which encodes the general behaviour of the targets within the scene. The probability of a state j after K time-steps, assuming the observation vector **O** is given by the following equation:

$$P(Q_{k+K} = S_j | O_1 O_2 ... O_k) = \sum_{i=1}^{N} \delta_k(i) \cdot a_{i,j}^{(K)} \quad (5.5)$$

where $a_{i,j}^{(K)}$ is the element {i,j} of the matrix A^K and $\delta_K(i)$, defined in the Appendix III, represents the maximum probability of the state i at the time-step K, given the observation vector.

In order to derive predictions about the destination of a target, (in other words the probability that the target will exit the scene at node j), HMM theory cannot provide a well-defined solution. This is because it must consider the infinite number of different sequences of states that lead to termination of object at the node j.

An implicit estimation is provided by a route-level interpretation and by classifying the part of the trajectory that has been observed to route models. More specifically, a trajectory is dynamically classified online using the ML trajectory classifier (§4.4.2). For each possible matching of the incomplete trajectory to a route model, a probability is assigned according to Eq.4.14, which shows the likelihood that the trajectory matches the specific route.

Activity recognition is performed by classifying the object trajectory to one of the existing routes. A Boolean classifier based only on the geometric model is sufficient to classify trajectories that match only one route for their entire length, but it cannot classify incomplete trajectories that entirely lie on the common part of overlapped routes. In this case, the ML classifier provides a more appropriate interpretation.

Figure 5.4 illustrates a partial trajectory (white) and two candidate routes (red and magenta). A plot of the likelihood of each route is shown, as derived by the MLTC. In the beginning of the sequence, the red route dominates due to its high popularity, but gradually the magenta route is matched, as the target moves on.

In the case that an object jumps from one route to another, the trajectory cannot be classified to only one route as different parts correspond to different routes. In this case, all the posterior probabilities of the trajectory become zero at a point k, which indicates that single matching is impossible. The problem can be solved by splitting the trajectory at the point k. Then, the rest of the trajectory is matched to a different route.

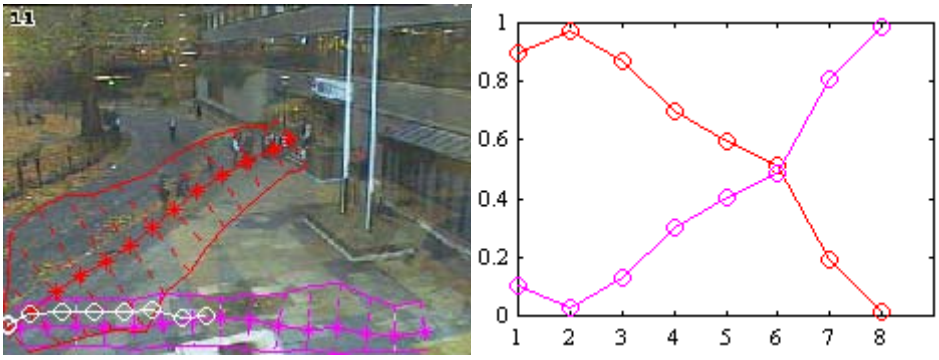


Figure 5.4: Classification of an incomplete trajectory (white) online by the ML classifier. The graph shows a plot of likelihood against time (in seconds) for matching the trajectory to the two route models.

5.4.3 Suspicious activity detection

Surveillance systems are required to provide an attention mechanism for "suspicious"/"dangerous" activities that will alarm operators. The benefit of such a capability is obvious, as the operators are not required to monitor the scene activity continuously, but they can only respond to the alarms.

Both supervised and unsupervised methods have been proposed to solve the problem. Supervised methods, e.g., Foresti et al [62], are based on the classification of trajectories to "normal" and "dangerous". The drawbacks of the supervised approach are that model learning is not automatic and that a "dangerous" model cannot represent the infinitive variety of "dangerous" trajectories. Unsupervised methods, e.g., by Johnson [53], allow the assessment of the typicality of the trajectories according to their consistency to an activity model. Therefore, atypical trajectories may be "suspicious" but not necessarily "dangerous". The approach presented in this section is based on the typicality assessment.

The learnt route models represent the set of typical activities of the scene. Therefore, activities that are not consistent with the learnt route models can be characterised as atypical. In this section, three different approaches for atypical activity detection are discussed.

The first approach is based only on the spatial extent of the union of the route models. Route models determine where activity is expected; therefore any activity

outside the routes can be characterised as atypical. For instance, if an intruder climbs a wall, the system can identify the suspicious activity, because the wall is a region where no activity is expected. However, this approach cannot detect atypical activities that occur within the set of the route models, for instance a suspicious person wandering around the scene, within the spatial extent of the set of the routes, but not consistent to any route.

An alternative approach could be to characterise as atypical those trajectories that do not match any single route for their entire length. This means that a trajectory may be characterised atypical, even if it occurs within the union of the valid route models. Figure 5.5 illustrates such examples of unmatched atypical trajectories. However, this approach cannot distinguish between atypical activities and typical complex activities not describable by a single route, all of which may be occurring within the area defined by the union of the route models.

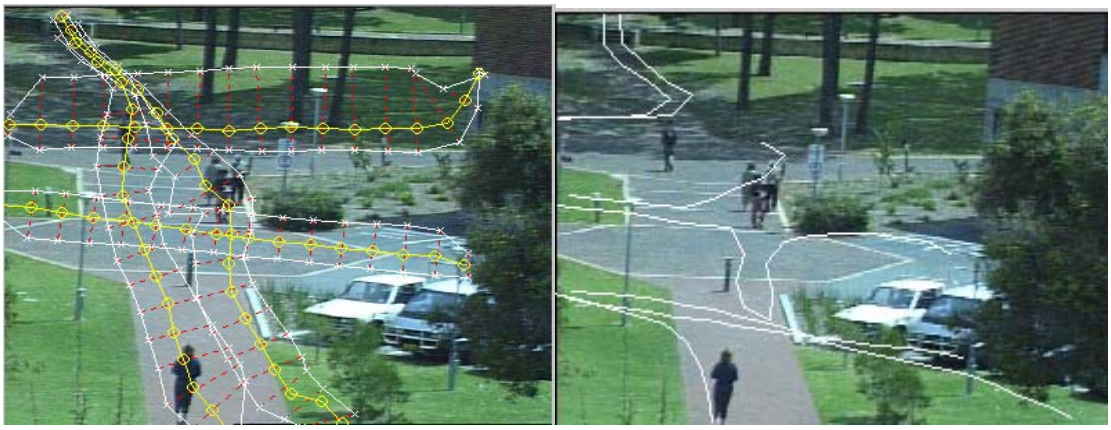


Figure 5.5: Unmatched trajectories (right) from a set of scene route models (left) are indicated as atypical.

Both approaches are based only on the spatial information of the route models and provide a Boolean mechanism to characterise activity. However, it is desirable to consider the statistics of the previously observed activity in order to provide a quantitative characterisation of the activity. To fulfil this requirement, atypicality detection is based on the HMM representation of the activity.

More specifically, the consistency of an observation vector **O** (trajectory) with a given HMM λ, is represented by the probability $P(\mathbf{O}|\lambda)$. The estimation of the above

probability is known as the evaluation problem, the first of the three basic problems in HMM theory. The solution, as described in [80], is given by the equation:

$$P(\mathbf{O} \mid \lambda) = \sum_{i=1}^{M} \alpha_M(i) \tag{5.6}$$

where M is the size of the observation vector **O** and α_M is the forward variable as explained in the Appendix III. The typicality criterion for the observation vector is given as:

$$l(\mathbf{O}) = \log\left(P(\mathbf{O}|\lambda)\right)/M \tag{5.7}$$

The typicality criterion l(**O**) is not related directly to the probability P(**O**|λ), but to its logarithm, because the probability P(**O**|λ), may have too low a value to be represented within the arithmetic range of the computer. Division by M is performed to normalise the criterion against the size of the vector. Atypicality is detected when l(**O**) is below a threshold.

While the criterion defined by Eq.5.7 indicates the typicality of an entire trajectory, the criterion defined in Eq.5.8 characterises a specific sample $\mathbf{O}_k$ of the trajectory:

$$l'(\mathbf{O}_k) = \log\left\{P\left(O_1 O_2 \ldots O_k \middle| \lambda\right)\right\} - \log\left\{P\left(O_1 O_2 \ldots O_{k-1} \middle| \lambda\right)\right\} \tag{5.8}$$

or equivalently:

$$l'(\mathbf{O}_k) = \log\left(\sum_{i=1}^{k} \alpha_k(i) - \sum_{i=1}^{k-1} \alpha_{k-1}(i)\right) \tag{5.9}$$

Figure 5.6 depicts three trajectories and Figure 5.7-Figure 5.8 present their evaluation according to the two criteria. The left trajectory is a common trajectory and this is verified by the values of the two criteria. (However, the same trajectory could be characterised as atypical, if it was observed late the night, according to §5.3). The middle trajectory is not so unusual, however it contains two samples (red circles) where the target speeds up. The criterion *l* estimates the overall typicality of the trajectory; therefore it does not find anything atypical. The criterion *l′* is able to detect the two suspicious samples. The right trajectory is a clearly atypical trajectory that could represent somebody climbing (red circle). The suspicious activity of the event is detected by both criteria.

Figure 5.6: Three trajectories are shown. The left trajectory is a very common one. The middle one contains two rather suspicious samples (red circles). The right one is a very uncommon one of somebody climbing. Actually the last trajectory is an error of the motion tracking algorithm.

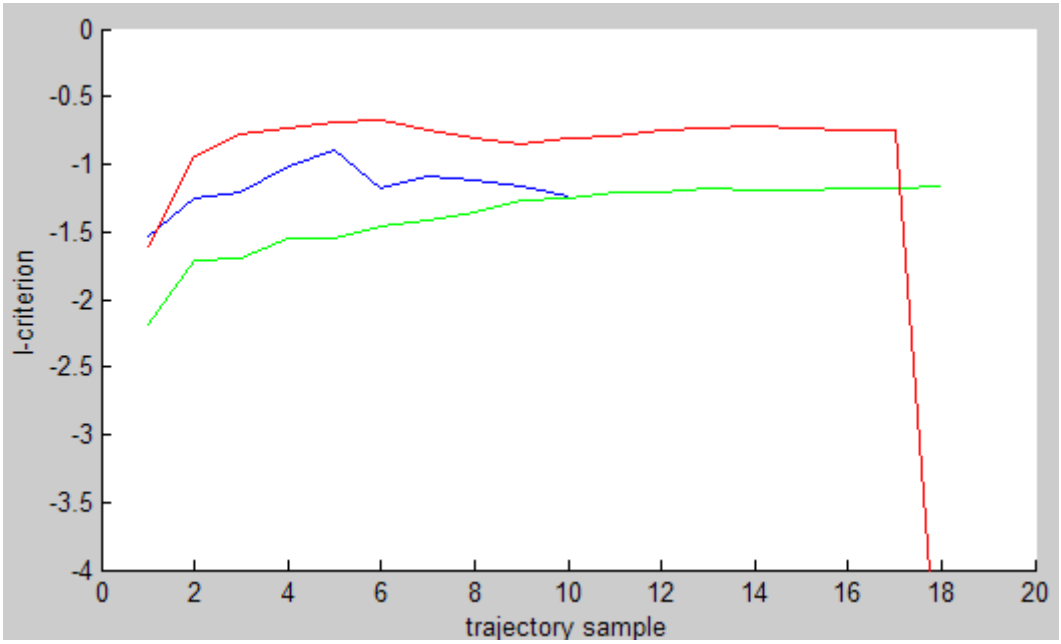


Figure 5.7: The evaluation of the three trajectories according to the *l* criterion. The left (green line) and the middle (blue line) trajectories of the Figure 5.6 are in general typical, while the right one (red line) is clearly atypical.

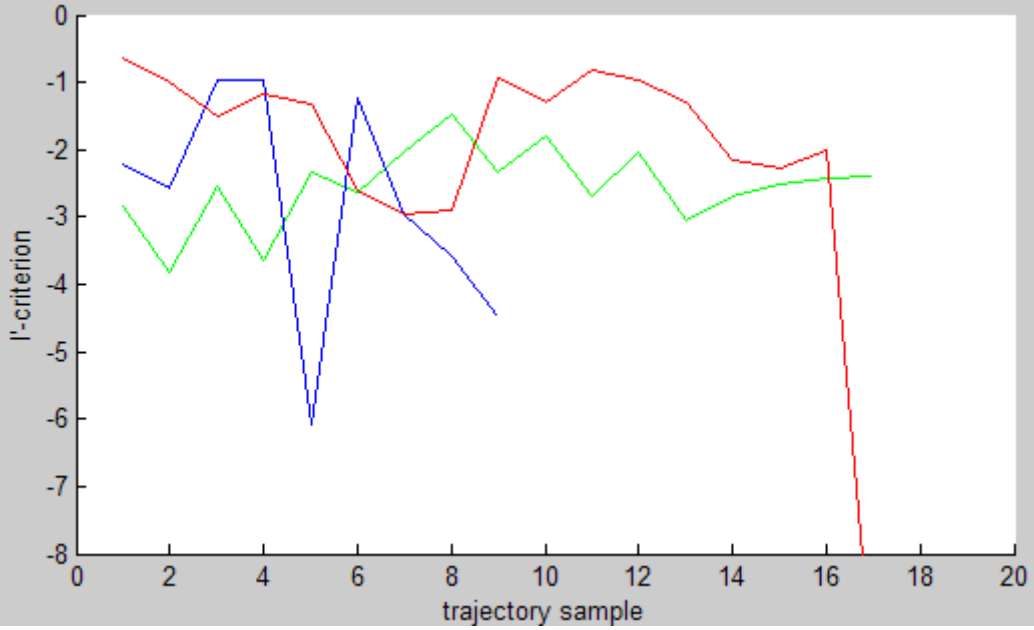


Figure 5.8: The three trajectories of Figure 5.6 are evaluated according to the *l'* criterion. All the samples of the left (green line) trajectory are typical; two of the samples of the middle (blue line) trajectory are characterized as atypical; the climbing sample of the right (red line) is clearly atypical.

5.5 Discussion

RBHMM is not another special type of HMM and the states of the proposed RBHMM are not really "hidden", in the sense that both the states and their membership functions are known from the semantic scene model. The RBHMM is characterised as a HMM because it is described by the HMM equations.

The benefits of using RBHMM are that its parameters can be easily learnt and that it provides human-like interpretation of activity in terms of scene features. The motivation for the development of the RBHMM is that it exploits the semantic scene model. Theoretically, any special variant of HMM can be overlaid on the scene model.

Activities are characterised as typical or atypical depending on how consistent they are to the RBHMM activity model of the scene. Characterisation of an activity as "suspicious" or "dangerous" implies the existence of a class of "suspicious"/"dangerous" activities. However, training such a model requires supervised learning, which exploits the human justification of the activities [62]. But even in this case, it is not guaranteed that the whole range of the "suspicious" activities will be represented.

Another issue with atypical activity detection is that erroneous trajectories are characterised as atypical. For example, the third trajectory of Figure 5.6 is actually an error of the motion tracking algorithm. This is a real problem, as any motion tracking error will be detected as atypical activity that will ring the alarm and unfortunately, such errors are common and unavoidable in real surveillance systems. Erroneous trajectories could be filtered, e.g., using the semantic scene model. However, in this case, the atypical activities would be filtered too and the surveillance system would become useless.

The next chapter discusses how to integrate scene models from a network of multiple cameras and how target motion can be modelled in the unseen areas of the network. To deal with these challenges, an activity model, based on the detected entry/exit zones, is proposed, which is derived without using correspondence information.

Chapter 6

Learning in Multi-Camera Surveillance

6.1 Introduction

The learning methods for the scene and activity models that were presented in the previous chapters were all demonstrated on single camera views. However, the majority of real surveillance systems consist of multiple cameras that cover a wide area. This chapter investigates the applicability of these methods in a multi-camera surveillance system. It also discusses the problem of integration of information derived from multiple cameras.

A case study of a multiple camera surveillance system is presented. Learning methods are validated by results on all the camera views of the system. Activity and scene models are also learnt on a common ground plane coordinate system. The ground plane model allows integration of the models derived from multiple cameras.

However, the ground plane model assumes that all the scene activity is co-planar, which is not always true. In addition, the ground plane model is based on geometric camera models that are derived by calibration.

This thesis proposes an alternative model of information integration that can be derived automatically. An activity model that covers the entire scene observed by multiple cameras is learnt by a correspondence-free method. This methodology is used to automatically estimate the camera topology and to integrate image-based scene models in temporal terms. Because the model represents the activity even across the gaps in between the camera views, it can be used to support tracking across these "blind" areas.

Section §6.2 reviews previous work related with this chapter. Section §6.3 discusses how a scene is viewed by a multi-camera system. Semantic scene models are learnt for the City University multiple camera surveillance system in section §6.4.

Section §6.5 introduces an activity model that covers the entire scene and an appropriate correspondence-free learning method. The chapter concludes with the discussion presented in section §6.6.

6.2 Previous Work

The ground plane constraint (GPC), introduced by Tan et al [96], is the method of information integration that is used by the majority of multiple camera surveillance. The ground plane (GP) model assumes that all the targets move on a plane and the cameras models are determined with respect to this plane, usually by manual calibration methods [98].

Automatic calibration of a single camera with respect to the ground plane is proposed by Renno et al [82]. Calibration is performed by measuring the heights of pedestrians and applying a linear model that relates these heights with the pedestrians' distances from the camera. However, their method requires the knowledge of the height of the camera and cannot be applied in all circumstances, e.g., not for top-down camera views. Also, the ground plane constraint is not always valid, especially for surveillance systems that cover wide areas.

Stein [93] and Black and Ellis [9] proposed a homography-based method to associate two camera views that are substantially overlapped, which also implies that all the activity is coplanar. The method compares the observations derived from the two camera views and uses the Least Median of Squares (LMS) algorithm to estimate a homography matrix that defines the relationship of the two camera views. Stauffer and Tieu [91] extended the method and proposed the Tracking Correspondence Model (TCM). Their method can automatically estimate which of the cameras have overlapped views and establish common TCMs for the clusters of the cameras that are linked through overlapped views. However, cameras or clusters of cameras that do not have overlapped views are treated as isolated and no model is proposed to link them.

Camera views can be related not only by their geometric relationships, but also by their activities. Kettnaker and Zabih [57] proposed a Bayesian framework for linking camera views using target information like transition times, transition probabilities and

appearance cues. The advantage of this framework is that allows linking of non-overlapped views. However, they do not define how the Bayesian Belief Network (BBN) can be trained. Javed et al [51] proposed that transition times and relationships between colour appearances in different cameras could be learnt using supervised learning. They have used an approach of target feature matching to track pedestrians across multiple cameras, learning a typical track's spatio-temporal transition probability using a Parzen estimator. Individual tracks are corresponded by maximising the posterior probability of the spatio-temporal and colour appearance, adapted to account for changes between cameras. The transition probabilities are learnt using a small number of manually labelled trajectories. But supervised learning is not assumed suitable for auto-calibrating surveillance systems. Porikli and Divakaran [79] proposed an unsupervised method of automatic colour calibration between cameras based on the colour appearances of matched targets. The method estimates conditional probabilities of colour appearances and formulates a Bayesian Belief Network. However, it is not always possible to establish such colour correspondences, e.g., a surveillance system may consist of cameras of different image modalities such as monochrome and/or thermal. In this case, colour information is not available at all.

In [16], a network-based activity model is constructed using network tomography [99]. This method constructs a network representation of the scene activity. It requires only counting the target transitions through the borders of manually defined areas of the scene. Therefore, correspondence is required only around the borders. However, it is not an absolute correspondence-free method and it cannot be used in areas where no correspondence can be established, such as the gaps in between cameras.

The method presented here (§6.5, based on [29] and [70]) shares similarities with the Amari [1]. Amari proposed information geometry to be used to derive the structure of a high-order Markov chain that could derive human brain structure from detected neuron triggers. The method does not use any correspondence information. It just attempts to statistically correlate the signals that appeared at the different nodes of the model.

Wren and Rao [103] proposed a method that can reveal the topography of the camera views concurrently with Ellis et al [29]. Both methods are correspondence free and just attempt to correlate events. Wren and Rao correlate target detections within

camera views, while Ellis et al correlate appearances and disappearances at entry/exit zones. However, Wren and Rao's method is able only to identify the topology of the cameras, whilst the method presented in this chapter identifies the distance between non-overlapped cameras and provides an activity model for the “gaps” of the virtual field of view (FOV).

6.3 Multi-camera Surveillance Systems

Multi-camera surveillance systems cover wide-area scenes and aim to track targets within this scene. The key issue in these systems is to effectively integrate information from multiple cameras in order to provide complete histories of targets' activities within the observed scene.

To integrate information in the spatial domain, a world coordinate system is required. Usually, a ground plane coordinate system [96] is used which is consistent with the assumption that all the scene activity is coplanar. A ground plane map is used to illustrate results in ground plane coordinates. For example, Figure 6.1 shows a synthetic ground map, while Figure 6.6 illustrates a ground map constructed from geometric calibration models and views from six cameras.

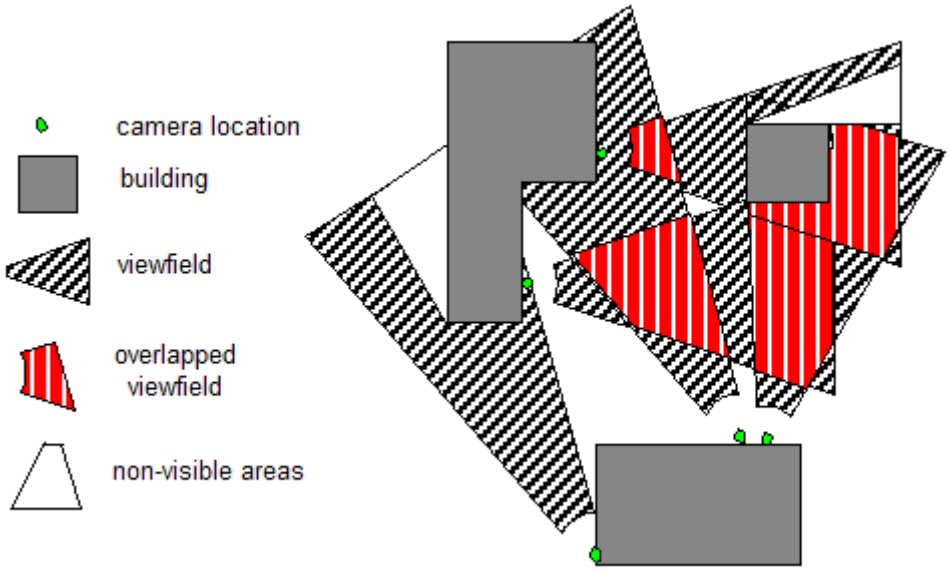


Figure 6.1: Visible FOVs of a network of cameras.

Four different field of views (FOV) are defined on the ground plane, to determine how the scene is viewed by a multiple camera surveillance system:

– Visible FOV defines the regions that a single camera images, excluding occluded areas and obscure areas where activity cannot be interpreted.

– Camera FOV encompasses all the regions within the camera view, including occluded regions.

– Network FOV is the union of all the visible FOVs of all the cameras of the network.

– Virtual FOV covers the network FOV and all the gaps in between the visible camera FOVs, within which targets may exist.

The visible FOVs of different cameras are related with different types of associations, depending on their spatial locations and how activity is seen. Specifically, two FOVs may be a) overlapped, i.e. they have common parts and a target may be seen simultaneously by the two cameras, b) adjacent, i.e. they do not have common parts, but they are quite close so targets exiting the one FOV may enter the other FOV, c) distant, i.e. they are far apart and there is no direct relationship of the targets activities in these two views. This thesis uses the term "camera network topology" to represent the set of all the relationships of the cameras of the network.

6.4 A multi-camera case study

The methods that were described in chapters 3 and 4 are applied in a specific multi-camera surveillance system, the City University Experimental Surveillance (CUES) system. Entry/exit zones and route models are learnt and represented on both individual image planes and a common ground plane.

The CUES system consists of six cameras that cover both pedestrian environments and traffic roads. The camera views of this system are shown in Figure 6.2. Each camera tracks targets within its own visible FOV and tracking information is passed to a central database [28].

The system has been geometrically calibrated with respect to a common ground plane coordinate system, using the Tsai algorithm [98]. A ground map (see Figure 6.6) has been constructed that shows the cameras FOVs and their relative location.

Figure 6.2 shows the views of the six cameras of the network (left column) and trajectory datasets that were derived by the motion tracking modules, attached to each camera (middle column). The trajectory datasets were derived during a 13-hour daylight period and they consist of 8417, 9985, 6799, 3060, 5022 and 15272 tracks for the cameras 1-6 respectively. Figure 6.3 and Figure 6.4 illustrates the derived entry/exit zones for the six cameras, according to the methodology described in §3.3. Entry/exit zones are used to remove noisy trajectories, as explained in §3.3.2. The cleaned trajectory datasets consist of 4282, 5579, 3853, 1922, 4230 and 10600 tracks for the cameras 1-6 respectively (Figure 6.2, right column).

Sets of 256, 500, 44, 500, 298 and 500 tracks (for the cameras 1-6 respectively) from the cleaned trajectory datasets were used as input to the route learning algorithm (§4.3). Figure 6.5 shows the results plotted onto the six camera views.

Figure 6.2: Views of the six cameras (left), trajectory datasets (middle) and cleaned trajectory datasets (right).

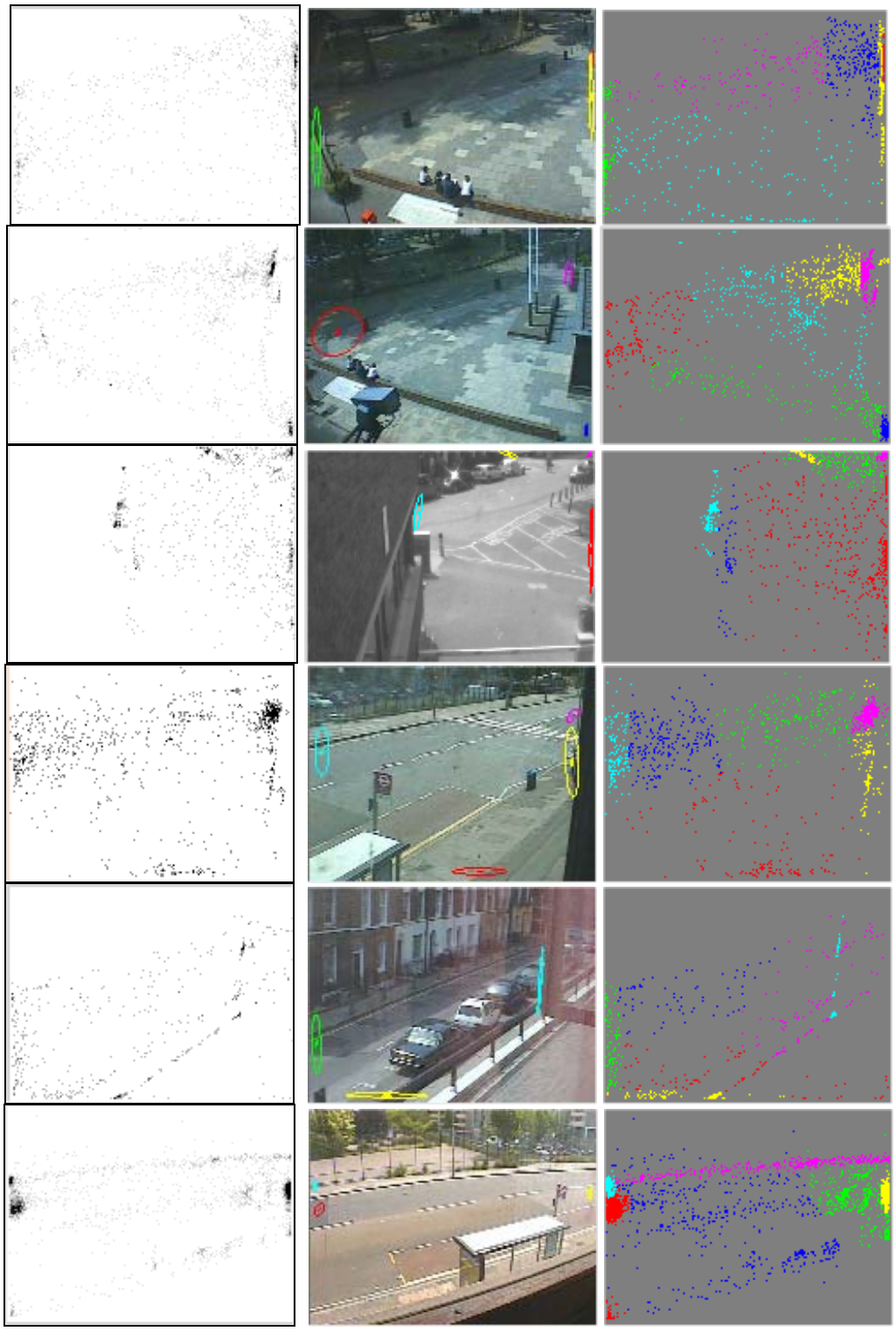


Figure 6.3: Detection of entry zones on image plane: Histograms of entry points (left), detected entry zones (middle) and classified points (right).

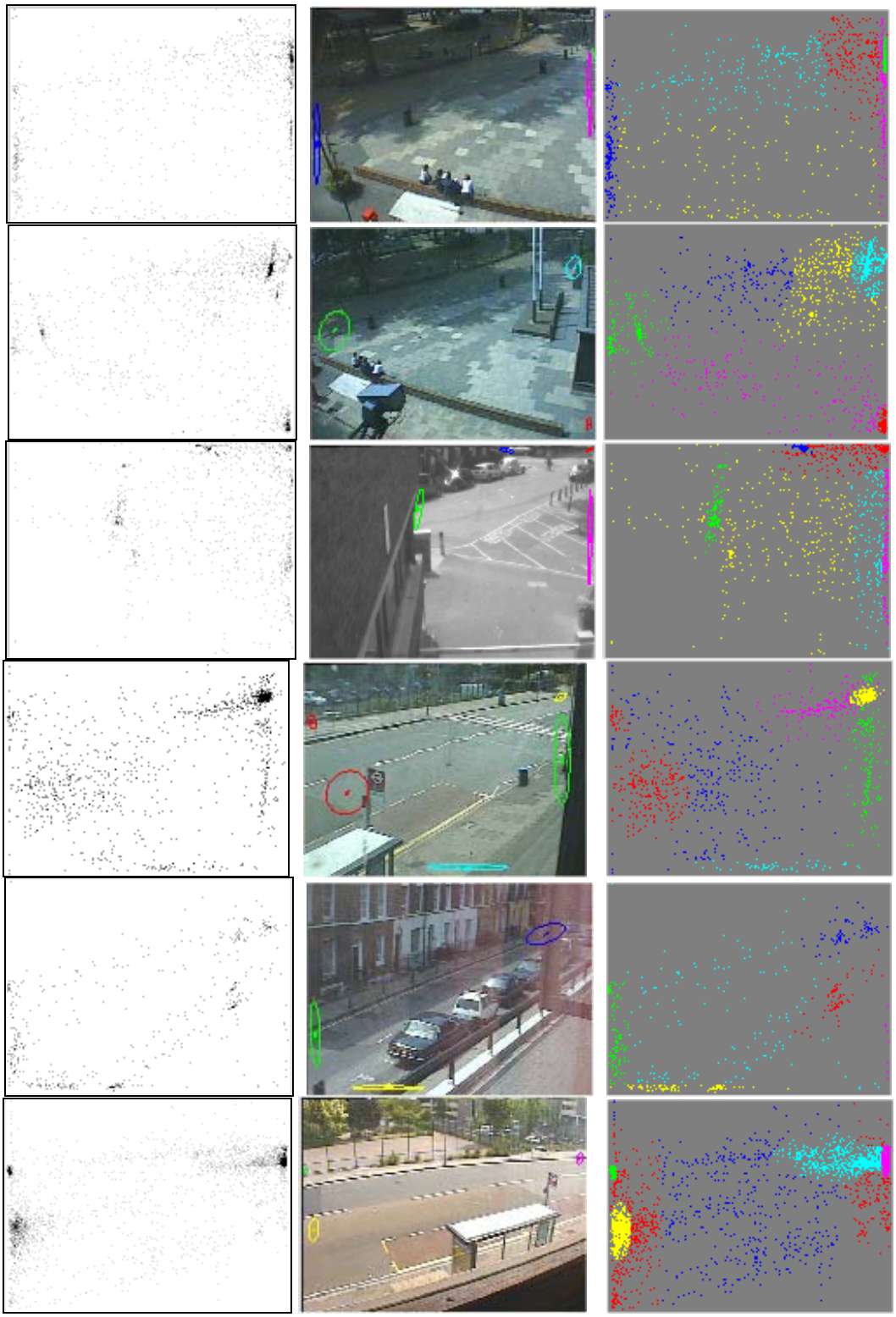


Figure 6.4: Detection of exit zones on image plane: Histograms of exit points (left), detected exit zones (middle) and classified points (right).

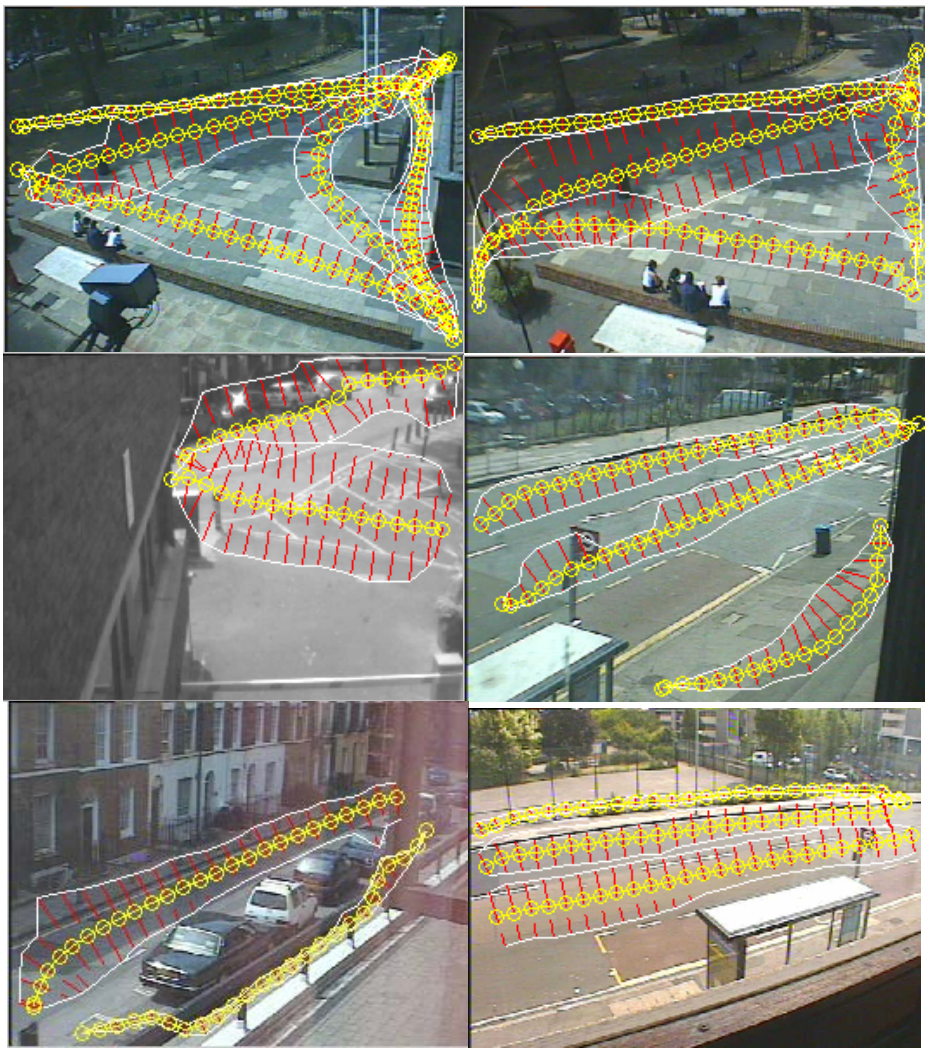


Figure 6.5: Detected routes on the image planes of the six cameras of CUES system.

Activity scene models can be applied to both the individual camera views and the common ground plane. Trajectory data has been converted to ground plane coordinates, as described in §2.5, and used as input to the entry/exit zones learning algorithm and the route learning algorithm. Results are illustrated in Figure 6.7 and Figure 6.8.

Two issues are related to the prescribed approach of constructing integrated models for the entire covered scene. Firstly, the method requires explicit geometric calibration of all the cameras of the system. Secondly the model covers only the network FOV, failing to represent activity on the "holes" of the virtual FOV. The next section introduces a novel technique that deals with these two issues.

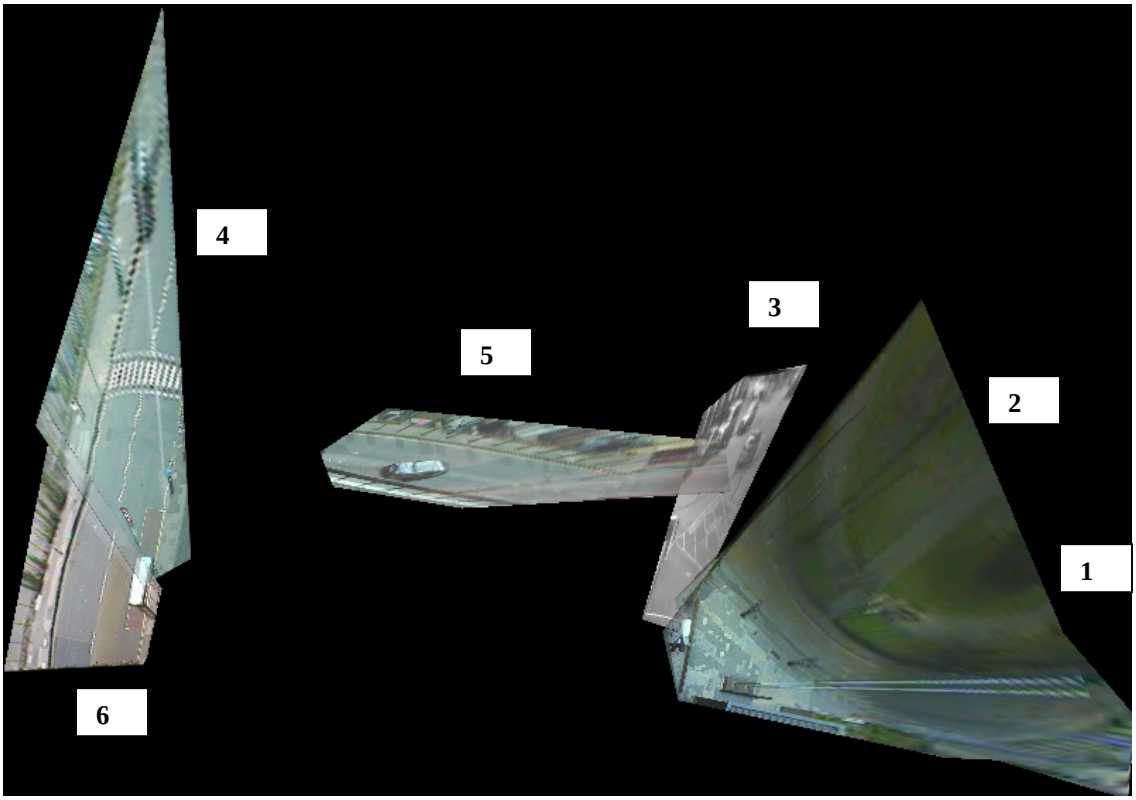


Figure 6.6: Ground map of the six-camera CUES system.

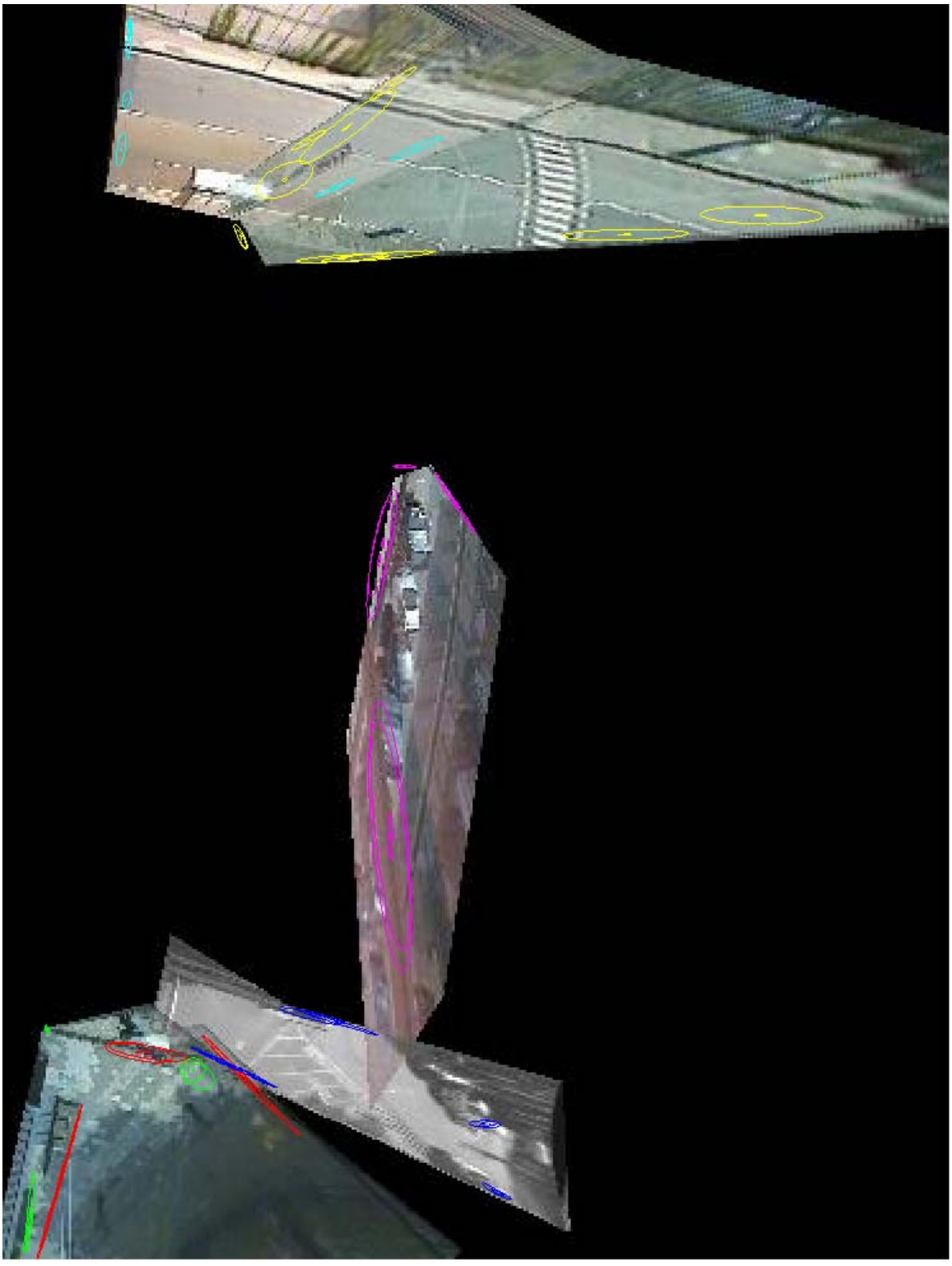


Figure 6.7: Entry/exit zones detected on the ground plane. Each colour (red, green, blue, yellow, magenta, cyan) corresponds to one of the six camera views.

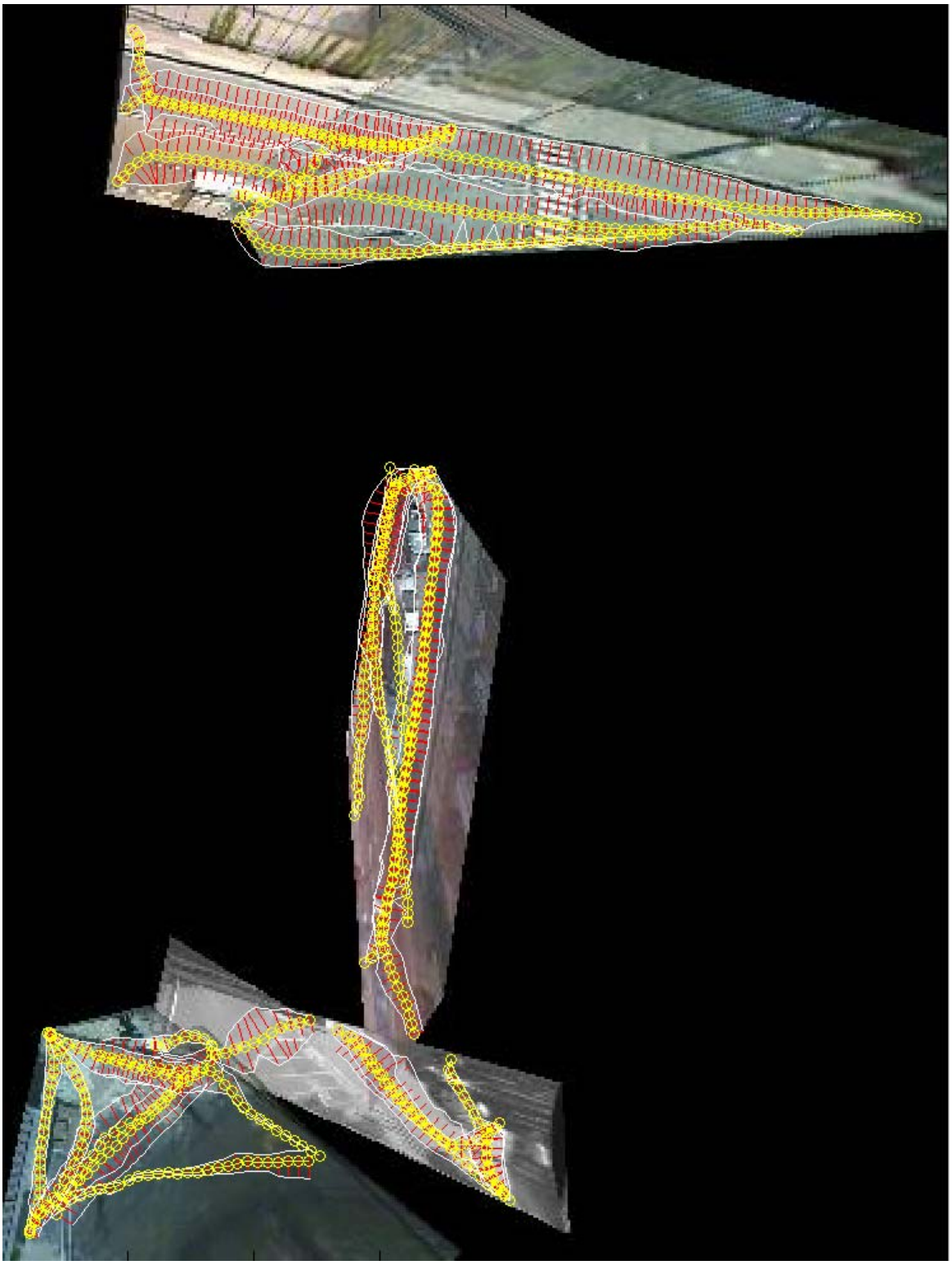


Figure 6.8: Routes learnt on the common ground plane.

6.5 Multiple Camera Activity Network (MCAN)

In the previous section, semantic scene models have been derived for individual camera FOVs and for the network FOV. However, it is desirable to extend the scene model to the entire virtual FOV, to cover activities that occur within the "blind" areas of the system. Although these activities are not directly viewed by the system, reasonable assumptions are derived.

Let us assume that no geometric information exists that allows localisation of the camera FOVs. Cameras with overlapped visible FOVs can be identified [93], [10] and calibrated using homography. A common semantic scene model can be derived for each set of cameras with overlapped FOVs. However, possible gaps in between cameras do not allow the semantic scene model to be established for the whole system, and isolated scene models are derived, instead.

To overcome the lack of a common scene model for the virtual FOV, the isolated scene models must be linked. However, no spatial linking is achievable, due to the lack of geometric calibration. Instead, this thesis proposes a probabilistic-temporal linking of the isolated views.

All the entry/exit zones of the camera FOVs are represented collectively as a network of nodes (similar to the topological representation in §3.2). The links of the network represent transitions between the entry/exit zones, either visible (through the Network FOV), or invisible (through the "blind" areas). A markovian chain or a HMM can be overlaid on the topology representation, which can provide a probabilistic framework for activity analysis and long-term predictions. (In this case, an extra "out-of-the-scene" state is required to indicate the event where targets exit the Virtual FOV of the system).

Visible links are learnt automatically (see chapter 4) using trajectories derived by a single-camera tracker [27] or overlapped multiple camera tracker [10] and are physically represented in spatial terms, according to the route model (§4.2).

However, the challenge is to identify the invisible links and this is the focus of the method proposed in this section. Invisible links are estimated in temporal terms and more specifically by pdfs that shows the distribution of the target transition periods through the blind areas.

6.5.1 Theoretical formulation

A MCAN is formulated by the set of all the entry/exit nodes that are detected within all the cameras of the system. No information is provided regarding the spatial relationship of these nodes. It is required to identify the directional links of this network expressed in probabilistic-temporal terms, which represent target transitions from one node to the other.

A graph model (shown in Figure 6.9) is used to represent a possible link between two nodes, i and j. Targets disappear from the node i with rate $n_i(t)$ and appear at the node j with rate $m_j(t)$. A third virtual node k represents everything out of the nodes i and j. Targets transit from node i to node j in time τ with probability $a_{ij}(\tau)$, otherwise they transit to the virtual node k with probability a_{ik}.

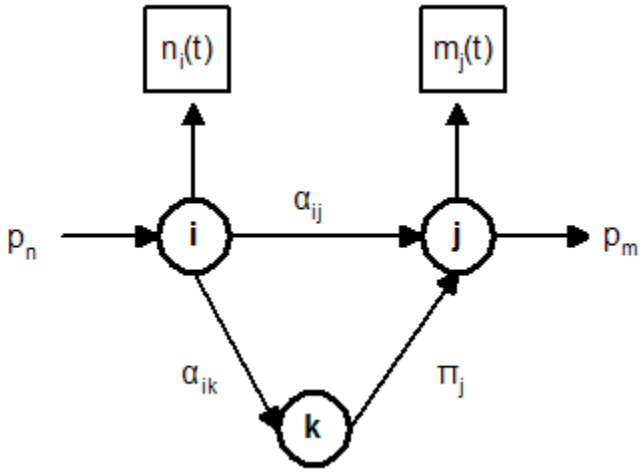


Figure 6.9: Graph model showing the probabilistic links between two nodes (i and j) and the virtual node k.

Transition probabilities fulfil the equation:

$$\int a_{ij}(\tau)d\tau + a_{ik} = 1 \qquad (6.1)$$

Also, targets from the node k move to node j with rate π_j; i.e. new targets can be "generated" at node k, and are detected on entering at node j.

The surveillance system is able to observe the signals $n_i(t)$ and $m_j(t)$. The two signals are assumed individually and jointly stationary. Therefore, the cross-correlation function is defined:

$$R_{ij}(\tau) = E\{(n_i(t) \cdot m_j(t+\tau)\} \qquad (6.2)$$

If it is assumed that the two signals $n_i(t)$, $m_j(t)$ are taking digital values from the set {0, 1}, then $p_n=p\{n_i(t)=1\}=E\{n_i(t)\}$ *and* $p_m=p\{m_j(t)=1\}=E\{m_j(t)\}$.

The cross-correlation $R_{ij}(\tau)$ and the covariance $C_{ij}(\tau)$ defined as:

$$C_{ij}(\tau)=R_{ij}(\tau)-p_n p_m \tag{6.3}$$

are used to identify possible links. If $\int C_{ij}(\tau)d\tau = 0$, then the two signals are uncorrelated and, because according to the proposed graph model, their relationship can be only linear, they are independent [77] and no real link should exist in between them. Otherwise, the two signals are dependent and a valid link i→j must exist. In this case, the transition probability is estimated by the formula:

$$a_{ij}(\tau) = C_{ij}(\tau) / p_n \cdot (1 - p_n) \tag{6.4}$$

The above equation is derived by the graph model in Figure 6.9. To prove this, let define the stationary digital signals $s_\tau(t)$ and $o_j(t)$ with means: $E\{s_\tau(t)\}= a_{ij}(\tau)$ and $E\{o_j(t)\}= \pi_j$. Then, according to the graph model in Figure 6.9, the signal $m_j(t)$ can be expressed as:

$$m_j(t+\tau) = s_{ij,\tau}(t) \cdot n_i(t) + o_j(t+\tau) \tag{6.5}$$

and the probability p_m as:

$$p_m = E\{m_j(t+\tau)\} = a_{ij}(\tau) \cdot p_n + \pi_j \tag{6.6}$$

Based on the Eq.6.5, the product $n_i(t) \cdot m_j(t+\tau)$ is written as:

$$n_i(t) \cdot m_j(t+\tau) = n_i(t) \cdot \left[s_{ij,\tau}(t) \cdot n_i(t) + o_j(t+\tau) \right] \tag{6.7}$$

Because the signal $n_i(t)$ is digital, $n_i^2(t) = n_i(t)$, therefore:

$$n_i(t) \cdot m_j(t+\tau) = n_i(t) \cdot s_{ij,\tau}(t) + n_i(t) \cdot o_j(t+\tau) \tag{6.8}$$

and the cross-correlation $R_{ij}(\tau)$ is written as:

$$R_{ij}(\tau) = \mathrm{E}\{n_i(t) \cdot m_j(t+\tau) = n_i(t) \cdot s_{ij,\tau}(t) + n_i(t) \cdot o_j(t+\tau)\} = p_n \cdot a_{ij}(\tau) + p_n \cdot \pi_j \tag{6.9}$$

From the Eq.6.3, Eq.6.6 and Eq.6.9, it is derived that:

$$C_{ij}(\tau) = p_n \cdot a_{ij}(\tau) + p_n \cdot \pi_j - p_n \cdot \left[a_{ij}(\tau) \cdot p_n + \pi_j \right] \tag{6.10}$$

which is equivalent to Eq.6.4.

Summarising, the MCAN is defined by the nodes, expressed as Gaussian distributions on the separate camera views and directional links between nodes, defined by transition probabilities that depend on the transition time.

6.5.2 Implementation

For each possible link i→j, from an exit zone i to an entry zone j, a cross-correlation function $R_{ij}(\tau)$ is estimated for $-T \le \tau \le T$, by accumulating disappearing events at the exit zone i and appearing events at the entry zone j, in a discrete time buffer. More specifically, for a given disappearing event at zone i at time t_1, searching is performed for appearing events at zone j, at time t_2, $t_2 \in [t_1\text{-}T, t_1+T]$, where T is a parameter that defines the time-search window. When such an event is detected, $R_{ij}(\tau)$ is updated:

$$R_{ij}(\lfloor t_2 - t_1 + 0.5 \rfloor) \leftarrow R_{ij}(\lfloor t_2 - t_1 + 0.5 \rfloor) + 1 \tag{6.11}$$

Theoretically, $C_{ij}(\tau)$ should be estimated by Eq.6.3. However, because the assumption that the signals are stationary is not accurate, the covariance $C_{ij}(t)$ is estimated as:

$$C_{ij}(\tau) = R_{ij}(\tau) - median(R_{ij}) \tag{6.12}$$

If a valid link exists, then $C_{ij}(\tau)$ has a clear peak which indicates the most popular time-transition value. Valid links are detected by detecting peaks above a threshold, defined as:

$$thr = \mu_{R_{ij}} + \omega \cdot \sigma_{R_{ij}} \tag{6.13}$$

where ω=3, in practice.

The transition probability $a_{ij}(\tau)$ is estimated using the Eq.6.4 and enforcing the constraint

$$0 \le \int a_{ij}(\tau) d\tau \le 1 \tag{6.14}$$

However, if the most popular transition time is negative, this is an indication of target transitions from j to i. Therefore, instead of $a_{ij}(\tau)$, we estimate $a_{ji}(-\tau)$:

$$a_{ji}(-\tau) = C_{ij}(\tau) / p_n \cdot (1 - p_n) \tag{6.15}$$

The topology of the camera views is determined by the set of the valid links and their transition times. If a link is detected between the zones of two cameras, the two cameras are either adjacent or overlapped. If the transition time between the exit zone i and the entry zone j is approximately zero, then the two zones of the two cameras are overlapped. If the transition time is positive, then the targets move from one zone to the other through an invisible path. Finally, if the transition time is negative, then the targets

move from one zone to the other through a path that is partially or entirely visible from the two cameras.

6.5.3 Results

Results are presented for a six-camera network. Trajectory data was derived by motion tracking modules [7], running during the daylight period. The dataset consists of 4282, 5579, 3853, 4230, 1922, 10600 trajectories for the cameras 1-6 respectively, derived during a 13-hour period. Only the first and the last point of each trajectory are used for these experiments and they are clustered to automatically to derive the entry and exit zones.

The entry and exit points are used to estimate the entry and the exit zones in each camera. The detected zones are the nodes of the MCAN and are numbered to illustrate the results (Figure 6.10). Entry and exit zones that were coincident have been merged for sake of simplicity.

The real spatial relationship of the zones is depicted in Figure 6.11, where the zones are overlaid onto a ground plane map of the network FOV. The ground plane map has been derived using geometric calibration information. However, this information is not used by the method presented here and the map is given only to provide the readers with the real geometry of the scene. The geometry of the scene is also shown in the simplified drawing of Figure 6.12.

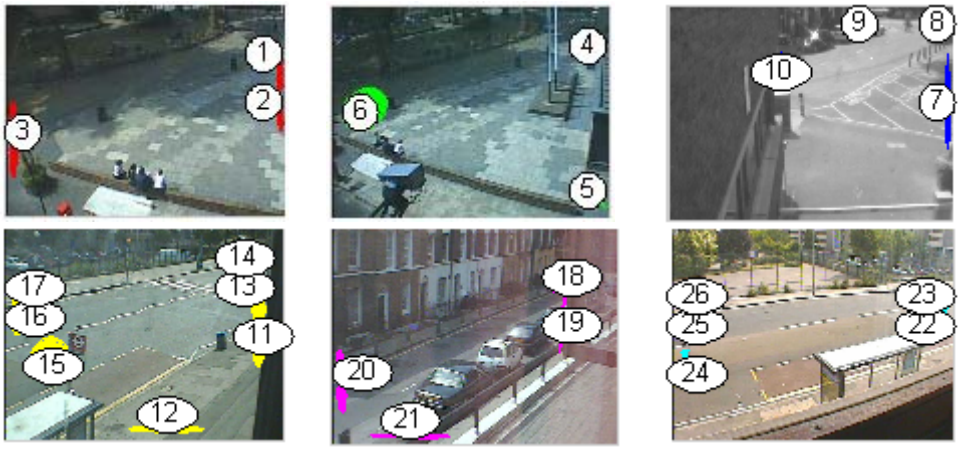


Figure 6.10: The detected entry/exit zones for the six cameras of the network. The zones are numbered as nodes of the activity network.

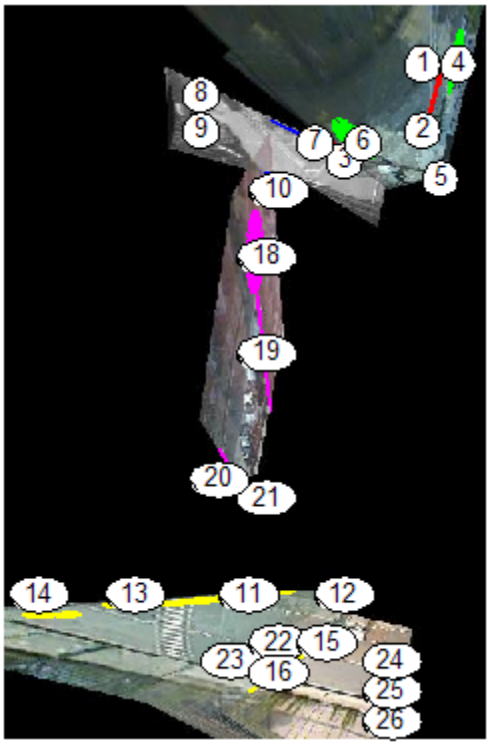


Figure 6.11: Entry/exit zones illustrated on a ground plane map of the network FOV.

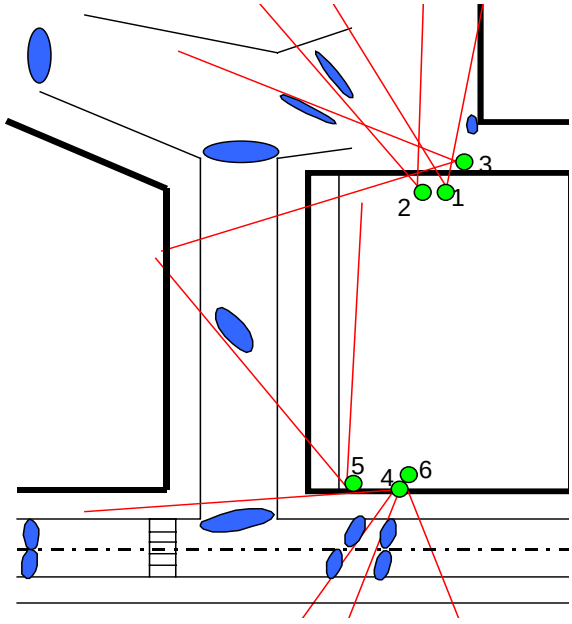


Figure 6.12. Geometric model of the camera environment showing the relative placement of the six cameras. Expected entry/exit zones are shown as blue ellipses.

Figure 6.13 depicts the cross-correlation function and Figure 6.14 the transition probabilities for six detected links (detailed results in Table 6.1). The black line in Figure

6.14 indicates the level of cross-correlation if the zones were uncorreletated, while the red line indicates the peak detection threshold (*thr*). In all the cases, the cross-correlation function has a clear peak above the threshold. On the contrary, Figure 6.15 depicts the cross-correlation function of invalid links where no clear peak exists.

The near-zero transition time between zones 3-7 (link 8) is an indication of overlapping. The small positive transition time between zones 18-10 (link 42) and between 23-20 (link 55) indicates that the two zones are closed each other and a "blind" region exists in between. Finally, the negative transition time between zones 1-3 (link 1), between 10-7 (link 25) and between zones 24-13 (link 53) indicates that the zones are close each other and their connection links are visible by at least one camera. Indeed, all conclusions are verified, as seen in Figure 6.11.

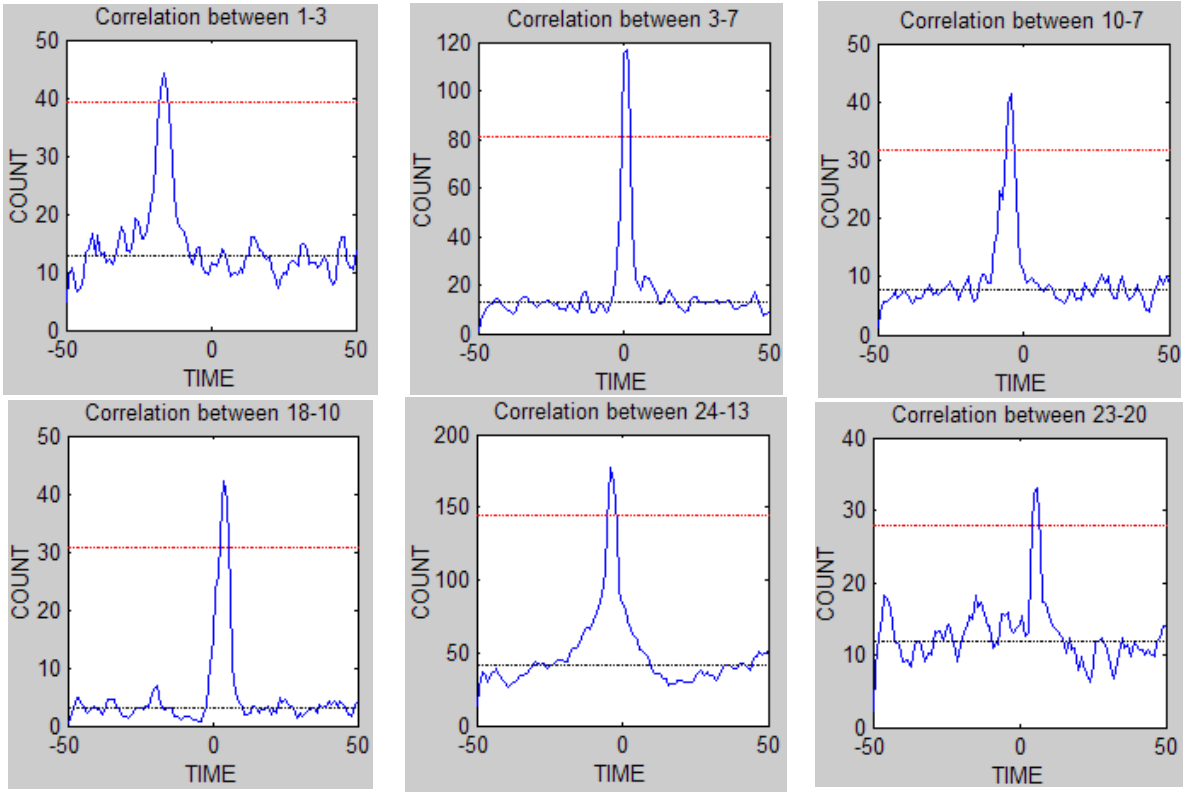


Figure 6.13: Cross-correlation functions for selected pairs of zones. Time is measured in secs. Red line indicates the peak detection threshold (thr), while the black line indicates the level of cross-correlation if the zones were uncorrelated. In all the cases, a clear peak exists on the most popular transition time.

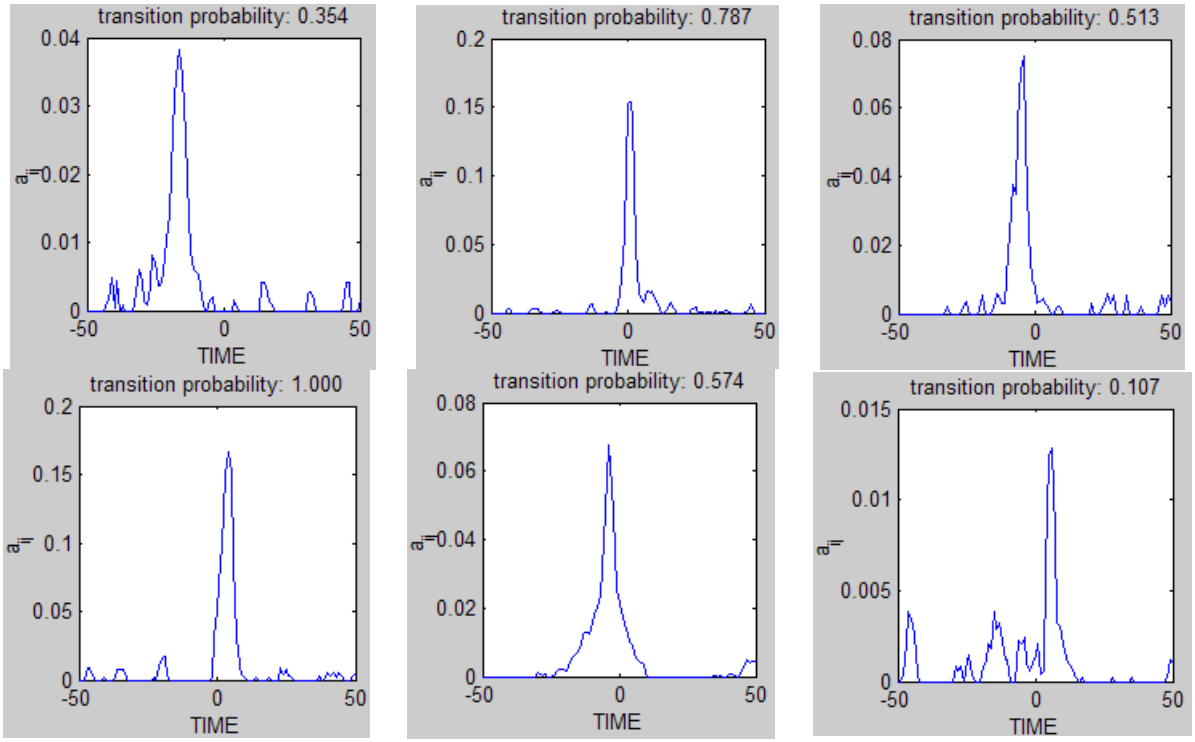


Figure 6.14: Transition probabilities derived by the cross-correlations of the Figure 6.13.

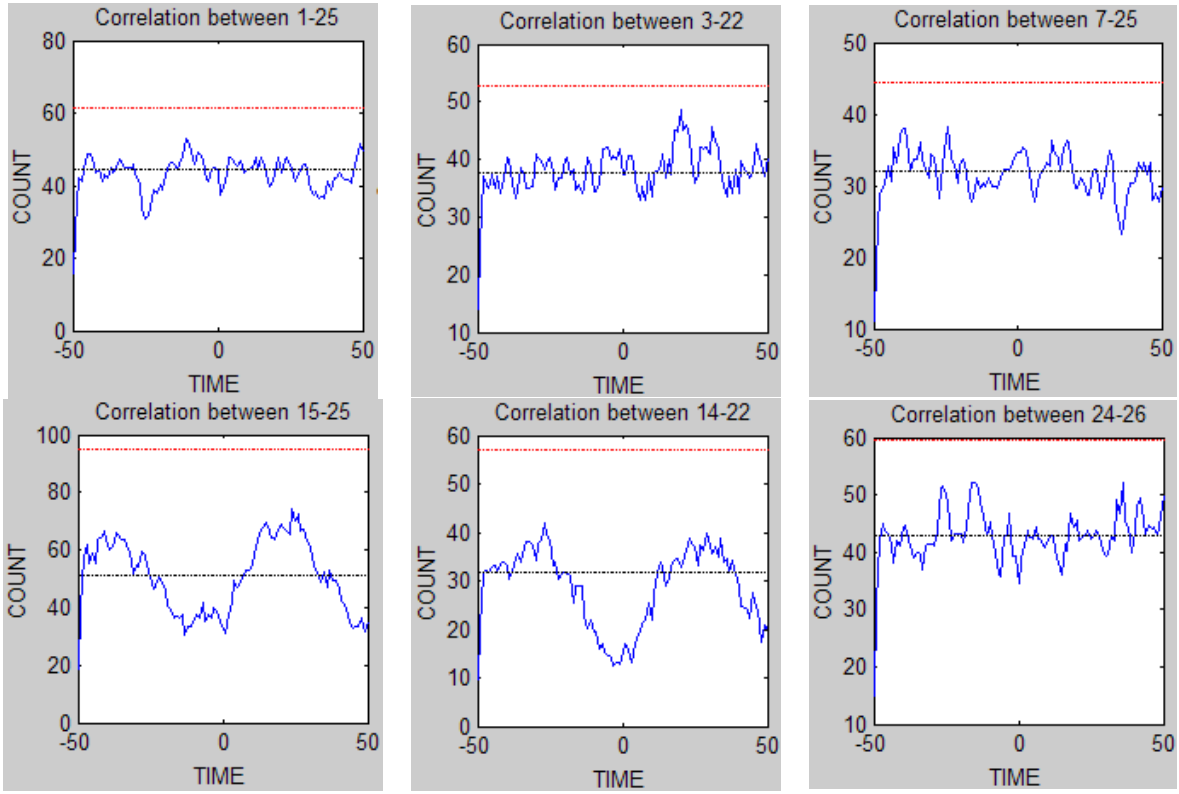


Figure 6.15: Cross-correlation for invalid links. Time is measured in secs. Red line indicates the peak detection threshold (thr), while the black line indicates the level of cross-correlation if the zones were uncorrelated. In all the cases, no clear peak exists.

Table 6.1 shows the 58 links that were automatically detected, between the zones shown in Figure 6.10. For each link the most popular transition time and the total transition probability were estimated. Additionally, the distances between the centres of the zones on the ground plane Figure 6.11 were used to provide an estimate of the most popular speeds across links, assuming linear velocity of the targets on the ground plane. The estimated speeds provide a quantitative validation of the method. For instance, links with low speed values (<3m/sec) correspond to pedestrian motion (e.g., most links between zones in cameras 1, 2 and 3), high speed values (>10m/sec) correspond to vehicle motion (e.g., most links between zones in cameras 4 and 6. Middle speed values (between 3m/sec and 10m/sec) correspond to vehicle motion on low-speed lanes (e.g., links between zones in cameras 3, 4, 5 and 6.).

Link	Exit	Entry	Time (sec)	Prob.	Dist. (m)	Speed (m/sec)	Link	Exit	Entry	Time (sec)	Prob.	Dist. (m)	Speed (m/sec)
1	1	3	-16	0.35	23.56	1.47	30	10	11	12	0.19	76.48	6.37
2	3	1	-15	0.37	23.26	1.55	31	8	20	-11	0.49	69.90	6.35
3	2	2	0	0.37	0.35	-	32	10	19	4	0.24	34.51	8.63
4	1	6	-14	0.54	20.06	1.43	33	15	13	-3	1.00	34.78	11.59
5	3	4	-17	0.40	25.11	1.48	34	11	12	-6	0.44	16.49	2.75
6	1	9	-39	0.14	43.65	1.12	35	14	16	-4	0.85	44.29	11.07
7	1	10	-21	0.16	34.38	1.64	36	12	11	-7	1.00	16.94	2.42
8	3	7	1	0.79	2.72	2.72	37	11	20	3	0.13	23.87	7.96
9	1	21	-48	0.11	82.98	1.73	38	15	22	-1	1.00	5.58	5.58
10	6	1	-13	0.46	20.39	1.57	39	14	25	-5	1.00	64.70	12.94
11	4	3	-17	0.19	26.33	1.55	40	20	7	-38	0.30	61.02	1.61
12	5	4	-13	0.68	19.07	1.47	41	20	8	-11	0.23	69.19	6.29
13	6	6	0	0.41	0.75	-	42	18	10	4	1.00	15.77	3.94
14	6	4	-14	0.55	22.24	1.59	43	20	11	6	0.22	23.97	3.99
15	4	5	-13	0.19	20.75	1.60	44	18	16	-9	0.44	74.49	8.28
16	4	4	-1	0.30	1.62	1.62	45	20	20	0	0.21	0.28	-
17	6	7	3	0.64	4.70	1.57	46	20	19	-4	0.41	19.46	4.87
18	4	9	-40	0.08	46.29	1.16	47	18	20	-4	1.00	38.46	9.61
19	10	3	5	0.16	11.42	2.28	48	21	21	-1	0.23	0.14	0.14
20	7	3	0	0.72	4.89	-	49	21	19	-8	0.66	21.45	2.68
21	7	6	3	0.72	7.08	2.36	50	20	22	8	0.39	32.14	4.02
22	8	9	-3	0.36	7.42	2.47	51	18	25	-13	1.00	81.25	6.25
23	8	8	0	0.23	0.94	-	52	23	10	12	0.16	87.04	7.25
24	8	10	-5	0.51	23.70	4.74	53	24	13	-4	0.57	50.67	12.67
25	10	7	-4	0.51	9.71	2.43	54	23	16	0	0.36	9.06	-
26	10	8	-5	0.21	22.62	4.52	55	23	20	6	0.11	34.08	5.68
27	9	8	-2	0.43	6.57	3.29	56	24	22	-2	1.00	21.46	10.73
28	7	9	-11	0.22	20.71	1.88	57	23	25	-2	1.00	29.48	14.74
29	7	10	-5	0.43	8.77	1.75	58	23	26	-4	0.14	30.71	7.68

Table 6.1: Detected links between entry/exit zones. Information about total transition probabilities (Prob.), most popular transition times (Time), real distance on the ground plane (Dist.) and an estimated target speed (Speed) are also shown.

When correlating signals from the same camera view, the transition times are almost always negative, because the links between the zones are actually visible from the camera. For example, the detected visible links of camera 3 are shown in Figure 6.16.

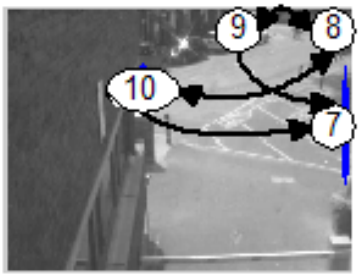


Figure 6.16: Detected visible links between zones of the camera 3.

Figure 6.17 illustrates some detected links between zones of different cameras. Both visible (22-15, 16-23, 25-14) and invisible links (23-20, 20-22, 11-20) were successfully detected. Most of the links are unidirectional, because they correspond to vehicle traffic flow. The link 25-14 is redundant in the sense that it can be described by a set of simpler links (25-16, 16-23, 23-14).

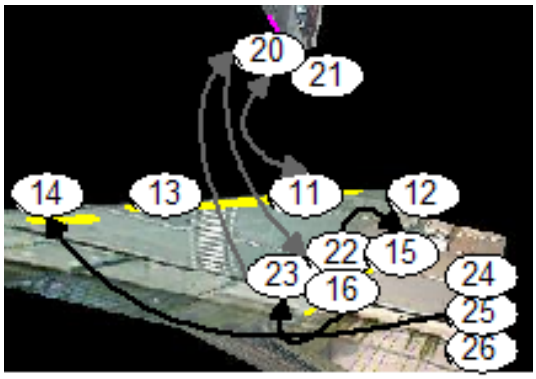


Figure 6.17: Detected visible and invisible links between zones of the cameras 4, 5 and 6.

The links of Table 6.1 were used to automatically determine the topology of the camera network, as shown in Figure 6.18. More specifically, any link between zones of two specific cameras, with a transition time that is not too long (e.g., more than 20secs), indicates that the cameras are either adjacent or overlapped. The sign of the transition times between zones of different cameras can indicate whether camera FOVs are adjacent or overlapped. If the transition time is too long or no significant correlation is detected, then the camera FOVs are assumed disjoint.

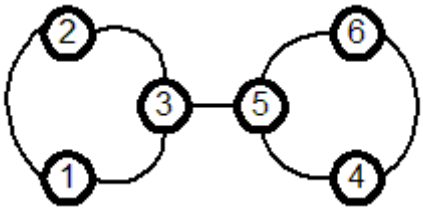


Figure 6.18: The topology of the six-camera network, as derived automatically.

Figure 6.19 depicts an activity model that contains both visible links (routes) detected in §6.4 and invisible links detected in this section. This illustration provides a model of activity of the entire virtual FOV of the CUES system.

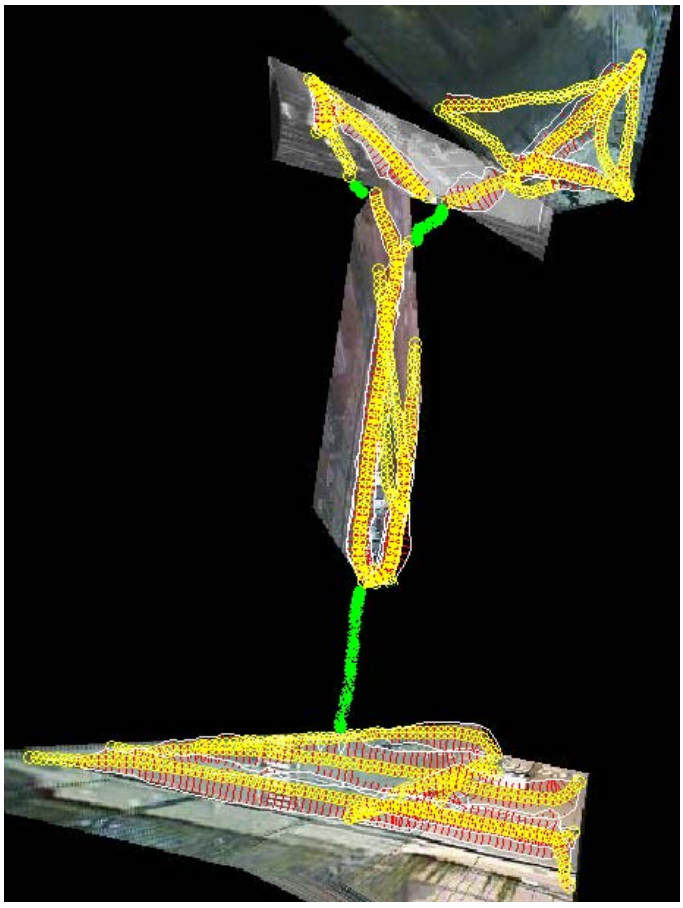


Figure 6.19: Visible routes (yellow-red) detected in §6.4 and invisible links (green) detected in this section.

6.6 Conclusions - Discussion

This chapter demonstrated how the learning methods of previous chapters can be applied in a real surveillance system. The results on the image plane were produced to validate the methods, while the results on a common GP illustrated how learnt models are integrated in the spatial domain.

However, the GP model assumes that all the activity of the wide area scene is co-planar, which is not always true. Additionally, the GP model requires that all the network cameras be properly calibrated, which is not consistent with the aim of this thesis for self-calibrated and adaptive surveillance systems.

An alternative approach of integrating scene models of different cameras has been presented. A MCAN that covers all the virtual FOV of the system has been created and an appropriate learning method was demonstrated. Again, the model has been learnt using unsupervised methods, without recourse to an unreliable correspondence process.

The MCAN is used to reveal the topology of the camera views of the system. It also links the entry/exit zones of each camera view using transition times and transition probabilities. This information can be also used to support tracking across the "blind" areas of the virtual FOV.

Other methods that support tracking across "blind" areas are based either on the dynamics or the appearance cues of the targets. A motion prediction filter (e.g., Kalman filter) uses the dynamics of the target to predict the camera view and the location that the target will re-appear. However, this method requires knowledge of both the geometric models of the cameras and the GP model and assumes that motion dynamics will not change during the target transition across the "hole" of the virtual FOV. This is not always true due to possible target turning or scene obstacles.

Using appearance cues, such as target geometry or colour, requires some kind of calibration between the cameras to ensure the correct correspondence of the cues at different cameras. Sometimes, this approach is problematic and unreliable, for example no reasonable colour correspondence can be established between colour cameras and monochrome cameras or thermal cameras. Also, the target geometry cues (size, shape) are heavily dependent of the camera geometric model, for example targets' sizes and

shapes are quite different when viewed by common CCTV cameras, CCTV cameras with top-down view or omnidirectional cameras.

The MCAN proposed in this chapter is independent on the camera types and does not require any calibration. A possible limitation of the method is that requires a large number of observations to statistically reveal the dependencies. Also, the variability of the targets' velocities, when they disappear or reappear is not considered, although the variability of transition times is modelled. In the case that the circumstances allow the usage of other cues (dynamics, appearances), the transition probabilities are still useful as they can be combined with other probabilistic information using a Bayesian framework [57].

Chapter 7

Conclusion

7.1 Summary

The aim of this thesis was to develop a methodology that allows surveillance systems to automatically derive semantic scene models. These models consist of activity-based semantic elements such as entry/exit zones, paths, junctions, routes and stop zones that are described in both spatial and probabilistic terms. Automatic methods that exploit consistencies in large datasets of observations and allow automatic learning of the elements of the scene models were presented.

The spatio-probabilistic character of the model aims to describe the spatial extent of the semantic features, similarly to the way that a human might interpret them, and to represent the variation of the usage of the semantic features and related uncertainty.

The proposed methodology can significantly benefit surveillance systems. The semantic nature of the spatial component of the model supports the implementation of applications such as video annotation and contextual databases. On the other hand, the probabilistic component of the model supports applications such as atypical activity detection, long-term activity prediction and tracking enhancement. The philosophy of unsupervised learning that was adopted enables some autonomy for the surveillance systems, as it allows them to construct the scene models without human intervention.

To successfully represent the spatio-probabilistic character of the scene model, a variety of representations were investigated, including known models such as Gaussian Mixture Models, accumulative maps and Hidden Markov Models and novel models such as the spline-based route model and the Multiple Camera Activity Network. Unsupervised training of these models was based on known methods, such as

Expectation-Maximisation and accumulative statistics whilst more novel methods were developed to learn the route model and the Multiple Camera Activity Network.

Some general conclusions can be drawn from the models that were used in this thesis. Accumulative maps provide a very general non-parametric representation of the data, however they lack compactness, comparing to parametric models. Gaussian Mixture Models are adequate to represent regions that are spatially limited and usually related to single-point events. However, they cannot represent more complicated regions that are related to trajectories. For this reason, the route model was developed that represents the average of a cluster of trajectories and their variation using spline representations. Hidden Markov Models were used to represent the dynamics of the targets, as they move around the scene. Finally, the Multiple Camera Activity Network provides a means to model and learn activity in a wide-area scene, covered by uncalibrated multiple cameras, without the explicit requirement of tracking.

Chapter 3 not only introduced the activity-based semantic model, but also investigated how semantic elements related to single-point events (such as entrance, exit, stop) can be represented using GMMs and learnt by EM. Entry/exit zones, in particular, are valuable because they can help to validate the trajectory dataset. Trajectories are assumed valid only if they start from an entry zone and exit at an exit zone. Additionally, the representation and learning of semantics not explicitly related to the activity in the scene, such as the occlusion regions and the semi-stationary motion noise sources, were investigated.

Chapter 4 addressed the problem of deriving semantic regions related to sequences of points (trajectories), such as routes, paths and junctions. To allow this, a novel route model was introduced that allows adequate spatio-probabilistic representation of clusters of similar trajectories. An appropriate unsupervised auto-initialising learning method was developed that allow automatic deriving of the route models of a scene to be automatically derived from a set of trajectories. Further analysis of the scene route models, based on computational geometry, segmented them into more primitive elements: paths and junctions. These primitive elements are valuable, because they can be used to represent any possible route either in a single camera view or in a wide-area scene

covered by multiple cameras. Additionally, junctions are useful because they represent the areas that targets may change their direction.

Chapter 5 demonstrated the idea of overlaying probabilistic networks onto the scene model, augmenting the representation to support analysis of the scene activity. As an example, the Route-Based HMM was introduced that is superimposed on the scene route models. The benefit of this approach is that learning, which is an issue for HMM, is quite fast and the nodes of the HMM are explicitly related to the semantics of the scene. The usage of the scene model in activity analysis applications, such as long-term predictions and atypical activity detection was also illustrated.

Finally, Chapter 6 investigated the applicability of the methodology in real multiple camera surveillance systems. More specifically, it dealt with the problem of integrating scene models from multiple cameras. Although semantic models from multiple cameras can be combined in a ground plane map, this geometric-based approach is not attractive as it assumes that the activity in the entire observed scene is coplanar and requires manual calibration of the cameras. An alternative non-geometric approach that relates camera views in terms of transition times and transition probabilities is introduced: the Multiple Camera Activity Network that is constructed using the set of the entry/exit zones of all the cameras. A "correspondence-free" method is introduced that allows encoding of activity even in the unseen regions of the scene. The Multiple Camera Activity Network is also useful because it automatically reveals the topology of the camera views of the surveillance system, i.e. determines whether the camera views are overlapped, adjacent or distant.

7.1.1 Contributions

The main contributions of the current thesis are:

- Developing an activity-based semantic scene model that consists of semantics like entry/exit zones, stop zones, routes, paths and junctions.
- Demonstrating how semantic regions that are related to single-point events (entry, exit, stop) could be learnt automatically from observations.

- Developing a novel route model that represents typical patterns of motion and an appropriate unsupervised algorithm.
- Augmenting the scene model with probabilistic networks to represent the activity in the scene.
- Developing a novel method that automatically learns the activity across a wide-area scene that can integrate information from multiple cameras, reveal the camera topology and encode activity in the unseen regions of the environment.

7.2 Future work

In general, the methodology described by this thesis achieved the initial aim. However, some further refinement of the methods can be considered. For example, automatically deriving the model order criterion for the GMM model of the entry/exit zones. Also, the route-model learning algorithm could be modified to remove the dependency of the outcome from the amount of training data. Redundant links of the MCAN could be removed.

More generally, the issue of the adaptability of the models over time was not explicitly investigated. However the probabilistic character of the proposed models and the soft-logic approach in the design of the proposed learning methods provide the foundations for adaptive variations of the current methodology.

Finally, another issue is that the semantic features that were learnt were based on a restricted set of events (“enter”, “exit”, “stop”, “move”) related to the dynamics of the targets. A richer set of semantics can be derived if a wider set of activity-based events is considered (“turn-right”, “turn-left”, “speed up”, “slow down”) that may be related to target classification (“route” of “pedestrians” → “pavement”, “route” of “vehicles” → “road”) or to more complicated rules (“targets moving slowly in a line” → “queues”, “queues” and “stop zone” of “pedestrians” and “stop zone” of “large vehicle” and “merge” of “pedestrian” with “large vehicle” → “bus stop”).

A range of applications in visual surveillance can benefit from the current work. The semantic character of the scene model can be exploited in applications related to interaction of the surveillance systems with their operators, such as conceptual databases

[12] and video annotation. The probabilistic character of the model can be used to enhance the motion tracking process, to identify atypical activities and to provide long-term prediction. Finally, the automatic learning of the camera topology can be used as part of a fully automatic multiple camera calibration process.

7.3 Epilogue

This thesis has developed a methodology that enables surveillance systems to automatically learn activity-based semantic scene models. Because the structure of the scene influences the activity in the scene, observing the activity provides cues related to the scene itself. A reverse-engineering method was adopted that exploits the potential consistency of large numbers of activity observations to build a scene model based on activity-based semantics.

Therefore, surveillance systems are able to build their own cognitive knowledge databases that enable a higher level of understanding of the scene activity. The unsupervised character of the learning methods is consistent with the concept of "plug 'n' play" surveillance systems.

The application of the methodology was investigated on both single-camera and multiple-camera surveillance systems. In particular for multiple-camera systems, an automatic method has been developed that allows integration of the scene models of individual cameras. Because the method does not require any manual camera calibration, it is useful for the installation and the adaptation of large surveillance systems.

The presented work was motivated not only by the potential applications in visual surveillance, but also by the ambition to build an autonomous cognitive computer vision system that senses, learns and understands its environment with a minimal support from human operators.

References

[1] Shun-ichi Amari, “Information geometry on hierarchical decomposition of stochastic interactions”, *IEEE Trans. on Information Theory*, vol.47(5), pp.1701-1711, July 2001.

[2] Hubert Austermeier, Georg Hartmann, Ralf Hilker, “Color-calibration of a robot vision system using self-organizing feature maps”, *International Conference on Artificial Neural Networks, ICANN96*, pp.257-262, Berlin, Germany, 1996.

[3] Ali Azarbayejani, Alex Pentland, “Real-Time Self Calibrating Stereo Person Tracking Using 2-D Shape Estimation from Blob Features”, *International Conference on Pattern Recognition, ICPR96*, vol.3, pp.627-632, Vienna, Austria, 1996.

[4] Kobus Barnard, Brian Funt, “Color Constancy Under Varying Illumination”, *European Conference on Computer Vision, ECCV96*, vol.2, pp.3-16, Cambridge, England, 1996.

[5] L. Baum, “An inequality and associated maximization technique in statistical estimation of probabilistic functions of Markov processes” *Inequalities*, vol.3, pp.1-8, 1972.

[6] A.M Baumberg, D.C. Hogg, “Learning Spatiotemporal Models from Examples”, *British Machine Vision Conference*, BMVC95, Birmingham, September 1995.

[7] A.M. Baumberg, “Learning deformable models for tracking human motion”. *PhD thesis, University of Leeds*, UK, 1995.

[8] J.C. Bezdek, “Pattern Recognition with Fuzzy Objective Function Algorithms”, Plenum Press, New York, 1981.

[9] James Black, Tim Ellis, “Multi Camera Image Tracking”, *Second International Workshop on Performance Evaluation of Tracking and*

Surveillance, PETS2001, Kauai, Hawaii, December 2001.

[10] James Black, Tim Ellis, Paul Rosin, “Multi View Image Surveillance and Tracking”, *IEEE Workshop on Motion and Video Computing*, pp 169-174, Orlando, December 2002.

[11] J. Black, T.J. Ellis, D. Makris, “Wide Area Surveillance With a Multi-Camera Network”, *IEE Intelligent Distributed Surveillance Systems IDSS'04*, London, February 2004.

[12] James Black, Tim Ellis, Dimitrios Makris, "A Hierarchical Database for Visual Surveillance Applications", *IEEE International Conference on Multimedia and Expo, ICME2004*, Taipei, Taiwan, June 2004.

[13] James Black, Dimitrios Makris, Tim Ellis, "Hierarchical Database for a Multi-Camera Surveillance System", *Pattern Analysis and Applications* (PAA), submitted.

[14] Aaron F. Bobick, “Movement, Activity and Action: The Role of Knowledge in the Perception of Motion”, *Royal Society Workshop on Knowledge-based Vision in Man and Machine*, London, UK, February 1997.

[15] A.F. Bobick, Y.A. Ivanov, “Action Recognition using Probabilistic Parsing”, *IEEE Conf. on Computer Vision and Pattern Recognition, CVPR98,* pp.196-202, Santa Barbara, CA, June 1998.

[16] J.E. Boyd, J Meloche, Y. Vardi, “Statistical Tracking in video traffic surveillance”, *International Conference on Computer Vision, ICCV99*, pp.163-168, Corfu, Greece, September 1999.

[17] M. Brand, N. Olivier, A. Pentland, “Coupled Hidden Markov Models for Complex Action Recognition”, *IEEE conf. On Computer Vision and Pattern Recognition, CVPR97*, Puerto Rico, 1997.

[18] S. Brock-Gunn, T.J. Ellis, “Using colour templates for target identification and tracking”, *British Machine Vision Conference, BMVC92*, Leeds, UK, 1992.

[19] H.H. Bui, S. Venkatesh and G. West, “Tracking and Surveillance in Wide Area Spatial Environments Using the Abstract Hidden Markov Models”, *International Journal of Pattern Recognition and Artificial Intelligence*, vol 15(1) pp. 177-196, February 2001.

[20] Christopher J.C. Burges, “A tutorial on support vector machines for pattern recognition”, *Data Mining and Knowledge Discovery*, vol.2(2), pp.121-167, 1998.

[21] Hilary Buxton, Shaogang Gong, “Advanced Visual Surveillance using Bayesian Networks”, International Conference on Computer Vision, ICCV95, Cambridge, MA, June 1995.

[22] Hilary Buxton, “Generative Models for Learning and Understanding Dynamic Scene Activity”, *Generative Model Based Vision Workshop 2002, GMBV2002*, Copenhagen, Denmark, June 2002.

[23] A. Chardin and P. Perez. “Unsupervised image classification with a hierarchical EM algorithm”. *International Conference on Computer Vision, ICCV99,* vol.2 , pp.969-974, Kerkyra, Greece, September 1999

[24] F. Cupillard, F. Bremond, M. Thonnat, “Behaviour Recognition for Individuals, Groups of People and Crowd”, *IEE Intelligent Distributed Surveillance Systems*, London, March 2003.

[25] A. Dempster, N. Laird, D. Rubin, “Maximum likelihood from incomplete data via the EM algorithm”, *Journal of the Royal Statistical Society,* B-39, pp.1-38, 1977.

[26] J.C. Dunn, “A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters”, *Journal of Cybernetics* vol.3, pp.32-57, 1973.

[27] Tim Ellis, Ming Xu, “Object detection and tracking in an open and dynamic world”, *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, PETS2001*, pp.31-38, Kauai, Hawai, December 2001.

[28] Tim Ellis, James Black, A Multi-View Surveillance System, IEE Intelligent Distributed Surveillance Systems, London, UK, February 2003.

[29] Tim Ellis, Dimitrios Makris, James Black, “Learning a Multicamera Topology”, *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, VS-PETS2003*, pp. 165-171, Nice, France, October 2003.

[30] Tim Ellis, James Black, Ming Xu and Dimitrios Makris, “Integration and learning of Information from multiple camera views”, *In Visual Surveillance: A Computational Approach, J. Ferryman (ed.),* Springer, to be published.

[31] Tim Ellis, James Black, Ming Xu, Dimitrios Makris, “A Distributed Multicamera Surveillance System”, *In Ambient Intelligence, A Novel Paradigm, P. Remagnino, G.L. Foresti, T. Ellis (ed.),* Kluwer Academic Publishers, to be published.

[32] Olivier Faugeras, “Three-Dimensional Computer Vision, A Geometric Viewpoint”, *The MIT Press*, Cambridge, MA, 1993.

[33] J.H. Fernyhough, A.G. Cohn, D.C. Hogg. “Generation of semantic regions from image sequences”, *European Conference on Computer Vision, ECCV96*, pp.475-478, Cambridge, UK, April 1996.

[34] John Fernyhough “Generation of Qualitative Spatio-Temporal Representations From Visual Input”, *PhD Thesis, University of Leeds*, 1997.

[35] J. Fernyhough, A.G. Cohn, D.C. Hogg, “Constructing qualitative event model automatically from video input”. *Image and Vision Computing*, vol.18, pp.81-103. 2000.

[36] J.M. Ferryman, A.D. Worrall and S.J. Maybank, “Learning Enhanced 3D Models for Vehicle Tracking”. *British Machine Vision Conference, BMVC98*, pp.873-882, Southampton, September 1998.

[37] R. Fraile, S.J. Maybank, “Vehicle Trajectory Approximation and

Classification*", British Machine Vision Conference, BMVC98*, Southampton, UK, 1998.

[38] A. Galata, N. Johnson, D.C. Hogg, "Learning Variable Length Markov Models of behaviour", *Computer Vision and Image Understanding*, vol 81, pp. 398-413, 2001.

[39] Darrel Greenhill, Paolo Remagnino, Graeme A. Jones, "VIGILANT, Content-Querying of Video Surveillance Streams", *2nd European Workshop on Advanced Video-Based Surveillance Systems, AVBS2001*, Kingston, UK, September 2001.

[40] W.E.L. Grimson, C. Stauffer, R. Romano, L. Lee, "Using adaptive tracking to classify and monitor activities in a site", *IEEE Conference on Pattern Recognition, CVPR98*, pp.22-29, Santa Barbara, USA, June 1998.

[41] A.R. Hanson, E.M. Riseman, "The VISIONS Image-Understanding System" *Advances in Computer Vision, ACV88*, vol. 1, pp. 1-114, 1988.

[42] B.K.P Horn, B.G. Schunk, "Determining optical flow", *Artificial Intelligence*, Vol. 17, pp.185-203, 1981.

[43] R.J. Howard, H. Buxton, "Analogical representation of spatial events, for understanding traffic behaviour", *10th European Conf. On AI*, pp. 785-789, 1992.

[44] R.J. Howarth, "Spatial Representation and Control for a Surveillance System", *PhD thesis, Queen Mary and Westfield College, The University of London*, 1994.

[45] Weiming Hu, Dan Xie, Tieniu Tan, "Learning Patterns of Activity Using Fuzzy Self-Organizing Neural Network", *First Chinese Workshop on Intelligent Visual Surveillance*, pp.168-182, 2002.

[46] A. Hunter, J. Owens, M. Carpenter, "A Neural System for Automated CCTV Surveillance", *IEE Intelligent Distributed Surveillance Systems,* London, March 2003.

[47] Infant Vision Lab, Phycological Research at the Eunice Kennedy Shriver Center “Milestones in Visual Develpoment”, *http://umassmed.edu/shriver/research/phycological/infantvision/milestones.cfm,* Umass Medical School, Waltham MA.

[48] Michael Isard, Andrew Blake, “Contour tracking by stochastic propagation of conditional density”, *European Conference on Computer Vision, ECCV96*, vol.1, pp. 343--356, Cambridge UK, 1996.

[49] Michael Acheson Isard, “Visual Motion Analysis by Probabilistic Propagation of Conditional Density”, *PhD Thesis, University of Oxford*, September 1998.

[50] R. Jain, "Dynamic Scene Analysis Using Pixel-Based Processes*," IEEE Computer*, vol. 14, no. 8, pp. 12-18, August 1981

[51] Omar Javed, Zeeshan Rasheed, Khurram Shafique, Mubarak Shah, “Tracking Across Multiple Cameras With Disjoint Views”, *IEEE International Conference on Computer Vision, ICCV03,* Nice, France, October 2003.

[52] N. Johnson, D.C. Hogg, “Learning the distribution of object trajectories for event recognition”, pp.583-592, BMVC95, Birmingham, UK, September 1995.

[53] Neil Johnson. “Learning Object Behaviour Models”, *PhD thesis, School of Computer Studies, University of Leeds,* UK, September 1998.

[54] S.C Johnson, “Hierarchical Clustering Schemes”, *Psychometrika*, vol.2, pp.241-254, 1967.

[55] Pakorn Kaewtrakulpong, “Adaptive Probabilistic Models for Learning Semantic Patterns*”, PhD Thesis, Brunel University*, 2002.

[56] R. E. Kalman, “A new approach to linear filtering and prediction problems”. *Trans. of the ASME – Journal of Basic Enginnering*, 82: pp.35-45, 1960.

[57] V. Kettnaker, R. Zabih . “Bayesian Multi-Camera Surveillance”. *IEEE*

Conference on Computer Vision and Pattern Recognition, CVPR99, Fort Collins, Colorado, pp 2253-2561, June 1999.

[58] Esther B. Koller-Meier and Luc Van Gool, “Modeling and Recognition of Human Actions Using a Stochastic Approached”, *2nd European Workshop on Advanced Video-Based Surveillance Systems, AVBS2001*, pp.17-28, Kingston, UK, September 2001.

[59] L. Lee, R. Romano, G. Stein, “Monitoring Activities from Multiple Video Streams: Establishing a Common Coordinate Frame”, *IEEE Trans. On Pattern Recognition and Machine Intelligence,* vol. 22(8), pp. 758-767, August 2000.

[60] S.E. Levinson, L.R. RabineL and M.M. Sondhi, “An Introduction to the Application of Probabilistic Functions of a Markov Process to Automatic Speech Recognition”, *The Bell System Technical Journal*, vol. 62, No. 4, pp. 1035-1074, April 1983

[61] Jiangung Lou, Qifeng Liu, Tieniu Tan, Weiming Hu, “Semantic Interpretation of Object Activities in a Surveillance System”, *International Conference on Pattern Recognition, ICPR02*, Quebec City, Canada, August 2002.

[62] Gian Luca Foresti, Giorgio Giacinto, Fabio Roli, “Detecting Dangerous Behaviors of Mobile Objects in Parking Areas”, *2nd European Workshop on Advanced Video-Based Surveillance Systems, AVBS2001*, Kingston, UK, September 2001.

[63] Sebastian Luhr, Hunh. H. Bui, Svetha Venkatesh, Geoff A.W. West, “Recognition of Human Activity through Hierarchical Stochastic Learning”, Int. Conf. on Pervasive Computing and Communications, PerCom’03, pp.416-422, Fort Worth, Texas, March 2003.

[64] J. MacQueen, “Some methods for classification and analysis of multivariate observations”, *Proc. Of 5th Berkeley Symposium on Mathematical statistics and probability, University of California Press,*

Berkley, USA, vol. 1, pp. 281-297, 1967.

[65] Dimitrios Makris, Tim Ellis, “Finding Paths in Video Sequences”, *British Machine Computer Vision, BMVC2001,* pp.263-272, Manchester, UK, September 2001.

[66] Dimitrios Makris, Tim Ellis. “Spatial and Probabilistic Modelling of Pedestrian Behaviour”, *British Machine Vision Conference, BMVC2002,* pp.557-566, Cardiff, UK, September 2002.

[67] Dimitrios Makris, Tim Ellis, “Path Detection in Video Surveillance”, *Image and Vision Computing*, vol.20(12), pp.895-903, October 2002.

[68] Dimitrios Makris, Tim Ellis, “Automatic Learning of an Activity-Based Semantic Scene Model”, *IEEE Int. Conf. On Advanced Video and Signal Vased Surveillance, AVSS2003*, pp.183-188, Miami, FL, July 2003.

[69] Dimitrios Makris, Tim Ellis, James Black, “Learning Scene Semantics”, *Early Cognitive Vision Workshop, ECOVISION 2004*, Isle of Skye, May 2004.

[70] Dimitrios Makris, Tim Ellis, James Black, "Bridging the Gaps between Cameras", *IEEE Conference on Computer Vision and Pattern Recognition, CVPR2004,* Washington DC, USA, June 2004.

[71] Dimitrios Makris, Tim Ellis, “Learning Semantic Scene Models from Observing Activity in Visual Surveillance*”, IEEE Transactions on Systems, Man and Cybernetics, Part B*, submitted.

[72] Dimitrios Makris, Tim Ellis, James Black, “Mapping an Ambient Environment”, *In Ambient Intelligence, A Novel Paradigm, P. Remagnino, G.L. Foresti, T. Ellis (ed.),* Kluwer Academic Publishers, to be published.

[73] M. Mohnhaupt and B. Neumann, “Understanding object motion: recognition, learning and spatiotemporal reasoning” *Journal of Robotics and Autonomous Systems*. vol.81, pp.65-91, 1991.

[74] B. Neumann. “Natural language description of time-varying scenes”.

Semantic Structures: Advances in Natural Language Processing. pp.167-207, 1989.

[75] Jeffrey Ng, Shaogang Gong, “Learning Pixel-Wise Signal Energy for Understanding Semantics”, *British Machine Vision Conference, BMVC2001*, pp.695-704, Manchester, UK, September 2001.

[76] J. Owens, A. Hunter, “Application of the Self-Organising Map to Trajectory Classification”, *3rd IEEE International Workshop on Visual Surveillance*, pp. 77-83, Dublin, Ireland, 2000.

[77] Athanasios Papoulis, “Probability, Random Variables and Stochastic Processes”, *Third Edition, McGraw-Hill*, 1991.

[78] I. Pavlidis (editor), Proceedings of “IEEE Conference on Advanced Video and Signal Based Surveillance, AVSS2003”, Miami, July 2003.

[79] F. Porikli, A. Divakaran, “Multi-Camera Calibration, Object Tracking and Query Generation”, *International Conference on Multimedia and Expo, ICME2003,* pp 653-656, Baltimore, July 2003

[80] Lawrence R. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”, *Proc. of the IEEE*, Vol. 77, no. 2, pp. 257-286, February. 1989.

[81] P. Remagnino, G.A. Jones, “Classifying Surveillance Events from Attributes and Behaviour”, *British Machine Vision Conference, BMVC2001*, pp.685-694, Manchester, UK, September 2001.

[82] J. Renno, J. Orwell, G.A. Jones, “Learning Surveillance Tracking Models for the Self-Calibrated Ground Plane, *British Machine Computer Vision, BMVC2002,* pp.607-616, Cardiff, UK, September 2002.

[83] Jorma Rissanen, “Stochastic Complexity in Statistical Inquiry”, *World Scientific Publishing Company*, Singapore 1989.

[84] Paul L. Rosin, Tim Ellis, “Image difference threshold strategies and shadow detection”, *British Machine Vision Conf*erence, *BMVC95*, pp. 347-

356, Birmingham, UK, 1995.

[85] Paul L. Rosin, “Thresholding for Change Detection”, *British Machine Vision Conference, BMVC97*, pp. 212-221, Colchester, UK, 1997.

[86] S. Roweis and Z. Ghahramani, “A unifying review of linear Gaussian models”, *Neural Computation*, vol. 11, pp. 305-345, 1999.

[87] K.R. Sasikala and Maria Petrou, “Properties of the generalised fuzzy aggregation operators”, *Pattern Recognition Letters*, vol.22(1), pp.15-24, January 2001.

[88] Chris Stauffer, “Scene Reconstruction Using Accumulated Line-of-Sight”, *Masters thesis, Massachusetts Institute of Technology*, MIT, 1997.

[89] Chris Stauffer, W.E.L Grimson, “Adaptive background mixture models for real-time tracking”. *International Conference on Computer Vision and Pattern Recognition, CVPR99*, Fort Colins, USA, 1999.

[90] Chris Stauffer, W. Eric. L.Grimson, “Learning patterns of activity using real-time tracking”, *IEEE Trans.on Pattern Analysis and Machine Intelligence*, pp.747-757, vol.22(8), August 2000.

[91] C. Stauffer, K. Tieu, “Automated multi-camera planar tracking correspondence modelling”. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2003,* vol.1, pp 259-266, Madison Wisconsin, June 2003.

[92] Chris Stauffer, “Estimating Tracking Sources and Sinks”, Second IEEE Event Mining Workshop, Toronto, Canada, July 2003.

[93] G.P. Stein, “Tracking from Multiple View Points: Self-calibration of Space and Time*”, Image Understanding Workshop,* Montery, CA, November 1998.

[94] Christos Stergiou and Dimitrios Siganos, “Neural Networks”, http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html, London, 1996.

[95] P.F Sturm, L.Quan, “Camera Calibration and Relative Pose Estimation from Gravity”, *International Conference on Pattern Recognition, ICPR00*, pp.72-75, Barcelona, September 2000.

[96] T. N. Tan, G. D. Sullivan, K.D. Baker, “Recognising Objects on the Ground-plane”, *Image and Vision Computing*, vol.12, pp.164-172, 1994.

[97] Kentaro Toyama, John Krumm, Barry Brummit, Brian Meyers, “Wallflower: Principles and practice of background maintenance”, *International Conference on Computer Vision, ICCV99*, pp.255-261, September 1999, Corfu, Greece.

[98] Roger Tsai, “An efficient and accurate camera calibration technique for 3D machine vision”, *Conference on Computer Vision and Pattern Recognition, CVPR ’86*, pp 323-344, Miami, FL, June 1986.

[99] Y. Vardi, “Network tomography: estimating source-destination traffic intensities from link data”, *Journal of the American Statistical Association*, vol.91(433), pp.365-377, 1996.

[100] A.J. Viterbi, “Error Bounds for convolutional codes and an asymptotically optimum decoding algorithm”, *IEEE Transactions on Information Theory*, vol. IT-13, pp.260-269, February 1967.

[101] Michael Walter, Alexandra Psarrou and Shaogang Gong, “Learning Prior and Observation Augmented Density Models for Behaviour Recognition”, *British Machine Vision Conference*, BMVC99, pp.23-32, Nottingham, UK September 1999.

[102] Steven Winikoff, “Genetic Learning”, http://alcor.concordia.ca/~smw/home/genetic.html, Montreal, Canada, 2001.

[103] C.R. Wren, S.G. Rao, “Self-configuring lightweight sensor networks for ubiquitous computing”, *UbiComp,* Seattle, 12-15 October 2003.

[104] Tao Xiang, Shaogang Gong, Dennis Parkinson, “Autonomous Visual Events Detection and Classification without Explicit Object-Centred Segmentation and Tracking”, *British Machine Vision Conference,*

BMVC2002, pp.233-242, Cardiff, UK, September 2002.

[105] Ming Xu, Tim Ellis, “Colour-Invariant motion detection under fast illumination changes”, *2nd European Workshop on Advanced Video-based Surveillance, AVBS2001*, pp. 335-345, September 2001, Kingston, UK.

[106] Ming Xu, Tim Ellis, “Illumination-invariant motion detection using colour mixture models”, *British Machine Vision Conference, BMVC2001*, pp.163-172, Manchester, September 2001.

[107] Ming Xu, Tim Ellis, “Partial observation vs. blind tracking through occlusion”, *British Machine Vision Conference, BMVC2002,* pp.777-786, Cardiff, UK, September 2002.

Appendix I

Gaussian Mixture Model

A single multivariate Gaussian model is defined by the probabilistic density function (pdf):

$$p(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{e^{-\frac{1}{2}\cdot(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mathbf{m})}}{(2\pi)^{\frac{d}{2}}\cdot|\boldsymbol{\Sigma}|^{\frac{1}{2}}} \tag{I.1}$$

where $\mathbf{x}$ is an observation in a d-dimensional space, $\boldsymbol{\mu}$ is the mean and $\boldsymbol{\Sigma}$ the covariance matrix of the distribution. The importance of the Gaussian model is that it approximates successfully a wide range of phenomena that are affected by a large number of random variables with arbitrary pdfs, according to the Central Limit Theorem.

A Gaussian Mixture Model is defined as

$$p(\mathbf{x}|\theta) = \sum_{j=1}^{K} p_j \cdot p(\mathbf{x}|\boldsymbol{\mu}_\mathbf{j},\boldsymbol{\Sigma}_\mathbf{j}) \tag{I.2}$$

where K is the number of the components of the mixture model, $\boldsymbol{\theta} = \{p_1,...,p_K,\boldsymbol{\mu}_\mathbf{1},...,\boldsymbol{\mu}_\mathbf{K},\boldsymbol{\Sigma}_\mathbf{1},...,\boldsymbol{\Sigma}_\mathbf{K}\}$ is the set of all the parameters and $\{p_j\}$ is the set of the prior probabilities of the components ($\sum_{j=1}^{K} p_j = 1$).

The posterior probability of the i-component is given by the formula:

$$p(i|\mathbf{x},\theta) = \frac{p_i \cdot p(\mathbf{x}|\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)}{\sum_{j=1}^{N} p_j \cdot p(\mathbf{x}|\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)} \tag{I.3}$$

A single Gaussian model can be visualised in the K-dimension space by a set of hyper-ellipsoidals, whose surfaces represent points with equal probabilities. The centre of these hyper-ellipsoidals is defined by the mean vector $\boldsymbol{\mu}$, while their orientation is defined

by the eigenvectors of the covariance matrix $\mathbf{\Sigma}$ and their shape by the associated eigenvalues. If the lengths of the axes of a hyper-ellipsoidal are set to equal to the square roots of the associated eigenvalues of the covariance matrix $\mathbf{\Sigma}$, then this hyper-ellipsoidal contains the 68.3% of the samples of the multivariate guassian distribution. In this case, the volume of the hyper-ellipsoidal is equal to the square root of the determinant of the covariance matrix, multiplied by π: $\pi\sqrt{|\mathbf{\Sigma}|}$.

If a Gaussian model is bivariate, then it is visualised by ellipses, defined similarly by the mean vector $\boldsymbol{\mu}$ and the covariance matrix $\mathbf{\Sigma}$. The ellipse that contains the 68.3% of the samples has area equal to the square root of the determinant of the covariance matrix, multiplied by π: $\pi\sqrt{|\mathbf{\Sigma}|}$.

Appendix II

Expectation-Maximisation

Let assume that a dataset $\boldsymbol{X}=\{\mathbf{x}_1,..,\mathbf{x}_N\}$, where N is the number of the samples, is drawn by a parametric model $p(\mathbf{x}|\Theta)$, where Θ represents the set of the parameters of the model. The likelihood $L(\Theta|\boldsymbol{X})$ of the parameters given the data is defined as:

$$L(\Theta|\mathbf{X}) = p(\mathbf{X}|\Theta) = \prod_{i=1}^{N} p(\mathbf{x}_i|\Theta) \tag{II.1}$$

The aim of the Expectation-Maximisation algorithm is to adapt the parameters Θ in order to maximise the likelihood $L(\Theta|\boldsymbol{X})$, given the incomplete data $\mathbf{X}$. It is assumed that a complete dataset exists $\mathbf{Z}=(\mathbf{X}, \mathbf{Y})$, where $\mathbf{Y}$ denotes the missing data. The missing data may represent missing data values in samples of a distribution. But usually, it used to model hidden variables.

The algorithm is iterative and each iteration consists of two steps, the expectation step and the maximisation step. In the expectation step, the expected value of the likelihood of the complete data $\mathbf{Z}$ is estimated with respect to the missing data $\mathbf{Y}$, given the observed data $\mathbf{X}$ and the estimated parameters from the previous step Θ^{old}.

$$Q(\Theta,\Theta^{old}) = E\left[\log p(\mathbf{X},\mathbf{Y}|\Theta)\middle|\mathbf{X},\Theta^{old}\right] \tag{II.2}$$

In the maximisation step, the set of parameters Θ are re-estimated so that the expectation of the first step is maximised:

$$\Theta^{new} = \arg\max_{\Theta} Q(\Theta,\Theta^{old}) \tag{II.3}$$

Dempster et al [25] have proved that each iteration is guaranteed to increase the likelihood and the algorithm is guaranteed to converge to a local maximum of the likelihood function.

To apply the EM algorithm to estimate the parameters of a GMM with order K, given the observed data $\mathbf{X}$, the missing data $\mathbf{Y}$ is used to represent the components of the

mixture model that generates **X**. The expectation and the maximisation steps can be performed simultaneously, and the new estimates of the model parameters at each iteration, are given by the following formulas:

$$p_j^{new} = \frac{1}{N}\sum_{i=1}^{N} p\left(j\middle|\mathbf{x_i}, \Theta^{old}\right) \tag{II.4}$$

$$\boldsymbol{\mu}_j^{new} = \frac{\sum_{i=1}^{N} x_i \cdot p\left(j\middle|\mathbf{x_i}, \Theta^{old}\right)}{\sum_{i=1}^{N} p\left(j\middle|\mathbf{x_i}, \Theta^{old}\right)} \tag{II.5}$$

$$\boldsymbol{\Sigma}_j^{new} = \frac{\sum_{i=1}^{N} x_i \cdot p\left(j\middle|\mathbf{x_i}, \Theta^{old}\right) \cdot \left(\mathbf{x_i} - \boldsymbol{\mu}_j^{new}\right) \cdot \left(\mathbf{x_i} - \boldsymbol{\mu}_j^{new}\right)^{\mathrm{T}}}{\sum_{i=1}^{N} p\left(j\middle|\mathbf{x_i}, \Theta^{old}\right)} \tag{II.6}$$

Appendix III

Hidden Markov Models

Let assume that a dynamic system may be in one of N possible "hidden" states $\{S_1, S_2, .. S_N\}$ at each time. The system may transit from one state to another at discrete times $\{t_1, t_2, ...\}$ and results to a sequence of "hidden" states $Q=\{q_1, q_2, ...\}$. Instead of the sequence of states, a sequence of observation vectors $\mathbf{O}=\{\mathbf{o_1}, \mathbf{o_2}...\}$ in a discrete or a continuous space is retrieved that is produced by the sequence Q.

If transition probabilities between states are assumed stationary and the procedure of producing an observation vector from a state is modelled by a pdf, then a Hidden Markov Model (HMM) is used to model the system. More precisely, the HMM is defined by:

- The state transition matrix $A=\{a_{ij}\}$, i,j=1..N, where a_{ij} represents the transition probability from state i to state j:

$$a_{ij} = P\left(q_{t+1} = S_j \middle| q_t = S_i\right) \tag{III.1}$$

- The observation probability distribution $B=\{b_i(\mathbf{v})\}$, i=1..N, where $b_i(\mathbf{v})$ represents the pdf that the observation vector $\mathbf{v}$ is produced by the state i:

$$b_i(\mathbf{v}) = P\left(\mathbf{o_t} = \mathbf{v} \middle| q_t = S_i\right) \tag{III.2}$$

- The initial state distribution $\pi=\{\pi_i\}$, i=1..N, where π_i represents the probability that the state i is the first state of the sequence Q:

$$\pi_i = P\left(q_1 = S_i\right) \tag{III.3}$$

According to the probability theory, the following constrains are valid:

$$\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i = 1 \ldots N \tag{III.4}$$

If $\mathbf{v}$ is a continuous variable, then:

$$\int_{\mathbf{v}} b_i(\mathbf{v}) d\mathbf{v} \quad \forall i = 1 \ldots N \tag{III.5a}$$

Otherwise if **v** is a discrete variable, then:

$$\sum_{\mathbf{v}} b_i(\mathbf{v}) \quad \forall i = 1 \ldots N \tag{III.5b}$$

Finally:

$$\sum_{i=1}^{N} \pi_i = 1 \tag{III.6}$$

In [80], three basic problems related to HMM are identified:

1) The evaluation problem: Given a HMM λ, what is the likelihood P(**O**|λ) of a sequence of observations **O**={$\mathbf{o_1}$,..$\mathbf{o_T}$}.

2) The "uncover the hidden part" problem: Given a HMM λ and a sequence of observations **O**={$\mathbf{o_1}$,..$\mathbf{o_T}$}, what is the optimal sequence of states Q={q_1,.. q_T) that maximises the conditional probability P(Q|λ, **O**).

3) The learning problem: Given a dataset of sequence of observations {O}, what is the HMM λ that best explains the dataset or equivalently maximises the conditional probability P(O|λ).

The solution to the first problem is given by the forward-backward procedure. More specifically, a forward variable is defined as:

$$\alpha_t(i) = P(\mathbf{o_1 o_2} \ldots \mathbf{o_t} \mid q_t = S_i, \lambda) \tag{III.7}$$

The forward variables are initialised as:

$$\alpha_1(i) = \pi_i \cdot b_i(\mathbf{o_1}) \quad 1 \le i \le N \tag{III.8}$$

Then, they are inductively estimated:

$$\alpha_{t+1}(i) = \left(\sum_{i=1}^{N} \alpha_t(i) \cdot a_{ij} \right) \cdot b_j(\mathbf{o}_{t+1}) \quad 1 \le j \le N \tag{III.9}$$

The solution to the first problem is given by the formula:

$$P(\mathbf{O} \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{III.10}$$

A similar solution can be derived using the backward variable which is defined as:

$$\beta_t(i) = P(\mathbf{o}_\tau \mathbf{o}_{\tau+1} \ldots \mathbf{o}_T \mid q_t = S_i, \lambda) \tag{III.11}$$

The solution to the second problem is given by the Viterbi algorithm [100], which uses the variables $\gamma_t(i)$ and $\delta_t(i)$, defined as:

$$\gamma_t(i) = \frac{\alpha_t(i) \cdot \beta_t(i)}{P(\mathbf{O} \mid \lambda)} \tag{III.12}$$

$$\delta_t(i) = \max_{q_1 \ldots q_{\tau-1}} P(q_1 \ldots q_t = i, \mathbf{o_1} \ldots \mathbf{o_t} \mid \lambda) \tag{III.13}$$

For the learning problem, no analytic solution is available, therefore iterative methods must be used. The most popular solution is given by the Baum-Welch method [5], which is an implementation of the EM algorithm. As a consequence, the Baum-Welch method converges to a local maximum of the likelihood function.