Classification of Vehicles for Urban Traffic Scenes

Norbert Erich Buch

Submitted in partial fulfilment of the requirements of
Kingston University for the degree of
Doctor of Philosophy

June, 2010

Collaborating partner:
Traffic Directorate at Transport for London

London Streets

Kingston University London

Supervision:

Dr. Sergio A. Velastin (Director of Studies)

Dr. James Orwell

Digital Imaging Research Centre
Faculty of Computing, Information Systems and Mathematics
Kingston University
Penrhyn Road
Kingston upon Thames
Greater London
KT1 2EE
United Kingdom

http://dirc.kingston.ac.uk

Collaboration and Sponsorship:

Technology Delivery Group
Traffic Directorate, London Streets
Transport for London (TfL)
Windsor House
42-50 Victoria Street
London
SW1H 0TL
United Kingdom

http://tfl.gov.uk

Abstract

An investigation into detection and classification of vehicles and pedestrians from video in urban traffic scenes is presented. The final aim is to produce systems to guide surveillance operators and reduce human resources for observing hundreds of cameras in urban traffic surveillance. Cameras are a well established means for traffic managers to observe traffic states and improve journey experiences. Firstly, per frame vehicle detection and classification is performed using 3D models on calibrated cameras. Motion silhouettes (from background estimation) are extracted and compared to a projected model silhouette to identify the ground plane position and class of vehicles and pedestrians. The system has been evaluated with the reference i-LIDS data sets from the UK Home Office. Performance has been compared for varying numbers of classes, for three different weather conditions and for different video input filters. The full system including detection and classification achieves a recall of 87% at a precision of 85.5% outperforming similar systems in the literature.

To improve robustness, the use of local image patches to incorporate object appearance is investigated for surveillance applications. As an example, a novel texture saliency classifier has been proposed to detect people in a video frame by identifying salient texture regions. The image is classified into foreground and background in real- time. No temporal image information is used during the classification. The system is used for the task of detecting people entering a sterile zone, a common scenario for visual surveillance. Testing has been performed on the i-LIDS sterile zone benchmark data set of the UK Home Office. The basic detector is extended by fusing its output with simple motion information, which significantly outperforms standard motion tracking. Lower detection time can be achieved by combining texture classification with Kalman filtering. The fusion approach

running on 10 frames per second gives the highest result of $F1$=0.92 for the 24 hour test data set.

Based on the good results for local features, a novel classifier has been introduced by combining the concept of 3D models with local features to overcome limitations of conventional silhouette-based methods and local features in 2D. The appearance of vehicles varies substantially with the viewing angle and local features may often be occluded. In this thesis, full 3D models are used for the object categories to be detected and the feature patches are defined over these models. A calibrated camera allows an affine transformation of the observation into a normalised representation from which '3DHOG' features (3D extended histogram of oriented gradients) are defined. A variable set of interest points is used in the detection and classification processes, depending on which points in the 3D model are visible. The 3DHOG feature is compared with features based on FFT and simple histograms and also to the motion silhouette baseline on the same data. The results demonstrate that the proposed method achieves comparable performance. In particular, an advantage of the proposed method is that it is robust against miss-shaped motion silhouettes which can be caused by variable lighting, camera quality and occlusions from other objects.

The proposed algorithms are evaluated further on a new data set from a different camera with higher resolution, which demonstrates the portability of the training data to novel camera views. Kalman filter tracking is introduced to gain trajectory information, which is used for behaviour analysis. Correctly detected tracks of 94% outperform a baseline motion tracker (OpenCV) tested under the same conditions. A demonstrator for bus lane monitoring is introduced using the output of the detection and classification system. The thesis concludes with a critical analysis of the work and the outlook for future research opportunities.

to my parents

Acknowledgments

Firstly, I would like to thank Sergio Velastin and James Orwell, my supervisors, for their effort, help, motivation and invaluable guidance. They both went out of their way to help with my professional development and chances to travel to different parts of the world. The specific balance between independence and useful feedback provided a prolific work environment.

I am grateful to Transport for London for the sponsorship of this project. In particular, many thanks to the Technology Delivery Group for making me feel very welcome during my time in the office in London and for many conversations on the practical aspects of traffic management. I would like to thank the people involved in the project: Mark Cracknell, Jeremy Evans, William Lowder, John McCarthy and Derek Renaud for their guidance and feedback.

An important aspect of my work was discussing ideas and technicalities of implementation with friends and colleagues. We developed tools together to make our 'research life' easier. I want to thank Justin Cobb, Alberto Colombo, Jesús Martínez del Rincón, Hussein Ragheb and Damien Simonnet. Furthermore, Fei Yin kindly provided his tracking performance evaluation framework to test parts of the system in addition to many valuable conversations across desks. The daily lunches of the research group provided a strong and international community for my time in Kingston. Thank you guys! There are many more people, who are not named in person, to whom I am grateful.

Love and gratitude to my fiancée Marina and all my family for their encouragement and support along my educational path. These thanks extend both to my family in Austria, and my future family in law in England.

Glossary of Terms

3DHOG	3D extended histogram of oriented gradients
ANPR	Automatic number plate recognition
BFM	Boundary fragment model
CCTV	Closed circuit television
CIF	Common intermediate format (video format: 352x288)
CLEAR	Classification of Events, Activities, and Relationships
DC	Direct current
DoG	Difference of Gaussians
FFT	Fast Fourier transform
GHz	Giga Hertz
GLOH	Gradient Location and Orientation Histogram
GMM	Gaussian mixture model
GMSM	Gaussian mixture shadow model
GPS	Global positioning system
GPU	Graphics processing unit
GUI	Graphical user interface
HMM	Hidden Markov model
HOG	Histogram of oriented gradients
HSV	Hue saturation value colour space
ICA	Independent component analysis
i-LIDS	Imagery library for intelligent detection systems
IRID	Image recognition and incident detection project (TfL)
ISM	Implicit shape model
LDA	Linear discriminant analysis
LLC	Locally linear coordination

LLE	Local linear embedding
MCMC	Markov chain Monte Carlo
MoG	Mixture of Gaussians
MPF	Marginal probability field
MRF	Markov random field
NIST	National Institute of Standards and Technology
NTSC	National television system committee (TV encoding)
OpenCV	Open computer vision library
PAL	Phase alternating line (TV encoding)
PCA	Principal component analysis
PDF	Probability density function
PF	Particle filter
RAM	Random access memory
RGB	Red green blue colour space
SIFT	Scale invariant feature transform
SNP	Statistical no- parametric model
SP	Statistical parametric model
ST-MRF	Spatial- temporal Markov random field
SURF	Speeded up robust features
SVM	Support vector machine
TfL	Transport for London
TV	Television
Viper	Video Performance Evaluation Resource
Viper-GT	Video Performance Evaluation Resource - Ground Truth
Viper-PE	Video Performance Evaluation Resource - Performance Evaluation
VOC	Visual object classes challenge

List of Figures

List of Tables

Content

1. Introduction

"Concern for man and his fate must always form the chief interest of all technical endeavours. Never forget this in the midst of your diagrams and equations."

- Albert Einstein

1.1. Video Analytics for Traffic Management

Intelligent image detection systems are part of a centralised approach to modern day traffic management. This has arisen from the need for more cost effective and efficient monitoring of traffic. In turn, this has increased the scope for automatic analysis of urban traffic activity from CCTV in recent years. This increase can be contributed in part to the additional numbers of cameras and other sensors, enhanced infrastructure and consequent accessibility of data. Also the advancement of analytical techniques to process the video (and other) data together with increased computing power has enabled new applications. Video analytics is defined as computer vision based surveillance algorithms and systems to extract contextual information from video. The main concept is to aid human operators in observing video data. This can allow online and post-event detection of events of interest, which is useful for traffic management due to additional data available. The current main bottleneck of surveillance is the limitation of human resources for observing hundreds of cameras. Automatic pre- processing allows efficient guidance for the operators to pick cameras to view and accumulate statistics, with the aim to improve traffic flow. Video cameras have been deployed for a long time for traffic and other monitoring purposes, because they provide a rich information source for human understanding. Video analytics may now provide added value to

those cameras by automatically extracting relevant information and easing the bottleneck of operators viewing all cameras.

With 1200 cameras and over 100 monitors it is not feasible to continuously monitor every CCTV camera installed within Transport for London's (TfL) road network which demonstrates the bottleneck described above. In fact, it has been shown that for manual monitoring the accuracy of detection significantly decreases over time. Therefore, the development of a technology that provides automatic and relevant real- time alerts to Traffic Co-ordinators can have an immediate and long term impact on traffic management through the implementation of responsive traffic strategies.

In urban environments, there are several monitoring objectives that can be supported by the application of computer vision and pattern recognition techniques. These include the detection of traffic violations (illegal turns, one way streets, *etc*) and the identification of road users. For the latter task, the most reliable approach to date is either through recognition of the number plates (ANPR) or radio frequency transponders, which of course cannot be used for pedestrians or bicycles. Nevertheless ANPR tends to be only effective for specialised camera views (zoomed on plates) and cannot provide wide-area observation or the measurement of the interactions between road users, *etc*. that may be possible with computer vision using standard cameras. Thus, for the monitoring objectives outlined above, the detection and classification of road users is a key task. However, using general purpose surveillance cameras, this is a demanding challenge. The quality of surveillance data is generally poor and the range of operational conditions (night-time, low angle and changeable weather that affects the auto-iris) require robust techniques. The significant difference between traffic surveillance and generic object recognition is important for the understanding of the methods used. Object recognition tasks typically focus on high resolution images (mega pixel range) with

Figure 1 Illustrations of camera installations in London and the resulting views.

few constraints on the viewing angle. The Visual Object Classes (VOC) challenge (Everingham *et al.,* 2009) gives precise definitions for classification, detection and segmentation problems. The computational cost for classifying those unconstrained images is usually high.

In contrast to the above, traffic surveillance systems deal with low camera resolution. Many current installations include analogue PAL and NTSC cameras, which only provide a limited amount of visual detail of road users. The monitoring objectives generally require real-time processing, which limits the complexity of proposed approaches. The scenes are usually more constrained than object recognition, with cameras mounted on poles above roads (illustrated in Figure 1). Cameras are typically assumed to be stationary, on a 'home' position, unless operators take control over a camera. Many algorithms use this assumption to extract information when no operator is observing the camera and information may

be lost otherwise. This is significant added value for existing installations. In general, this surveillance task itself is not as well defined as it is for example for image retrieval, and no scientific benchmarking challenge has yet taken place. Prospective users of this technology have to evaluate such technology on a per case basis.

In early 2006, TfL launched the Image Recognition and Incident Detection (IRID) project. This project was tasked to review the current image processing market and see how it met TfL's detection requirements. Testing was carried out on the following criteria: congestion, stopped vehicles, banned turns, vehicle counting, subway monitoring and bus detection (Cracknell, 2007, Cracknell, 2008). Results from this testing show good performance in congestion detection (80% precision), but poor performance in tracking based detection (~20% precision), clearly showing limitations in capability. This PhD project was sponsored by TfL to investigate low level imaging techniques for urban traffic monitoring. Robustness is an important consideration to overcome the limitations of currently available systems and to aid TfL's traffic managers in their task of *Keeping London Moving*.

1.2. Scope and Outline

This thesis focuses on video analysis from urban traffic management cameras. To address the bottleneck of a limited number of operators, algorithms are to be developed for the detection and classification of vehicles from currently installed CCTV cameras in London. This provides different challenges than highway monitoring, which will be described in more detail in the literature review. The cameras can be assumed stationary and in home position if no operator is taking control of them.

Based on requirements of TfL, five generic classes are identified for the classifier:

- Bus/Lorry
- Van
- Car/Taxi
- Motorbike/Bicycle
- Pedestrian.

Portability of the system between cameras is essential, as extensive setup or training for every single camera is not feasible. Camera ground plane calibration can be assumed to be provided, as maps of roads are available or can be quickly generated manually. A demonstrator for tracking and some behaviour analysis is in the scope of the thesis, whereas more complex analysis of the generated meta data will be future work.

Literature relevant to traffic monitoring in urban and highway scenes together with generic visual surveillance technology is reviewed in chapter 2. The general framework used throughout the thesis will be introduced in section 2.6 based on the literature. Vehicle detection and classification is first solved by motion estimation and 3D models for vehicles in chapter 3. The motion estimation is state of the art for stationary surveillance cameras. The 3D model approach allows portability between cameras and the estimation of real world coordinates of vehicles. Camera calibration is sufficient to use the models for any view. However, the classification relies on motion silhouettes (binary mask) which is noisy and can be affected by camera shake, shadows, occlusion and so on.

The robustness of motion silhouettes is limited, as only very little information of the input image is available in the binary mask. The appearance information of objects is unavailable to the classifier. The use of local image patches for appearance modelling is investigated in chapter 4 to overcome the mentioned problems. In particular, the scenario of human intrusion detection for fence monitoring is solved. Local image patches with fast Fourier transform (FFT)

features show good performance for discriminating people from background based on appearance in single frames.

Local appearance from the previous chapter is integrated with the 3D framework in chapter 5. This on the one hand uses the complete image information allowing it to work on still images and on the other hand provides portability due to the novel integration with 3D models. For features, FFT as before, histogram and histogram of oriented gradients (HOG) are used. In this way, the approach moves beyond the traditional concept of motion estimation by incorporating concepts of the object recognition domain into video surveillance. The algorithm implicitly handles variable viewpoints of vehicles and camera resolutions based on a single training set.

The proposed algorithms are evaluated in more detail in chapter 6. Portability and occlusion performances are evaluated on a new data set. Tracking is incorporated into the framework and a demonstrator for bus lane monitoring is introduced. The thesis concludes with chapter 7 with a critical discussion of the work presented and the outlook for future work. Additional implementation details of the proposed framework and tables not included in the main text are available in the appendix.

1.3. Contribution

A comprehensive review of visual traffic analysis systems and related methods of computer vision is presented in chapter 2. The main gaps in literature identified are firstly the classification of vehicles based on richer information than the motion silhouette size. Secondly, coverage for urban environments is historically less than for highways. This section will introduce the contributions of the thesis in respect to the literature and the problem definition in section 1.2.

The issue of portability and vehicle classification with motion silhouettes is addressed in chapter 3 resulting in three main contributions. Firstly the use of 3D models for road user classification. In particular, matching those 3D models with closed contours extracted from motion foreground is novel. Several methods for background refinement are available. All main road users are detected and classified with a single framework. The second contribution is a unified framework for the use of 3D models, which will allow seamless integration of appearance later on. The third contribution is the system evaluation on a public data set. Evaluation results are presented on the i-LIDS data set from the UK Home Office which can be licensed by research institutions and manufacturers (iLIDS, nd). This will provide the baseline for the following chapters.

For chapter 4, the contribution is three-fold addressing the evaluation of local appearance features in 2D for visual surveillance applications. Firstly a novel saliency classifier is proposed for human intrusion detection in still images. People are not assumed upright, as is the case for most pedestrian detectors in the literature *e.g.* (Dalal and Triggs, 2005, Jones and Snow, 2008). Salient objects are detected in real- time, based on spectral texture features of local image patches. The basic classifier is extended with a novel fusion of the saliency and a simple inter-frame difference motion mask. A second extension uses Kalman filtering and allows motion silhouettes to initialise tracks to reduce detection time. The second contribution is the testing of the algorithms on the i-LIDS sterile zone data set (full 24 hours), which is used to benchmark visual surveillance systems. Comparative results with the state of the art OpenCV blob tracker are provided. Finally, detailed runtime and complexity analysis for the framework is presented.

The integration of chapters 3 and 4 addresses the issue of using appearance features for vehicle classification, which leads to the following contributions in chapter 5: Firstly, the 3D spatial models are extended to incorporate the location of

interest points from which local features are extracted. The local features are alternatively constructed from histograms of oriented gradients (HOG), standard image histograms or FFT features. The combination of 3D interest points and HOG is hence introduced as the novel 3DHOG feature. Image patches are arranged on a 3D surface rather than a 2D grid, which preserves the advantages of both 3D model and local features. Performance is evaluated, comparing 3DHOG with FFT and histogram-based local features. The second contribution is a training and classification framework based on the 3DHOG feature, which allows classification using a variable number of interest points (previous approaches required a fixed number of interest points). This framework therefore seamlessly handles variable visibility due to vehicle orientation and occlusion by always providing the same normalised model match response. This approach works independently of motion silhouettes and can be applied to stationary objects, still images or moving cameras and is therefore in principle less likely to be affected by motion segmentation issues. The third contribution is an extensive evaluation of the proposed method on the i-LIDS data set providing comparative results for chapter 3.

The algorithms proposed are tested to demonstrate capabilities in portability, under occlusion, with tracking and behaviour analysis in chapter 6. The first contribution is the evaluation on a new data set and comparison with an industrial tracker. The novel viewpoint of a high resolution camera is handled outperforming the industrial classifier. The second contribution is the extension of the 3D vehicle detector and classifier by tracking on the ground plane. A variable sample rate Kalman filter is introduced to accommodate missed observations. The classification of vehicles is used during tracking due to the novel approach of classifying before tracking. The evaluation framework of (Yin *et al.*, 2007) is used to generate rich performance figures based on ground truth containing image bounding boxes. The performance of the 3D model based ground plane tracker is

compared to a state of the art blob tracker. The final contribution of the chapter is the introduction of a bus lane monitor to generate alarms for prohibited vehicles entering a restricted zone.

In summary, the central contribution is the integration of local features with 3D models for object detection and classification (3DHOG). 3D models and local features are evaluated independently in the surveillance domain first. The 3DHOG algorithm is then tested for urban traffic analysis scenarios and its properties are investigated. The next chapter will discuss related work as background for the remainder of the thesis.

2. Review

2.1. Introduction

This chapter will focus on recent approaches for road side cameras in urban environments used by human operators, to provide automated solutions to the monitoring problems introduced in chapter 1. A previous survey (Kastrinaki *et al.*, 2003) focused on highway surveillance and on-vehicle systems. A more comprehensive review of on-vehicle vision systems for driver assistance and autonomous driving can be found in (Sun *et al.*, 2006) and a conference paper (Sun *et al.*, 2004). A review of general surveillance systems is provided in (Kumar *et al.*, 2008) and (Valera and Velastin, 2005) with a particular focus to distributed surveillance systems. Figure 2 shows some example camera views from the i-LIDS data set (iLIDS, nd) provided by the Home Office of the United Kingdom.

The remainder of the chapter is organised as follows. First, deployment of systems for video analytics is considered in section 2.2. The generic elements of traffic analysis systems are introduced with examples in section 2.3. Considering complete systems in section 2.4, the full surveillance task from reading a video stream to classifying vehicles and event recognition is described. Detailed discussions and the outlook for future research are provided in section 2.5. The chapter concludes with an overview of the thesis in respect to the review in section 2.6.

Figure 2 Example frames from the i-LIDS parked car data set (iLIDS, nd). a,b) sunny conditions with shadows and reflections on cars c) image saturation in the upper part of the image d) detail of a light car in the saturated area where only dark elements remain visible e) interlacing artefacts are commonly dealt with by removing every second video line and therefore halving the resolution f) raining condition with reflections g) rain during dusk h) headlight reflections during night.

2.2. Video Analytics deployed in the Traffic Domain

This section reviews applications and existing systems for traffic monitoring. The first part in section 2.2.1 will focus on vehicle counting, which is mainly applied to highway scenes. Automatic number plate recognition is a very specialised application typically used for tolling and discussed in section 2.2.2. The most challenging and least solved problem holding the highest research potential is incident detection in section 2.2.3.

2.2.1. Vehicle Counting

The problem of vehicle counting is most commonly solved by deploying inductive loops. Those loops provide high precision, but are very intrusive to the road pavement and therefore come with a high maintenance cost. Most video analytics systems on highways focus on counting and possibly classification to allow for more detailed statistics (Traficon, nd, Citilog, nd, Ipsotek, nd, Autoscope, nd, CRS, nd). Some systems have also been adapted for urban environments, with cameras mounted on high poles. This provides a higher viewing angle, which limits the occlusion between densely spaced vehicles, which results in similar conditions to highways. However, those highly mounted cameras are specifically for video analytics, because standard CCTV cameras for human operators are mounted lower.

2.2.2. Automatic Number Plate Recognition

ANPR is a very specialised and well researched application for video analytics. There is a vast range of companies *e.g.* (CRS, nd, Virage, nd) providing solutions for tolling, congestion charging, vehicle identification or vehicle tax verification. Cameras are highly zoomed to provide a high resolution image of the number plate, but therefore losing the context of the scene. Active infrared lighting is often used to exploit the reflective nature of the number plate. The task is simplified by the fact

that the number plate is intended to be communicated and uniquely identifiable. Toll stations of freeways have dedicated lanes with cameras, where registered users can pass slowly without stopping. In contrast, inner city congestions charge systems (*e.g.* Stockholm, London, Singapore) have to be less intrusive and operate on the normal flow of passing traffic. Point to point travel time statistics are obtained from re- identification of vehicles with time stamps across the road network.

2.2.3. Incident Detection

Work on incident detection focuses on a higher level of scene understanding than the two above approaches. Examples for highways are the detection of accidents (Traficon, nd, Citilog, nd, Ipsotek, nd, Autoscope, nd, CRS, nd) and stopped vehicles. Tunnel surveillance also focuses on smoke detection for warning of tunnel fires. Hard shoulder running has been rolled out as a pilot in the UK including video analytics from (Ipsotek, nd). The hard shoulder of a motorway is turned into a running lane during peak time, which requires reliable inspection for obstacles and monitoring for incidents during operation.

Urban environments involve a much wider range of incident detection systems and require an even higher level of scene understanding. Congestion detection is being rolled out in London (Cracknell, 2008) based on existing CCTV cameras including systems from (Ipsotek, nd). Existing systems could not demonstrate acceptable results for practical deployment for other scenarios at the time of the study. Detection of illegal parking is the objective for one data set from i-LIDS (iLIDS, nd). A high level of position accuracy is required for illegal turning, bus lane monitoring and box junctions. Those target applications also require classification of vehicles like in chapters 3 and 5 and significant context information from a zoomed out camera. A system for detecting 'car park surfing' is available from (Ipsotek, nd), which monitors if pedestrians move from car to car. This is

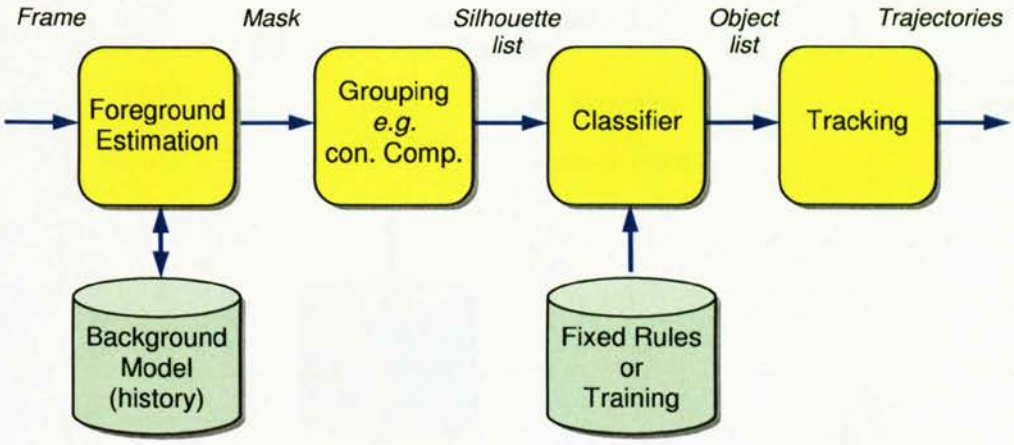


Figure 3 Block diagram for a top-down surveillance system. The grouping of pixels in the foreground mask into silhouettes that represent objects is done early with a simple algorithm without knowledge of object classes.

regarded as usual behaviour before a theft to identify target vehicles. The next section will focus on algorithmic aspects of systems, whereas section 2.4 will revisit applications in respect to the literature.

2.3. Elements of Traffic Analysis Systems

In this section, generic elements required in a traffic analysis system will be introduced. To structure the presentation, the literature has been grouped into top-down and bottom-up approaches. A typical video analytics application uses a pipeline of foreground estimation (section 2.3.1), classification (section 2.3.2) and tracking (section 2.3.4). See Figure 3 for a block diagram. A statistical model typically estimates foreground pixels, which are then grouped with a basic model (*e.g.* connected regions) and propagated through the system until the classification stage *e.g.* (Gupte *et al.*, 2002, Morris and Trivedi, 2006a, Hsieh *et al.*, 2006, Bloisi and Iocchi, 2009, Creusen *et al.*, 2009, Gao *et al.*, 2009b). Classification then uses prior information (previously learned or pre-programmed) about the object classes to assign a class label. For the remainder of the review, this class of algorithms will be referred to as 'top-down' or 'object-based', because pixels are grouped into

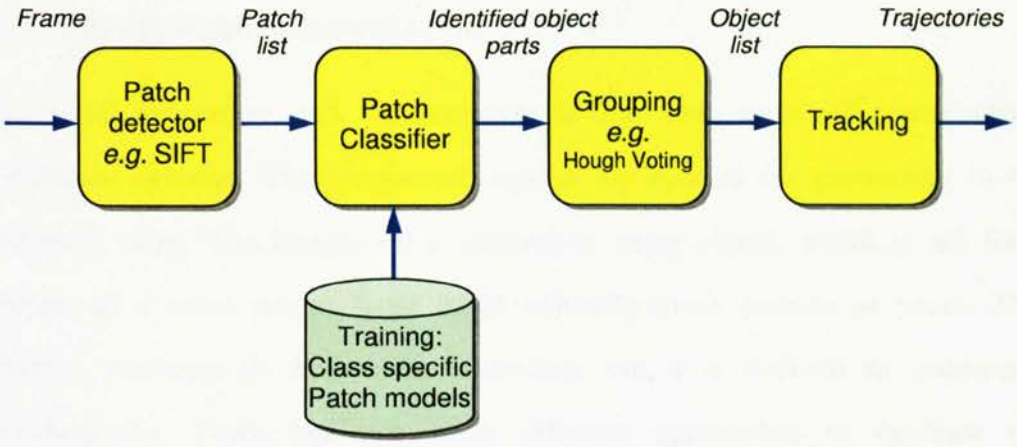


Figure 4 Block diagram for a bottom-up surveillance system. Local image patches are first extracted from the input image and classified as being a specific part of a trained object class. Those identified parts are combined into objects based on the class through a grouping or voting process. Advanced tracking concepts (Leibe *et al.*, 2008b) allow this grouping to be performed in the spatial- temporal domain, which directly produces an object trajectory rather than frame per frame object detections.

objects early during the processing. In contrast, a 'bottom-up' approach is defined as one which detects and classifies parts of an object first (block diagram in Figure 4). This initial classification of the parts uses learned prior information about the final object classes, *e.g.* an image area is classified to be a car wheel or a pedestrian head based on previously learned appearances of wheels and heads. The combination of those parts into valid objects and trajectories is the final step of the algorithm *e.g.* (Leibe *et al.*, 2004, Leibe *et al.*, 2008b, Opelt *et al.*, 2006b). This type of approach is typically used in generic object recognition. In the next section, the top-down approach will be described in more detail, including *Foreground Segmentation* and *Top-Down Vehicle Classification*. This is followed by relevant *Bottom-up* Classification approaches for traffic surveillance. The last section considers *Tracking*, which can equally be applied to both classification methods.

2.3.1. Foreground Segmentation

Foreground estimation and segmentation is the first stage of many visual surveillance systems. The foreground regions are marked for processing in the subsequent steps. The foreground is defined as every object, which is not fixed furniture of a scene where fixed could normally mean months or years. This definition conforms to human understanding, but it is difficult to implement algorithmically. There are two main different approaches to estimate the foreground, which both use strong assumptions to comply with the above definition. Firstly, a background model of some kind can be used to accumulate information about the scene background of a video sequence. The model is then compared to the current frame to identify differences (or 'motion'), provided that the camera is stationary. This concept lends itself well for computer implementation, but leads to problems with slow moving traffic. Any car should be considered foreground, but stationary objects are missed due to the lack of motion. The next sections discus different solutions for using motion as the main cue for foreground segmentation. The second approach performs segmentation based on whole object appearances (see section 2.3.1.6). This approach can be used for moving as well as for stationary cameras, but requires prior information for foreground object appearances.

2.3.1.1. Frame Differencing

Possibly the simplest method for foreground segmentation is frame differencing. A pixel by pixel difference map is computed between two consecutive frames. This difference is thresholded and used as foreground mask. This algorithm is very fast, however, it can not cope with noise, abrupt illumination changes or periodic movements in the background like trees. In (Park *et al.*, 2007), frame differencing is used to detect street parking vehicles. Special care is taken in the algorithm to suppress the influence of noise. Motorcycles are detected in (Nguyen and Le, 2008) based on frame differencing. However, using more information than just the last

frame for subtraction is preferable. This leads to the background subtraction techniques described in the next sections.

2.3.1.2. Background Subtraction

This group of background models estimates a background image which is subtracted from the current video frame. A threshold is applied to the resulting difference image to give the foreground mask. The threshold can be constant or dynamic as used in (Gupte *et al.*, 2002). The methods described below differ in the way the background picture is obtained.

2.3.1.2.1. Averaging

In the background averaging method, all video frames are summed up. The learning rate specifies the weight between a new frame and the background. This algorithm has little computational cost, however, it is likely to produce tails behind moving objects due to contamination of the background with the appearance of the moving objects. (Gupte *et al.*, 2002) and (Huang and Liao, 2004) use the *instantaneous background*, which is the current frame with detected objects removed. The regions of detected objects are filled with the old background pattern. By averaging the *instantaneous background* instead of the current frame, the tails generated by moving objects are reduced. The feedback of the motion mask could however lead to erroneous background estimations, if the threshold is set poorly. A dynamic threshold is applied to reduce this problem of never updating a region detected as foreground. Other papers report the use of averaging, usually for computational reasons: (Kanhere *et al.*, 2005, Chen and Zhang, 2007, Kanhere, 2008, Kanhere and Birchfield, 2008)

2.3.1.2.2. Single Gaussian

To improve robustness, a single Gaussian model can be used for the background. Instead of only the mean value as for averaging, the variance of the background pixels is calculated additionally. This results in a *mean image* and *variance image* for the background model. A new pixel is classified depending on the position in the Gaussian distribution, which is the statistical equivalent to a dynamic threshold. (Kumar *et al.*, 2003, Morris and Trivedi, 2006a, Morris and Trivedi, 2006b, Su *et al.*, 2007) use a single Gaussian background model.

2.3.1.2.3. Mode Estimation

(Zheng *et al.*, 2005) use the mode of the temporal histogram of every pixel to estimate the background image, which is a non parametric method. The mode estimation takes place in a constant time window. Robustness to illumination changes and long term operation are not demonstrated in the paper. The described algorithm took 230 seconds for processing 600 frames on a Pentium 4 at 3 GHz and 1 GB RAM. For the mode in the histogram to correctly represent the background, the background has to be visible dominantly during the observation period to produce a dominant peak in the histogram. This is a similar assumption as for a Gaussian Mixture Model (GMM) and holds for typical traffic surveillance applications, but fails for parked vehicles or heavy congestion. However, the algorithm is sensitive to the bin size. If the size is too small and the input pixel values vary over several bins, no distinct peak would appear. The GMM (see section 2.3.1.3) in comparison is a parametric method and models the width of the distributions. The individual Gaussians conceptually represent the bins in the histogram. A Gaussian filter is applied to the histogram in (Song and Tai, 2008).

2.3.1.2.4. Kalman Filter

A Kalman filter can be used to estimate the background image, where the colour of each pixel is modelled by one filter. The foreground can be interpreted as noise for the filter state. However, illumination changes are non Gaussian noise and violate basic assumptions for the use of Kalman filters. (Messelodi *et al.*, 2005a) proposes a Kalman filter approach which can deal with illumination changes. The illumination distribution over the image is estimated and used to adjust the individual Kalman filter states. The foreground estimation was tested in (Messelodi *et al.*, 2005b) indicating superior performance compared to the Kalman filter based algorithm proposed in (Boninsegna and Bozzoli, 2000).

2.3.1.2.5. Wavelets

A wavelet based background model is introduced in (Gao *et al.*, 2009b) in the context of urban traffic surveillance. The evaluation indicates better performance than the GMM (Stauffer and Grimson, 1999), however the test data is very limited in size.

2.3.1.3. Gaussian Mixture Model

The GMM was introduced in the seminal paper of (Stauffer and Grimson, 1999) and (Stauffer and Grimson, 2000). Each pixel is modelled as a mixture of two or more Gaussians and updated online. The stability of the Gaussian distributions is evaluated to estimate if they are the result of a more stable background process or a short term foreground process. Each pixel is classified to be background if the distribution representing it is stable above a threshold. The model can deal with lighting changes and repetitive clutter. The computational complexity is higher than standard background subtraction methods. Two images per Gaussian distribution used (typically 3 to 5) have to be kept in memory, which leads to 50MB for a 720x576 colour frame. (Veeraraghavan *et al.*, 2002) uses the GMM for observing

an intersection. (Martel-Brisson and Zaccarin, 2007) extend the GMM to deal with shadows (see section 2.3.1.5). For an introduction to Gaussian mixture models see (Power and Schoonees, 2002). The implementation of (KadewTraKuPong and Bowden, 2001) is available in the OpenCV library (OpenCV, nd) and is commonly used in research. Many researchers have adapted this model for traffic analysis (Veeraraghavan *et al.*, 2002, Zhang *et al.*, 2008b, Bloisi and Iocchi, 2009, Wang *et al.*, 2009b, Johansson *et al.*, 2009). The limitation of the approach remains the computational complexity and therefore higher time requirements compared to the simpler approaches in section 2.3.1.1 and 2.3.1.2.

An alternative to the GMM is given by (Tanaka *et al.*, 2007). A Probability Density Function (PDF) is used to detect objects in the picture. No explicit background image is kept. A pixel process is estimated for every pixel. Based on the estimated PDF, the probability for an observed pixel to occur is calculated. If the probability is high, nothing unexpected happened and the pixel is assumed to be background. If the probability is low, the pixel is assumed to be foreground. The algorithm is very cost effective, as only an estimation for a Gaussian Mixture Model is calculated. The computation for every frame involves only an update and not a recalculation of the model. At a resolution of 320x240, the algorithms takes less than 80 ms on a Pentium 4 at 3.3 GHz and 2.5GB RAM to segment a new video frame.

2.3.1.4. Graph Cuts

The foreground segmentation problem can be represented as a graph of a Markov Random Field (MRF). Every pixel of the images is represented by a node in the graph. The vertices between nodes and sources are set to a weight related to the data (data constraint). Sources represent the labels for a pixel, in this case foreground and background. Vertices between nodes are used to introduce a smoothing constraint. The graph cut separates source and sink nodes completely and leaves the

nodes connected to either to indicate that this pixel corresponds to the respective label. The advantage of graph cuts is that the optimal solution can be found in polynomial time. (Boykov and Veksler, 2005) give a general introduction to graph cuts. Applications for image restoration, stereo imaging and video blending are mentioned. (Torr, 2007) is a tutorial for applications of graph cuts. Recent applications use graph cuts for scene understanding from moving vehicles (Sturgess *et al.*, 2009). A new more general Marginal Probability Field (MPF) has been introduced in (Woodford *et al.*, 2009). MRF is a special linear case of this new MPF.

2.3.1.5. Shadow Removal

An evaluation of moving shadow detection is given in (Prati *et al.*, 2003). The authors grouped the literature into four different categories. The first category is statistical non- parametric (SNP) which considers the colour consistency of the human eye to detect shadows. An example of this is (Cucchiara *et al.*, 2001) which is used in several traffic systems (Zhang *et al.*, 2008b, Johansson *et al.*, 2009). The statistical parametric (SP) approach imposes additional spatial constraints to SNP. Two different deterministic non- model based approaches are described which use a combination of statistical and knowledge based assumptions. No single approach performs best, furthermore, the type of applications determines the best suited algorithm. Deep cast shadow positions are predicted in (Johansson *et al.*, 2009) based on GPS location, time information and 3D vehicle models. With this additional prior information, qualitative improvements are demonstrated, but no quantitative evaluation is provided.

A shadow removal technique using Gaussian Mixture Models (GMM) is introduced in (Martel-Brisson and Zaccarin, 2007). Instead of using colour consistency, the authors use the stability of states in the GMM to determine shadows. In contrast to the two groups of states in (Stauffer and Grimson, 1999),

one background state, several shadow states and several foreground states are used. The concept assumes, that shadow states are less stable than background states but more stable than foreground states. Converged shadow states are copied into a Gaussian mixture shadow model (GMSM) to prevent them from being overridden by foreground states. This model calculates the shadow volume in the RGB space rather than assuming it to be a cylinder like for the colour consistency assumption in (Cucchiara *et al.*, 2001).

2.3.1.6. Object based Segmentation

Object based segmentation relies on object detection to identify the foreground. In this section, methods are considered which detect objects in a holistic way by searching for full objects. (Sullivan *et al.*, 1996) convert the wire frame of a 3D vehicle into a gradient image by assigning a triangular grey level profile to every edge. The projected image is compared to the gradient image of the camera to find a match. This work has been followed up in (Hu *et al.*, 2004, Lou *et al.*, 2005, Remagnino *et al.*, 1998, Zhang *et al.*, 2008c). Optical flow is used in addition to wire frames in (Ottlik and Nagel, 2008) to segment vehicles in the image.

Different methods are proposed to find correspondences between 3D model projections and new images. (Messelodi *et al.*, 2005b) generates the convex hull for 3D vehicle models in the image. The ratio between convex hull overlap of model and image normalized by the union of both areas generates a matching score. Similar 3D vehicle models are matched with a motion segmented input video in (Song and Nevatia, 2007) for detection. An extension is provided in (Johansson *et al.*, 2009), which also adapts the size of vehicles. A method for rendering 3D vehicle models for matching at new viewing angles is proposed in (Guo *et al.*, 2008).

An approach with edges is used in (Kim and Malik, 2003). Horizontal and vertical edges are grouped into vehicles using a probabilistic framework. The

grouped vehicles are used for tracking in a highway surveillance application. All methods employing 3D modelling trade off additional prior information against higher computational complexity. A constant background colour is assumed for highway scenes in (Yoneyama *et al.*, 2005). This allows vehicle detection by simply taking the difference between the mean colour and a pixel. The approach would not work in urban environments with street clutter.

2.3.2. Top-Down Vehicle Classification

Classification is the task of assigning a new instance to a group of previously seen instances called the class. The classifier needs information about a new instance which is usually referred to as features. Features are extracted from the whole object according to top-down methods discussed earlier. In section 2.3.2.1 a selection of possible features is described. A machine learning algorithm is trained with instances of known classes (hence this is referred to as supervised learning) to extract discriminative information from the features (see section 2.3.2.2). The classifier then uses this learned information to assign a class label to a new instance.

2.3.2.1. Features

Classification and tracking relies on a feature extraction process, which ideally produces similar values for the instances of a given class throughout the video stream. This section gives an overview of different kinds of features, grouped by the support in the image as either a binary foreground region, the contour of this region or larger image patches.

2.3.2.1.1. Region based

Region based features are usually extracted from the whole image region of an object. In video sequences, this is mainly the area of the foreground silhouette extracted by the foreground segmentation algorithm. Image moments are often used

to generate a feature vector for the silhouette. Without any feature generation, the convex hull of the silhouette (binary mask) can be used for comparison. (Messelodi *et al.*, 2005b, Song and Nevatia, 2007, Johansson *et al.*, 2009) use such an approach for region matching. (Gupte *et al.*, 2002, Zhang *et al.*, 2007a) use length and height to classify vehicles on a highway. Rule based approaches are common, *e.g.* (Hsieh *et al.*, 2006) use size and a *linearity* feature for vehicle classification. The *linearity* feature is a measure for the roughness of the vehicle silhouette. (Huang and Liao, 2004) use size, area and length with a set of rules to classify vehicles in a highway scene. Occlusions between vehicles can produce similar effects on the silhouette, which is demonstrated in (Zhang *et al.*, 2008b) where a similar measure is used for occlusion reasoning. For vehicle classification, (Morris and Trivedi, 2006a) and (Morris and Trivedi, 2006b) use 17 different region features including 7 moments for 7 classes. A comparison between image based features (IB) like pixels and image measurements features (IM) like region size is given. Both feature types are used with Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) as dimensionality reduction technique. IM with LDA was used for the final algorithm as it gave the best performance. The features are classified using a weighted *k*- nearest neighbour algorithm (section 2.3.2.2.4). A Kalman filter (section 2.3.4.1.1) is used to track the foreground regions based on the centroids. The evaluation on a 24 hour test sequence recorded by the authors shows a classification accuracy of 74.4% for independent tracking and classification. The accuracy can be increased to 88.4% by combining tracking and classification and therefore rejecting single miss- classification. Further work of the authors incorporates HOG features for in- vehicle systems (Gandhi and Trivedi, 2007). In (Alonso *et al.*, 2007), initial bounding boxes for vehicles are generated based on edges, which assumes that street clutter does not exhibit similar edge patterns. The bounding boxes are verified by symmetry and corner detection inside this region.

2.3.2.1.2. Contour based

Contour based features only take the edge of a silhouette into account. The distance between contour points is used as a similarity measure. Processing is performed on closed contours as extracted from video sequences. The contour including edges is used in (Hu *et al.*, 2004, Lou *et al.*, 2005, Remagnino *et al.*, 1998).

A common problem when dealing with contours is occlusion between vehicles, especially in urban environments. An algorithm proposed in (Zhang *et al.*, 2008b) can resolve occlusions between two vehicles by considering convexity of the shape. The convex outline of a contour of two vehicles will have dents, which can be identified to separate two vehicles, if the occlusion is not severe. The contour signature is used in (Zhang *et al.*, 2008a) for vehicle classification from side views.

2.3.2.2. Machine Learning

Machine learning techniques are used to generate a discriminative classifier from training data and to assign class labels to unseen data. An important property of the learning technique is the supervision during learning. This describes the amount of labelling information required of the training data. Labelling can range from simply tagging an image with a class to completely segmenting the image manually and labelling individual parts of objects. Weak supervision might involve unlabelled or wrongly labelled images. Ground truth is similar information and required for evaluation. The classifier output for test data is then compared to this manually generated ground truth. Large amounts of ground truth are required to provide evaluation with high statistical confidence. Section 2.5.2 will look into common data sets, which is important to share the effort in generating this ground truth. A good overview of machine learning techniques can be found in (MacKay, 2003). In the next sections, first distance measures and clustering for training are introduced before discussing different classifier architectures.

2.3.2.2.1. Distance Measures

Features are commonly represented as vectors in an *N*- dimensional feature space. This representation allows the definition of a distance (*i.e.* difference) between two vectors, which can be used during clustering and especially classification to measure similarity between features. Many distance measures are available with various properties. Firstly, the Manhattan distance calculates the sum of the absolute difference along every coordinate axis between the vectors. This results in the least computational effort but complex mathematics. Secondly, the Euclidean distance (Gao *et al.*, 2009a) returns the geometric distance between two vectors. Due to the square and square root, computational complexity is increased. It is possible to normalise the Euclidean distance along every axis to reduce the effect of non spherical point clouds. The Mahalanobis distance is similar to a normalised Euclidean. The variance of the data along a coordinate axis is used for normalisation. The covariance matrix of the data needs to be calculated for this reason. This normalisation transforms the data cloud into a spherical shape. This distance is used for the training in chapter 5. For histogram comparison, the Bhattacharyya distance is used in (Acunzo *et al.*, 2007). The χ^2 - distance is used in (Ma and Grimson, 2005) as a distance measure. It has similar properties to the Mahalanobis distance, however, it does not require the calculation of the covariance matrix. The paper describes a system for distinguishing between two classes of vehicles. The vehicles are presented centred in the image and at the same scale at high resolution to the algorithm which is a high degree of supervision. A set of modified SIFT features (Scale Invariant Feature Transform in section 2.3.3.1.2) is calculated on edge points to give a rich representation for the image. Generated feature vectors are labelled according to the training vector with the smallest χ^2 - distance. A constellation model is used to find the most probable vehicle class based on the positions of the observed feature vectors. This was evaluated for two separate

cases of binary classification. In the first case, 50 cars and 50 minivans were randomly chosen from the sample pool for training. The testing was performed on 200 samples from each class taken from their own data. About 98% accuracy is reported for that case. The test between sedans and taxis resulted in a slightly lower accuracy. More detailed results are given for different shape models of the probabilistic framework (refer to section 2.3.3.3 for shape models).

2.3.2.2.2. Dimensionality Reduction

For feature vectors, not all dimensions are necessarily statistically independent. Dimensionality reduction can be applied to reduce the data to the significant dimensions and in this way speed up processing or simplify classification. The classic method is Principal Component Analysis (PCA). This technique performs an orthogonal coordinate transformation of the feature space. The eigenvectors of the covariance matrix of the training data with the highest eigenvalues are used as new coordinate axes. This transformation ensures that the largest data variance is represented along the coordinate axes. Neglecting small eigenvalues which correspond to less significant deviations in the data reduces the dimensionality of the feature space. (Zhang *et al.*, 2005) uses this concept with SIFT feature vectors to generate PCA- SIFT features. PCA has been applied directly on candidate images for vehicle detection at night- time in (Thi *et al.*, 2008, Robert, 2009a, Robert, 2009b). (Morris and Trivedi, 2006a) use Linear Discriminant Analysis (LDA), which is a similar concept, for vehicle classification. (Chen and Zhang, 2007) use Independent Component Analysis (ICA) which separates the data into independent sources in addition to the orthogonal coordinate base of PCA. The paper introduces a vehicle classification algorithm. Standard foreground segmentation is performed to get bounding boxes of vehicles. The pixel values inside the bounding boxes form the feature vector. Independent Component Analysis (ICA) is performed on the

training images to reduce the dimension of the feature space. This is similar to the image based feature (IB) described in (Morris and Trivedi, 2006b) in section 2.3.2.1.1. To assign one of the three class labels to new feature vectors during operation, three one class Support Vector Machines (SVM) are used (see section 2.3.2.2.4). The SVMs are trained with 50 vehicles each. Three tests are conducted with 150 sample vehicles randomly chosen from the author's own sample pool. The reported performance is 65% recall at 75% precision. The ICA based algorithm is shown to outperform a Principal Component Analysis (PCA) based baseline algorithm, however, (Thi *et al.*, 2008, Robert, 2009a, Robert, 2009b) report much higher performance with their PCA based approach.

Several non- linear embedding methods are compared in a review (van der Maaten *et al.*, 2009): Isomap, Maximum Variance Unfolding, Kernel PCA, diffusion maps, Locally Linear Embedding (LLE), Laplacian Eigenmaps, Hessian LLE, Local Tangent Space Analysis, Locally Linear Coordination LLC, and manifold charting.

2.3.2.2.3. Clustering

Clustering is performed on the training data. If the training data only contains object features, unsupervised clustering would need to identify the number of classes or clusters in the data and the correspondence of the training samples to those clusters. As this general clustering problem has not been solved satisfactorily, *k*- means clustering is commonly performed. This clustering technique groups the training samples into a specified number of groups based on the distance between features. (Morris and Trivedi, 2006a) uses this clustering technique for vehicle classification. Lighting conditions are clustered in (Acunzo *et al.*, 2007). Hierarchical clustering builds a cluster tree, which allows cutting off branches at different levels and sizes.

Metrics other than the final cluster number can be used for this cutting, which allows more flexibility.

A related technique is the generation of a codebook or alphabet for object classification. This is usually applied if several local feature vectors are used to specify an object. The class label for every feature vector is known from supervision or from a previous clustering of the objects. The distance between feature vectors is used to group them together. Every group of feature vectors is replaced by one codebook entry holding all class labels of the individual vectors. This approach can increase the speed of the final classifier and reduce the amount of training and data storage, as shown in (Leibe *et al.*, 2004, Opelt, 2006, Leibe *et al.*, 2008b) for bottom-up object detection. The same concept is termed 'visual dictionary' in (Serre *et al.*, 2007) and used for vehicle classification in (Wijnhoven *et al.*, 2008, Wijnhoven and de With, 2009, Creusen *et al.*, 2009).

2.3.2.2.4. Classifiers

Classifiers map a new unknown object instance with extracted feature vector to a known class or perhaps no class. This mapping process depends on what was previously learned from training data. Different ways for generating and performing this mapping are outlined in the next sections.

Nearest Neighbour Classifier

The nearest neighbour classifier is the simplest non parametric classifier for a feature vector. The distance between a new feature vector and every vector of the training set is calculated. Any distance measure can be used for that purpose. The class label of the closest training vector is assigned to the new vector. To improve robustness, the *k*- nearest neighbour algorithm can be used. The class label for the new class is determined by the *k*- nearest training vectors. Both methods require

many distance calculations and do not scale very well for large training sets in terms of computational complexity and memory requirements. There is no time requirement for training, however, the classification time increases with the training size. (Morris and Trivedi, 2006a, Hsieh *et al.*, 2006) use this method to classify vehicles based on binary foreground features. In the seminal paper for SIFT (Lowe, 1999), corresponding interest points are found using the nearest neighbour algorithm in the feature space. A further extension is the weighted *k*- nearest neighbour algorithm. For this case, the class membership is defined by weights which results in a softer decision boundary. (Morris and Trivedi, 2006b) use this algorithm to improve robustness against outliers.

Support Vector Machines

An introduction and review of kernel based learning used for Support Vector Machines (SVM) can be found in (Muller *et al.*, 2001). SVM perform classification using linear decision hyper-planes in the feature space. During training, the hyper-planes are calculated to separate the training data with different labels. (Dalal and Triggs, 2005, Chen and Zhang, 2007, Serre *et al.*, 2007, Thi *et al.*, 2008, Wijnhoven and de With, 2009, Creusen *et al.*, 2009) use a SVM for vehicle classification. If the training data is not linearly separable, a kernel function can be used to transform the data into a new vector space. The data has to be linearly separable in the new space. Support vector machines scale well for large training sets. The complexity for training increases with the number of training samples, however, the classification is independent of it. The generic approach does not provide confidence measures for the classification. There are extensions which derive a confidence based on the distance of a feature vector to the hyper-planes, which is not always reliable.

Probabilistic Frameworks

Given that real world measurements have uncertainty, probabilistic frameworks estimate the (posterior) probability based on observed data and prior knowledge. For example, the posterior probability of a vehicle belonging to class A is calculated from the image data and the prior knowledge of how frequent vehicles of class A are observed. The vehicle detection system presented in (Song and Nevatia, 2007) uses a Bayesian framework with Markov chain Monte Carlo sampling. First, a foreground map is computed using background subtraction. A proposal map is computed from the foreground map, indicating likely vehicle centroids. The distance of points from the boundary of the foreground map indicates the likelihood in the proposal map. A Bayesian problem is formulated for the vehicle positions. The proposal eliminates overlapping vehicles in 3D space and is evaluated by the match between foreground map and projection silhouettes of the 3D models. A Markov chain Monte Carlo (MCMC) algorithm is used to search for several good solutions. The MCMC generates new states by changing the number of vehicles, the positions and orientations. Tracking between frames is performed by a Viterbi optimisation algorithm which finds the optimal track through the set of solutions for every frame. Other works (Kim and Malik, 2003, Hsieh *et al.*, 2006) use probabilistic frameworks for vehicle detection and tracking.

2.3.3. Bottom-up Classification

This section discusses literature for bottom-up approaches. An introduction to this concept, which is traditionally used for generic object recognition is given in (Pinz, 2005). As discussed at the beginning of section 2.3, this involves detecting parts of objects and classifying them, before they are grouped to objects. The next section introduces interest point descriptors, which are used to extract discriminative features from images patches. Section 2.3.3.2 covers the learning technique of

boosting, which has proved to be very powerful when used with interest points. Sections 2.3.3.3 and 2.3.3.4 introduce spatial models for interest points.

2.3.3.1. Interest Point Descriptors

Interest points (also referred to as key points) are image positions, from which features are extracted. Those points may be uniformly sampled in the image space (Dalal and Triggs, 2005, Dalal *et al.*, 2006), in a 3D surface space (chapter 5) or defined by a saliency detector as in Harris corners, Difference of Gaussians (Lowe, 1999), Hessian (Bay *et al.*, 2006), *etc.* A comprehensive comparison of local patch features can be found in (Mikolajczyk and Schmid, 2005, Zhang *et al.*, 2007b), including a temporal extension in (Wang *et al.*, 2009a) where it is shown, that the performance of interest point descriptors is mostly independent of interest point detectors.

2.3.3.1.1. Basic Patch Based Descriptors

The simplest patch based feature vector is the collection of values of the image pixels. In (Agarwal *et al.*, 2004) this approach is used to generate an alphabet of patches for object classification. The distance measure between patches is defined by the cross- correlation of them. The correlation function is very sensitive to size and illumination changes of the image. This fact encourages other feature transformations which can deal with changing conditions. The following paragraphs introduce several solutions.

Using a histogram rather than pixel values allows for more spatial invariance. The seminal paper for those concepts is (Lowe, 1999) followed up by many other algorithms (Dalal and Triggs, 2005, Mikolajczyk and Schmid, 2005, Bay *et al.*, 2006).

Binary edges can provide normalised input for feature descriptors. Illumination conditions are mostly removed during edge detection. (Opelt *et al.*,

2006a, Opelt, 2006, Ma and Grimson, 2005, Kim and Malik, 2003) use the Canny edge detector to generate features.

2.3.3.1.2. Scale Invariant Feature Transformation (SIFT)

The Scale Invariant Feature Transformation (SIFT) was introduced in the seminal paper of (Lowe, 1999). The local features generated are invariant to image scaling, translation and rotation and partially invariant to illumination changes and affine projection changes. The feature vectors are generated at maxima of the scale space of the gradient input image. In addition to the 160- dimensional feature vector, the characteristic scale and orientation of every interest point is calculated. Conceptually, a SIFT feature uniquely describes the appearance of salient points in the image, which will remain salient even if the image is resized, rotated or the illumination is changed. The SIFT features can be used to find point to point correspondences in two different images of the same object. (Opelt *et al.*, 2006a) combines SIFT features and other local features for generic object recognition. (Zhang *et al.*, 2005) uses a derivation of SIFT, the PCA-SIFT (Principal Component Analysis- SIFT) for generic object recognition. The local features are used in combination with global edge features in an AdaBoost (adaptive boost) classifier. Modified SIFT descriptors are used in (Ma and Grimson, 2005) to generate a rich representation of vehicle images. (Gao *et al.*, 2009a) uses re-identified SIFT interest points between frames for tracking vehicles in urban scenes.

2.3.3.1.3. Speeded Up Robust Features (SURF)

The SURF descriptors are introduced by (Bay *et al.*, 2006). The descriptor aims for applications of correspondence finding between images, as in SIFT and similar descriptors. However, the design focuses on computational speed hence allowing loss of performance. The use of box filters instead of Gaussian filters in the case of

(Lowe, 1999) reduces computational complexity. Haar wavelet responses in sub-regions around an interest point are used to generate the feature vector, which can be calculated with integral images in this way.

2.3.3.1.4. Histograms of Oriented Gradients (HOG)

The concept of grids of Histograms of Oriented Gradients (HOG) was introduced by (Dalal and Triggs, 2005). To calculate the feature vector, the gradient input image window is divided into a grid of cells. For every cell, a histogram of the gradient orientation in pixels is calculated. The histogram represents an eight dimensional local feature vector. The vectors of all cells are concatenated to give one global feature vector for the image window. In the original paper, this vector is used to detect pedestrians. This concept is extended to vehicle detection in chapter 5 by introducing 3DHOG which uses 3D model surfaces rather than 2D grids of cells. This allows for the algorithm to resolve scale and use a single model for variable viewpoints of road users.

2.3.3.1.5. Other Descriptors

There have been a wide range of other descriptors introduced in the literature. The Boundary Fragment Model (BFM) is introduced in the seminal paper of (Opelt *et al.*, 2006b). The model uses only segments of contours for generic object recognition. The idea of local interest point features as used in (Lowe, 1999, Leibe *et al.*, 2004, Crandall *et al.*, 2005, Opelt *et al.*, 2006a) is extended to boundary elements. The Chamfer distance measure is used to generate a codebook of fragments in training and to classify newly seen boundary fragments to codebook entries. The use of a Canny edge detector to generate the boundary fragments allows the model to be used with still images.

Another extension of the SIFT descriptor is gradient location and orientation histogram (GLOH) in (Mikolajczyk and Schmid, 2005). A larger feature vector with finer quantisation than SIFT is extracted and the dimensionality is reduced using PCA based on a large training set.

2.3.3.2. Boosting

Boosting is a method to improve the performance of a simple (possibly poor) classifier. It is very popular in conjunction with local feature descriptors to also improve computational speed by selecting an optimal subset of input features. Adaptive Boosting (AdaBoost) was first introduced in (Freund, 1995) as an extension to boosting. An introduction to AdaBoost is given in (Freund and Schapire, 1999). AdaBoost uses weak classifiers, which only need to perform better than random. Weights for those weak classifiers are learned during training. Every round of training changes the weights of training images to force the classifier to be trained on difficult examples. The weighted weak classifiers result in a final strong classifier. The basic AdaBoost algorithm performs binary classification and is robust against over fitting. The original paper (Viola and Jones, 2004) uses a cascade of AdaBoost classifiers with underlying Haar filters for face detection. The success of this face detector increased the popularity of AdaBoost for computer vision. The same authors used a temporal extension of their algorithm for pedestrian detection in road surveillance (Jones and Snow, 2008). (Zhang *et al.*, 2005) perform generic object recognition with a binary multi layer AdaBoost network. In (Opelt *et al.*, 2006a, Opelt *et al.*, 2006b), binary AdaBoost is used for generic object recognition where boosting automatically performs the feature selection. An extension to multiple classes and incremental learning is introduced in (Opelt *et al.*, 2006c, Opelt, 2006). (Khammari *et al.*, 2005) uses boosting of gradient features to detect vehicles in road scenes. (Acunzo *et al.*, 2007) uses a boosted classifier for illumination condition detection (day, night, *etc*).

2.3.3.3. Explicit Shape

Explicit shape implies direct modelling the spatial relationship between parts of objects detected. Various different models for the shape are introduced here with relevance for traffic surveillance.

2.3.3.3.1. k-fans

The *k*-fan model was first introduced in (Crandall *et al.*, 2005) to schematise part based object recognition. The parts of an object are divided into reference nodes and regular nodes of a graph. The parameter *k* represents the number of reference nodes. Every reference node has a spatial relation to every other node in the graph. By changing *k* from 0 to the total number of nodes, the spatial prior can be changed from no shape modelled to a full rigid structure. Most shape models are related to *k*-fans. (Kim and Malik, 2003) use a 1-fan model to group edges of a highway scene into vehicles. The camera calibration is used to model the 3D appearance of vehicles. (Ma and Grimson, 2005) use a constellation model similar to 1-fan for vehicle detection based on Scale Invariant Feature Transformation (SIFT) features. The HOG (Dalal and Triggs, 2005) and 3DHOG (chapter 5) algorithms use a fully connected graph.

2.3.3.3.2. Implicit Shape Model (ISM)

The implicit shape model (1- fan) is introduced in (Leibe *et al.*, 2004) and explained in more detail in (Leibe *et al.*, 2008a). Image patches at key points of objects are learned during training. In addition to the object label, a probability density function for the relative position in the object is provided. The evidence for object positions is accumulated based on those positions through generic Hough voting. In the case of Hough transform for line detection, every pixel of the image contributes to possible lines in the angle and position space. If many pixels vote for one angle and

one position, this line is detected. A similar concept is used for object voting in (Lowe, 1999). Every detected SIFT interest point votes for its corresponding object centroid in *x-y* voting space. The maximum in this space defines the detected and classified object at a position. This method is extended using different features and distance measures in (Leibe *et al.*, 2004, Leibe *et al.*, 2005, Leibe *et al.*, 2007, Cornelis *et al.*, 2008, Leibe *et al.*, 2008b, Opelt *et al.*, 2006a, Opelt *et al.*, 2006b, Opelt *et al.*, 2006c).

A similar approach is used in (Agarwal *et al.*, 2004), however, the relations between detected parts are used to generate a feature vector. Both methods use pixel values of the image patches. A good example for bottom-up surveillance based on ISM is (Leibe *et al.*, 2008b), where road users are tracked from a static urban surveillance camera. The framework was first introduced in (Leibe *et al.*, 2007) based on a generic object detector (Leibe *et al.*, 2004) with implicit shape model for vehicle detection from a moving camera. This work shows how bottom-up object detection approaches can be used for traffic analysis. The algorithm is demonstrated to perform, in urban environments, similar to the state of the art on moving stereo, while most foreground segmentation methods discussed in section 2.3.1 would not work for such a scenario. The limitations of this approach are lower detection ratios compared to typical bottom-up approaches and higher computational complexity.

2.3.3.3.3. Alphabets

The concept of alphabets is introduced to reduce the number of training samples. Instead of using every single feature vector from training, similar vectors are combined. The resulting entry holds a list of class labels and could take several positions in a shape model. This concept is used in (Leibe *et al.*, 2005, Ma and Grimson, 2005, Opelt, 2006, Wijnhoven *et al.*, 2008, Wijnhoven and de With, 2009, Creusen *et al.*, 2009).

2.3.3.4. Object Classification without Explicit Shape Structure

A solution for generic object recognition without shape structure is given in (Opelt *et al.*, 2006a) and commonly referred to as 'bag of words' (0- fan). A large set of different key point features is extracted from images. An AdaBoost classifier is trained with those features. This training procedure automatically selects the most discriminative features for the final classifier. An additional boosting layer is introduced in (Zhang *et al.*, 2005). This second layer uses global features to improve the classification.

2.3.3.4.1. Object Recognition with Hierarchy

The introduction of hierarchy in object recognition is mainly related to biological research. For example, (Ullman, 2007) discusses the structure of the human visual cortex and derives a tree style object hierarchy. The features of objects are based on image patches. (Serre *et al.*, 2007) present something that is more relevant for computer vision applications. There, a complete object recognition and segmentation system is implemented using a visual cortex structure. Four layers are used, which perform simple filtering, complex searching and a repetition of those two. A comparable result to state-of-the-art computer vision is achieved with that biologically inspired system. This concept is used in traffic surveillance in (Wijnhoven and de With, 2007, Wijnhoven *et al.*, 2008, Creusen *et al.*, 2009, Wijnhoven and de With, 2009). A standard foreground estimation method and motion tracker (no details are provided) generate vehicle images, which are passed through a sequence of simple and complex layers represented by Gabor filters and a support vector machine (SVM) classifier. A different appearance classifier is trained for every 90 degree of viewing angle. In contrast, the algorithm in chapter 5 can operate on arbitrary viewing angles. On the same data set, (Wijnhoven and de With, 2007) outperform (Ma and Grimson, 2005). The authors have moved this

concept towards a bottom-up approach in (Wijnhoven *et al.*, 2008, Wijnhoven and de With, 2009, Creusen *et al.*, 2009). Feature vectors are now extracted from interest point locations rather than a uniform density over the whole image patch.

2.3.4. Tracking

Tracking is used to measure vehicle paths in video sequences. This is performed in two steps: Firstly, features for the object or foreground regions are generated in every video frame (see section 2.3.2.1). Secondly, a data association step has to provide correspondences between the regions of consecutive frames based on the features and a dynamic model. Temporal consistency constraints are required to avoid confusion of tracks and to smooth noisy position outputs of detectors. The data association step can use the same distance measure as machine learning algorithm, see section 2.3.2.2.1. The classification result and location in the image is typically included in the feature for this association. The next sections discuss motion models for tracking in traffic applications and possible data association based on prediction.

2.3.4.1.1. Kalman Filter

The Kalman filter was originally introduced in (Kalman, 1960) and has been successfully used in many applications including missile tracking. The optimal state of a linear time invariant motion model is estimated assuming Gaussian process and measurement noise. The prediction stage of the Kalman filter is used to extrapolate the position of objects in a new frame based on a constant velocity constraint. The prediction can be associated with new measurements or can be used to trigger detectors. A correction step uses the detection as measurement and updates the filter state. This concept is used in (Morris and Trivedi, 2006b, Messelodi *et al.*, 2005b, Rad and Jamzad, 2005, Song and Nevatia, 2007, Johansson *et al.*, 2009, Bloisi and

Iocchi, 2009) for tracking. Kalman filters propagate a single object state between frames compared to multiple hypotheses for particle filters in the next section. The extended Kalman filter (EKF) can facilitate non- linear models.

2.3.4.1.2. Particle Filter

The particle filter is a generalisation of the Kalman filter introduced in (Gordon *et al.*, 1993). A recent tutorial (Doucet and Johansen, 2009) reviews the filter and relevant concepts. It allows for multiple hypotheses to be propagated between frames by modelling arbitrary probability density functions by sample particles. This overcomes the constraint of a single Gaussian distribution of Kalman filters. (Isard and Blake, 1998) introduced the particle filter into the computer vision domain. The filter is used for traffic videos in (Nummiaro *et al.*, 2003, Bardet and Chateau, 2008, Mauthner *et al.*, 2008, Nguyen and Le, 2008, Wang *et al.*, 2009b, Gao *et al.*, 2009a).

2.3.4.1.3. Spatial- Temporal Markov Random Field

The Spatial- Temporal Markov Random Field (S-T MRF) is introduced by (Kamijo *et al.*, 2000, Kamijo *et al.*, 2001a, Kamijo *et al.*, 2001b) for vehicle tracking in urban traffic scenes. The input image of resolution 640 x 480 is divided into blocks of 80 x 60 pixels. Every block is represented by a node in a S-T MRF, which is modelled as a graph like in section 2.3.1.4. The S-T MRF is used to generate vehicle labels for the blocks. Adjacent blocks as well as blocks in consecutive frames are considered neighbours for the model. A solution for the object map (nodes of the S-T MRF) of the current frame is found based on the current image, the previous image and the previous object map. The result is used in a Hidden Markov Model (HMM) to detect events like vehicle passes or collisions. (Kamijo

and Sakauchi, 2002, Kamijo *et al.*, 2004) is an extension to the earlier work introducing incident detection in tunnels.

2.3.4.1.4. Graph Correspondence

A system for region tracking based on graph correspondence is introduced in (Gupte *et al.*, 2002) for vehicle tracking. Every region in a frame is represented by a node in the graph similar to MRF. One vertex leaving every node is generated for two consecutive frames. The destination node of the vertex is determined by the best overlap score of the image regions. Due to this bidirectional structure of the graph, splitting and merging of region during tracking can be handled. To avoid conflicts in the graph, adding conflicting vertexes is suppressed. (Huang and Liao, 2004, Veeraraghavan *et al.*, 2002) use the graph correspondence for vehicle tracking and classification. In (Taj *et al.*, 2008), vehicle and pedestrian tracking is evaluated on the CLEAR data set (CLEAR, 2007) and uses greedy graph correspondence tracking based on (Shafique and Shah, 2005). Dynamic programming approaches can be used to find an optimal path through nodes of several frames. (Song and Nevatia, 2007) uses the Viterbi algorithm to find the optimal vehicle constellations over several frames.

2.3.4.1.5. Event Cones

The concept of event cones to find space time trajectories is introduced in (Leibe *et al.*, 2007). Every object observation in a frame is assigned an event cone, which in turn represents a volume of possible object positions in the future and the past. The shape of the cone is determined by the dynamic model of the object, similar to Kalman filters. Object detections of all frames are accumulated to allow a probabilistic framework with an optimisation step to select the optimal set of trajectories to explain the full history of observations. This allows tracks to be split

retrospectively, which is traded off against optimisation of a growing data set for long video sequences. In addition, a real time scene understanding system might be presented with a continuously changing interpretation of the past video. Performance of this approach is demonstrated for an urban surveillance task in (Leibe *et al.*, 2008b, Cornelis *et al.*, 2008).

2.4. Complete Traffic Analysis Systems

This section covers traffic surveillance systems which could be used in a control room environment for traffic management. By distinguishing between urban and highway scenes, a higher coverage of highway applications in the literature is shown similar to the deployment discussed in section 2.2. This is partly due to the easier conditions on a highway with usually more homogeneous and constant flow than in urban areas. In addition, the distance between vehicles is larger and reduces the amount of occlusion. Figure 2 on page 11 shows some challenging examples from an urban environment.

2.4.1. Urban

The challenge for monitoring urban traffic is the high density of vehicles and the low camera angle. The combination of both leads to a high degree of occlusion. In addition, the clutter on the streets increases the complexity of scenes. The literature is divided up into 2D approaches, which operate in the domain of the camera view and 3D approaches (section 2.4.1.2) that employ some degree of 3D modelling or reconstruction. Both demonstrate comparable performance.

2.4.1.1.Analysis in the Camera Domain

This section deals with systems that work directly in the camera coordinate domain. An early real time monitoring system for intersections is proposed in (Veeraraghavan *et al.*, 2002). A standard Gaussian Mixture Model is used for

foreground segmentation. Tracking of foreground regions is done with graph correspondence. The tracked objects are classified into pedestrians and vehicles based on the main orientation of the bounding box. Example images for different weather conditions are shown; however, there is no quantitative evaluation of the performance of the system. A support vector regression based background model is used in (Wang *et al.*, 2006). Shape based data association in tracking feeding back to the detection shows to significantly improve the results. A multi agent framework performs tracking under occlusions in (Guha *et al.*, 2006). Tracking performance of 61% is reported on 30 minutes of the authors' surveillance video.

A system for detecting parked vehicles is introduced by (Park *et al.*, 2007). Camera homography is used to generate a normalised ground plane view. Based on a frame differencing motion map, *parking in* and *out* conditions are calculated. A state machine is used to track the speed changes of vehicles until stopping to generate those conditions. The system is evaluated with 24 hours of video data from two different sites. A good detection rate of 94.7% is reported on their own data.

Interest points are tracked independently at urban intersections in (Saunier and Sayed, 2006). This provides robustness against errors in the background estimation and can deal with changing viewing angle, as no prior assumption to the constellation of feature points is made. The tracking performance is between 85% and 94% depending on the data set. Whole vehicle parts rather than individual points are tracked with particle filters in (Mauthner *et al.*, 2008) operating on very low resolution images.

Finally, there are two papers looking at specialised urban traffic applications. (Nguyen and Le, 2008) focuses on motorcycle tracking with multi-modal particle filters. A recall rate for counting of 99% is demonstrated for videos from Vietnam. In an urban setting in Venice, boats are tracked in (Bloisi and Iocchi, 2009). GMM is combined with optical flow and a Kalman filter to track and count

boats along the Grand Canal. Counting accuracy is 94% for a 2 hour sequence, which is particularly challenging due to waves on the water.

2.4.1.2. 3D Modelling

Systems in this section use explicit 3D modelling. A real time system is introduced in (Messelodi *et al.*, 2005b) to track and classify vehicles at intersections. 3D models are used to initialise an object list for every fifth frame based on the convex hull overlap of model projection and motion map. Camera calibration is required for this operation. A feature tracker follows the detected objects along some frames before a new initialisation takes place. The tracker is used to speed up operation, as the 3D operation would not be fast enough to operate on every frame in real time. The objects are classified into 8 classes based on a two-stage classifier. The first stage evaluated the convex hull, the second layer uses pixel appearance (colour) for classes with similar convex hull. The performance is evaluated on 45 minutes of video data from two different sites. The total classification rate is given with 91.5% for the test data of the authors.

The use of 3D wire frame models for vehicle detection and classification was proposed in (Sullivan *et al.*, 1996, Tan *et al.*, 1998). First, a hypothesis for a vehicle position is generated in a search window. To do that, 1D profiles along the three axes of cars (horizontal forward, sideways and vertical) are correlated with trained templates. The hypothesis is verified by correlating the gradient input image with the wire frame image. The wire frame image is generated using the camera calibration to project the wire frame and replacing every line with a three pixel wide triangular grey level profile. This line of research is followed up in (Remagnino *et al.*, 1998, Hu *et al.*, 2004, Lou *et al.*, 2005). A similar work using optical flow to find detection regions is presented in (Ottlik and Nagel, 2008) with previous work in (Dahlkamp *et al.*, 2006, Dahlkamp *et al.*, 2004). The 3D wire frames of vehicles are used in a Hough transform to provide additional cues for vehicle detection. Only

4 vehicle models are provided, which leads to a low detection rate of 65% on video data from (Nagel, nd).

(Song and Nevatia, 2007) use a Bayesian framework with Markov chain Monte Carlo sampling. A proposal map is computed from the foreground map, indicating likely vehicle centroids based on a constant size vehicle model. The evaluation on two video sequences shows detection rates of 96.8% and 88%. The method is extended to predict (and hence remove) shadow projections. However, only shadows from known lighting conditions can be dealt with, *e.g.* sun light. This work is extended in (Johansson *et al.*, 2009) by incorporating multiple vehicle models but requiring manual setup of vehicle orientation. The performance of this algorithm is not evaluated quantitatively.

There is a completely different 3D approach presented in (Kim and Malik, 2003). An edge detector is applied in an entry window to the side view highway image to retrieve horizontal and vertical lines of vehicles. Those line features are grouped together, using a probabilistic method, to form vehicles based on a 3D line model. Once vehicles are detected in the entry window of the scene, they are tracked using cross correlation between frames. The detection rate compared to hand counting is reported to be 85%.

The problem of collision detection in urban intersections is tackled in (Atev *et al.*, 2005; Atev and Papanikolopoulos, 2008). Multiple cameras, calibrated according to (Masoud and Papanikolopoulos, 2007) with road primitives, are used to identify 3D ground plane locations of vehicles by projecting all foreground masks to the road plane. 85% of 273 vehicles are detected successfully on the authors' data.

2.4.2. Highways

Observing highway scenes usually gives the advantage of high camera angle and homogeneous traffic flow. A comprehensive review (Kastrinaki *et al.*, 2003) focuses on this topic. Newer references are discussed here and divided up into detection (section 2.4.2.1) and classification (section 2.4.2.2).

2.4.2.1. Detection

A region- based vehicle detection and classification system is proposed in (Gupte *et al.*, 2002). The main focus of the paper is on detection with effort put in a fast background estimation using the *instantaneous background* to get good segmentation. Tracking is performed using graph correspondence based on motion silhouette overlap. The proposed classifier uses two classes (cars, non cars) with size based features. The camera calibration is required to normalise those features. On the 20 minutes validation sequence, 70% of vehicles are correctly classified. Tracking using particle filters is introduced in (Wang *et al.*, 2009b). The system is motivated by generic surveillance, but results are shown for their own highway video sequence. A Markov chain Monte Carlo particle filter (MCMC PF) is used in (Bardet and Chateau, 2008) to track vehicles detected with simple frame differencing as background model. On a short 50 frame sequence, up to 89% of vehicles are tracked correctly.

The traffic system proposed in (Hsieh *et al.*, 2006) allows vehicle detection and classification into four classes. The camera is assumed to be in axis with the highway. This assumption allows the estimation of the lane centres by using the tracks (by Kalman filter) of vehicle centroids. The lane centres are used to calculate the lane division lines. Those lines are used to separate vehicle blobs merged due to shadows. The detected vehicles are classified based on size and the *linearity* feature. This *ad hoc* feature is a measure of the roughness of the blob. A Bayes classifier based on the Mahalanobis distance between feature vectors with constant prior is

used for classification. The performance is evaluated on 10 minutes of video data from three different sites. The reported detection accuracy is 82%. Out of the detected vehicles, 93% are classified correctly using cues from multiple frames. Higher detection accuracy is reported in (Wang *et al.*, 2004), however, the test video exhibits less occlusion. A rule based framework to deal with shadows and occlusions is introduced in (Su *et al.*, 2007). A recall of 95.6% is reported on 500 frames from a proprietary video of non occluded vehicles on a highway.

In contrast, vehicle detection from cameras on the roadside using height features is introduced in (Kanhere *et al.*, 2005) and followed up in (Kanhere, 2008, Kanhere and Birchfield, 2008). With camera calibration, the height of interest points is estimated throughout the video based on a foot point constraint of the bottom of a motion silhouette. This allows effective grouping of points into cars and trucks. The segmentation and tracking performance exceeds 90%.

2.4.2.2. Classification

(Rad and Jamzad, 2005) propose a system to track and classify vehicles on highways. Vehicles are first classified into three classes based on the width of the bounding box and the travelling speed. The classified bounding boxes are tracked using a Kalman filter. The reported tracking error rate is 5.4%.

(Huang and Liao, 2004) describe a motion segmentation and classification algorithm. Seven vehicle types are classified from side view motorway images. Blob features like length and compactness are used with a rule based classifier. The *instantaneous background* update model is used. Merged blobs of different vehicles are separated using dense optical flow fields. However, this method only works if there is a speed difference between occluding vehicles. The performance is evaluated with a test sequence lasting for 463 seconds which results in 91% overall classification rate.

2.5. Discussion

This section will discuss challenges in the field of traffic surveillance, especially in the urban domain. One major aspect is common data sets, which are analysed in section 2.5.2. Future research directions are given in section 2.6 in relation to this thesis.

Classical visual surveillance approaches of background modelling and tracking have been successfully applied for highway surveillance (Bardet and Chateau, 2008, Morris and Trivedi, 2006b, Kanhere and Birchfield, 2008). There are attempts to overcome the problem of occlusion and shadows for that type of scene. Urban environments are more challenging due to denser traffic, variable orientation of vehicles at intersections and lower camera position. More advanced approaches have been suggested including 3D models (Lou *et al.*, 2005, Messelodi *et al.*, 2005b), shadow prediction (Song and Nevatia, 2007, Dahlkamp *et al.*, 2006), appearance models (Kim and Malik, 2003, Ma and Grimson, 2005), *etc.* Algorithms developed for the generic object recognition domain have been applied and show promising results in the urban traffic domain (Leibe *et al.*, 2008b, Wijnhoven and de With, 2007, Wijnhoven and de With, 2009).

2.5.1. Challenges

From an application perspective, the main technical challenge is the diversity of camera views and operating conditions in traffic surveillance. In addition, a large variety of observation objectives like vehicle counting, classification, incident detection or traffic rule enforcement can be useful. This has generated a large and diverse body of work, where it is difficult to perform direct comparison between proposed algorithms. It would be beneficial for the community to define a set of clear tasks like it has been done in object recognition with (project PASCAL, nd). The main contribution of a challenge like this is a public data set. The next section

introduces a few available data sets. One possible reason for the lack of a common framework is the diversity of traffic rules, car classes, *etc.* around the world. Research seems always tailored to local environments, even if it only means adopting vehicle classes according to local traffic regulations. There is very limited literature dealing with night time (Robert, 2009a) and difficult light (Johansson *et al.*, 2009). To cover all possible situations, there might be the requirement for a bank of detectors, which are switched based on illumination (Acunzo *et al.*, 2007, Thi *et al.*, 2008).

The main technical challenge in urban environments is occlusions and dense traffic. There are many solutions for occlusion handling in highway scenes (Hsieh *et al.*, 2006, Su *et al.*, 2007, Kanhere and Birchfield, 2008) for relatively sparse traffic, which can not necessarily be transferred to urban environments. The introduction of 3D models shows promising results and allows occlusion prediction or at least modelling of a non overlapping 3D constellation of vehicles for a given scene.

2.5.2. Data Sets

Public data sets and evaluation would allow the field to objectively compare algorithms. In addition, labelled training data is essential for the training of machine learning algorithms discussed in section 2.3.2.2. Unfortunately, most authors use their proprietary data, which is rarely made available on the web. Even with videos available, ground truth is scarcer and very often application dependent. The i-LIDS data set (iLIDS, nd) is an attempt by the UK Home Office to benchmark visual surveillance systems based on requirements of end users. One scenario deals with illegally parked cars in urban roads and consists of 24 hours of video. There is only event based ground truth, which is of limited use for evaluation of low level algorithms. Tracking ground truth is available for parts of those videos through

(CLEAR, 2007) with a vehicle and pedestrian tracker evaluated in (Taj *et al.*, 2008). Greyscale images of urban intersection from a long distance high vantage point view are provided at (Nagel, nd) and are used in (Dahlkamp *et al.*, 2004, Dahlkamp *et al.*, 2006). Image patches used in (Ma and Grimson, 2005) are available[1] as Matlab data files. Similar image patches are used repeatedly in (Creusen *et al.*, 2009, Wijnhoven *et al.*, 2008, Wijnhoven and de With, 2009, Wijnhoven and de With, 2007) but no direct download is provided. Data for more general visual surveillance with some traffic related scenes is available from (VISOR, nd).

2.6. Future Research and Thesis Outline

A comprehensive review of computer vision technology for traffic analysis systems with a specific focus on urban environments was presented in this chapter. Research is expanding from the highway environment to the more challenging urban domain. This opens many more application possibilities with traffic management and enforcement. Traditional methods use background estimation and perform top-down classification, which can raise issues under urban conditions. Methods from the object recognition domain (bottom-up) have shown promising results, but not sufficient reliability yet. Clearer definitions of scenarios and applications are required to generate a more consistent body of work, which uses common data for comparable evaluation. Better fusion of top-down and bottom-up algorithms will be beneficial.

There is a larger body of work dealing with vehicle detection than with classification. For many applications, knowing the class of road users is essential. Some combined detectors and classifiers have been proposed (Leibe *et al.*, 2008b, Wijnhoven and de With, 2009, Lou *et al.*, 2005). Future classifiers should be able to take tracking prediction into account. According to several studies (Wang *et al.*,

[1] http://people.csail.mit.edu/xiaoxuma/proj/

2006, Morris and Trivedi, 2006a), the combination of both improves the results. The 3DHOG algorithm introduced in chapter 5 is a good example for an appearance based classifier, which can incorporate tracking predictions as initial hypotheses for new frames. The task of classification of vehicles should be pursued in order to increase the capabilities to the level of detection and tracking.

After the low level detection and tracking is tackled, there is significant potential for traffic rule enforcement. Current systems mainly focus on basic counting in highway and urban scenes. More sophisticated analysis of road user interaction is desirable in urban environment, especially including cyclists and pedestrians. Intelligent traffic light timing could benefit from a measurement of the state (position, velocity, class, *etc.*) of all road users at an intersection. The currently common installations of inductive loops in many cities cannot provide such comprehensive data.

A unified framework for detecting and classifying all road users is introduced in chapter 3 and used throughout the remainder of the thesis. The use of 3D models is introduced in general and combined with traditional background estimation. The classification task is the main objective for the algorithms presented. Local appearance models will be considered for a human detector in chapter 4 as a representative bottom-up method. The 3DHOG classifier in chapter 5 is integrated with the unified classification framework. This structure allows the guidance of the classifier with hypotheses generated from tracking information. A module hierarchy and parameters of the implemented framework are provided in appendix A.

3. Motion Silhouette Classifier

3.1. Introduction

This chapter presents work done by the author to detect and classify vehicles and pedestrians (called collectively 'road users') in urban traffic scenes. The problem tackled is road user classification on a per frame basis of a video stream. Every frame is treated independently for classification and no reasoning about the movement of road users is performed. Silhouettes (closed foreground regions) extracted by foreground analysis are the input to the classifier. The classification process is based on 3D models for road users. Related work has been introduced in chapter 2 with a specific focus on 3D model based methods in section 2.4.1.2 on page 44. Because of the overall context of CCTV monitoring, the detection of the system can be restricted to specific region(s) of the camera view (also referred to as "region of interest"). The framework introduced for using 3D models and evaluation is also used for the work described in a subsequent chapter where a more sophisticated detection mechanism is proposed and evaluated.

The following assumptions are made: Every silhouette corresponds to one road user being fully visible. This implies no occlusion in the scene and between road users. The orientation of the road users *on the ground plane* throughout the scene remains approximately constant, which implies that road users follow a straight road. The viewing direction of road users towards the camera can change, however, particularly if vehicles move from the back to the front of the camera view. The assumption of constant orientation clearly does not hold for pedestrians, who could be walking in any direction on the road. However, because of their posture (walking) and size (on typical road monitoring CCTV), the appearance of

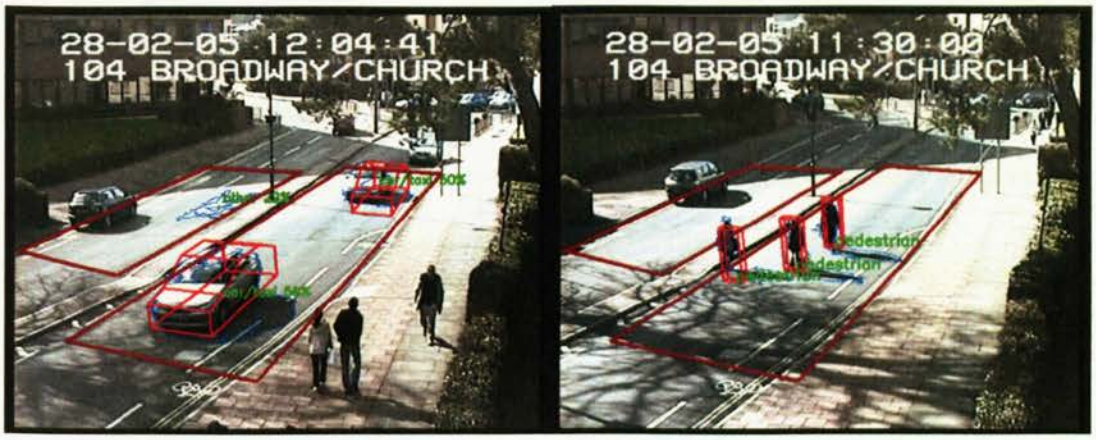


Figure 5 Example views from the i-LIDS data set with detected vehicles and pedestrians. The left image also shows an ambiguous foreground region (thin blue outline) on the top left, which was classified as class 'other' and that consequently, has no wire frame. The outlines of regions of interest R are shown as dark red rectangles on the road.

their silhouettes does not change significantly with direction and so it turns out that what seems as an unrealistic assumption does not have a major effect on the detection of pedestrians, as will be shown later with the results. Every region of interest R in Figure 5 can be associated with a different orientation. These regions R are a binary image mask defined manually for road areas where detection and classification will take place.

There are five classes used for the classifier as indicated in chapter 1 plus an additional class for objects not belonging to any defined class.

- Bus / Lorry
- Van
- Car / Taxi
- Motorbike / Bicycle
- Pedestrian
- Other (class for objects not belonging to any of the above classes).

The rest of the chapter is organised as follows. Section 3.1.1 gives an overview of the method. The detector is introduced in section 3.2. Section 3.3 covers the classifier and models used. The evaluation of the proposed system is given in section 3.4. Finally, a summary can be found in section 3.5.

3.1.1. Outline of the Proposed Approach

In the system introduced later in this chapter, background estimation generates foreground silhouettes. Silhouette centroids are used to generate road user hypotheses. A classifier verifies a hypothesis of a road user being present in the scene by matching silhouettes with road users' models. This is done by placing candidate 3D models on the scene's ground plane and projecting it to the camera view. A match measure is calculated for every hypothesis by comparing the model with the foreground silhouette. Every model is placed on a grid of positions on the ground plane to produce the match measure for every silhouette. This represents an algorithm based on image measurement features, which are better than image features according to Morris and Trivedi, 2006b (see discussion in section 2.3.2.1 on page 23). The highest match measure indicates the most likely position of the road user given the silhouette. The highest match measures of different classes are compared to make a decision about the class of a silhouette. Silhouettes with low match measures for all classes are classified as being of the class 'other' (see example in Figure 5). To use the 3D models, cameras are calibrated by means of a map and a minimum of five corresponding points with the image. A system block diagram is shown in Figure 6, where each block will be explained individually in the next sections.

The ground plane orientation of all road users is assumed to be more or less fixed, as pointed out earlier. One ground plane orientation is defined for every region of interest. A single object is assumed for every silhouette (*i.e.* no overlap).

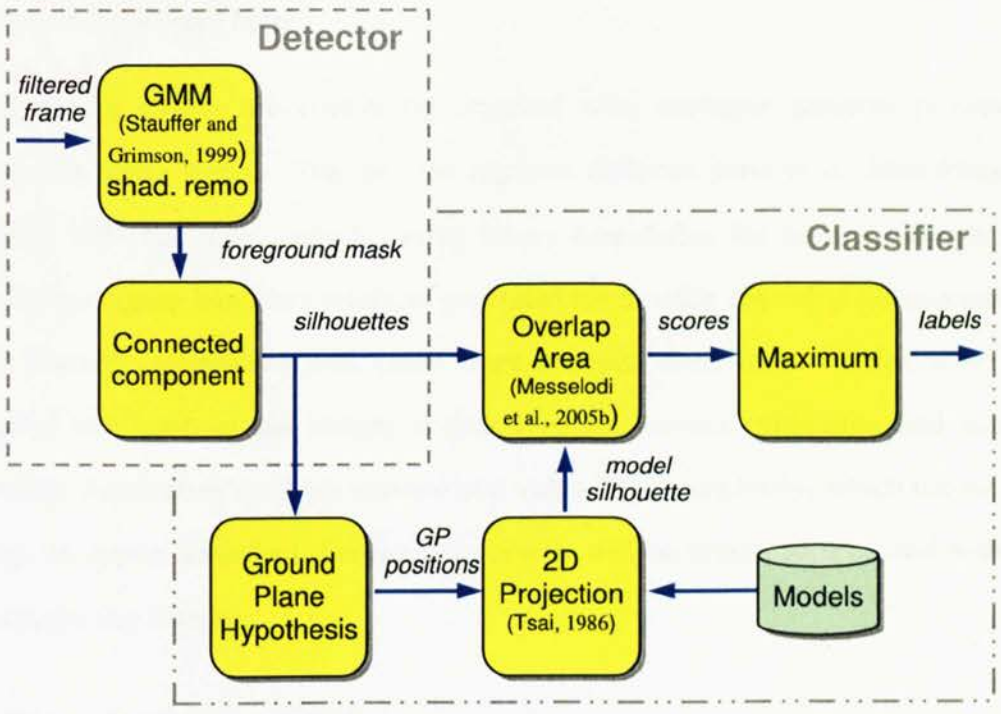


Figure 6 Block diagram of the detection and classification system

With those assumptions, the score is a size and overlap measure between the detected silhouette and a projected silhouette from the wire frame mask of the candidate model.

3.2. Detection

The detector uses background estimation to extract motion silhouettes from a video frame. Those silhouettes will later be used by the classifier. See the detector part of Figure 6 for a block diagram. Every block of the detector is described in more detail in this section, followed by the classifier in section 3.3, which takes the silhouettes as input. This structure will be expanded by a tracker in chapter 6, but will keep the same generic structure.

De-interlacing input filter

Surveillance videos are commonly captured with analogue cameras producing interlaced video signals. This process captures different parts of a video frame at slightly different times, which causes blurry boundaries for moving objects. To rectify the zigzag boundary artefacts generated for moving objects, a pre-processing step linearly interpolates odd video lines between even lines. In this way, the original size ratio of the images is preserved for camera calibration and human viewing. Alternative methods remove odd video lines completely, which causes the image to appear squashed. Performance results will be presented with and without de-interlacing filtering.

Background estimation (GMM)

A Gaussian Mixture Model implementation (KadewTraKuPong and Bowden, 2001) from the OpenCV library (OpenCV, nd) was used. The GMM, first introduced in the seminal paper of (Stauffer and Grimson, 1999), is used to generate an initial foreground mask. The software is set to estimate five Gaussians using a background threshold of 0.7 , which is the default value. The temporal window size, which is the inverse of the learning rate, is chosen at 50 to allow fast adaptation to illumination changes. With this value, stationary changes are in practice incorporated into the background within 15 seconds. The outdoor scene recorded with an auto iris function of the camera requires fast learning to accommodate illumination changes. Large objects in the scene can change the overall illumination conditions due to this gain control.

Shadow removal

The foreground pixels are post processed with the constant chromaticity shadow removal algorithm presented in (Cucchiara *et al.*, 2001). Every foreground pixel is

removed from the foreground mask, if its shadow condition is true: The pixel colour in the background image (most stable Gaussian from GMM) is compared to the pixel colour in the current image. For the comparison, both colour values are transformed into the HSV (Hue, Saturation, and Value) colour space. Value reductions down to 55% of the current pixel with respect to the background pixel are considered shadows, and the pixel is removed from the foreground mask. This algorithm assumes that for shadowed surfaces hue and saturation stay constant and only the value changes (all compared to the background image). This assumption holds for light shadows as seen in overcast condition if the camera is not saturated.

Connected components

Binary masks connected components are extracted from the final foreground mask. The purpose is to generate silhouettes, which are connected components and will be processed by the classifier later. Each silhouette is denoted by S. A filter operation is used to produce a set of final silhouettes $\mathbb{S}$ considering size and location with respect to the region of interest R as explained below. This set of final silhouettes $\mathbb{S}$ is used as input for classification. The length operator $\mathrm{L}(S)$ of a silhouette computes its perimeter in pixels. The area operator $\mathrm{A}(S)$ computes the number of foreground pixels in a silhouette. The overlap ratio operator $\theta(S,R)$ defined below gives the overlap of a silhouette with the region of interest R (*e.g.* red outlines in Figure 5):

$$\theta(S,R)=\frac{\mathrm{A}(S\cap R)}{\mathrm{A}(S)} \tag{1}$$

To be considered for classification, silhouettes S have to satisfy that their length is greater or equal than a threshold τ_L and that the overlap is greater or equal than a threshold τ_O. Values of $\tau_L = 200\,\text{pixel}$ and $\tau_O = 0.25$ are used for the experiments. Then:

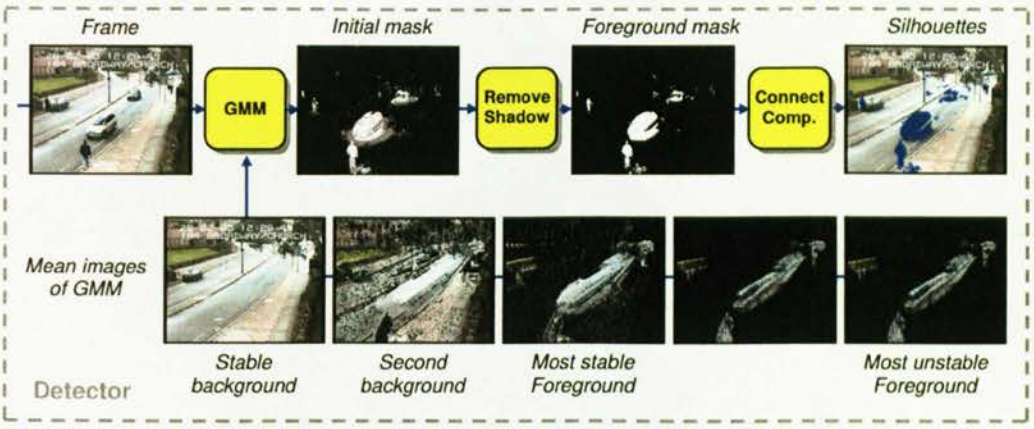


Figure 7 Example pictogram structure of the detector corresponding to the block diagram in Figure 6. The mean background images of the GMM modes are shown along the bottom, followed on top by the foreground mask and connected components S.

$$\mathbb{S} = \left\{ S \middle| \mathrm{L}(S) > \tau_L \wedge \theta(S,R) > \tau_O \right\} \tag{2}$$

The length threshold τ_L should be made equal the smallest road user in the scene. This means, that smaller silhouettes corresponding to noise are filtered out. If the value is chosen too large, smaller road users might be wrongly filtered out. The choice of overlap threshold τ_O only affects the silhouettes entering at the edge of the region of interest R. Practically, road users are fully contained in the region of interest R for most of the time, where the choice of overlap threshold τ_O has no effect. The data flow of the detector is illustrated in Figure 7 as a pictogram.

3.3. Classification

This step classifies each silhouette from the detection to be one of the set of road user types shown in Figure 8. This will be achieved by finding the match between the projected model and a silhouette. The classifier is divided into four steps shown in the classifier block diagram in Figure 6: ground plane hypothesis generation, 2D

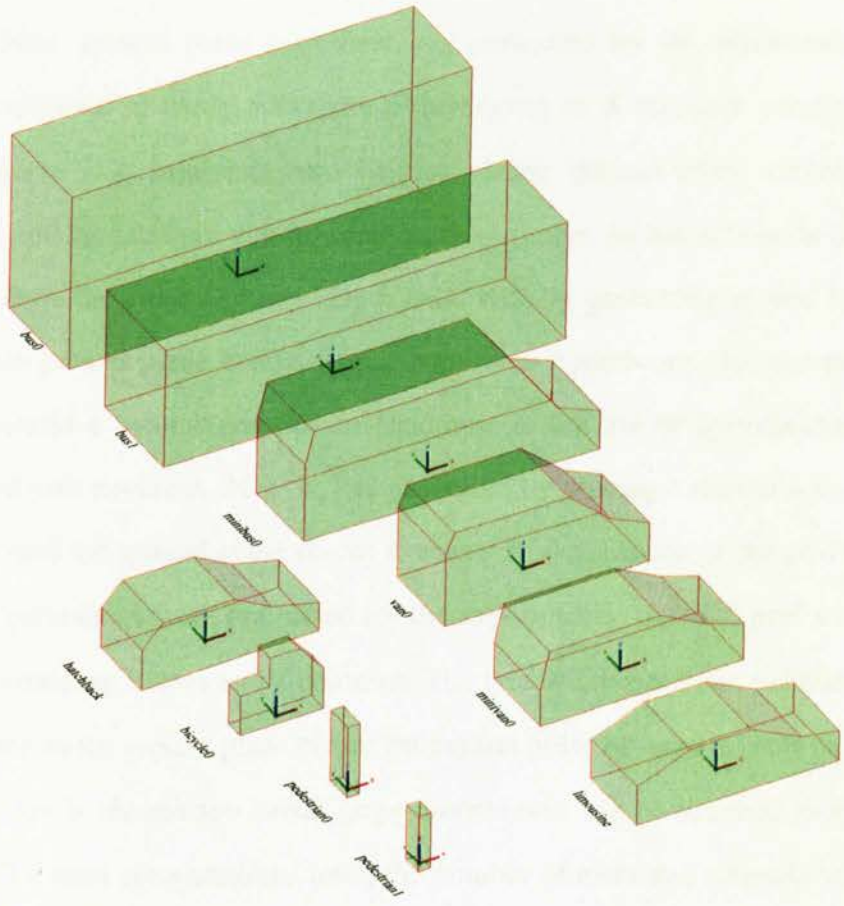


Figure 8 Wire frame models F_i used for classification. Refer to Table 1 for model and class correspondences.

model projection, overlap of model with silhouettes and maximum search. This section follows this structure.

Ground plane hypothesis generation

The camera requires calibration to be able to operate in the ground plane space and to use 3D models. The algorithm of (Tsai, 1986) is used to obtain the ground plane calibration for the camera using a map of the road and defining at least five corresponding points between the map image and the camera image. Based on the calibration, ground plane coordinates $\mathbf{g}=(x,y,z)$ can be converted to image coordinates. Back projection from the image to the ground plane implies that points are located on the ground plane in 3D world space.

First, ground plane hypothesis are generated for the silhouettes. The 2D image centroid $\mathbf{c}$ of every silhouette S belonging to $\mathbf{S}$ are back projected to the ground plane (*i.e.* implying zero height) giving ground plane centre $\mathbf{r}$. This projection of the centroid $\mathbf{c}$ introduces position noise, as the silhouette centroid $\mathbf{c}$ may not lie in the ground plane. This is dealt with by generating several hypotheses around the ground plane centre. Those additional hypotheses also compensate for noisy centroid $\mathbf{c}$ estimations due to shadows. A full set of hypothesised ground plane road user positions $\mathcal{H} = \{\mathbf{h}_p\}$ is generated by placing a regular square grid of points around the ground plane centre $\mathbf{r}$ where p is the index of the grid positions. The grid parameters were optimised for the experiments: the total grid width was 7 metres containing 7 rows and 7 columns. The grid width has to be sufficiently large to compensate for ground plane centre estimation noise of large models (*e.g.* bus). If this grid size is chosen too small, large models will not be matched at the correct location. To limit computational time, the number of rows and columns was chosen as low as possible. If the number of rows and columns is chosen low, the localisation of road users will be coarse.

2D model projection

The 2D projection generates model masks $M_{p,i}$ for every ground plane hypothesis $\mathbf{h}_p$. Figure 8 shows the full set of wire frame models $\mathcal{F} = \{F_i\}$ used for classification, where the model index i is in the range of 0 to 9 (see Table 1). The model dimensions are based on current vehicle manufacturers' information. The model mask is generated by

$$M_{p,i} = \mathrm{SIL}(F_i, \mathbf{h}_p), \tag{3}$$

where $\mathrm{SIL}(F_i, \mathbf{h}_p)$ is the projection of model F_i at ground plane location $\mathbf{h}_p$ according to the following two steps: Every model point of wire frame F_i is projected to the camera view (mask image) and the projected wire frame is drawn in

Model ID i	Wire frame Model	Model name	Class ID j	Class name
0	F_0	Pedestrian1	0	Pedestrian
1	F_1	Pedestrian0		
2	F_2	Bicycle0	1	Bike
3	F_3	Hatchback	2	Car / Taxi
4	F_4	Limousine		
5	F_5	Minivan0		
6	F_6	Van0	3	Van
7	F_7	Minibus0		
8	F_8	Bus1	4	Bus / Lorry
9	F_9	Bus0		
			-1	Other

Table 1 Class configuration table T showing correspondences between model ID i and class ID j (note that a class may correspond to more than one model)

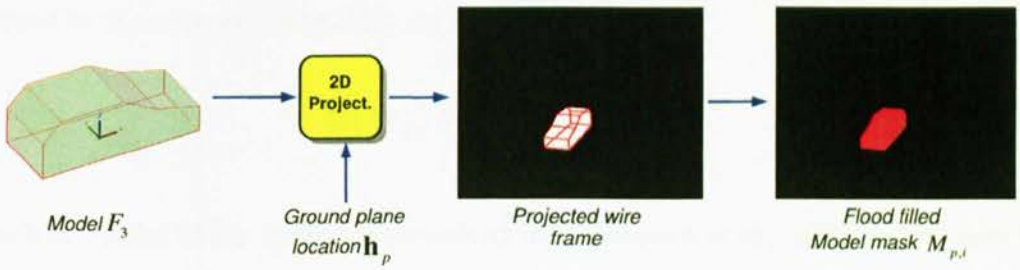


Figure 9 Illustration of the projection process of models. The wire frame of models is projected to the camera view and flood filled.

the mask image between the projected points. The binary mask $M_{p,i}$ is generated by flood filling the projected wire frame. This process is illustrated in Figure 9. The above algorithm changes model masks' size and shape implicitly according to the ground plane location.

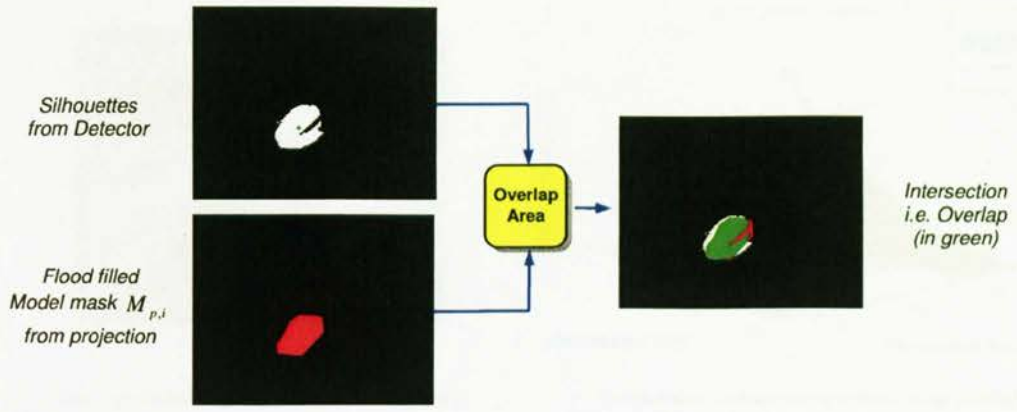


Figure 10 Illustration of model matching process. The normalised overlap between silhouettes and model mask is calculated.

Match measure as overlap of model with silhouette

A measure of quality of fit between the silhouette S and model masks $M_{p,i}$ is defined by the normalised overlap area $\Theta(M_{p,i}, S)$:

$$\Theta(M_{p,i}, S) = \frac{\mathrm{A}(M_{p,i} \cap S)}{\mathrm{A}(M_{p,i} \cup S)} \tag{4}$$

which is similar to the approach presented in (Messelodi *et al.*, 2005b). The area of the intersection of both masks is divided by the area of the union of both masks, which results in a match measure in the range $[0,1]$. Figure 10 gives an illustration of the overlap calculation.

Maximum search for best fitting model

Perform a global search of ground plane positions $\mathbf{g} = (x, y)$ and model indices i to find the best fit Θ'_S for every silhouette S

$$\Theta'_S = \max_{p,i} \Theta(M_{p,i}, S) \tag{5}$$

where $\mathbf{g}_S$ and i_S are the arguments that generated the maximum Θ'_S *i.e.*

$$\mathbf{g}_S, i_S = \arg\max_{p,i} \Theta(M_{p,i}, S). \tag{6}$$

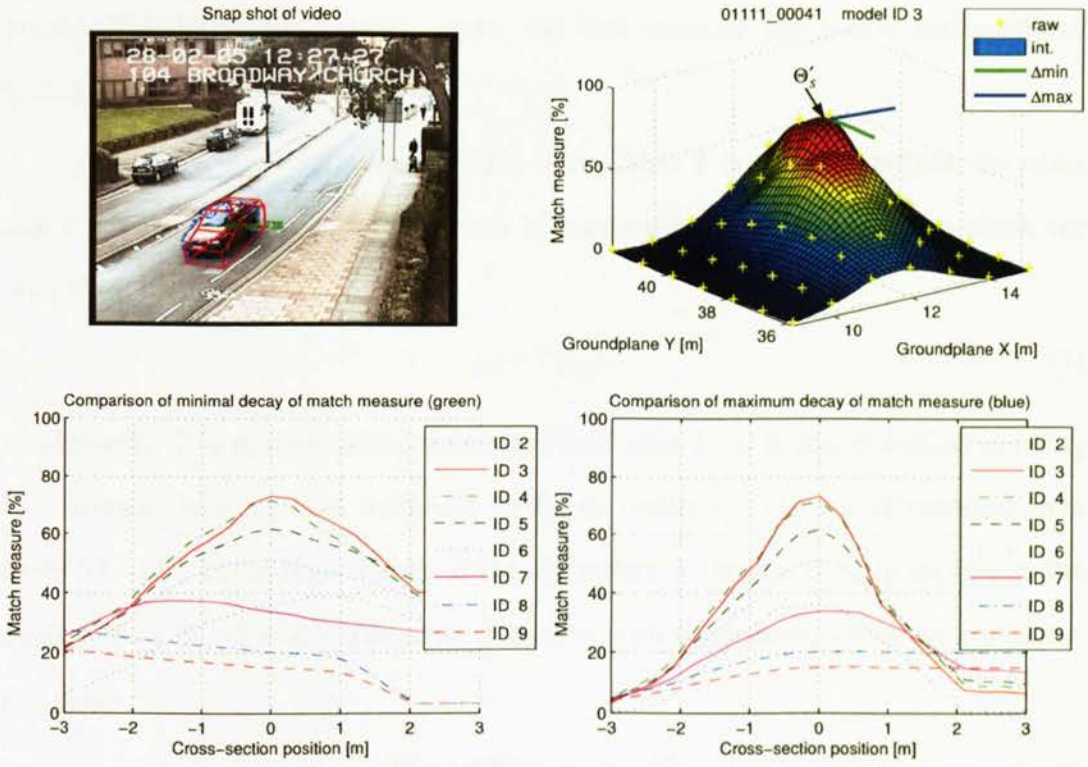


Figure 11 Match measure for one silhouette S. The upper left image shows the silhouette and best fitting model i_S at ground plane position (x, y). Top right: the winning match surface $\max_i \Theta(M_{p,i}, S)$ with data points. Bottom: Cross- section through every model's match surface $\Theta(M_{p,i}, S)$ along the minimum and maximum decay direction at (x, y). The legend label ID corresponds to the model index i in Table 1.

The overlap response is illustrated in Figure 11. Note the well shaped peak of the overlap function in respect to the ground plane positions $\mathbf{h}_p$. The peak is elliptic rather than circular, which can be observed by the different gradients in the bottom graphs in Figure 11. This can be expected due to the perspective angle of the camera. A shift along the x-axis (sideways on the road) produces a large horizontal shift in the image, which generates a sharp drop in overlap (bottom right graph). A shift along the y-axis (along the road) produces a less distinct vertical shift in the image, which causes a slower drop and therefore lower accuracy (bottom left

graph). The lower the camera angle, the less accurate the y-axis measurement becomes.

Finally, the configuration table T in Table 1 is used to retrieve the class index j_S for the model i_S as there can be many models for one class to allow for intra class variability

$$j_S = \mathrm{T}(i_S). \tag{7}$$

A silhouette S is promoted into a detected road user D, if it was classified as being of a known class, *i.e.* has sufficient fit Θ'_S to model i_S. The set of detected road users $\mathfrak{D} = \{D\}$ is the final output of the algorithm. A threshold τ_P is applied to the quality of fit Θ'_S of every silhouette S to deal with silhouettes, which do not match any class:

$$\mathfrak{D} = \{S | \Theta'_S > \tau_P, S \in \mathfrak{S}\}. \tag{8}$$

The threshold was optimised as $\tau_P = 0.48$ to provide an even balance between missed road users and wrongly detected road users. For completeness, the intermediate results and internal steps of the whole classification algorithm are illustrated as a pictogram in Figure 12 showing mask and silhouette images.

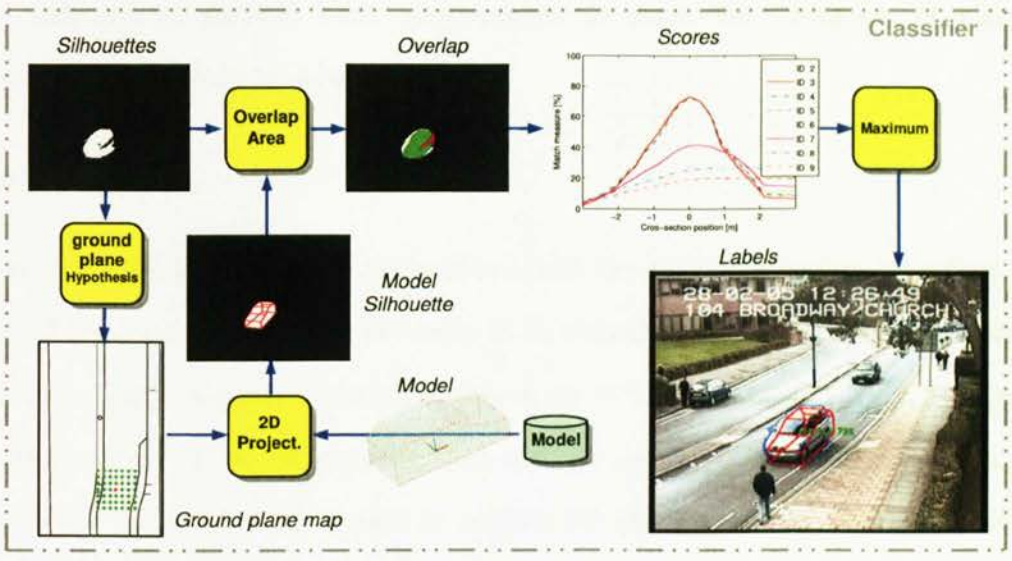


Figure 12 Illustration of data flow for the classification framework. This corresponds to the classifier block in Figure 6. One example silhouette S with centroid $\mathbf{c}$ in green is shown. The map on the bottom left illustrates the ground plane hypotheses $\mathbf{h}_p$ as green crosses. The model projected on the red position results in the red flood filled model mask M, shown as example for a single hypothesis. The normalised overlap operation $\Theta(M_{p,i}, S)$ is illustrated in the middle of the classifier for the example silhouette S and model mask M.

3.4. Evaluation

The proposed system has been evaluated on video from the i-LIDS data sets (iLIDS, nd). The set of ground truth $\mathcal{GT} = \{GT\}$ was partly provided by i-LIDS (ground truthed by NIST) in Viper format (Viper,) consisting of bounding boxes and class labels for road users. It had to be converted from NTSC to PAL indexing and was extended for pedestrians. The classifier produces bounding boxes and class labels for classified road users $\mathcal{D}$ also in Viper format. The next section introduces the metrics used, followed by the data set in section 3.4.2. Section 3.4.3 gives results for vehicle only detection and classification. Joint operation of all road user classes

is evaluated in section 3.4.4. Experiments to asses the influence of weather conditions are shown in section 3.4.5.

3.4.1. Metrics

Every classified road user D, is matched with the best overlapping bounding box GT of the ground truth $\mathcal{GT}$. The entry in an extended confusion matrix depends on the class labels. A general confusion matrix for N classes $(C_1, C_2, \ldots, C_N)$ is shown in equation (9). If no overlapping road user GT is found in the ground truth, the classified road user D is entered in column FP (false positive). All non matched road users in the ground truth $\mathcal{GT}$ within the region of interest are entered in row FN (false negatives). All the metrics used for evaluation can be derived from an extended confusion matrix. To allow evaluation of the detector in addition to the classifier, row FN (false negative) and column FP (false positive) are added to the confusion matrix.

		Groundtruth				
		C_1	C_2	$\cdots$	C_N	FP
Detected	C_1	$c_{1,1}$	$c_{1,2}$	$\cdots$	$c_{1,N}$	$c_{1,N+1}$
	C_2	$c_{2,1}$	$c_{2,2}$	$\cdots$	$c_{2,N}$	$c_{2,N+1}$
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	C_N	$c_{N,1}$	$c_{N,2}$	$\cdots$	$c_{N,N}$	$c_{N,N+1}$
	FN	$c_{N+1,1}$	$c_{N+1,2}$	$\cdots$	$c_{N+1,N}$	0

(9)

The metrics used for the evaluation of the whole system (detector and classifier) will be precision, recall and the F_1 measure. The definitions are taken from the i-LIDS trial (iLIDS, nd) specifications. Precision P and recall R are calculated independently for every class C_i and jointly for all classes. The following definitions are used for i-LIDS:

$$R = \frac{TP}{TP + FN} \tag{10}$$

$$P = \frac{TP}{TP + FP} \tag{11}$$

$$F_1 = \frac{(\alpha + 1)RP}{R + \alpha P} \tag{12}$$

The recall bias α can be set according to the application. The values for the above equations can be read from the confusion matrix (9). The true positive (TP) for any class C_i is the corresponding diagonal element $c_{i,i}$. The ground truth for recall is the column sum of all classes. The number of detections used for precision is the row sum of all classes C_i. Equations (10) to (12) can be expressed in terms of the confusion matrix from equation (9) with a matrix element defined as $c_{i,j}$. The total number of classes is N. The recall $R_{S,i}$ of the whole system (index S) per class C_i and the precision $P_{S,i}$ of the whole system per class C_i are defined as follows:

$$R_{S,i} = \frac{c_{i,i}}{\sum_{j=1}^{N+1} c_{j,i}} \tag{13}$$

$$P_{S,i} = \frac{c_{i,i}}{\sum_{j=1}^{N+1} c_{i,j}}. \tag{14}$$

Joint values for recall R_S and precision P_S for all classes can be calculated by summing up all diagonal elements and the corresponding rows or columns. Every class has an implicit weight according to the number of occurrences.

$$R_S = \frac{\sum_{i=1}^{N} c_{i,i}}{\sum_{i=1}^{N} \sum_{j=1}^{N+1} c_{j,i}} \tag{15}$$

$$P_S = \frac{\sum_{i=1}^{N} c_{i,i}}{\sum_{i=1}^{N} \sum_{j=1}^{N+1} c_{i,j}} \tag{16}$$

Precision $P_{C,i}$ for the classifier only (index C) per class C_i can be calculated by ignoring the column for FP. This equation deals with the classification result of correct detected objects only. The classifier recall $R_C = P_C$ when considering all classes jointly. The recall $R_{C,i}$ and precision $P_{C,i}$ for the classifier per class and the joint precision P_C are defined as:

$$R_{C,i} = \frac{c_{i,i}}{\sum_{j=1}^{N} c_{j,i}} \tag{17}$$

$$P_{C,i} = \frac{c_{i,i}}{\sum_{j=1}^{N} c_{i,j}} \tag{18}$$

$$P_C = \frac{\sum_{i=1}^{N} c_{i,i}}{\sum_{i=1}^{N} \sum_{j=1}^{N} c_{i,j}}. \tag{19}$$

Finally, precision P_D and recall R_D can be calculated for the detector only (index D). The classification performance is ignored by summing over all classes to calculate the true positives. The column sum is used for recall and the row sum is used for precision. Once again, the values can be calculated for each class C_i ($P_{D,i}$, $R_{D,i}$) or jointly for all classes (P_D, R_D).

$$R_{D,i} = \frac{\sum_{j=1}^{N} c_{j,i}}{\sum_{j=1}^{N+1} c_{j,i}} \tag{20}$$

$$R_D = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} c_{j,i}}{\sum_{i=1}^{N} \sum_{j=1}^{N+1} c_{j,i}} \tag{21}$$

$$P_{D,i} = \frac{\sum_{j=1}^{N} c_{i,j}}{\sum_{j=1}^{N+1} c_{i,j}} \tag{22}$$

$$P_D = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} c_{i,j}}{\sum_{i=1}^{N} \sum_{j=1}^{N+1} c_{i,j}} \tag{23}$$

This full set of metrics allows comparison with many published results. Often in the literature only a subset of those metrics is provided in a single paper. These metrics allow comparison of different aspects of the proposed algorithm with the corresponding publications.

3.4.2. Data set

The i-LIDS data sets (iLIDS, nd) are licensed by the UK Home Office for image research institutions and manufacturers. Each data set comprises 24 hours of video sequences under a range of realistic operational conditions. They are used by the UK government to benchmark video analysis products. They are useful for evaluating and comparing algorithms by the computer vision community and there is a gradual increase in take-up. Out of the Parked Car data set, what i-LIDS calls "scenario 1" was chosen, because it complies with the assumptions in section 3.1 and provides road users with large scale variations. Refer to Figure 5, Figure 13 and Figure 14 for example views. There is no public data set commonly used for urban traffic analysis. This makes direct comparison of reported results difficult. One contribution in this chapter is the use of this public data set to allow quick future comparison of systems in the same environment. Approximately one hour of video for sunny, overcast and changing conditions has been selected for the evaluation: (PVTRA10xxxx) 1a03, 1a07, 1a13, 1a19, 1a20, 1a21, 2a04, 2a05, 2a06, 2a08, 2a09, 2a10, 2a11 and 2a15. The recordings use a camera with an auto iris function that

keeps the average illumination of the view constant. Large vehicles with a predominant colour can cause adjustments in the iris and noticeable changes in the background. In addition, the overcast videos contain saturated areas in the middle and far end of the view. These are useful challenges to test the limit of the proposed approach(es).

Some ground truth usable for the tests (the data is normally used for event detection tests) was provided with the data set, however it had to be converted and extended. This limited the total length of video used for the evaluation. The total number of vehicle and pedestrian appearances is 782 as in Table 4. The proportions for each class are as follows: 47% car/taxi, 31% pedestrian and 8% each for van and bus/lorry and 6% for motorbike/bicycle.

3.4.3. Detection and Classification without Pedestrian Models

This section provides results for the proposed system without pedestrian models. The pedestrian model has been removed for this section just to make the results comparable to state of the art solutions in the literature, which usually do not consider pedestrians. The author's results compare to the state of the art, but for practical reasons are not evaluated on the same data for vehicle detection and classification. Using the shadow removal filter without the de-interlacing filter gives the best performance. Comparison of the filters is provided at the end of this section with a detailed analysis in section 3.4.4. Table 2 shows an extended confusion matrix including FP (false positives) and FN (false negatives) for the evaluation of detector and classifier and Table 3 shows results for the classifier and details per class. All values are normalised to the ground truth count per class displayed at a bottom row. The overlap indicates the overlap between ground truth bounding box and detection bounding box, which is obtained as the bounding box of the detected wire frame model. The whole system evaluates to a recall R of 87%

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.87	.03	.02	.00	.36
car/taxi	.00	.86	.10	.00	.05
van	.00	.02	.84	.02	.10
bus/lorry	.00	.02	.03	.98	.05
FN	.13	.07	.02	.00	.00
count	45	370	63	62	
overlap	.64	.66	.69	.72	

Symbol	*Value*
Recall R	87.0%
Precision P	85.5%
Classifier P_C	92.9%
Detector R_D	93.7%
Detector P_D	92.0%
GT Overlap	0.67

Table 2 Confusion matrix and overall silhouette classifier performance for vehicle only operation using shadow removal

ground truth / *detection*	bike	car/taxi	van	bus/lorry
bike	1.00	.03	.02	.00
car/taxi	.00	.92	.10	.00
van	.00	.02	.85	.02
bus/lorry	.00	.03	.03	.98
count	39	343	62	62

	bike	car/taxi	van	bus/lorry
R_j	86.7%	85.7%	84.1%	98.4%
P_j	59.1%	92.7%	79.1%	81.3%
R_{Cj}	100.0%	92.4%	85.5%	98.4%
P_{Cj}	78.0%	98.1%	86.9%	84.7%
R_{Dj}	86.7%	92.7%	98.4%	100.0%
P_{Dj}	75.8%	94.4%	91.0%	96.0%

Table 3 Confusion matrix for the silhouette classifier using shadow removal and per class evaluation

at a precision P of 85.5%. The classifier achieves a precision P_C of 92.9%. The detector has a recall R_D of 93.7% at a precision P_D of 92%. For qualitative results, refer to Figure 13 for true positive examples and Figure 14 for wrong classification. The higher number of false positives for the class bike is due to pedestrians being classified as bikes. At this stage, no pedestrian model was used and all the motion silhouettes resulting from pedestrians in the scene should have been classified as belonging to class 'other'.

Figure 13 Examples of correct detections and classification of vehicles using the silhouette classifier with shadow removal filter

Figure 14 Top: Two examples for false positives due to pedestrians being detected as bike and as car due to occlusion in a group. The bottom left image shows a car being misclassified as bike as it turns into the car park. The last image shows a missed car due to its similar colour compared to the saturated road area

3.4.3.1. State of the art literature

Direct comparison of quantitative results with the literature is difficult due to the lack of a common data set for vehicle classification. A detailed introduction to state of the art algorithms has been given in section 2.4 on page 42. The total recall R and precision P of the proposed system appears to outperform the following systems in terms of their reported results on their own data sets. The first reference is (Messelodi *et al.*, 2005b); with a system performance of 82.8% for detection and classification of urban road users into 8 classes is reported. All the following systems use highway imagery, which highlights the lack of work of vehicle classification using urban data. Total system R 65% at P 75% for classifying 150 car samples into 3 classes after detection and tracking is achieved in (Chen and Zhang, 2007). On 20 minutes test video, 70% of vehicles are classified (cars/non cars) after detection and tracking in (Gupte *et al.*, 2002). A classifier accuracy of 74.4% is reported for a 24 hour test sequence in (Morris and Trivedi, 2006b) using 3 classes. The same authors extended the system to 7 classes with a classification accuracy of 88.4% in (Morris and Trivedi, 2006a).

3.4.3.2. Input filter comparison

Results of the algorithm proposed in this thesis are compared for four different scenarios using input filters for shadow removal and de-interlacing. The effect of using different combinations of those filters is shown in Figure 15. Shadow removal is essential for good performance, whereas de-interlacing has a negative effect. This is partly due to smoother outline of silhouettes and the additional noise introduced in the background modelling from the interpolation process. The de-interlacing filter will be more important in chapter 5 where the appearance of road users is used for classification. The next section discusses all four cases in more detail when pedestrian detection is also considered.

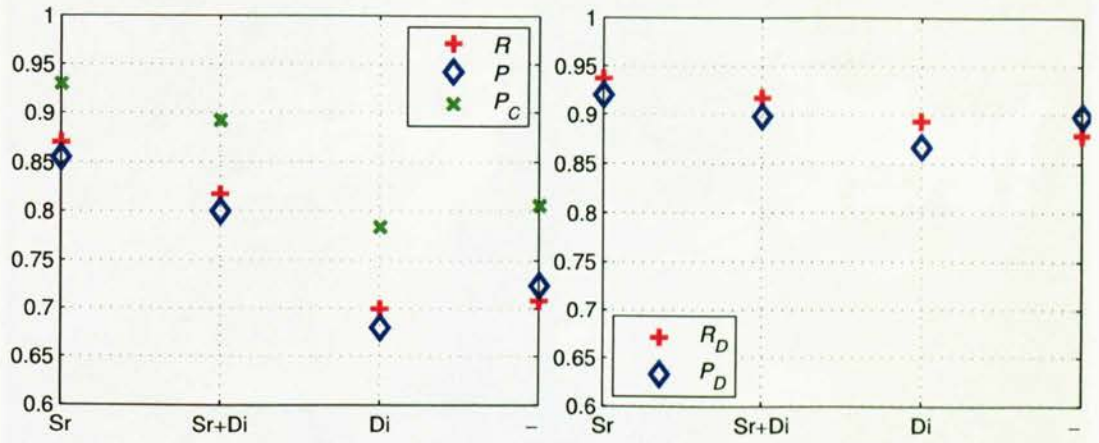


Figure 15 Performance comparison for the classification framework without pedestrian models using 4 different filter algorithms: shadow removal (Sr), shadow removal with de-interlacing (Sr+Di), de-interlacing (Di) and no filter (-). The left diagram shows system recall R, precision P and classifier precision P_C. The right diagram indicates the detector recall R_D and precision P_D

Figure 16 Two example views of the classification framework including pedestrian models for two different filter configurations from top: shadow removal (Sr) and bottom: shadow removal with de-interlacing (Sr+Di). The car at the bottom left is missed, because of a tighter silhouette of the car in addition to the saturation artefact.

3.4.4. Detection and Classification with all Road User Models

This section shows results for the proposed algorithm, when all road users are classified with the same framework. Results are given for the same four filter configurations (*i.e.* a) shadow removal, b) shadow removal with de-interlacing, c) de-interlacing and c) not filters) introduced in the last section with a qualitative comparison in Figure 16 and Figure 17. Best performance can be seen for shadow removal.

Figure 17 Two example views of the classification framework including pedestrian models for the remaining filter configurations top: de-interlacing (Di) and bottom: no filter (-). Too large silhouettes (their perimeters shown in blue) can be observed when shadow removal is not carried out, causing missed vehicles and wrong classifications (last two columns).

ground truth / *detection*	pedestrian	bike	car/taxi	van	bus/lorry	FP
pedestrian	.71	.49	.01	.02	.00	.10
bike	.02	.47	.03	.00	.00	.02
car/taxi	.02	.00	.85	.10	.00	.04
van	.00	.00	.02	.84	.02	.08
bus/lorry	.00	.00	.02	.03	.98	.03
FN	.24	.04	.07	.02	.00	.00
count	241	45	371	63	62	
overlap	.57	.60	.66	.69	.72	

Symbol	*Value*
Recall R	79.5%
Precision P	83.9%
Classifier P_C	89.8%
Detector R_D	88.6%
Detector P_D	93.5%
GT Overlap	0.64

Table 4 Confusion matrix of the motion silhouette classifier when using the shadow removal filter including overall performance figures for pedestrians

ground truth / *detection*	pedestrian	bike	car/taxi	van	bus/lorry
pedestrian	.94	.51	.01	.02	.00
bike	.02	.49	.03	.00	.00
car/taxi	.03	.00	.92	.10	.00
van	.01	.00	.02	.85	.02
bus/lorry	.01	.00	.03	.03	.98
count	182	43	344	62	62

	pedestrian	bike	car/taxi	van	bus/lorry
R_j	71.0%	46.7%	85.2%	84.1%	98.4%
P_j	77.4%	58.3%	92.4%	79.1%	81.3%
R_{Cj}	94.0%	48.8%	91.9%	85.5%	98.4%
P_{Cj}	87.2%	60.0%	96.6%	85.5%	83.6%
R_{Dj}	75.5%	95.6%	92.7%	98.4%	100.0%
P_{Dj}	88.7%	97.2%	95.6%	92.5%	97.3%

Table 5 Classifier and class wise performance figures for the motion silhouette classifier when using the shadow removal filter

3.4.4.1. Shadow removal filter

The framework used with the shadow removal filter gives the best performance for road user classification. Refer to Table 4 for an extended confusion matrix with overall performance figures and to Table 5 for class wise results. Very good classification performance is observed for the vehicles classes, whereas confusion occurs between bikes and pedestrians. This is due to very similar motion silhouettes of both road users, especially in the far region of the camera view when bicycles are

ground truth / *detection*	pedestrian	bike	car/taxi	van	bus/lorry	FP
pedestrian	.82	.55	.02	.05	.00	.12
bike	.02	.45	.04	.00	.01	.02
car/taxi	.01	.00	.81	.26	.00	.06
van	.01	.00	.02	.63	.04	.09
bus/lorry	.00	.00	.02	.02	.94	.06
FN	.14	.00	.09	.05	.00	.00
count	277	51	373	65	68	
overlap	.57	.58	.66	.73	.70	

Symbol	*Value*
Recall R	78.5%
Precision P	79.4%
Classifier P_C	86.4%
Detector R_D	90.9%
Detector P_D	91.9%
GT Overlap	0.63

Table 6 Confusion matrix and system performance of the motion silhouette classifier for shadow removal and de- interlacing filter

seen front on (see Figure 13). The higher false positive rate for the bike class observed earlier for the classifier without pedestrian models (Table 2) does not appear here, as a pedestrian model was used. The low detection performance of pedestrians is due to their non-rigid nature. The basic cube-like models do not match motion silhouettes of pedestrian as well as they do cars, which required the detection threshold to be halved for pedestrians. In addition, the interlacing of the cameras does affect smaller object more, which explains the performance increase of pedestrians when using a de-interlacing filter in the next section. However, using a single algorithm for all road users is beneficial in terms of system complexity.

3.4.4.2. Shadow removal and de-interlacing filter

The framework with both input filters indicates best performance for pedestrians. The confusion matrix in Table 6 shows system recall 82% for pedestrians, which is an improvement of 11% compared to shadow removal filtering only. The additional de-interlacing filter allows a better match of motion silhouettes compared to the last section. However, the classification performance for vehicles degraded, particularly the recall of vans from 84% to 63%.

ground truth / *detection*	pedestrian	bike	car/taxi	van	bus/lorry	FP
pedestrian	.63	.43	.03	.31	.00	.24
bike	.12	.35	.01	.01	.00	.08
car/taxi	.01	.02	.70	.09	.00	.08
van	.01	.00	.09	.45	.00	.06
bus/lorry	.00	.00	.07	.14	1.00	.10
FN	.22	.20	.10	.00	.00	.00
count	242	51	382	109	52	
overlap	.57	.42	.57	.59	.53	

ground truth / *detection*	pedestrian	bike	car/taxi	van	bus/lorry	FP
pedestrian	.63	.47	.00	.08	.00	.08
bike	.07	.31	.01	.00	.00	.06
car/taxi	.02	.02	.72	.11	.00	.06
van	.00	.00	.09	.63	.00	.13
bus/lorry	.00	.00	.05	.19	1.00	.11
FN	.27	.20	.13	.00	.00	.00
count	208	51	370	80	53	
overlap	.58	.44	.57	.57	.54	

Table 7 Confusion matrix for the motion silhouette classifier with de-interlacing filtering and with no filter. More tables for those cases are provided in appendix C.1.

3.4.4.3. De-interlacing filter and no filter

For these experiments, only the de-interlacing filter or no filters were used. In both cases, the performance is significantly worse than the experiments that include the shadow removal filter, which can be observed in Table 7. Compared to the best performance in section 3.4.4.1, recall drops by 11.7% to 67.8% and precision drops by 10.5% to 73.4%. This is due to oversized motion silhouettes, which can be seen in Figure 17. Therefore, this demonstrates that shadow removal is essential for this framework to perform well.

3.4.5. Influence of Weather Conditions

Robust operation under varying realistic weather conditions is important. This section compares the performance of the classifier without pedestrian model for sunny, overcast and changing conditions. Direct comparison is given in Figure 18 indicating that the approach performs best for sunny conditions. This may be due to the high contrast in the videos and therefore good foreground estimation. The following sections give more details about each condition. Some performance tables

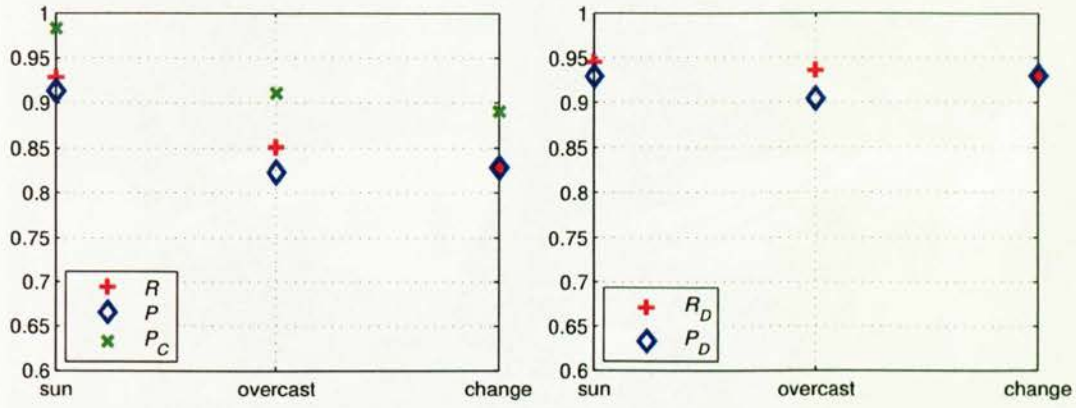


Figure 18 Performance comparison for the motion silhouette classifier under three different weather conditions

have been omitted here for space reasons and are provided in appendix C.1.

3.4.5.1. Sunny conditions

The best individual performance is achieved for sunny conditions, with the confusion matrix shown in Table 8. Many researchers have reported that sunny conditions degrade system performance due to shadows. This unexpected case of sunny conditions outperforming overcast conditions for classification may be explained by the dynamic range of the images. The high contrast and the deep shadow can be seen in examples of Figure 19. The sun allows a precise detection of the outline of road users; however it includes a deep shadow. The classifier can deal with that shadow as the silhouette is only extended in a single direction which reduces the overlap match measure for all models but keeps the ordering. In contrast, the lower dynamic range and the tendency of image saturation for overcast conditions introduce more noise to the road user's silhouette. This noise has a greater variability on the size of the silhouettes which can then lead to matching of a wrong model. However, due to the shadow, the mean overlap measure of the winning class in sunny conditions is 0.65, lower than the corresponding figure in overcast conditions (0.69). This means that the accuracy of the detected location for road users under sunny conditions is lower compared to overcast conditions.

Figure 19 Sunny examples top: true positive, bottom: false positive car and missed car

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.71	.00	.00	.00	.18
car/taxi	.00	.95	.00	.00	.04
van	.00	.01	.95	.00	.25
bus/lorry	.00	.00	.05	1.00	.00
FN	.29	.04	.00	.00	.00
count	17	135	20	9	
overlap	.44	.66	.68	.67	

Table 8 Confusion matrix for sunny conditions

Figure 20 Overcast examples top: two correct frames and bottom: one misclassified frame and one wrong detection due to saturation in the image.

3.4.5.2. Overcast condition

The performance for overcast conditions is second best after sunny. The confusion matrix in Table 9 shows many false positives for bikes. The false positives are observations of pedestrians, which should have been classified as 'other'. The miss classifications are mainly due to missed foreground areas due to saturation and low dynamic range of the scene. Refer to Figure 20 for examples.

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.96	.03	.00	.00	.39
car/taxi	.00	.78	.05	.00	.05
van	.00	.03	.91	.03	.05
bus/lorry	.00	.06	.05	.97	.06
FN	.04	.10	.00	.00	.00
count	28	119	22	33	
overlap	.72	.66	.76	.69	

Table 9 Confusion matrix for overcast conditions

3.4.5.3. Overcast changing to sunny

The worst performance can be observed for changing conditions. During those sequences, the sun appears several times which causes the auto iris of the camera to adjust. This produces ambiguous foreground silhouettes for short periods of time resulting in lower performance. Refer to Table 10 for the extended confusion matrix for this case with example views in Figure 21. The low performance of vans is due to their predominant white colour, which causes reduced foreground areas during times of saturation. This problem can be dealt with by exploiting the constraint that the same road users are present in the scene for many frames. Temporal filters and tracking are discussed in chapter 6.

Figure 21 Changing weather examples: Two correct frames at the top and two misclassified frames at the bottom.

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.00	.05	.05	.00	2.00
car/taxi	.00	.83	.24	.00	.07
van	.00	.02	.67	.00	.00
bus/lorry	.00	.02	.00	1.00	.05
FN	.00	.09	.05	.00	.00
count	0	116	21	20	
overlap	1.00	.66	.61	.80	

Table 10 Confusion matrix for changing conditions

3.5. Summary

This chapter presented a new algorithm for road user detection and classification using 3D models. The target application is urban traffic analysis which has different requirements compared to highway surveillance. 3D models based on car manufactures' dimensions are projected onto the image plane to search for a silhouette match measure (maximum). This match measure produces a distinctive peak at the right ground plane position and distinguishes different classes. This method has the potential of being useful for different applications (e.g. assembly, intelligent spaces) to generate camera specific object templates for visual matching and searching.

Evaluation was performed on the public i-LIDS data sets. Results have been provided for several input filters and weather conditions. Good overall performance of a recall of 79.5% at a precision of 83.9% is achieved for the whole detection and classification algorithm. The classifier achieves a high precision of 89.8% which is higher than reported results in the literature, but evaluated on different data sets. The lack of a common data set makes a direct comparison really difficult. The good performance can be contributed to the prior knowledge of 3D shape and therefore the knowledge of expected motion silhouettes. The simple camera calibration used here allows the application of the same models to be used across cameras. The full model incorporates vehicles and pedestrians into the same framework. This gives slightly lower performance (precision 89.8%) compared to the classifier without pedestrian models (precision 92.9%). Some confusion can be observed between bicycles and pedestrians, where 49% of bicycles are classified as pedestrians. This is due to their similar size and motion silhouette. The evaluation of input filters indicates that shadow removal filtering gives the best overall performance while de-interlacing improves pedestrian detection. Regarding weather conditions, the best classification performance of $P = 98.2\%$ is achieved for sunny

conditions outperforming overcast conditions and changing conditions. This result is due to the higher contrast and therefore less noise of the silhouettes in sunshine. As the author's classifier can deal with deep shadows, this condition gives the best results.

The described algorithm using 3D models could be compared to a method fitting ellipses to detected silhouettes. Those ellipses could be changed in size according to the camera perspective, which appears to be a simpler algorithm to give arguably similar results but no ground plane road location of road users. The setup of the above would; however, require more specialist knowledge and require a new setup for every camera. In contrast, the proposed algorithm uses a single set of models (3D volumes for road users) for all cameras. The dimensions are taken from real car dimensions in metres, which does not require any domain knowledge of computer vision. To set up a new camera, the camera calibration is simply obtained by clicking corresponding points on a road map and image.

The method is ultimately limited by the quality of the motion silhouettes. If the noise or imperfections of the silhouettes exceed the size variations between models, the classification will be erroneous. New vehicle shapes with similar size will not degrade performance. This is because the overall match between models and silhouettes might be lowered, but the rank order which determines the class would not be affected.

Personally, during this initial period of my research, I improved my understanding of the scientific method to investigate and evaluate methods and developed an appreciation of real world surveillance video data. The challenges faced when processing the size and variety made me shift my focus from expert like systems (rule based) as proposed in this chapter toward learning based methods like the next chapters. In this way, the variety and challenges of the data can be automatically tackled, provided that representative training data can be gathered.

The remainder of the thesis is dedicated towards more robust detection and classification of road users. 3D models show good performance, but the use of motion silhouettes alone is a distinct limitation for robustness and occlusions. This requires a radical change of concept to focus on incorporating local feature information into the classifier to have additional information (such as texture and appearance) apart from the motion foreground, which has shown limitations particularly during changing weather conditions. Local features indicated good performance in (Ma and Grimson, 2005) and are commonly used in object recognition style approaches *e.g.* (Leibe *et al.*, 2007, Leibe *et al.*, 2008b). The next chapter will focus on evaluating local features for surveillance tasks by considering the seemingly simple problem of detecting human intrusion in sterile zones. Those concepts will afterwards be integrated with the 3D framework presented in this chapter to generate what the author will call 3DHOG features in chapter 5 in an effort to overcome the limitations of the classifier presented here. In this way, the appearance of road users will be incorporated into the 3D models.

4. Local Features for Human Detection

4.1. Introduction

The previous chapter illustrated how motion (foreground) cues can effectively be used for road user classification. The summary then identified some shortcomings of estimating motion foreground. The use of local features (such as edges, textures, *etc.*) might be a way of overcoming these limitations. So, this chapter describes work done by the author to identify ways in which such local texture features (extracted in patches) can be used to detect intrusion in sterile zones under a range of environmental conditions. The work incorporates appearance into the models and moves away from the motion approach. This means that segmentation and object detection could take place using individual images (a task normally carried out without any apparent difficulty by human beings). The integration of both concepts (motion and local features) is then demonstrated in chapter 5. In this chapter the author takes a seemingly simple scenario (something one would assume would have been fully solved by now): the detection of people entering a sterile zone. This is a common task for surveillance *e.g.* a fence along a railway line, warehouse perimeters or similar. Such scenes contain a protected area typically with a physical barrier (*e.g.* fence) and a restricted (sterile) zone bordering the barrier. The author also uses the training and stringent testing framework given by the i-LIDS sterile zone test data set of the United Kingdom Home Office (iLIDS, nd). This data set is associated with a formal process of benchmarking commercial automatic surveillance systems and contains a wide range of environmental conditions (spanning all seasons and all weathers) as well as intrusion situations (walking,

crawling, running, rolling, *etc.*) in two camera views, referred to in here as View1 and View2 (Figure 22). The i-LIDS programme was inspired by a government need (informed by CCTV users) to rank systems so that those with an appropriate level of performance could be recommended to government departments, such as police forces. At the same time, the i-LIDS data set provides a common set of data that researchers can use to compare results. Although some of its definitions of what constitute true and false detections might seem arbitrary and even idiosyncratic, in this work we fully adopt the i-LIDS definitions so that researchers and even end-users could consider our results in that common context. The main challenge for the academic and industrial communities in such scenarios is to demonstrate robust operation over a wide range of environmental conditions. Those conditions include camera shake, illumination changes, auto iris (adaptive gain), 24 hour operation, rain, snow, wild animals, *etc.* Commonly used methods like motion estimation have problems dealing with those conditions. Those problems arise from the need of such methods to maintain and continuously update a background model, which often assumes a static camera view. Such limitations might be overcome by finding a method able to detect regions of interest (in this case intruders) on single images. In this context, it is observed that in many cases sterile zones contain greenery, gravel (railway) or other mostly homogeneous surfaces within which intrusion takes place. The author therefore formulates the intrusion detection problem as one of detecting "saliency", where saliency refers to a local significant difference in local texture features (in this case corresponding to the intruder). Later in the next chapter it will be explored, how the use of local features might improve the detection of road users in urban conditions.

Thus, this chapter presents a new texture saliency classifier for intrusion detection in still images. Salient objects are detected in real- time, based on spectral texture features of image regions. This means in practice, that people are detected

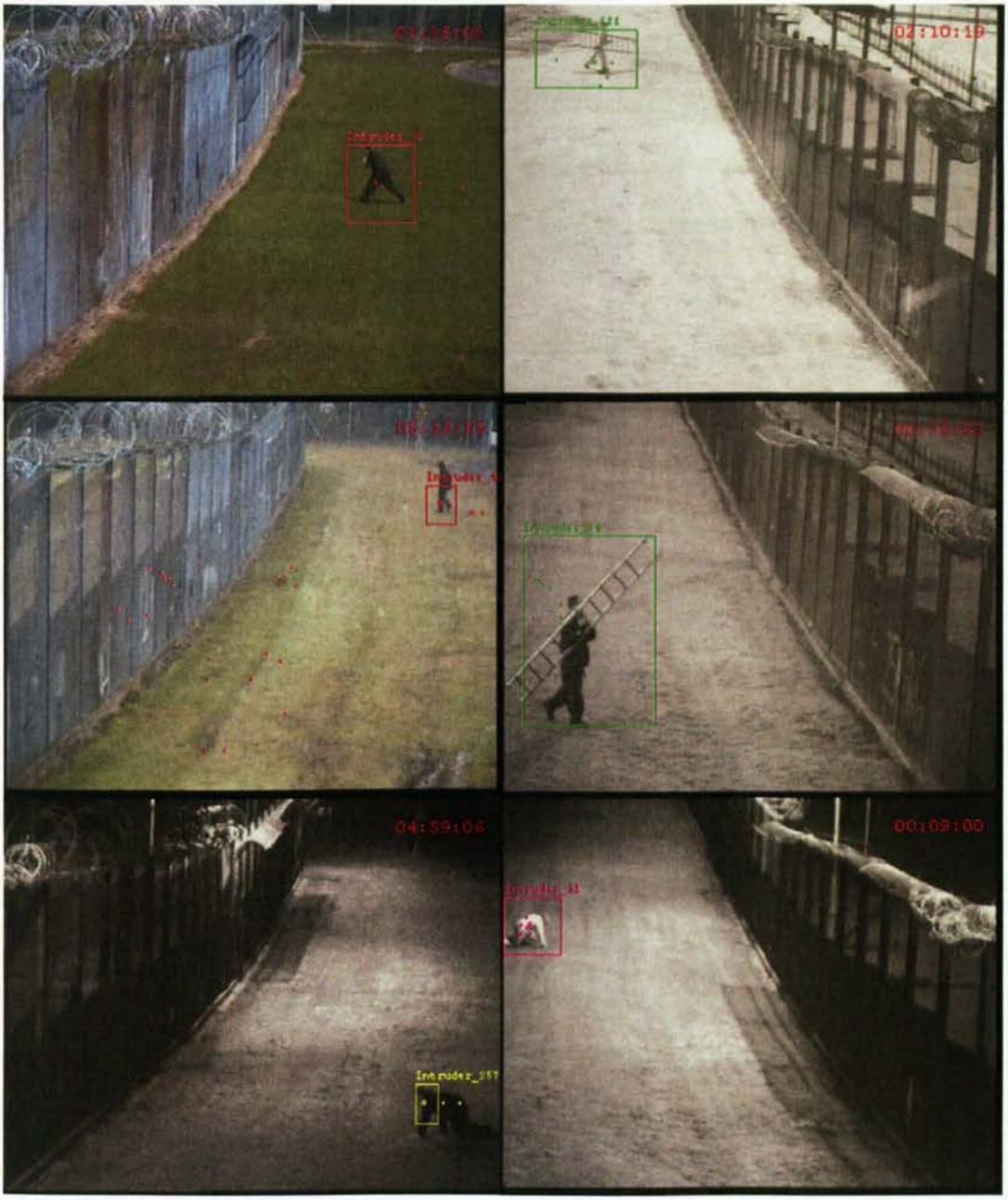


Figure 22 Examples of the i-LIDS data set showing the two camera views, different environmental conditions (falling snow in the middle left) and ways the fence is approached.

due to their texture difference compared to their surrounding texture. The basic detector is then extended with a combination of the texture saliency and an inter-frame difference motion mask so as to improve robustness. A further extension uses Kalman filtering and allows motion silhouettes to initialise tracks so as to reduce

detection time (this is particularly relevant to the i-LIDS benchmark that allows only up to 10 seconds to detect an intruder, no matter how slowly the intruder is moving). The algorithms are tested on the i-LIDS sterile zone data set and comparative results with the state of the art (at the time the work was done) OpenCV blob tracker are presented.

The remainder of this chapter is organised as follows: The next section discusses relevant work. The intrusion detector is introduced in section 4.3. Extensions to the detector are introduced in section 4.4. Section 4.5 describes the data set and provides details on the implementation including timing analysis. Full results are provided in section 4.6. Section 4.7 concludes the chapter.

4.2. Related Work

The most relevant literature on intrusion detection (*i.e.* object detection) can be divided into two categories: Methods exploiting temporal consistency by modelling background and methods operating on single frames. The first methods are usually fast to compute, but robustness to realistic conditions is limited. The proposed solution belongs to the second group, which gains robustness by solving the harder problem of foreground reasoning when considering single frames only. The recent body of pedestrian detectors like HOG (Dalal and Triggs, 2005), AdaBoost (Jones and Snow, 2008) or edgelets (Wu and Nevatia, 2005) are not applicable, because pedestrians are assumed to be upright. In the data set used here, people are also crawling, rolling sideways, *etc.* which breaks this assumption.

A common solution for utilising temporal consistency is to generate a pixel wise background model with which to estimate motion foreground and perform tracking. The background model can be a mixture of Gaussians as in the OpenCV1.0 blob tracker (OpenCV, nd). A background model based on the mode in the temporal histogram is given in (Zheng *et al.*, 2005). The disadvantage of using a

histogram is the slow adaptation for changed background when a high mode is established. The seminal papers of (Stauffer and Grimson, 1999, Stauffer and Grimson, 2000) introduce a Mixture of Gaussians background model per pixel to deal with multiple background illumination characteristics by trading off computational speed and memory size. This approach generally provides good results for outdoor scenes. (Sheikh and Shah, 2005) consider a probabilistic approach to model regions of pixels jointly. This allows the local spatial structure to be considered in a Markov Random Field (MRF). (Monnet *et al.*, 2003) use Principal Component Analysis (PCA) and an auto regressive model to predict a dynamic scene. This work is extended by (Culibrk *et al.*, 2009) who estimate stable texture regions. Periodically changing backgrounds are modelled in (Colombo *et al.*, 2007) to incorporate periodic distractions like escalators into the background model. A model based on texture blocks is proposed in (Heikkila and Pietikainen, 2006) and used for tracking in (Takala and Pietikainen, 2007). Pixel and block based approaches are combined in (Chen *et al.*, 2007) for a hierarchical method.

All background modelling approaches are affected by camera shake or fast scene changes which are typical for realistic conditions. Detection based on single frames may overcome those problems, however it increases the difficulty of detection as there is no temporal information available. Regression trees are used in (Davies and Lienhart, 2006) to classify pixels into road and non road for vehicle mounted cameras, assuming known road and non road seed areas. This does not require an offline training phase but has additional input from a laser range scanner. Based on training and structure from motion (Sturgess *et al.*, 2009) propose a segmentation system using graph cuts for road scene understanding. Texton, colour, location and HOG (Histogram of Oriented Gradients) descriptors are used in a boosting framework. A review of invariant pattern features is given in (Zhang and Tan, 2002); these are commonly used for classification of images and content based

retrieval. Recent work by (Shotton *et al.*, 2009) uses Textons to segment a single image and perform multi object recognition based on initial training.

4.2.1. Overall Approach

A new texture saliency classifier is proposed here for intrusion detection in still images. Intruders are detected because of their differing texture compared to the surrounding texture in the image. This is achieved through the analysis of the texture of local image patches in a video frame (the use of local patches is then taken forward, as discussed in the next chapter, to the more general problem of road user detection and classification). Thus, to analyse local texture the input image is divided up into patches, for which individual spectral texture features are generated. To identify image areas with similar texture, the patches are clustered in spatial and feature space. Comparing the texture features of those clusters gives an indication of homogeneity of the image (or conversely, of saliency), if some clusters' features significantly differ from the rest. Those differing clusters (*i.e.* the corresponding image patches) are likely to correspond to intruders and are labelled as "foreground" (or salient) and grouped into objects. In this way, image patches are evaluated for saliency within a single frame based on the overall homogeneity of the image. Object detections (intruders) per frame are remembered over time to build trajectories of object centres in the image space, which are then evaluated to determine if an intrusion condition has occurred. An intrusion takes place, if an object approaches the barrier. How this is defined and used will become clearer later.

The detection approach does not rely on temporal consistency of the frames. In this way, the algorithm is robust to camera shake, illumination changes and similar practical issues discussed earlier. In addition, complexity and runtime is still

low and no training is required, which is a typical limitation of single frame detectors, *e.g.* (Dalal and Triggs, 2005, Shotton *et al.*, 2009).

In a second variant of the algorithm, information is fused from the above basic detector and an interframe mask (that indicates possible motion by identifying significant changes in illumination in consecutive frames). For a region of the image to be labelled as foreground, the basic algorithm concentrates only on patches with significant interframe difference. The combination aims at adding robustness against distracting appearances (*e.g.* fence shadows) that mainly only affect texture and also against camera shake/illumination changes that mainly only affect the differencing mask. To decrease the alarm response time of the system, a Kalman filter is then introduced. The i-LIDS trial specification defines a hard ad-hoc time limit for alarms of 10 seconds after the first appearance of an intruder. This is probably driven by end-user demands and it means that a detected event is only a true positive if detected within that time, otherwise it becomes a false positive. By tracking partly visible people at the edge of the camera with a Kalman filter based on motion, this early evidence allows faster alarm triggers within the specified time. Having outlined the approach, in what follows, more detailed descriptions of the algorithm are given.

4.3. Intrusion Detector

This section describes the author's intrusion detector based on texture saliency. Section 4.3.1 discusses the five steps of foreground estimation and shows how the spectral features of image patches are used to detect salient texture foreground regions. Those regions are then combined into objects, for which trajectories are built as described in section 4.3.2. The last section also describes the rules used to trigger intrusion alarms based on the trajectories.

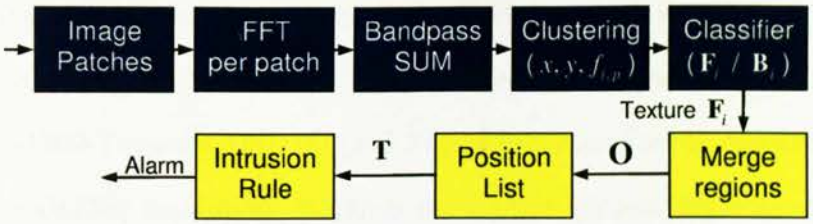


Figure 23 Block diagram of the intrusion detector

4.3.1. Foreground Estimation

Potential intruding objects are estimated from foreground that corresponds to inhomogeneities of local texture features in a single image. These potential objects are then passed to the intrusion rule described in section 4.3.2. The spatial distribution of features for local image patches in a single frame is analysed for saliency. No temporal background information needs to be maintained. To do this, the input image is first divided into patches, for which texture features are calculated. The foreground estimation using this local texture is performed in five steps shown as blue blocks in Figure 23:

- Local patch generation from region masks
- Fourier transform of individual patches
- Noise reduction and feature generation from frequency spectrum
- Clustering
- Classification of patches into foreground and background

Those steps are each described in detail in the next sections after discussing the acquisition of input images first.

A practical specification of i-LIDS is that it has to process analogue (PAL) video signals in real-time and activate a physical relay on detection of an intrusion. A demonstrator system was then built consisting of a video player (the original Quick Time MJPEG i-LIDS sequences were converted to MPEG-2 mpg files and

played using a hard disc-based commercial player with a standard composite (CVBS) video output). This output was then fed to a frame grabber based on a Philips TM-1300 Trimedia DSP (Digital Signal Processor) set to digitize the video at a CIF (360x288) resolution. Because the i-LIDS dataset includes challenging night-time footage, a decision was made to use only the luminance channel (a similar argument can be made for weather conditions such as snow and fog that have very little chrominance). Therefore the CIF monochrome (256 levels) output of the frame grabber is passed (via the PCI bus) to the main algorithm that runs as a normal PC application, thus this digitised video feed is the main input to the algorithm described here. The monochrome input image is histogram stretched to ensure that the full dynamic range of the image is used during all lighting conditions. The original shape of the histogram is preserved during this transformation. This early normalisation increases the signal strength, which is important to produce consistent foreground detection. In what follows we will refer to this normalised image as simply the input image.

4.3.1.1. Local patch generation from region masks

The i-LIDS data includes the characterisation of two distinct regions: the *approach* (in this case the grass) and the *boundary* (in this case the fence). Those two regions are defined by two binary pixel masks $\mathbf{R}_i$, which will drive the generation of two populations of local image patches from the input image. Those two populations will be analysed for saliency independently, which is why the region index i is introduced to distinguish them. The region index i has value 0 for the *boundary* (*i.e.* fence) and value 1 for the *approach* (*i.e.* grass). The masks $\mathbf{R}_i$ are taken directly from the sterile zone benchmark definition of i-LIDS (*i.e.* from the data set) and were not chosen by the author. Figure 24 shows an example of such a mask.

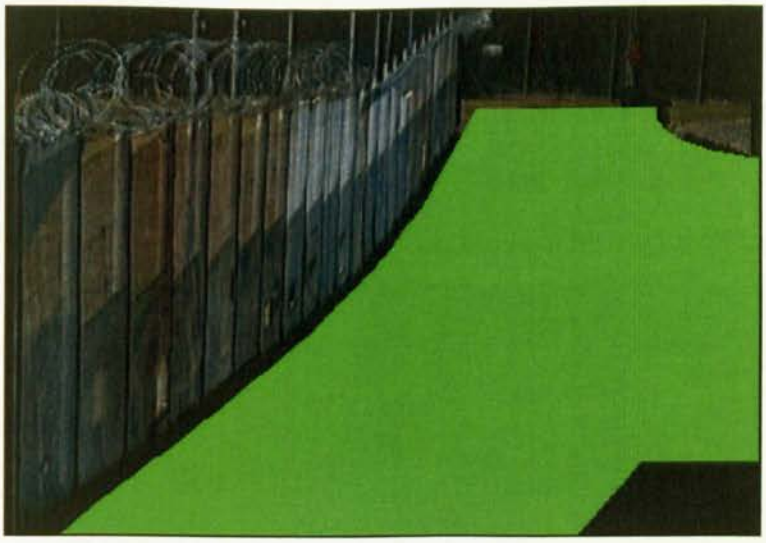


Figure 24 Example region mask $\mathbf{R}_i$ for the approach (green)

Image patches $\mathbf{P}_{i,p}$ are efficiently fitted to the region mask $\mathbf{R}_i$. The index p enumerates patches for each region $\mathbf{R}_i$ independently. The fitting process starts by placing patches at the top left inside the mask and continues to the bottom right by populating the mask with patches. The number of patches fitted depends on their size and overlap, which will be discussed in the next paragraph. If the boundary of the region is not vertical, the fitting will produce an unaligned grid of patches, as a new row of patches always starts at the edge of the region mask $\mathbf{R}_i$, which can be seen in Figure 25.

The patches $\mathbf{P}_{i,p}$ are 16x16 pixels and have 20 percent overlap between them. The patch size is chosen as a power of 2 to enable the use of the Fast Fourier Transform (FFT). The size should be chosen to be as small as possible to allow for a fine foreground resolution. On the other hand, the patches have to be sufficiently large to capture texture information. For the specific camera views, it was found through evaluation, that 16x16 pixel patches are sufficient to detect intrusion reliably. Increasing the overlap of patches is another way of increasing the foreground resolution and practically the sensitivity to small objects. The upper limit for increasing the overlap is ultimately limited by the required frame rate. The

computation time of the algorithm increases with the square of the overlap, because the number of patches increases with the square of the overlap. Figure 25 shows a frame with patches $\mathbf{P}_{i,p}$ for both masks $\mathbf{R}_i$ in different colours. The overlap can be best observed for patches in the bottom row highlighted by blue arrows.

Figure 25 Input frame with overlapping patches $\mathbf{P}_{i,p}$

4.3.1.2. Fourier transform of individual patches

To capture the texture and generate features of image patches $\mathbf{P}_{i,p}$, Fast Fourier Transform (FFT) is performed on each patch

$$\tilde{\mathbf{P}}_{i,p} = \mathrm{FFT}\left(\mathbf{P}_{i,p}\right). \tag{24}$$

The centre of the spectral patch $\tilde{\mathbf{P}}_{i,p}$ corresponds to the highest frequency, whereas the border corresponds to the lowest frequency. How noise can be reduced from the spectrum is explained in the next section.

4.3.1.3. Noise reduction and feature generation from frequency spectrum

The spectral patches $\tilde{\mathbf{P}}_{i,p}$ may contain noise, which may affect the foreground detection. This step first reduces noise and then calculates texture features for every image patch. Low frequencies contain the illumination conditions of the patch, which can differ significantly during night *e.g.* Figure 22 on page 91 on the far

right. The average brightness of a patch determines the DC (direct current) component and illumination gradients contain only low frequencies. On the contrary, high frequencies contain noise from the original (analogue) video feed. The random analogue noise (thermal noise) causes spatially small distortions also called snow (Ciciora *et al.*, 2004). This is introduced into the video player, cables and capture card. Both low and high frequency components have to be removed from the spectrum $\tilde{\mathbf{P}}_{i,p}$ resulting in a filtered spectral patch $\hat{\mathbf{P}}_{i,p}$. To reduce all the above noise, band pass filtering is applied as follows: Pixels along the border of spectral patches $\tilde{\mathbf{P}}_{i,p}$ are blanked (pixel value 0) to fully remove low frequencies and pixels in the centre of the patches are also blanked to fully remove high frequencies. The width of the outer border area to be blanked is 2 pixels; in addition the central square of 8 pixels width is removed. In this way, the noise is removed and texture information is preserved in the remaining filtered spectral patch $\hat{\mathbf{P}}_{i,p}$. Changing the width of blanked pixels by one pixel does not impact on performance noticeably. An illustration of the filtered spectral patches $\hat{\mathbf{P}}_{i,p}$ is given in Figure 26.

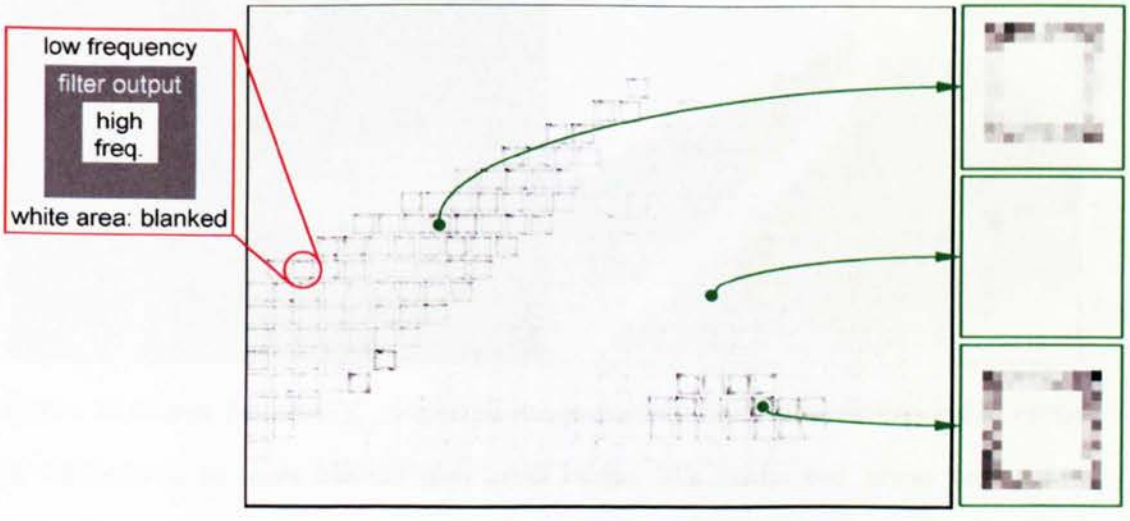


Figure 26 Filtered Fourier spectrum patches $\hat{\mathbf{P}}_{i,p}$ The spectral value range is normalised across regions to span the grey level range (the two regions are normalised independently for display purposes). The patches on the right illustrate the filtering by blanking the inner and outer area of the spectral patches $\hat{\mathbf{P}}_{i,p}$.

To generate a scalar feature $f_{i,p}$ for each patch $\mathbf{P}_{i,p}$, the sum of the all the elements of filtered spectral patch $\hat{\mathbf{P}}_{i,p}$ is calculated by

$$f_{i,p} = \sum \hat{\mathbf{P}}_{i,p} . \tag{25}$$

The feature $f_{i,p}$ can be used to discriminate people from background due to their different texture, while at the same time it gives a similar response over the whole background in typical sterile zone scenarios as defined by (iLIDS, nd). The example in Figure 27 shows the difference of feature value of the intruder compared to the grass background in the right area of the image.

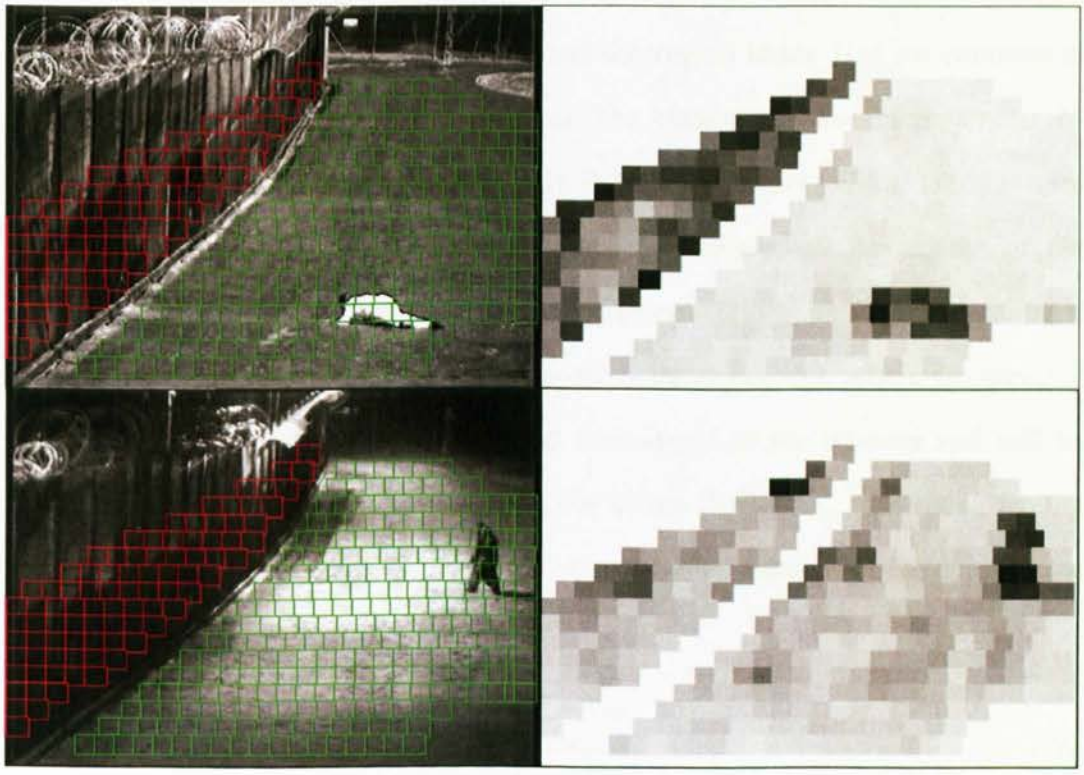


Figure 27 Scalar features $f_{i,p}$ (right) of image patches (left). The feature value range is normalised to span the full grey level range. The fence and grass region are normalised independently. The grass area shows an area distinctly different from the average, which corresponds to an intruder. The second example along the bottom illustrates, that the method is applicable for inhomogeneous illumination at night where intruders can be darker than the background.

4.3.1.4. Clustering features

After calculating the feature for each image patch, salient patches *i.e.* patches containing intruders have to be identified for potential alarms. To find significantly large salient regions in the image, neighbouring patches $\mathbf{P}_{i,p}$ are clustered with respect to their location and the feature scalar value $f_{i,p}$. The mean feature value of clusters $\bar{f}_{i,k}$ is used to detect an intruder. By considering clusters rather than single patches, larger support for saliency is accumulated. This reduces the detection of outliers. In addition, whole objects or large object fragments are represented by clusters. For the clustering itself, a hierarchical cluster tree is generated to find N clusters $\mathbf{C}_{i,k}$ with cluster index $k \in [1, N]$ and the region index i (as we continue to maintain two separate populations of data). The choice of value of parameter N will be discussed in the next section. Ward's linkage algorithm (Ward, 1963) is used to combine clusters in the tree which effectively minimises the square of the Euclidian distance between elements in the clusters. The clusters of the example frame are illustrated in Figure 28 as dots with different colours, where the elevated clusters (red and green) in the right graph correspond to the intruder and will be classified as foreground in the next step. For every cluster $\mathbf{C}_{i,k}$, the mean feature $\bar{f}_{i,k}$ is calculated by dividing the sum of features by the number of elements in the cluster

$$\bar{f}_{i,k} = \frac{\sum_{\mathbf{P}_{i,p} \in \mathbf{C}_{i,k}} f_{i,p}}{\left| \mathbf{P}_{i,p} \in \mathbf{C}_{i,k} \right|}. \tag{26}$$

The clusters for both regions (*i.e.* fence and grass) are illustrated in Figure 28.

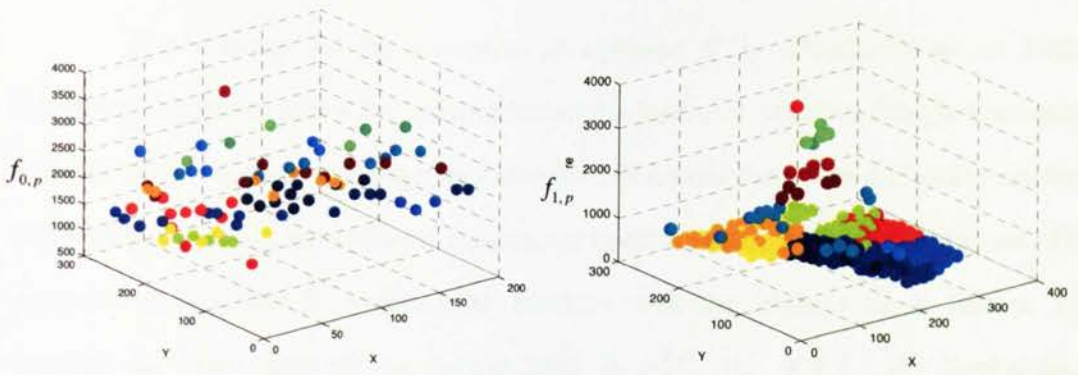


Figure 28 Clusters $\mathbf{C}_{i,k}$. The left image shows the clusters for the fence and the right image the clusters for the grass. Every image patch is represented by one dot, where the colour indicates the cluster label.

4.3.1.5. Classification of patches into foreground and background

The resulting clusters $\mathbf{C}_{i,k}$ with mean features $\bar{f}_{i,k}$ are now classified into foreground $\mathbf{F}_i$ and background $\mathbf{B}_i$. It is important to emphasise again, that this classification is based on the local texture feature statistic of a single frame. It is assumed that most of the image contains background and only a maximum of M patches are foreground $\mathbf{F}_i$. This is a valid assumption for typical sterile zone scenarios where a camera covers a large area with a limited number of people entering the scene. The i-LIDS trial allows for algorithm training, which could be used to find parameters. The values for patches in the foreground M and the number of clusters N empirically represent the scale and perspective of the camera view and can be obtained from analysing the scene by the following procedure: The smallest foreground object should occupy approximately one cluster. This results in $N = 15$ by estimating the ratio between the number of image patches $\mathbf{P}_{i,p}$ for the smallest object and the total number of patches $\mathbf{P}_{i,p}$ in region $\mathbf{R}_i$. The number of foreground clusters $M = 4$ is calculated as the ratio between the smallest and largest object, please refer to Figure 22 on page 91 for examples of size variations.

The concept of the *potential foreground* $\tilde{\mathbf{F}}_i$ is introduced as an initial foreground guess to allow the calculation of background statistics (single Gaussian) without contamination of foreground clusters. *Potential foreground* clusters are then evaluated again this background statistic to confirm them as final foreground. The *potential foreground* $\tilde{\mathbf{F}}_i$ contains M clusters with the highest mean feature $\bar{f}_{i,k}$ leaving all other clusters as background $\mathbf{B}_i = \left\{\mathbf{C}_{i,k} \middle| \mathbf{C}_{i,k} \notin \tilde{\mathbf{F}}_i\right\}$. As background statistic, the mean feature value $\bar{f}_i$ of the background clusters and their variance σ_i^2 is calculated, separate for both populations of data, i by:

$$\bar{f}_i = \text{mean}\left\{\bar{f}_{i,k} \middle| \mathbf{C}_{i,k} \in \mathbf{B}_i\right\} \tag{27}$$

$$\sigma_i^2 = \text{var}\left\{\bar{f}_{i,k} \middle| \mathbf{C}_{i,k} \in \mathbf{B}_i\right\}. \tag{28}$$

The final foreground $\mathbf{F}_i$ consists of salient clusters of $\tilde{\mathbf{F}}_i$ fulfilling the following saliency condition

$$\mathbf{F}_i = \left\{\mathbf{C}_{i,k} \middle| \mathbf{C}_{i,k} \in \tilde{\mathbf{F}}_i \wedge \bar{f}_{i,k} \geq T \cdot \sigma_i^2 + \bar{f}_i\right\} \tag{29}$$

with saliency threshold $T = 5$. This implies that the foreground patches have to have higher feature values than the background. The threshold was optimised for sample videos from the dataset's testing set. The choice of T is not very sensitive, as T is multiplied by the variance σ_i^2, which is recalculated and adapted to every frame. Figure 29 shows the final foreground $\mathbf{F}_i$ highlighted in the example frame.

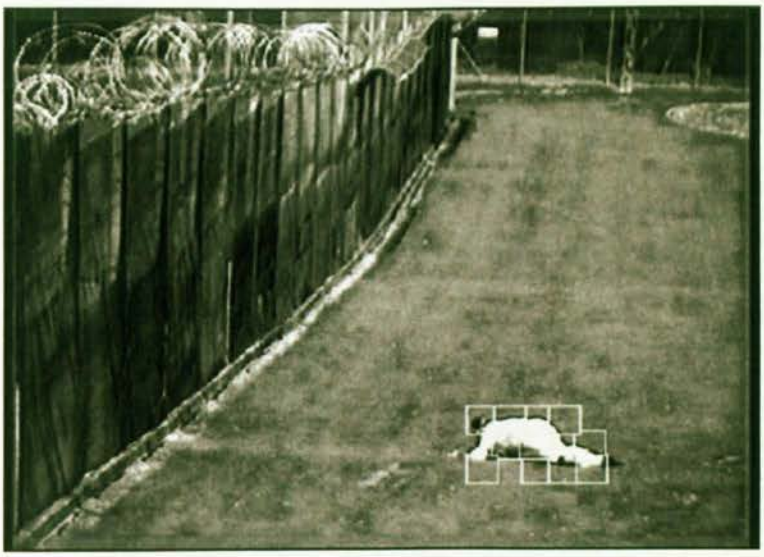


Figure 29 Final foreground patches $\mathbf{F}_i$ detected in the example frame

The approach for foreground evaluation is similar to the mixture of Gaussians as used in (Stauffer and Grimson, 1999) to model foreground and background. Their background threshold corresponds to the saliency threshold used here. The fact that the background is not modelled temporally here requires the threshold to be applied to the feature value rather than the distribution proportion. The use of a threshold like in (Stauffer and Grimson, 1999) would result in the foreground being always the same fraction of the whole image. The graphs in Figure 28 show similar absolute values for both fence and grass clusters, but the clusters of the person in the grass are significantly elevated above the background clusters. This shows that the feature value with respect to background statistics may be an indication for foreground.

4.3.2. Intrusion Rule

The intrusion rule part of the algorithm first generates objects from the foreground, computed as described in the previous section, and then evaluates their temporal trajectory to check for an intrusion condition. This process is illustrated as yellow blocks in Figure 23 on page 96. Firstly, spatially close foreground clusters are

merged into single objects **O**. This means, that foreground clusters with patches overlapping each other are merged. Large objects close to the camera are usually segmented with several clusters due to the camera perspective as discussed in section 4.3.1.4. This merging process addresses this problem.

The positions of objects **O** are logged over time to generate trajectories **T**. Those trajectories are analysed to detect genuine intrusions. Objects are associated with the closest trajectory based on the Euclidean distance in image coordinates. This is sufficient due to the low false detection rate of the intrusion detector and typical low number of trajectories. If there is more than one object in a frame, multiple trajectories are generated or updated, if some trajectories already exist. This simple accumulation of positions will be extended by a Kalman filter in section 4.4.2. All trajectories **T** are considered for an alarm condition. The alarm rule requires a trajectory to have accumulated support from the texture saliency detection for 2 seconds and the horizontal motion component has to be consistently towards the barrier (*i.e.* left or right, depending on the side of the fence). For different camera setups, a different motion direction could be used (*e.g.* vertical for a barrier along the top). A longer time window would increase the performance due to the increased evidence of an intruder; however, the stringent time window defined by the i-LIDS specification requires raising alarms fast. The fence location (left or right) is obtained from the i-LIDS scenario definition together with the sterile zone masks. An example frame with intruder and trajectory is shown in Figure 30.

Figure 30 Intruder with trajectory T for the example frame

4.4. Detector Extensions

Two extensions are made to the base intrusion detector. The first incorporates simple inter- frame difference motion estimation to reduce false detections by information fusion. The second introduces a Kalman filter to improve trajectory quality and shorten the alarm triggering time. The information fusion resulted in a significant performance increase, which may be due to the different noise dependencies. The extension with Kalman improved the time, but degraded performance in general as potentially noisy motion information was considered to start trajectories.

4.4.1. Motion Extension

The algorithm described in section 4.3.1 does not use any temporal information for detecting objects. The main reason for false detections is the existence of semi permanent objects in an image. Examples of those would be fence shadows, small clouds, *etc.* The algorithm can be improved by incorporating motion information and fusing the information with the result of texture analysis described from section

4.3.1. The motion extension incorporates temporal information by inter-frame differencing for pixel wise motion foreground estimation; please see Figure 31 for a block diagram (orange colour).

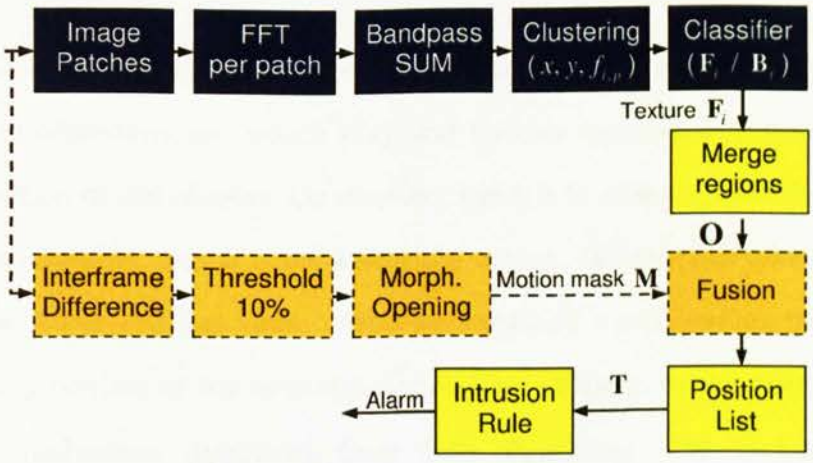


Figure 31 Block diagram for the intrusion detection with motion extension in orange.

The output of the motion estimation step is a final motion mask **M** and is generated as follows. A dynamic threshold is applied to the absolute pixel-to-pixel frame difference of two consecutive frames, so that a fixed proportion of 10% of pixels are selected as interframe motion. This enforces that only part of a frame (*e.g.* a person if present) can be foreground at any given time, considering the scenario assumptions discussed in section 4.3.1. Consequently, significant global changes in image conditions (*e.g.* illumination change due to sun) can be dealt with by focusing on the most significant moving objects. If there are no moving objects, the interframe motion corresponds to uniform noise pixels which will then not be considered further due to their small size. Finally, morphological opening with a 3x3 kernel is applied to eliminate such small noise and join up larger regions to result in the final motion mask **M**. This motion mask tends to contain only edges of moving objects loosing the middle section due to the crude interframe difference. The fusion will be able to use this mask **M**, because the fusion does not require complete coverage of the object, which is discussed in the next step. For

completeness, a pictogram of the intrusion detector with motion extension is shown in Figure 32 with an example of the motion mask at the bottom left.

Information fusion

On the one hand, motion information is affected by camera shake, fast changing illumination conditions, *etc.* which is typical for this application as pointed out in the introduction of this chapter. On the other hand, it is robust against the existence of stationary objects, which could affect the texture saliency detection only. The information fusion requires valid objects to contain at least 5 motion pixels inside the bounding box(es) of the object(s) detected by saliency. In this way, an object requires simultaneous detections from both algorithms. The texture saliency bounding box tends to always fully enclose the moving intruder due to the coarse structure of image patches and therefore encloses corresponding motion pixels comfortably. The number of motion pixels required was chosen as low as possible to avoid rejection of slowly moving intruders, but larger than the typical number of noise pixels in texture bounding boxes for the testing dataset. The fusion approach reduces false detections, as will be shown in the results section, as noise for appearance and motion is independent and therefore less likely to occur jointly. This allows lower detection thresholds for both detectors, which significantly reduces false negatives (missed intrusions) by simultaneously increasing the false positives (ambiguous alarms) of both detectors. The fusion of both algorithms eliminates those additional false positives and avoids an overall increase.

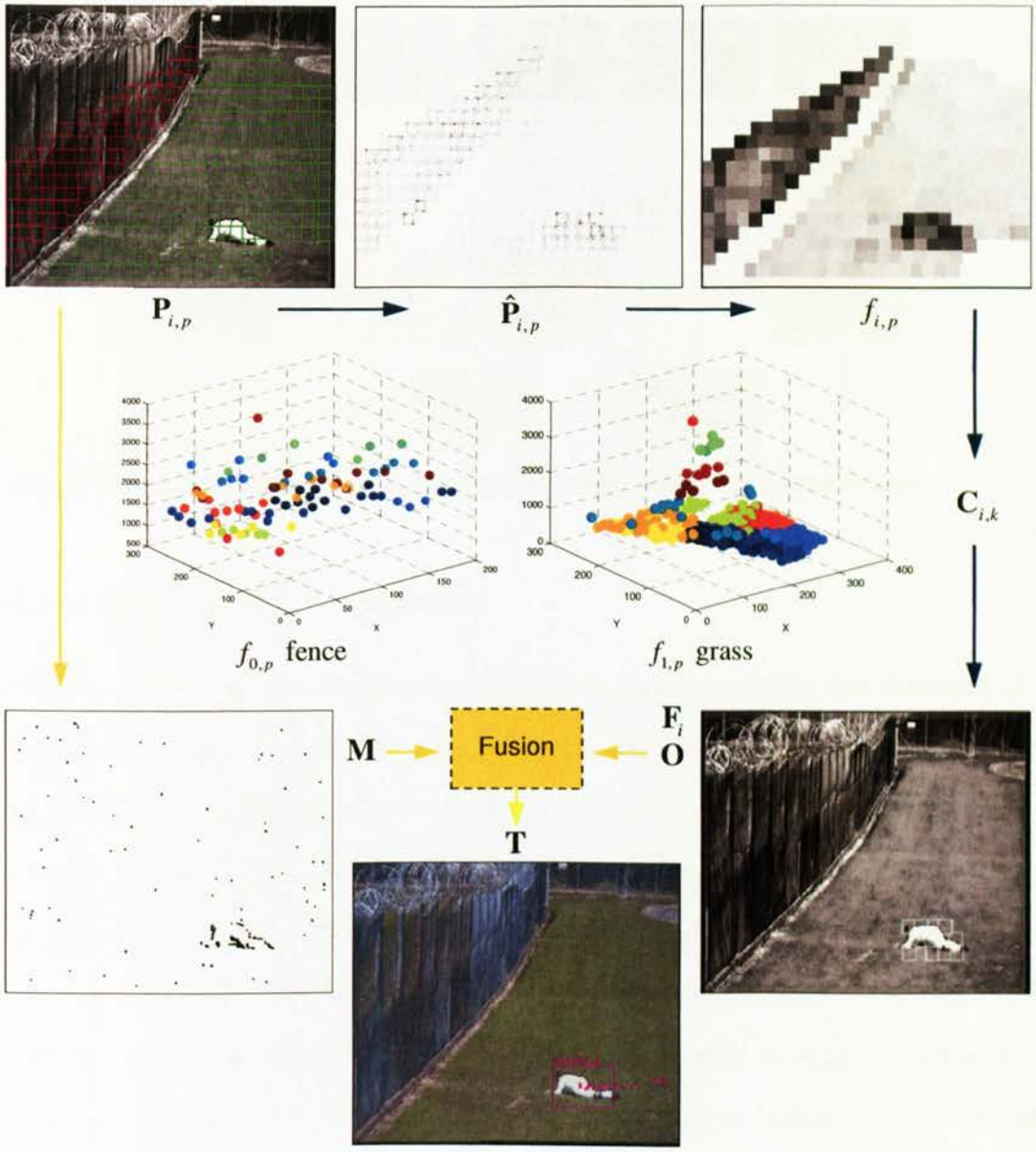


Figure 32 Pictogram of data flow of intrusion detection with motion extension. This corresponds to the block diagram in Figure 31 and uses the same colour code. The blue path gives an overview of the basic intrusion detection described in 4.3.1. The individual images are described in section 4.3.1. The orange path shows the extension with a motion mask $\mathbf{M}$ included on the bottom left.

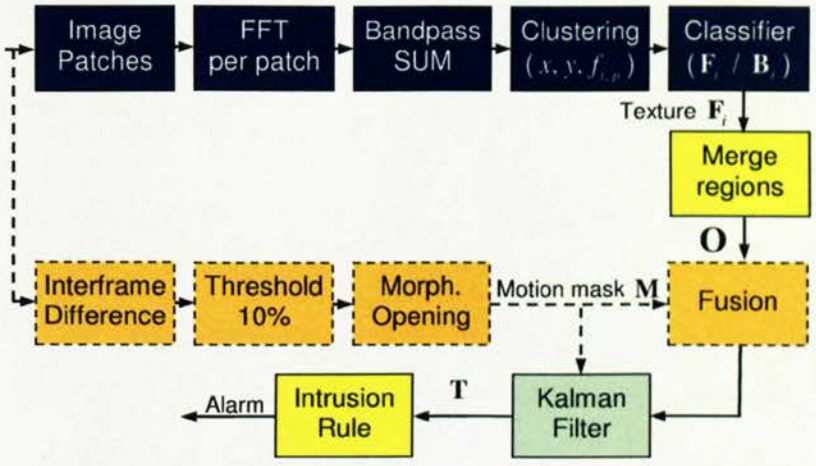


Figure 33 Block diagram for intrusion detector with Kalman Filter extension

4.4.2. Kalman Filter Extension

The algorithms described here so far suffer from a delay until the first detection of a person (*i.e.* latency). Very slow moving people stay partly occluded by the edge of the camera for a significant time which potentially delays detection. The second extension with a Kalman filter (Kalman, 1960) addresses this problem by allowing trajectories to be initialised purely by small motion regions. This motion estimation is very noisy. In contrast to the basic intrusion detection system, some filtering is required to provide consistency for trajectories. Please refer to Figure 33 for a block diagram and to Figure 34 for visual results. Examples in Figure 35 show people who may stay partly occluded until the latest possible alarm triggering time. The trajectory generation is now performed by a Kalman filter with a constant velocity model. First, silhouettes **S** are extracted from the motion mask **M** as connected components. This allows salient objects **O** as well as silhouettes **S** to update trajectories, but alarms still require saliency detection in addition to motion silhouettes at at least one point of a trajectory.

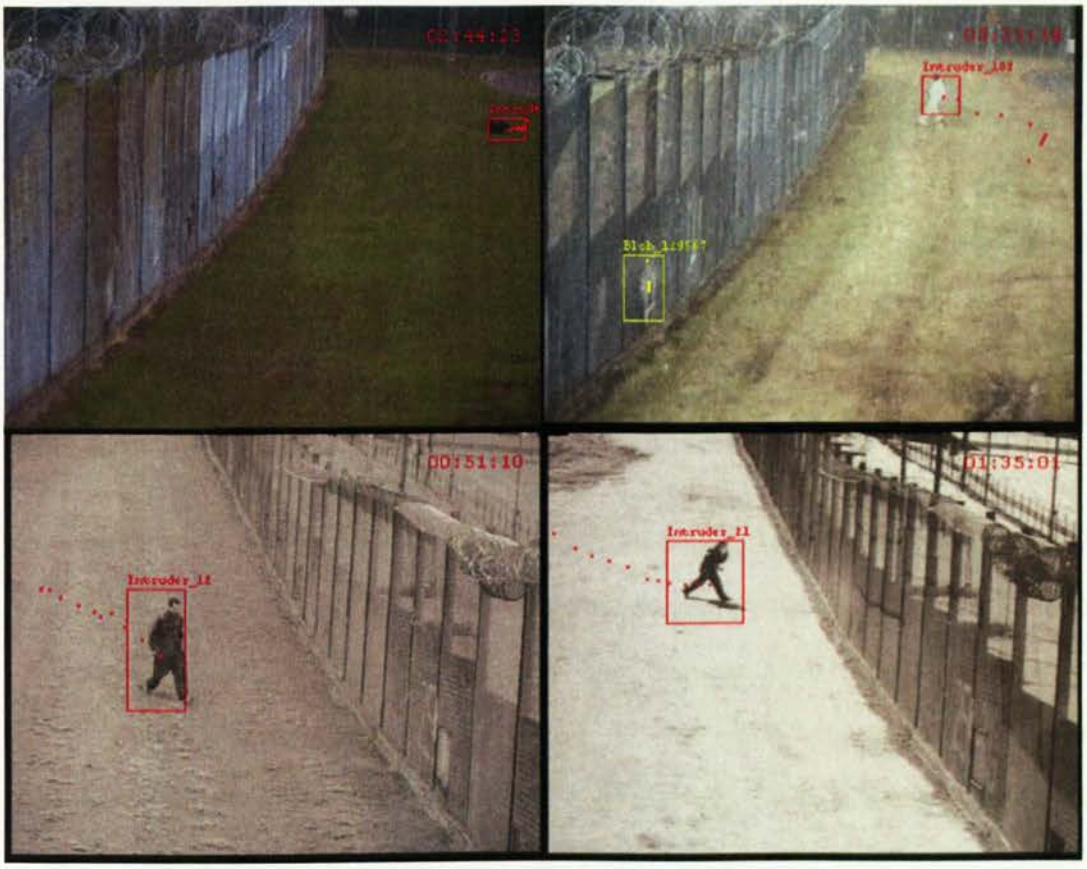


Figure 34 True positive examples of Kalman extension showing smooth trajectories. The second extension with a Kalman filter overcomes the problem of late detection by allowing trajectories to be initialised purely by motion. Note the person rolling sideways in the image on the left, which indicates the various ways the fence is approached in the i-LIDS data set.

New trajectories are initialised for both of those inputs (motion silhouettes or objects). Allowing motion silhouettes **S** to initialise trajectories **T** requires motion silhouettes of minimum size τ pixels to eliminate (analogue) video noise as discussed in section 4.3.1.3. Trajectories contain a sequence of object locations (x, y) over time, where the centroid of a motion silhouette **S** becomes the first object location in the trajectory. The saliency detector in comparison has a much higher precision and in practice does not require a minimum size filter. All trajectories **T** have an associated Kalman filter. To update those filters, a

measurement $\mathbf{z} = (x_m, y_m)$ of an object location is required. To associate trajectories and objects, the distance between a Kalman filter prediction $(\hat{x}, \hat{y})$ and object locations is evaluated. The closest object is used for the update according to equation (30). Positions of salient texture objects **O** are denoted (x_o, y_o) and for motion silhouettes (x_s, y_s) which defines the measurement selection as

$$\mathbf{z} = \begin{cases} (x_o, y_o) & \text{if } \min_o\left(\sqrt{(\hat{x}-x_o)^2+(\hat{y}-y_o)^2}\right) < \min_s\left(\sqrt{(\hat{x}-x_s)^2+(\hat{y}-y_s)^2}\right) \\ (x_s, y_s) & \text{else} \end{cases}. \quad (30)$$

The update with the motion silhouettes **S** allows trajectories **T** to start at the first appearance of a person at the edge of the camera and to fill temporal gaps in the saliency detection. The alarm delay time is thus reduced by this early detection of partly occluded people at the edge of the camera before the saliency detector triggers for the first time (see Figure 35). Saliency detection is mandatory for an alarm to be raised to overcome the limitations of motion only based systems discussed in section 4.2.

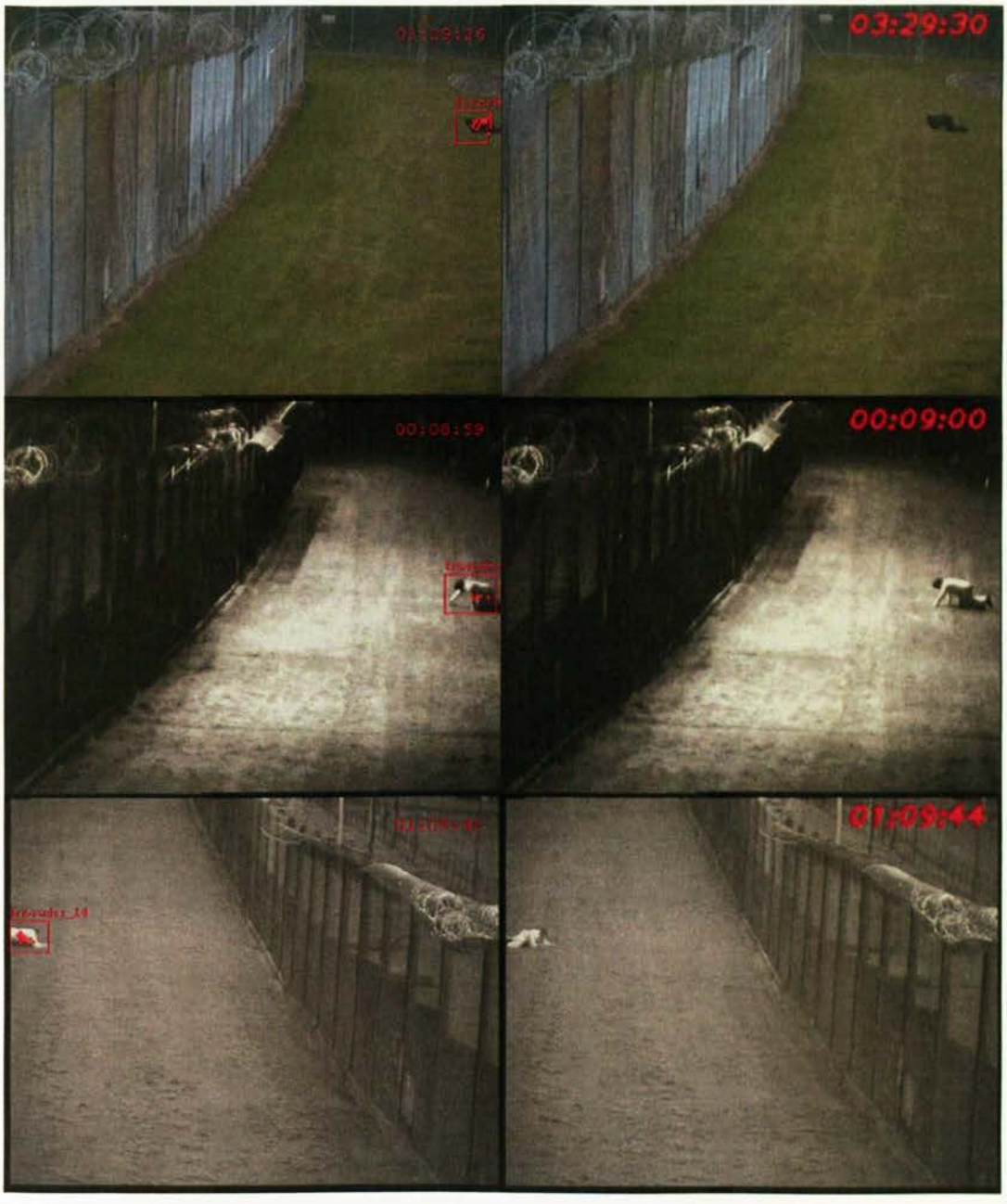


Figure 35 Comparison of alarm triggering time. The left column shows the frame when the system with Kalman filter triggered an alarm. The right column shows later alarms of the system without the filter, especially when intruders are partly occluded by the edge of the camera for a long time.

4.5. i-LIDS Testing

This section describes the framework and implementation of the proposed algorithm. The system has been tested on the i-LIDS data set, which is described with the particular requirements for system design. A runtime analysis for the real-time performance of the system is provided.

4.5.1. The Data

The i-LIDS challenge aims at providing a benchmark for systems which is defined by end users of the technology. The fact that the problem definition and data is generated by users ensures relevance and applicability of tested systems. Each data set comprises 24 hours of video sequences (2.160.000 frames) under a range of realistic operational conditions. The data set is limited in terms of number of views; however, producing a new view with the same variation of conditions carries a significant cost. The sterile zone test data set is used, which consists of two views (one colour, one black&white) during day and night with various weather conditions (rain, snow, fast moving shadows, *etc.*). The test requires to raise one alarm for every intrusion event and to compare the response with the provided ground truth. According to i-LIDS specifications, a valid alarm has to be raised no later than 10 seconds after the first appearance of an intruder. Each of the two camera views (View1 and View2) is split into a sequence with alarms (208 total) and a sequence without alarms but with various distractions (birds, rabbits, *etc.*) recorded over the duration of a whole year. Refer to Figure 22, Figure 34 and Figure 35 for detection examples.

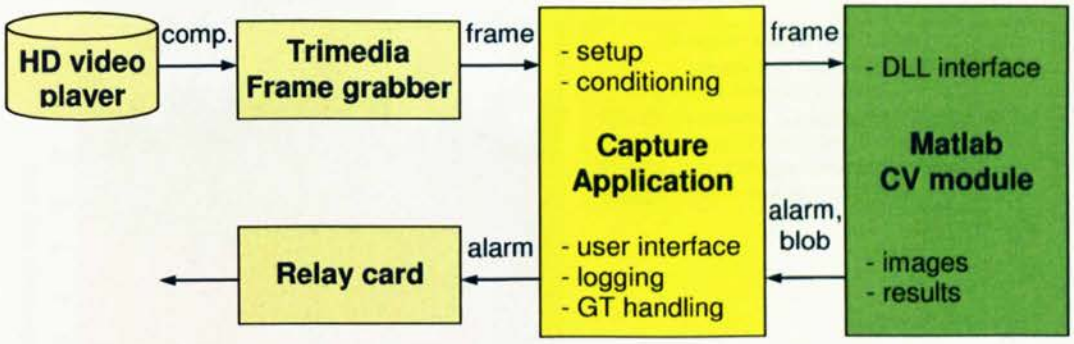


Figure 36 Block diagram of system implementation with frame grabber, capture application and Matlab computer vision module.

4.5.2. Framework

The system was designed according to i-LIDS requirements receiving an analogue video input with 25 frames per second at PAL resolution and providing a relay alarm output (see Figure 36). For the tests, the video was played back to the computer with a hard drive video player as composite video signal. A Trimedia frame grabber (NXP, nd) was used to sample the video and provide it to a capture application. The image processing was performed in a Matlab library, which is compiled and dynamically linked to the capture application. The capture application provides access to the hardware and performs conditioning of the input frames *e.g.* by allowing full control of brightness and contrast. This application also contains the user interface for ground truth handling and setting up of experiments. The Matlab module contains the algorithm described in this chapter by taking frames as input and providing alarms and trajectories as outputs.

4.5.3. Runtime Analysis

The system was tested on a Pentium 4 with 2.4 GHz and 1GB RAM. Real- time performance of 9 to 10 frames per second can be achieved with an average processing time of 81ms. Figure 37 shows the capture application's execution time over 200 processed frames. Overhead for the frame grabber is not shown. There is a

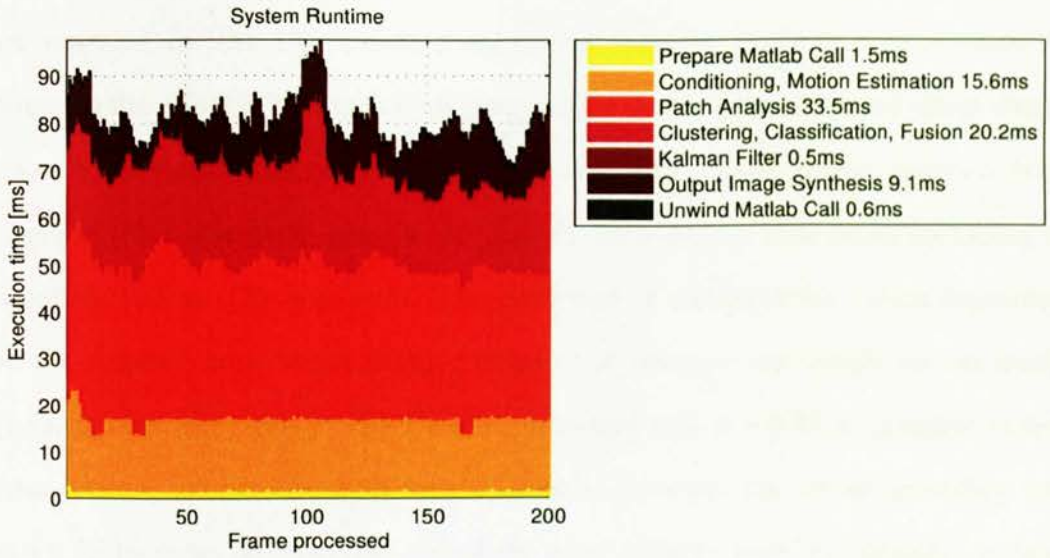


Figure 37 Runtime analysis of the whole system implementation with average runtime of every module.

little overhead for performing the Matlab call from the capture application of 1.1ms. The majority of time is spent for the patch analysis (FFT) and the subsequent clustering, classification and information fusion. The Kalman filter takes little time (0.5ms), but additional connected component analysis of the motion mask decreases the frame rate to 9 fps (from 10 fps) for the intrusion detector.

4.6. Results

This section describes the evaluation metrics, the baseline algorithm and gives qualitative results with analysis.

4.6.1. Metrics

The i-LIDS challenge defines event based evaluation, where only alarms reported within a window of 10 seconds of ground truth events are considered true positives (TP). Any alarms reported outside this window are false positives (FP). This is a somewhat arbitrary specification of the i-LIDS benchmark, especially as it does not consider the speed (*e.g.* slow) or location of an intruder. Later in this chapter, results

are reported for the 10 seconds window but also for a 20 seconds window to illustrate the effect of this metric. A person who might cause a second alarm due to a lost track would also count as false positive. Any missed person causes a false negative (FN). Recall R, precision P and $F1$ measure are calculated according to equations (10) to (12) on page 66. The recall bias α can have two values depending on the system's role, where higher values of α increase the weight on the recall. For a system to be considered for an operational role $\alpha = 0.65$ to penalise excess false alarms, which could disturb operators. Systems for event recording use $\alpha = 0.75$ to focus on detecting intrusions more reliably with less penalty on false alarms.

4.6.2. Baseline

The baseline used is a standard Kalman filter blob tracker with Gaussian background modelling based on the OpenCV library (OpenCV, nd) blob tracker (parameters FG_1, BD_CC, CCMSPF, Kalman). This algorithm belongs to the first class mentioned in the related work section, which estimates a stationary background with a Gaussian mixture model (GMM). Connected components are extracted from the foreground mask. Blobs are tracked by mean shift and resulting trajectories are post processed with a Kalman filter. The intrusion rule framework from section 4.3.2 is then applied to the trajectories. The main reasons for false detections are camera shake, fast illumination changes due to clouds, birds and changes from black & white to colour of the camera. This tracker is not without limitation, but it has been exposed to many applications and the behaviour is well understood so that the performance figures can be interpreted more easily.

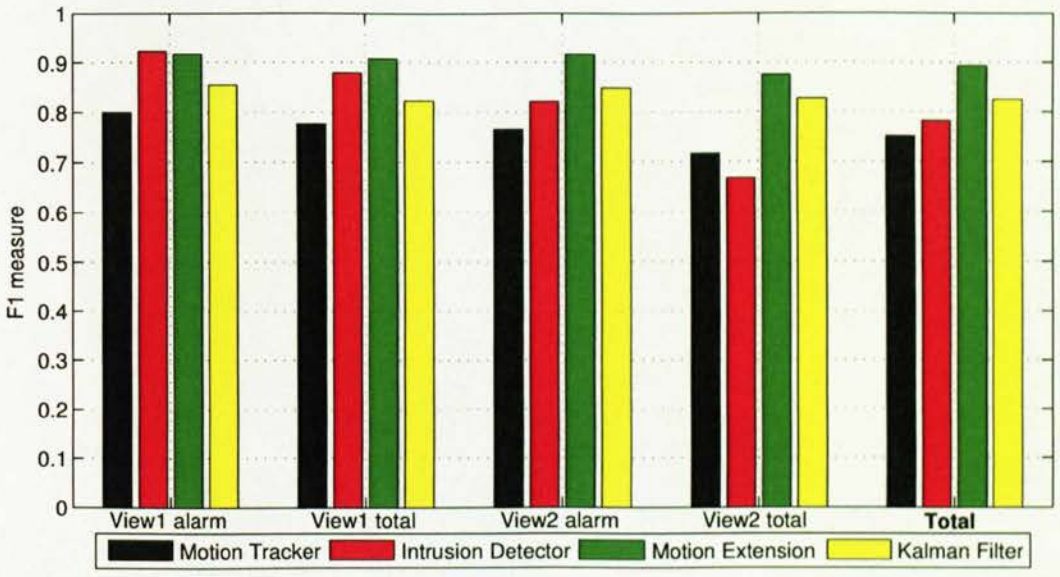


Figure 38 Performance for 10 seconds alarm window. Results are shown for alarming sequences, total per view including the non alarm sequences and total of the whole data set.

4.6.3. Analysis

Four algorithms are compared in this section (see Figure 38). The performance data is split into two camera views (View1, View2) and into sequences containing alarms and the total performance for the whole camera view. First is the baseline followed by the intrusion detector based on texture saliency. The final two algorithms incorporate the motion extension and the Kalman filter into the intrusion detector. All performance values are for operational alert $\alpha = 0.65$ unless stated differently. The baseline system achieves $F1 = 0.75$ in comparison to $F1 = 0.78$ of the intrusion detector. This outperforms the motion tracker, however, there are errors related to texture when shadows of the fences are detected. Low image contrast is the most common error cause and the reason for lower performance on View2, see Figure 39 for false positives from texture and false negatives from low contrast. A high detection threshold is required to eliminate false positives.

Figure 39 The top left image shows a wrongly detected bird flying towards the fence. The top right images shows a false detection due to fast moving clouds present at the same time as fence shadows, both errors are caused by texture. The bottom images show missed intruders due to low lighting conditions at night.

The intrusion detector with motion extension fuses information of those two approaches. It significantly outperforms both individual systems with $F1 = 0.89$ by exploiting the independence of the noise sources. A low threshold for saliency and motion detection allows reduction of false negatives from 35 to 17. To achieve this result, the saliency threshold was optimised resulting in $T = 2$, because lower thresholds produced arbitrary detection when no intruders were present in the image. With the fusion, the false positives are also reduced from 44 to 16. One disadvantage of the fusion is the increased time to generate an alarm which sometimes extends past 10 seconds for slow moving people.

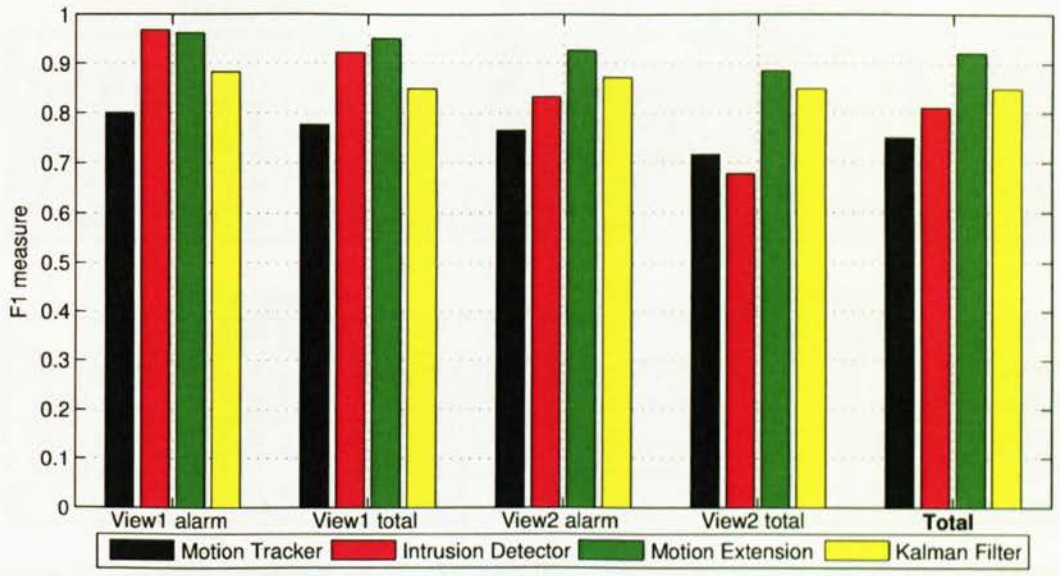


Figure 40 Performance for 20 seconds alarm window. An improvement compared to 10 seconds is noticeable for both intrusion detectors due to later correct detections of slow moving people.

This increased alarm time inspired the second extension by using Kalman filtering and initialising tracks from motion silhouettes $\mathbf{S}$ in the inter- frame difference mask $\mathbf{M}$. It is the last system shown in the figures. The minimum silhouette size $\tau = 5$, which is larger than the typical noise observed in the data (*e.g.* Figure 32). The performance of the Kalman filter extension is lower compared to the motion extension. This is due to a larger number of false positives particularly during the snow sequence. To keep those false positives down, the catch area for trajectories is kept small, which trades off some fragmented trajectories for people. Those trajectories are too short to alarm on which causes false negatives. The average alarm time in the 10 seconds window is lowered from 3.4 seconds for motion extension to 3 seconds for Kalman filtering, which was the aim of the extension.

Motion Tracker

	TP	FP	FN	*R*	*P*	*F1*
View1 alarm	85	17	28	0.75	0.83	**0.80**
View1 total	0	5	0	0.75	0.79	**0.78**
View2 alarm	62	10	33	0.65	0.86	**0.76**
View2 total	0	9	0	0.65	0.77	**0.72**
total	147	41	61	0.71	0.78	**0.75**

Intrusion Detector

	TP	FP	FN	*R*	*P*	*F1*
View1 alarm	107	2	6	0.95	0.98	**0.97**
View1 total	0	9	0	0.95	0.91	**0.92**
View2 alarm	66	3	29	0.69	0.96	**0.83**
View2 total	0	30	0	0.69	0.67	**0.68**
total	173	44	35	0.83	0.80	**0.81**

Motion Extension

	TP	FP	FN	*R*	*P*	*F1*
View1 alarm	108	4	5	0.96	0.96	**0.96**
View1 total	0	2	0	0.96	0.95	**0.95**
View2 alarm	83	3	12	0.87	0.97	**0.93**
View2 total	0	7	0	0.87	0.89	**0.88**
total	191	16	17	0.92	0.92	**0.92**

Kalman Filter

	TP	FP	FN	*R*	*P*	*F1*
View1 alarm	97	11	16	0.86	0.90	**0.88**
View1 total	0	7	0	0.86	0.84	**0.85**
View2 alarm	78	8	17	0.82	0.91	**0.87**
View2 total	0	4	0	0.82	0.87	**0.85**
total	175	30	33	0.84	0.85	**0.85**

Table 11 Detailed numbers of TP, FP and FN with *F1* measures for all four systems with alarm window setting of 20 seconds.

Performance figures improve for a larger alarm window of 20 seconds (Figure 40 and Table 11) which is caused by late correct detections. A late detection carries a high penalty according to the i-LIDS specification, as it is counted as false negative and false positive at the same time. The best overall performance is $F1 = 0.92$ for the intrusion detector with motion extension with the best performance for View1 of $F1 = 0.95$. View2 suffers from very low contrast, which is a particular problem for the detector; however, the motion extension improves the performance significantly by reducing the false positives from 30 to 7.

Finally, the $F1$ measure is compared for the two values of recall bias α. When using the event recording setting $\alpha = 0.75$ the motion tracker performance is reduced by 0.3%. In contrast, the intrusion detector has increased performance of 0.1%. The other two systems are not affected by α due to an even balance between false positives and false negatives in the results.

4.7. Summary

This chapter proposed a new texture saliency classifier to detect objects in still images of the i-LIDS sterile zone data set. The intrusion detector has been implemented in C++ and Matlab to operate in real time from analogue video input. This approach overcomes the typical limitations of background modelling based solutions. The runtime of 9 frames per second of the implementation is discussed in detail. The detector outperforms the OpenCV blob tracker as baseline. A first extension with information fusion of appearance and motion increases the performance significantly to $F1$=0.92 on the 24 hour test data set. The second extension using a Kalman filter is used to improve alarm response times, however it degrades the overall performance due to more false positives. The false positives are caused by noisy motion based foreground estimation. The results demonstrate good performance for local features in surveillance tasks with minimal reliance on motion information.

This part of the work was important because it gave rise to the concept of local feature patches which is carried forward to the work described in the next chapter, where it is combined with 3D spatial models. This allows the combination of operation on still images reducing reliance on motion estimation with 3D spatial modelling. This additional concept extends the application range from intrusion detection to vehicle classification.

5. 3DHOG Classifier

5.1. Introduction

This chapter describes the 3D extended Histogram of Oriented Gradients (3DHOG) classifier for vehicle and pedestrian detection. This new concept, developed by the author, extends 3D models from chapter 3 with the idea of local features evaluated in chapter 4. The overall concept including camera calibration remains the same as in chapter 3. The model matching process is changed and solved by also considering the appearance of objects. The spatial 3D models are combined with patch based appearance models to make the model matching independent of motion silhouettes (example results in Figure 41). Given a hypothesised (or known during training) object position and orientation on the ground plane, the model matching takes place in normalised 3D space. To do this, a 3D appearance representation of objects is constructed from 2D images for matching using pre-defined spatial models. Those 3D appearance representations are incomplete and only contain data from the visible part of objects in the 2D frame, which can vary depending on the view point. The classifier uses this incomplete representation of a new object to match it against a trained and complete 3D appearance model of the known classes. In this way, the classifier requires only a single but complete 3D appearance model (containing data from many viewing angles) to deal with any object view point.

Figure 41 Example views from the i-LIDS data set with detected and classified pedestrians and vehicles using 3DHOG

Firstly, the spatial 3D models have to be extended to incorporate interest points[2] on the surface of the models. Appearance information (local features) at the location of those interest points will be extracted to construct the appearance models and to classify newly seen objects. The local features themselves are constructed out of histograms of oriented gradients (HOG). The combination of those 3D interest points and HOG is hence introduced as the novel 3DHOG feature. Performance is evaluated, comparing 3DHOG with FFT and histogram-based local features. Given a hypothesised object location and orientation, the new feature allows model matching using a variable number of interest points (depending on visibility and self occlusion). Trained models can be matched against objects in any given viewing direction. The framework can deal with part occlusions *e.g.* by the edge of the camera, which is shown in section 6.2.

The remainder of this chapter is organised as follows: The next section discusses related work. Section 5.3 describes the spatial models and how interest points are used in those models. Section 5.4 describes the appearance feature

[2] In this work, the term interest point is used to refer to a point or location, around which local features will be extracted (*i.e.* they define the location of an image patch for computing features). The location of the point is not directly determined from the image data, but defined by the models. This is a slightly different use than it is common in generic object recognition, where interest points are extracted from images by interest point detectors (*e.g.* Corners, SIFT, Hessian).

extraction process using the spatial models. The training of the appearance models is described in section 5.5. The classification framework (extended from section 3.3) is given in section 5.6 with performance evaluation in section 5.7. The chapter is summarised in section 5.8.

5.2. Related Work

The process of classifying images or objects in images can be generally categorised either as 'top-down' (usually visual surveillance) or 'bottom-up' (usually object recognition) approaches, as outlined in chapter 2. For 'top-down', the whole context is analysed simultaneously or used to verify a hypothesis during searching. Motion silhouettes are generated from background modelling and classification is performed based on motion silhouette measurement features (Morris and Trivedi, 2006a, Song and Nevatia, 2007, Messelodi *et al.*, 2005b). This approach is vulnerable to inaccurate foreground segmentation, which is inherent to urban environments due to low camera angles, occlusions, *etc.* The above 2D approaches can be extended to 3D for vehicle detection and classification as in Song and Nevatia, 2007, Messelodi *et al.*, 2005b, Park *et al.*, 2007, Ottlik and Nagel, 2008 and chapter 3.

In contrast to the above, 'bottom-up' approaches are usually targeted at object categorisation and classification, especially of still images. A constellation model with SIFT features is used in (Ma and Grimson, 2005) for vehicle classification. The *implicit shape model* is used in (Leibe *et al.*, 2005), (Leibe *et al.*, 2007) and (Leibe *et al.*, 2008b) for pedestrian detection and shows the object recognition community moving towards surveillance applications (Liebelt *et al.*, 2008, Pingkun *et al.*, 2007). 'Top-down' and 'bottom-up' approaches are combined by (Dalal and Triggs, 2005), using local features with 2D fixed spatial constraints.

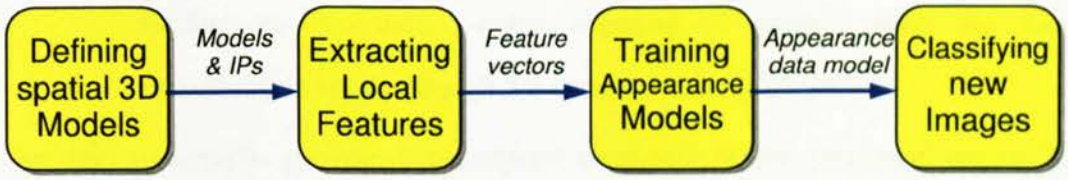


Figure 42 Overview of the algorithm. This block diagram outlines the relationship between the different stages of the algorithm.

This is used for pedestrian detection and for action recognition with temporal extension in (Kläser *et al.*, 2008, Wang *et al.*, 2009a).

5.2.1. 3DHOG Detector and Classifier

The new approach takes the good results from 3D models into account (Song and Nevatia, 2007, Messelodi *et al.*, 2005b) and incorporates local appearance features into the models. The whole framework is outlined in Figure 42 and follows four steps, which are described in the subsequent sections:

- *Defining spatial 3D models* in section 5.3 describes how models from chapter 3 are extended to include interest point locations.
- *Extracting local features* in section 5.4 deals with the feature extraction process based on the spatial models above. The feature extraction will then be used for both training and classification.
- *Training appearance models* in section 5.5 shows how manually labelled training images are used to generate appearance data models from features.
- *Classification framework* in section 5.6 brings all the above together: given a road user hypothesis from motion information as in chapter 3, the feature extraction uses the spatial models to generate local features for a new image. Those features are matched against the appearance models to provide match measures for the hypothesis. The process of

finding the maximum match measure of models gives the final classification result through the same process as chapter 3.

The next paragraphs provide a conceptual overview of the proposed use of 3D spatial and appearance models to qualify the above four steps. The new method defines the local features and the spatial relationship between them in 3D world space. In this way, the appearance-based model matching can be performed in 3D space as outlined in the introduction. This is beneficial as it is a normalised space where scale is defined. It also allows a single complete appearance model to be used for any viewing angle. In general, the method of histogram of oriented gradients (HOG) using a planar 2D search window (Dalal and Triggs, 2005) is generalised to 3D by conceptually 'wrapping' the camera image around the models (Figure 43) like in (Starck and Hilton, 2005). Using calibrated cameras, obtained in a relatively straightforward way given a plan map of the scene, the scale of objects is determined directly, in contrast to the multiple scale search in (Dalal and Triggs, 2005). The search space is now the ground plane as it was in chapter 3. By introducing a model match framework that deals with variable numbers of visible interest points, a single appearance model can be used to match objects from any angle. The trained classifier is portable between different cameras, only requiring the calibration of a new camera or a new camera position. This will be shown in section 6.2.

The algorithm detects rigid vehicles and pedestrians in the same way and does not use special cases. Texture is used to generate local features which do not rely on potentially noisy motion information. This implies that the method could be applicable in cases where reliable motion information is not available, *e.g.* stationary objects, single frames and moving cameras. The match measure is calculated in feature space in contrast to chapter 3, where the calculation is based on visual overlap directly calculated from the motion silhouette.

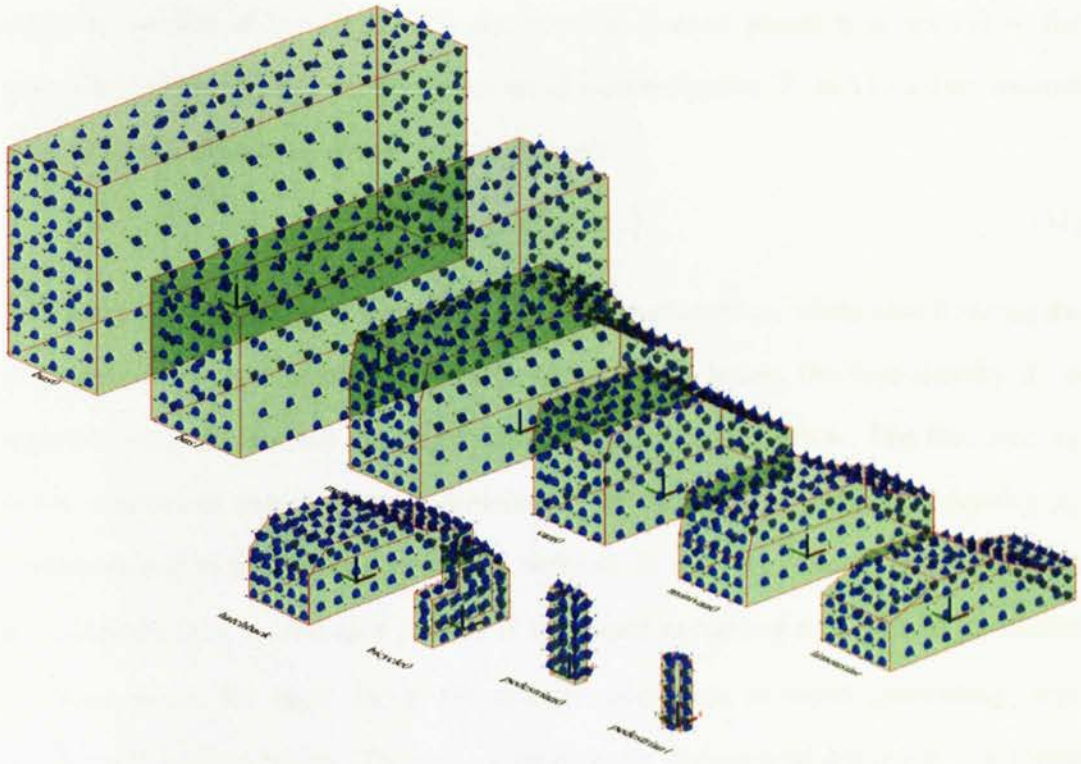


Figure 43 3D spatial models taken from chapter 3 extended with interest points. The interest points are illustrated as cones, which signify the position and also normal direction of interest points. The diameter of the cone will later be used to visualise the interest point's weight.

5.3. Defining Spatial 3D Models

The road users' models from chapter 3 will be used as input to define interest point locations in model space. Those locations will then be used, as explained in the next section, to extract features. The positions of a set of interest points are defined to be on a grid, located on the faces (also known as polygons) of 3D models (Figure 43 similar to chapter 3). The method, described in more detail next, is applied to all models and to keep the expressions succinct there is no model index subscript. An interest point in model space $\mathbf{p} = (x, y, z, \mathbf{e})$ is determined by its 3D location (x, y, z) in model space and orientation $\mathbf{e} = (e_x, e_y, e_z)$. A set of interest points $\mathcal{P} = \{\mathbf{p}\}$ on a face is defined on a regular grid with linear face density d_f around an

origin $\mathbf{p}_0$ (centre of the face). The direction of interest points $\mathbf{p}$ is normal to the face. The function grid( ) produces the set of interest points $\mathcal{P}$ as a two dimensional array of points filling the whole face (polygon):

$$\mathcal{P} = \text{grid}\left(\mathbf{p}_0, d_f\right). \tag{31}$$

To ensure good coverage for small faces of *e.g.* pedestrians, while also limiting the total number of interest points for large faces of *e.g.* buses, the face density d_f is adjusted according to face size s_f following equation (32) below. The face size s_f is the maximum extent (largest dimension) of the face. A reference density d_0 corresponding to a reference size s_0 is defined. If the face size s_f is smaller than the reference size s_0, the face density is increased according to a growth parameter γ. Vice versa, for large faces, the density decreases to avoid generating large numbers of interest points. The rate of increase or decrease of density is controlled by growth parameter γ:

$$d_f = \frac{d_0}{(1-\gamma)+\gamma\frac{s_f}{s_0}} \tag{32}$$

For the experiments $s_0 = 4\text{m}$, $\gamma = 0.35$ and $d_0 = 4$ interest points per metre were used. This trades off over sampling against very sparse interest points. In the case of low reference density d_0 and therefore sparse interest points, image patches would need to be made large to cover the spatial model without gaps between patches. This would then in turn lead to global rather than local features and hence to loss of discriminating power.

The sets of interest points on individual faces $\mathcal{P}$ are combined to form a full set of interest points $\mathcal{P}'$, which contains all the interest points of a given model. A typical car contains 300 interest points in this set $\mathcal{P}'$. The full set $\mathcal{P}'$ contains interest points for any viewing direction of the model. This can be observed in

Figure 43, where interest points are displayed as cones. The orientation of the cone corresponds to the orientation vector $\mathbf{e}$ of interest points $\mathbf{p}$.

5.4. Extracting Local Features

Local features are extracted from input images, using interest points as defined in the previous section, given an object location in the image. The object location might be hand labelled during training or might be a hypothesis during classification of new images. This same feature extraction is used in both appearance model training (section 5.5) and classification (section 5.6). First, for a candidate object location, image patches are obtained for interest points that are sufficiently visible as explained in the next section. Feature vectors are then calculated from those patches, as explained in section 5.4.2.

5.4.1. Extracting Normalised Image Patches

The visibility of interest points is first confirmed, before image patches are extracted at interest point locations. The extraction process automatically resolves the scale (*i.e.* depth) and perspective distortion (orientation) of the observation and presents a constant size image patch for feature extraction.

Visibility of interest points

The locations of interest points in real world coordinates are used to extract visible image patches. A given object ground plane location $\mathbf{x} = (x, y, z, r)$ with ground plane orientation r is required as input. Road users are assumed to be on the ground plane, which results in only one degree of freedom for the orientation angle r. Using this location $\mathbf{x}$, points in the model coordinate system can be transformed to the real world coordinate system by adding the location $\mathbf{x}$ to the model coordinates (Dunn and Parberry, 2002). All further descriptions will be in reference to the real world coordinate system. Let $\mathbf{v}$ be the unit vector of the viewing direction in real

world coordinates of interest point $\mathbf{p}$. The visible set of interest points $\mathcal{P}'' \subseteq \mathcal{P}'$ is a subset of all interest points $\mathcal{P}'$ of a model determined by the visibility threshold $\tau_v = 0.65$ to ensure some minimum visibility:

$$\mathcal{P}'' = \{\mathbf{p} | \langle \mathbf{e}, \mathbf{v} \rangle > \tau_v, \mathbf{p} \in \mathcal{P}'\}. \tag{33}$$

The above equation calculates the dot product between the direction $\mathbf{e}$ of the interest point and the viewing direction $\mathbf{v}$. The dot product will be 1, if both vectors are pointing in the same direction and the interest point is viewed head on. The product decreases, if the interest point is less aligned and is viewed further from the side. If the interest point direction is perpendicular to the viewing direction, the dot product will be 0 and the point will be invisible at the same time.

Patch image extraction

A set of 2D square image patches $\mathcal{I} = \{I\}$ in real world space is extracted for every visible interest point. Due to the perspective, this patch may not correspond to a square area in the original input image. An affine transformation will be estimated to map this distorted part of the input image to a square normalised patch in ground plane space. One square image patch I is defined for every visible interest point $\mathbf{p} \in \mathcal{P}''$ with constant pixel width $l_p = \delta \cdot \rho$ using constant 3D world resolution ρ in pixels per metre and width δ in metres, allowing some overlap of patches. Values for all parameters will be provided in the evaluation section 5.7. An affine transformation is used to map pixels of the input image with coordinates (u, v) to patch images I with coordinates (x, y). This produces the set of visible image patches $\mathcal{I}$ illustrated in Figure 45. The affine transformation between coordinates (u, v) and (x, y) used for image mapping is defined as

$$\begin{aligned} u &= c_0 x + c_1 y + c_2 \\ v &= c_3 x + c_4 y + c_5 \end{aligned}. \tag{34}$$

The transformation has 6 parameters $\mathbf{c}=(c_0,c_1,c_2,c_3,c_4,c_5)$, which need to be calculated. By providing three corresponding points between both coordinate systems, the resulting set of 6 equations can be directly solved for the 6 parameters $\mathbf{c}$. Figure 44 illustrates the transformation process in red. The next paragraph describes how the blue points are generated.

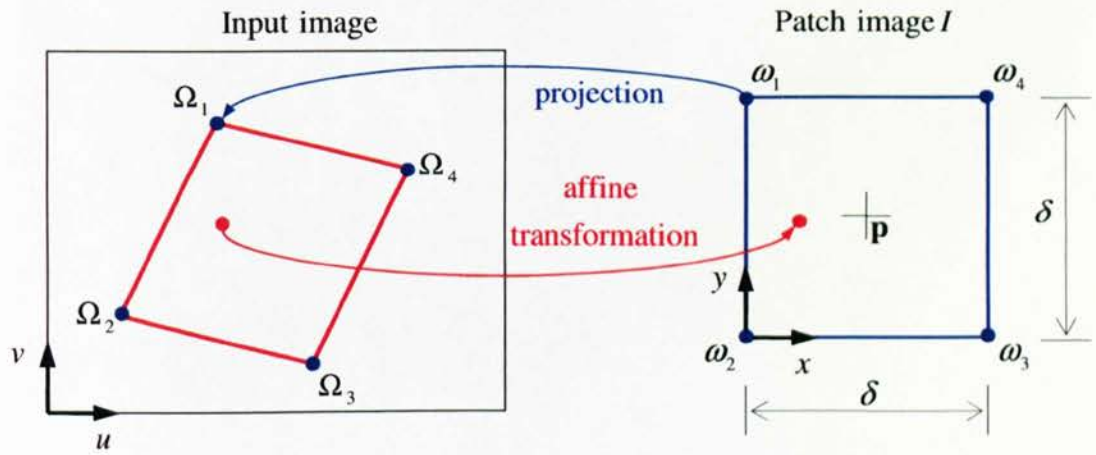


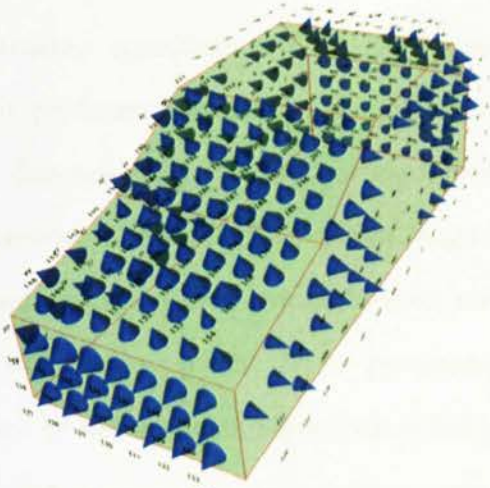
Figure 44 Illustration of patch image extraction

The corresponding points are generated by projection from real world space to image space. Visible interest points $\mathbf{p}\in\mathcal{P}''$ specify the centres of corresponding image patches $I\in\mathcal{I}$. Four corner points $\omega_1..\omega_4$ of image patch I are calculated by generating a square shape with width δ in a plane (2D) perpendicular to the orientation $\mathbf{e}$ of interest point $\mathbf{p}$. Corner points $\omega_1..\omega_4$ can be projected to the camera image coordinates generating points $\Omega_1..\Omega_4$. The same projection as in section 3.3 on page 58 was used. The transformation parameters $\mathbf{c}$ can be calculated from any 3 of the above point pairs.

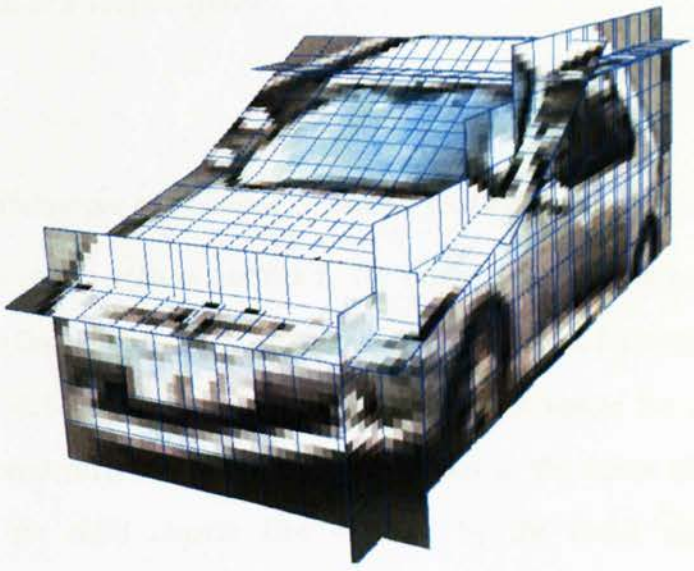
The cardinality Ic of the set of extracted image patches $\mathcal{I}$ is variable depending on the viewing direction of the model. The overall process can be viewed as one of wrapping the camera image around the model resulting in invariant representations for any 3D location and viewpoint. As a final step, histogram stretching is applied to the individual image patches in I to achieve additional

illumination independence and normalisation. So at the end of this process we have a set of image patches I with normalised images and we are now ready to extract features from those images, as explained in the next session. Please refer to Figure 45 for an example of extracted image patches I.

a)



b)



c)

Figure 45 a) Input image and b) hatchback model. The radii of cones indicate the weights q (described later) of interest points $\mathbf{p}$. c) shows the set of extracted image patches $I \in \mathfrak{I}$.

5.4.2. Generating Patch Features

The image patches I extracted as explained in the previous section are used to generate normalised feature vectors $\hat{\mathbf{f}}_k, k = 1..Ic$ with examples shown in Figure 46. Those features are then used to train appearance models and also to classify new images in subsequent sections. Three alternative algorithms (HOG, FFT and Histogram) will be considered here and their performances will be compared in section 5.7. The length of feature vectors $\hat{\mathbf{f}}_k$ depends on the individual algorithm used, but the training and classification framework is independent of that length. The index k of vectors $\hat{\mathbf{f}}_k$ enumerates the image patches I from which features are generated. This is used to emphasise, that every patch I is processed independently. Vectors $\mathbf{f}_k$ provided by any one of the available algorithms (HOG, FFT or Histogram) are normalised by the Euclidean norm for better performance, following (Dalal and Triggs, 2005):

$$\hat{\mathbf{f}}_k = \frac{\mathbf{f}_k}{\|\mathbf{f}_k\|} \tag{35}$$

5.4.2.1. 3D Histogram of Oriented Gradients (3DHOG)

The generation of the feature vectors $\mathbf{f}_k$ for image patches I is performed in the same way that (Dalal and Triggs, 2005) generate the vectors for single cells. First, a Sobel kernel $(-1,0,1)$ is used to compute the gradient image for all three colour channels independently. The angles are calculated in the range $[0, 2\pi]$ as this is recommended for rigid objects like vehicles by the Dalal and Triggs. The alternative would be the interval $[0, \pi]$, which would consider gradients from light to dark identical to dark to light. A single histogram is generated for every image patch with η bins. The highest gradient magnitude of the three colour channels is used for the histogram. In the process described in section 5.4.1 earlier, the visible part of 3D models is used to extract warped 2D patches, which can be conceptually seen as '3D surface windows' generalising the concept of planar 2D windows in the

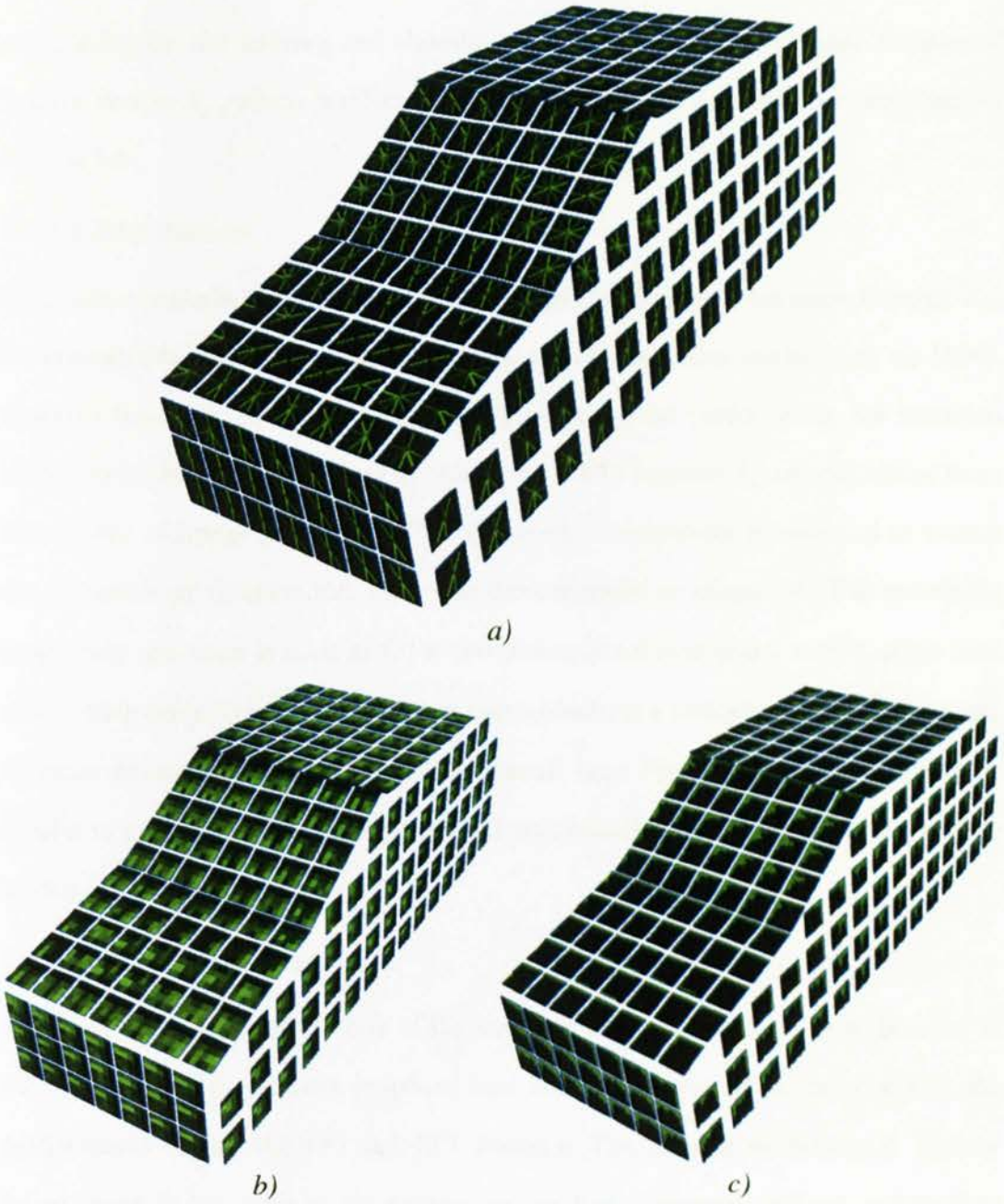


Figure 46 Feature vectors $\hat{\mathbf{f}}_k$ generated from the set of image patches $\mathcal{J}$ in Figure 45. a) 3DHOG features, b) spectral features (FFT) and c) image histogram.

seminal paper of (Dalal and Triggs, 2005). Due to changes in the viewpoint of objects, the number of visible interest points changes according to equation (33). This directly changes the number of feature vectors $\mathbf{f}_k$ and therefore makes the concatenation to a single constant size feature vector impossible. This fact adds

complexity for the training and classification to allow for this variable number of feature vectors $\mathbf{f}_k$, which is efficiently dealt with by a new framework described in section 5.6.

5.4.2.2. FFT feature

Alternative features to HOG are used to compare performance between features and to demonstrate that the overall framework is not dependant exclusively on HOG. Features based on FFT have previously provided good performance for intrusion detection in chapter 4. Fast Fourier transform (FFT) features $\mathbf{f}_k$ are calculated from the spectra of image patches I. The DC (direct) component is removed to reduce the influence of illumination, as it was demonstrated in chapter 4. The remaining magnitude spectrum is used to fill a two dimensional histogram with η angle bins and ν frequency bins. Every angle bin corresponds to a sector in the spectral image, whereas frequency bins correspond to annuli (see Figure 47). This approach is similar to using banks of Gabor filters and accumulating the responses into a feature vector.

5.4.2.3. Histogram feature

The grey level histogram is one of the simplest image features that can be used in the classification framework proposed here and thus it is used to compare with the performance of the 3DHOG and FFT features. The number of bins is η. Colour information is not used in the histogram, so that a general, colour independent model is learned for road user classes and described in the next section.

So at this point we have three possible candidate features on which to test performance. The next section will deal with how these appearance features can be used to train a system to recognise given classes of road users.

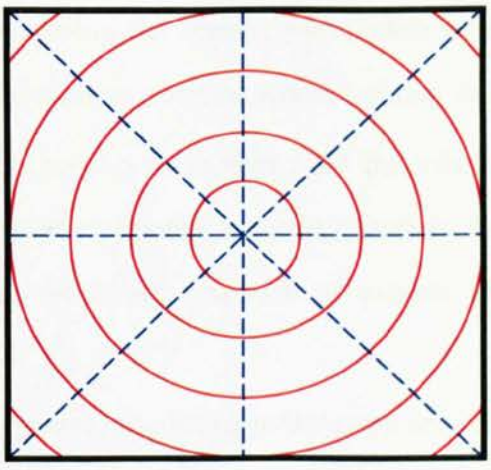


Figure 47 Illustration of histogram in the FFT feature extraction process. The continuous red lines indicate the frequency borders, whereas the dashed blue lines indicate the angle borders.

5.5. Training Appearance Models

Annotated training images are used to generate training feature vectors (representing the appearance of image patches) to obtain trained appearance models. The annotations consist of road user positions in the training images. These appearance models are then used in the classification of new images. Training takes place using the following five steps (Figure 48):

- *Training data and annotation* in section 5.5.1 defines the source of training data and the object location annotation required for training.
- *Feature extraction* was covered previously in section 5.4 and generates feature vectors for the training data.
- *Gaussian appearance models* in section 5.5.2 shows how Gaussian models are generated from feature vectors, so that distances can be calculated between the appearance models and newly seen images.

- *Sigmoid parameters for model normalisation* in section 5.5.3 provides a means of improving the appearance models by enforcing a normalised model match response over the whole training data for all interest points.
- *Interest point weights* in section 5.5.4 describes a further improvement by weighting interest points according to their ability to localise objects. In this way, the match response of models for new images can be improved.

The next sections are organised according to the steps above.

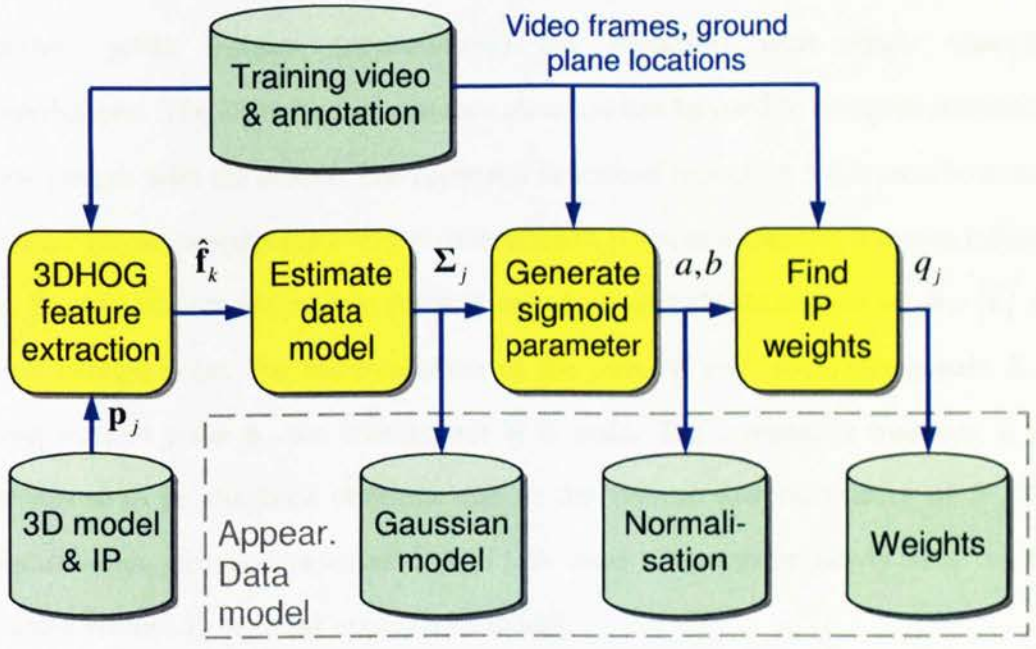


Figure 48 Block diagram for the training of appearance models. Features are extracted from training videos given object location annotation and the 3D models with interest points. A Gaussian model and subsequently normalisation coefficients and weights are calculated from the features.

5.5.1. Training Data and Annotation

The training set comprises frame images from the i-LIDS data set with labelled road user locations. A set of model locations on the ground plane $\mathcal{L} = \{\mathbf{x}\}$ represents the

annotation for the training. Those positions $\mathbf{x}$ were generated with the algorithm in chapter 3 for the training videos and manually refined, where necessary. Alternatively, the location of road users in training videos could be hand labelled. There are 20 to 30 labelled images for every type of road user model in the training data. It is important to ensure, as far as possible, that every interest point $\mathbf{p} \in \mathcal{P}'$ of models will be visible in the training data, so that their appearance can be learned as described in the next section.

5.5.2. Gaussian Appearance Model for Interest Points

Interest point features (appearances) are modelled with single Gaussian distributions. The Mahalanobis distance measure can be used to compare features of new images with the model. The approach described in section 5.4 is used to extract feature vector samples for every visible interest point in a training frame at location $\mathbf{x}$. Sample vectors per interest point $\mathbf{p}$ are accumulated into sample set $\mathcal{S} = \{\hat{\mathbf{f}}\}$ for each interest point. For the estimation of the mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ of each interest point $\mathbf{p}$, the training set $\mathcal{S}$ is used. The covariance matrices $\boldsymbol{\Sigma}$ are estimated to be diagonal matrices due to the typical low cardinality of $\mathcal{S}$. The Mahalanobis distance measure $\mathrm{DM}(\mathbf{f})$ is used to compare newly seen visible feature vectors $\hat{\mathbf{f}}_k$ with the appearance model:

$$\mathrm{DM}(\mathbf{f}) = \sqrt{(\mathbf{f}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{f}-\boldsymbol{\mu})}\,. \tag{36}$$

The above equation will be used for classification and also for model refinement as discussed in the next sections. For a joint day/night classifier, the single Gaussian model could be expanded to multiple Gaussians to capture the potentially changed appearance of road users at night.

5.5.3. Sigmoid Parameters for Model Normalisation

After estimating the Gaussian model for every interest point, the appearance model match performance of every individual point can be improved considerably by normalising their response to training images with a sigmoid function. Individual interest points have to be dealt with, because Support Vector Machine (SVM) classification as in (Dalal and Triggs, 2005) is not possible due to the variable number of visible interest points here. The variability stems from variable viewpoints of spatial models. As appearance match responses of different interest points can vary, the average response for different appearance models can be inconsistent. The normalisation with a sigmoid function addresses this limitation. First a surface of model match responses (distance surface) is generated by moving spatial models slightly away from training positions and checking the response as explained in the next section. This surface is then used to parameterise the sigmoid function as described in section 5.5.3.2.

5.5.3.1. Distance surface at training position

To generate a distance surface per interest point $\mathbf{p}$, features are calculated with spatial models moved away from the exact training position. In this way, the change of appearance model match in respect to ground plane movements of spatial models is evaluated. Good interest points should exhibit a strong drop in match response when moved away from the training position. A regular grid of positions $\mathbf{g}_M$ with $M = -M_{\max}..M_{\max}$ is generated for every position $\mathbf{x}$ in training set $\mathcal{L}$, similar to the hypothesis grid in section 3.3 on page 58. The size of the grid is set to 4m with 9 steps. This corresponds to a shift between grid points of approximately half an image patch and a total displacement of twice the patch size in every direction. Based on those dimensions, the location sensitivity of interest point appearance models can be assessed for normalisation in the next step and weight estimation as in section 5.5.4. The Mahalanobis distance $\mathrm{DM}(\hat{\mathbf{f}}_M)$ in equation (36) between the

interest points' models $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ and the extracted feature vectors $\hat{\mathbf{f}}_M$ at model positions $\mathbf{g}_M$ gives a distance surface $\mathbf{D}_M$ for every interest point $\mathbf{p}$ at every training position $\mathbf{x}$:

$$\mathbf{D}_M = \mathrm{DM}\left(\hat{\mathbf{f}}_M\right) \tag{37}$$

A mean distance surface $\bar{\mathbf{D}}_M$ (Figure 49) over all training samples $x \in \mathcal{L}$ of interest points $\mathbf{p} \in \mathcal{P}'$ is defined by the sum of all distance surfaces $\mathbf{D}_M$ generated from training samples $x \in \mathcal{L}$ divided by the number of elements in training set $\mathcal{L}$

$$\bar{\mathbf{D}}_M = \frac{\sum_{\mathcal{L}} \mathbf{D}_M}{|\mathcal{L}|}. \tag{38}$$

The above distance surface represents the average match response between the trained appearance model and the training data of an interest point. The next step will be to find parameters of a sigmoid function to normalise this response across all interest points.

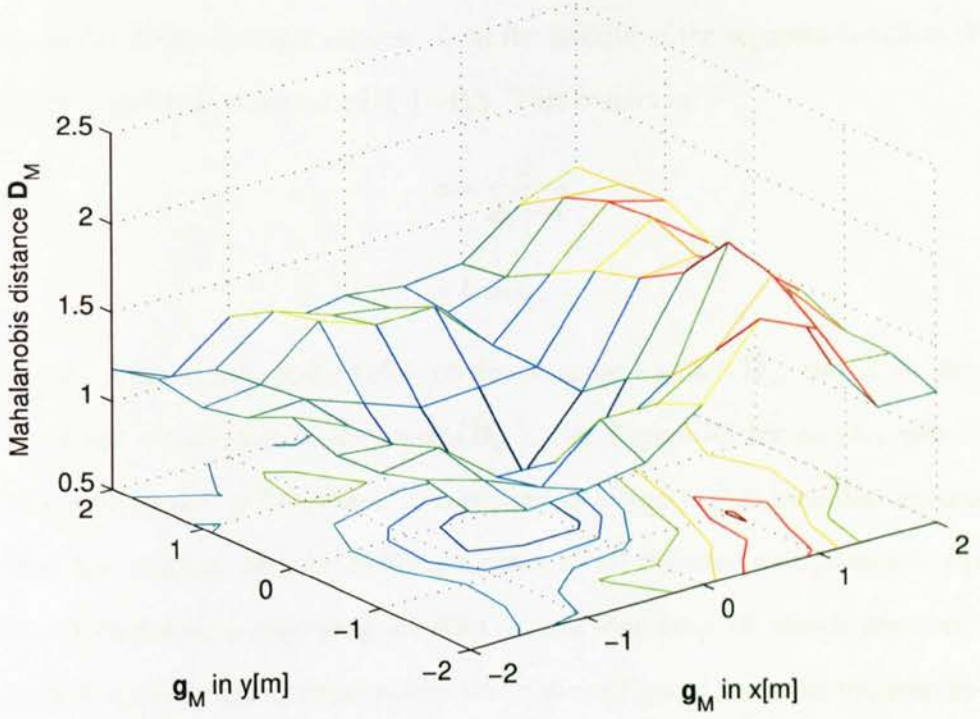


Figure 49 Example average feature distance surface $\bar{\mathbf{D}}_M$. The centre d_C at position $(0,0)$ corresponds to the training position x and has usually the lowest value. The feature distance increases for coordinates further away from the training position.

5.5.3.2. Sigmoid function

A logistic sigmoid function is calculated to transform a given Mahalanobis distance measure between model and observation to fit a fixed response interval. This normalisation will be used during weight estimation in section 5.5.4 and then classification. A distance $d_k = \mathrm{DM}(\hat{\mathbf{f}}_k)$ of visible feature vector $\hat{\mathbf{f}}_k$ is normalised to a match measure m_k in the interval $[0,1]$:

$$m_k = \mathrm{s}(d_k) \tag{39}$$

The sigmoid function $\mathrm{s}(d)$ for the normalisation (logistics function) is defined as

$$\mathrm{s}(d) = \frac{1}{1+e^{a(b-d)}} \tag{40}$$

and uses two parameters a and b. The parameters can be estimated from the distance surface $\bar{\mathbf{D}}_M$ for every model. The proposed parameterisation places the

centre point of the distance surface d_C at the middle of the sigmoid function (Figure 50) with a match measure of $s(d_C) = 0.5$. This results in

$$a = \frac{2}{d_C - \bar{d}} \tag{41}$$

$$b = d_C, \tag{42}$$

where d_C is the centre point value of the distance surface $\bar{\mathbf{D}}_M$ and $\bar{d}$ in the mean value of the whole surface $\bar{d} = \text{mean}(\bar{\mathbf{D}}_M)$. See Figure 49 for an example surface of an interest point. A full proof for equations (41) and (42) is included in Appendix section A.1 on page 190. By using the mean of all distance data points in equation (41) and therefore considering all data, a uniform drop of match measure m_k is generated for different interest points when moved away from the training position $\mathbf{x}$. The impact of feature outliers is limited due to the nature of equation (40), which is bound to the interval $[0,1]$. Any subset of interest points will provide the same match measure for appearance models after this normalisation, which is essential during self and part occlusion. The normalised match measure response $\mathbf{M}_M$ at training positions is given as

$$\mathbf{M}_M = s(\bar{\mathbf{D}}_M), \tag{43}$$

where $s(d)$ is the sigmoid function from earlier. An example output of the match measure $\mathbf{M}_M$ can be seen in Figure 51 showing a distinct peak, which is of the same height for all the interest points.

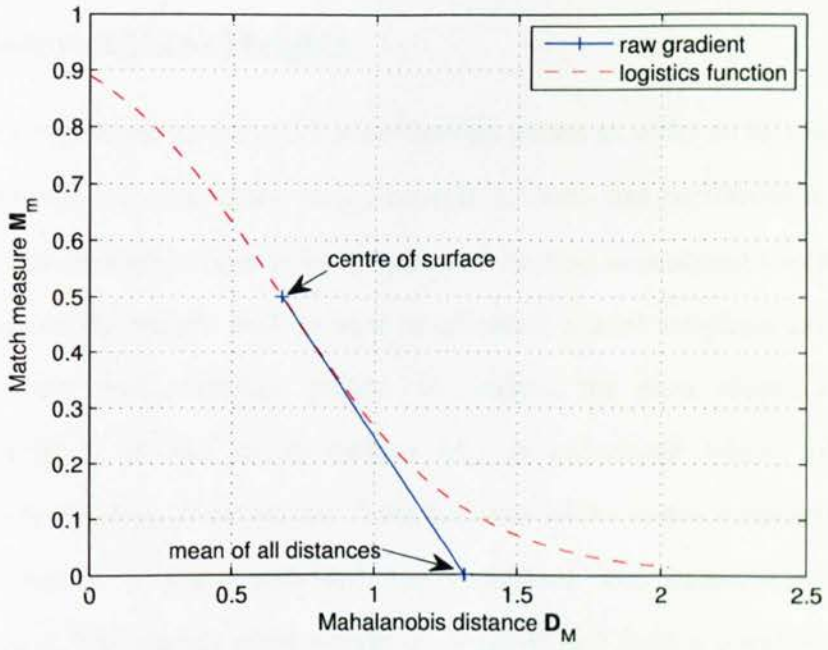


Figure 50 Estimated sigmoid function shown as a dashed line. The continuous line is the gradient of the sigmoid function defined by the centre value d_C of the distance surface $\overline{\mathbf{D}}_M$ and the mean distance $\overline{d}$ of all grid points.

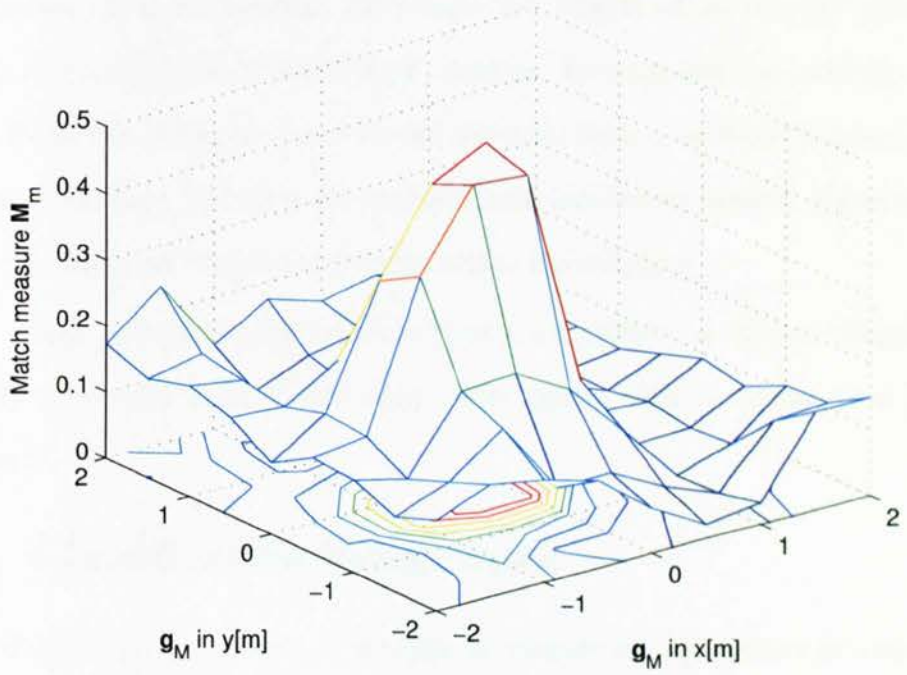


Figure 51 Final match measure surface $\mathbf{M}_M$ after application of the sigmoid function. A distinct peak at the training position can be observed. This peak is set to the same value for all interest points of all models.

5.5.4. Interest Point Weights

Relative weights are given to different interest points in order to favour those with good localisation performance and remove those with bad performance. The shape of the match measure response in the previous section is analysed for this task. For classification, the weight will be used to calculate a total weighted average match measure over visible interest points. To analyse the peak shape, a histogram $\mathbf{H}_h = \underset{M}{\text{hist}}(\mathbf{M}_M)$ of the match surface $\mathbf{M}_M$ is calculated where every bin h corresponds to a ring of the surface. Low variance of the match measures $\mathbf{H}_h$ inside such a ring is a good indicator for consistent and symmetric localisation performance. The interest point weight q_k is calculated from a weighted average of those variances using the element count C_h of histogram bins:

$$q_k = 1 - \sum_h \frac{\text{var}(\mathbf{H}_h)}{C_h}. \tag{44}$$

The above equation penalises (decreases) the weight of an interest point, if the match measure surface exhibits local maxima. To complete the training, the best 80% of interest points are used for the classifier with q as their weights. Refer to Figure 45 on page 134 for a car example with marked up interest points as cones, where the diameter of the cone corresponds to the weight q_k.

Once the training has been completed, a classifier can use the trained data to classify previously unseen road users. How this is done is discussed in the next section.

5.6. Classification Framework

The classification uses feature extraction to compare the appearance of a new object against the trained appearance models. The classification framework used here is based on the framework described in chapter 3. The difference lies in the way models are matched against observations for every road user hypothesis. Feature

vectors are extracted according to what was explained in section 5.4. Those feature vectors are then matched against previously trained appearance models as described in section 5.5. The matching will be described in section 5.6.1 after a short overview of the overall classification framework.

Background estimation with a Gaussian mixture model (KadewTraKuPong and Bowden, 2001) and shadow removal is used to generate motion silhouettes. For each silhouette, a grid of ground plane object hypotheses is generated from the centroid and scored by the classifier using equation (45) from the next section. Please refer to Figure 52 for a block diagram. The silhouettes are often noisy due to the challenging video data in urban environments with changing lighting conditions and low camera angle, but are a good indicator for the existence of a road user. (Example is shown in results of Figure 54).

The classifier sweeps through models and locations by scoring hypotheses based on matching appearance models against new features to find the highest match measure above the detection threshold τ_M. In the process, the 3DHOG framework is used to extract visible image patches and features for every hypothesis (*i.e.* ground plane location $\mathbf{x}$) as described in section 5.4. To handle variable visibility and occlusion, an average match measure per hypothesis is calculated according to equation (45) producing a match surface for hypotheses shown in Figure 53. To limit the search space, orientations of road users are assumed to align with the road direction, which is realistic for many road videos. The classification is performed on a per frame basis without tracking or temporal refinement.

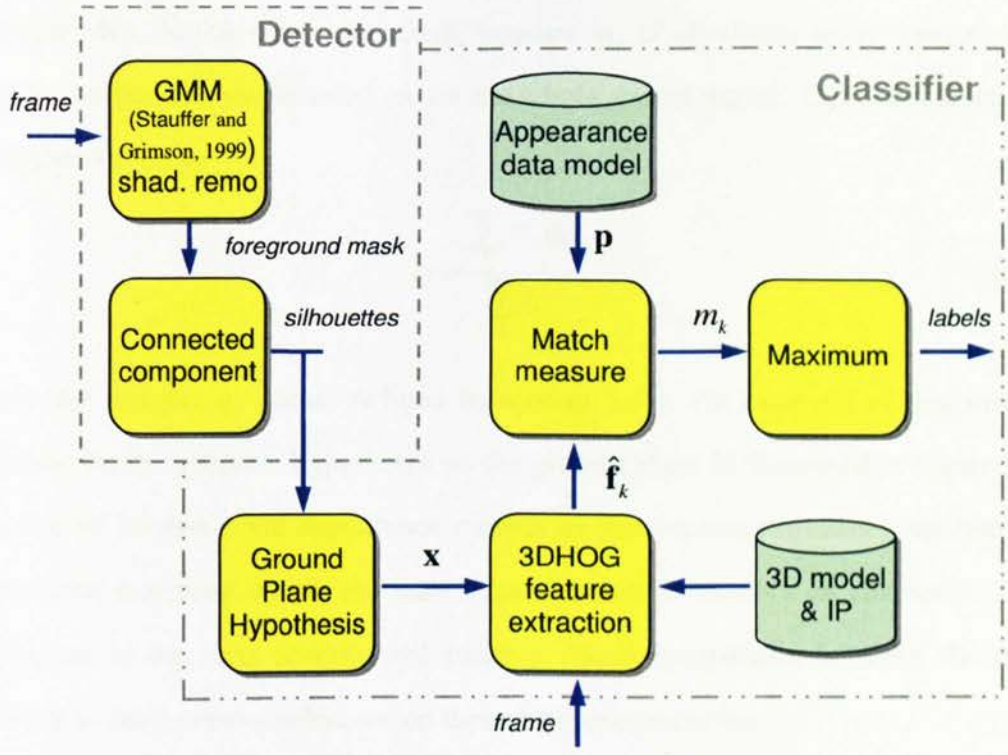


Figure 52 Block diagram for the 3DHOG classifier. The general structure is identical to the motion silhouette classifier in chapter 3. 3DHOG features are extracted directly from the input frame based on the ground plane hypothesis. The match measure operates in appearance feature space in contrast to the image space for the silhouette classifier (please compare to Figure 6 on page 55).

5.6.1. Match Measure between Model and Image

The match measure is calculated from the comparison between new feature vectors and the trained appearance models by summation of the match measure responses of individual interest points. First, feature vectors $\hat{\mathbf{f}}_k$ are generated for visible interest points $\mathbf{p} \in \mathcal{P}''$, where the spatial model location for the extraction is the ground plane hypothesis location. The index k enumerates the visible interest points of the given hypothesis. Every feature vector $\hat{\mathbf{f}}_k$ is matched against its appearance model by calculating the distance $d_k = \mathrm{DM}\left(\hat{\mathbf{f}}_k\right)$ according to equation (36). The distance is then normalised to a match measure $m_k = \mathrm{s}\left(d_k\right)$ according to

equation (40). At this point, the match measure m_k of all visible interest points has to be combined to a single value m for the whole spatial model. This is achieved by a weighted average

$$m = \frac{\sum_k m_k q_k}{\sum_k q_k}, \tag{45}$$

where the weights q_k are as defined in section 5.5.4. An example of this match response for the different hypotheses on the ground plane is illustrated in Figure 53. The use of interest point appearance models in this section provides a method of appearance matching within the same classification framework as chapter 3. The evaluation in the next section will make a direct comparison between the two methods to later draw conclusions on their different properties.

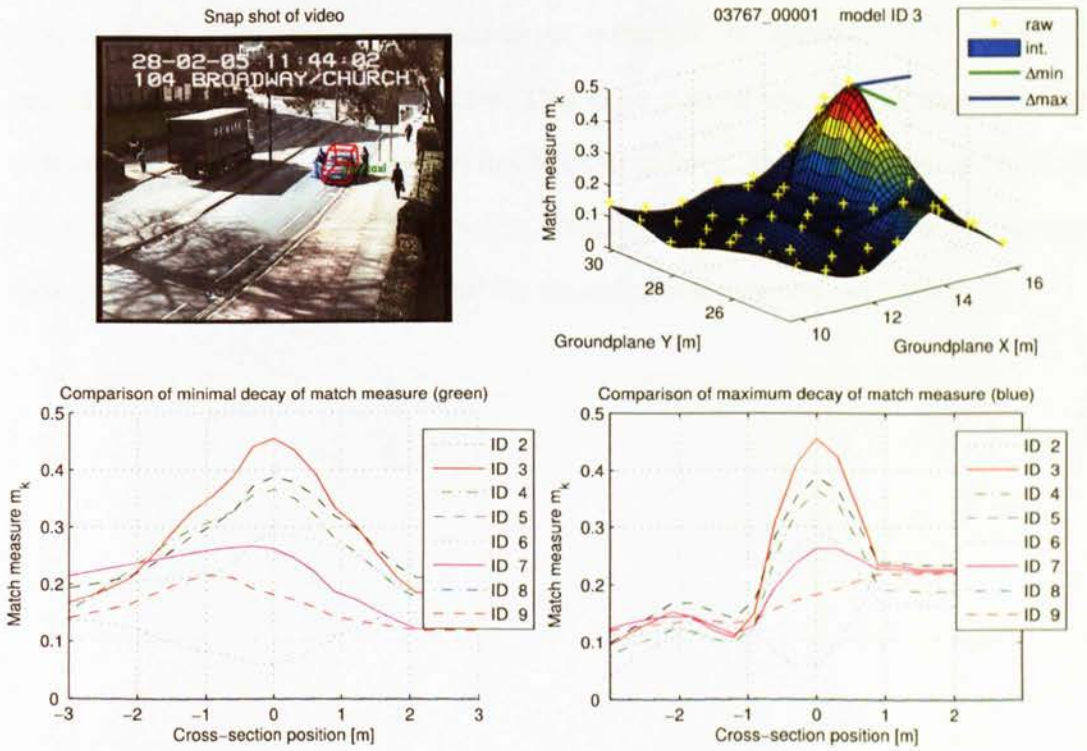


Figure 53 Example of car detection with occlusion of pedestrians showing a match measure surface with a good peak.

symbol	*value*	*Unit*
γ	0.35	[0,1]
d_0	4	#InterestPoints/m
τ_v	0.65	[0,1]
ρ	32	Pixel/m
δ	1	M
η	10	
v	4	
τ_M	0.38	[0,1]

Table 12 Parameters used during evaluation of the 3DHOG classifier

5.7. Evaluation

Evaluation was performed on realistic (operational quality) videos for traffic surveillance. All three algorithms are compared to the motion silhouette baseline in section 5.7.1. Full performance including pedestrian detection is discussed in section 5.7.2 with analysis of parameter influence in section 5.7.3. Table 12 provides a parameter list for the tests. The same part of the i-LIDS data sets that was used as described in section 3.4 has been used here. Approximately one hour of video for sunny, overcast and changing conditions was selected. Some illustrative examples are shown in Figure 54 and for classification problems in Figure 55.

Figure 54 True positive examples for vehicles and pedestrians using 3DHOG.

Figure 55 Two examples of errors generated with 3DHOG. Left: Missed car due to low contrast of the vehicle bonnet and roof. Right: Misclassified SUV as van due to similar size and appearance.

5.7.1. Feature Comparison for Vehicle Detection and Classification

Out of the three features discussed in section 5.4.2 (HOG, FFT, Histogram), the best performing algorithm is 3DHOG with a total recall of 77.3% at precision of 73.8% (Table 13) and classification accuracy of 87.9%. This compares to recall of 87% for a precision of 85.5% for the motion silhouette baseline from chapter 3 run on the same data set, but 3DHOG should be better at dealing with silhouette noise and particularly occlusion. A qualitative example for miss-shaped silhouette is shown later in Figure 57 on page 157 and the occlusion analysis is performed as part of the applications in section 6.2. The bounding box overlap of 3DHOG of 0.69 outperforms the algorithm from chapter 3 with 0.67, which indicates better localisation performance for 3DHOG. The system using FFT features showed lower performance (Recall 48.4% at precision 42.3% from Table 15) similar to the histogram features (Recall 48.9% at precision 42% from Table 16). The localisation performance for those two features is identical with 0.64. From those numbers, it is clear (as the rest of the framework is the same) that the 3DHOG provides a more discriminative and descriptive feature than the FFT and the Histograms. The

ground truth / detection	bike	car/taxi	van	bus/lorry	FP
bike	.95	.01	.00	.05	.88
car/taxi	.00	.83	.35	.04	.12
van	.00	.00	.58	.47	.12
bus/lorry	.00	.00	.00	.44	.02
FN	.05	.16	.07	.00	.00
count	43	374	60	57	
overlap	.68	.67	.75	.81	

Symbol	*Value*
Recall R	77.3%
Precision P	73.8%
Classifier P_C	87.9%
Detector R_D	88.0%
Detector P_D	83.9%
GT Overlap	0.69

Table 13 Extended confusion matrix for 3DHOG with total system performance

ground truth / detection	bike	car/taxi	van	bus/lorry
bike	1.00	.01	.00	.05
car/taxi	.00	.99	.38	.04
van	.00	.00	.63	.47
bus/lorry	.00	.00	.00	.44
count	41	316	56	57

	bike	car/taxi	van	bus/lorry
R_j	95.3%	83.4%	58.3%	43.9%
P_j	47.7%	82.3%	50.7%	96.2%
R_{Cj}	100.0%	98.7%	62.5%	43.9%
P_{Cj}	85.4%	93.1%	56.5%	100.0%
R_{Dj}	95.3%	84.5%	93.3%	100.0%
P_{Dj}	55.8%	88.4%	89.9%	96.2%

Table 14 Classifier confusion matrix for 3DHOG and class wise results

classification performance of 87.9% for 3DHOG compared to 56% indicates that gradient features are most descriptive for vehicle detection compared to FFT and histogram features. The classifier performance for 3DHOG in Table 14 shows a tendency for detecting smaller vehicles, which is highlighted by the low classifier recall rate of 43.9% for the bus/lorry class. This can be attributed to two effects. Firstly, the predominately frontal or rear view of vehicles, which allows a good fit of models from the next class of smaller vehicles. Secondly, there are only limited numbers of training samples for the larger vehicles in the data set. In contrast, the classifier recall for cars is 98.7% for the dominant class in the data set.

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.95	.27	.46	.79	2.90
car/taxi	.00	.54	.21	.05	.07
van	.00	.01	.25	.05	.11
bus/lorry	.00	.00	.00	.11	.02
FN	.05	.19	.08	.00	.00
count	39	355	61	62	
overlap	.60	.63	.77	.86	

Symbol	*Value*
Recall R	48.4%
Precision P	42.3%
Classifier P_C	56.3%
Detector R_D	85.9%
Detector P_D	75.1%
GT Overlap	0.64

Table 15 Extended confusion matrix for FFT features with total system performance

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.54	.26	.35	.64	2.95
car/taxi	.03	.60	.33	.34	.13
van	.00	.00	.24	.02	.03
bus/lorry	.00	.00	.00	.00	.00
FN	.44	.14	.08	.00	.00
count	39	401	63	61	
overlap	.52	.64	.76	1.00	

Symbol	*Value*
Recall R	48.9%
Precision P	42.0%
Classifier P_C	56.8%
Detector R_D	86.2%
Detector P_D	74.0%
GT Overlap	0.64

Table 16 Extended confusion matrix for histogram features with total system performance

The confusion matrixes for FFT and Histogram features in Table 15 and Table 16 show a confusion of most classes with the class bike. This is due to the fact that a smaller model can be mistakenly fit more easily to an arbitrary image region resembling features of the model than a larger one. 3DHOG is sufficiently discriminative also for the smaller models as not to show such a dominant confusion.

ground truth / *detection*	pedestrian	bike	car/taxi	van	bus/lorry	FP
pedestrian	.51	.12	.04	.07	.52	.14
bike	.05	.85	.03	.09	.07	.10
car/taxi	.00	.00	.71	.13	.00	.07
van	.00	.00	.01	.63	.15	.07
bus/lorry	.00	.00	.00	.00	.25	.01
FN	.43	.02	.22	.09	.00	.00
count	221	41	356	56	67	
overlap	.56	.64	.68	.75	.84	

Symbol	*Value*
Recall R	60.7%
Precision P	72.0%
Classifier P_C	80.4%
Detector R_D	75.6%
Detector P_D	89.6%
GT Overlap	0.66

Table 17 Extended confusion matrix for 3DHOG with total system performance including pedestrians

ground truth / *detection*	pedestrian	bike	car/taxi	van	bus/lorry
pedestrian	.90	.13	.05	.08	.52
bike	.09	.88	.03	.10	.07
car/taxi	.01	.00	.91	.14	.00
van	.01	.00	.01	.69	.15
bus/lorry	.00	.00	.00	.00	.25
count	125	40	277	51	67

	pedestrian	bike	car/taxi	van	bus/lorry
R_j	50.7%	85.4%	70.5%	62.5%	25.4%
P_j	55.7%	50.7%	88.1%	67.3%	94.4%
R_{Cj}	89.6%	87.5%	90.6%	68.6%	25.4%
P_{Cj}	65.5%	53.8%	96.9%	72.9%	100.0%
R_{Dj}	56.6%	97.6%	77.8%	91.1%	100.0%
P_{Dj}	85.1%	94.2%	90.9%	92.3%	94.4%

Table 18 Classifier confusion matrix for 3DHOG and class wise results including pedestrians

5.7.2. Simultaneous Operation for All Road Users

Full quantitative performance figures for road user detection and classification (including pedestrians) are shown in Table 17 and Table 18. The overall recall degraded to 60.7% due to the non rigid nature of pedestrians, which increases the complexity for the detection task. The same effect can be observed for the motion silhouette baseline from chapter 3. In contrast, precision is not affected by the

Figure 56 Left: Correctly classified lorry with 3DHOG despite shadows and oversized motion silhouette. Right: wrongly detected pedestrian at the front edge of a lorry due to vertical edges.

addition of the pedestrian model. The localisation performance expressed with the overlap measure of 0.66 outperforms the motion silhouette baseline with 0.64. A qualitative comparison in Figure 57 highlights the advantage of 3DHOG in this case. The motion silhouette classifier exhibited confusion between pedestrians and bikes in Table 4 due to similar size. This issue does not arise with 3DHOG; however, wing- mirrors and front corners of lorries are misclassified as pedestrians due to a similar appearance (see Figure 56). This leads to a low systems recall of 25.4% for lorries. Combining information from both classifiers could resolve some of the misclassifications due to the different failure modes. The classifier precision for cars, the predominant class in the data set, is 96.9%, which is slightly higher than the motion silhouette baseline with 96.6%.

5.7.3. Influence of Patch Size

This section analyses the sensitivity of the 3DHOG algorithm to the patch size. The size is reduced to $\delta = 0.8\,\mathrm{m}$ at resolution $\rho = 20\,\mathrm{Pixel/m}$ and $\delta = 0.5\,\mathrm{m}$ at $\rho = 16\,\mathrm{Pixel/m}$. Figure 58 shows the change of performance for this parameter variation. The full list of confusion matrixes and performance figures is included in

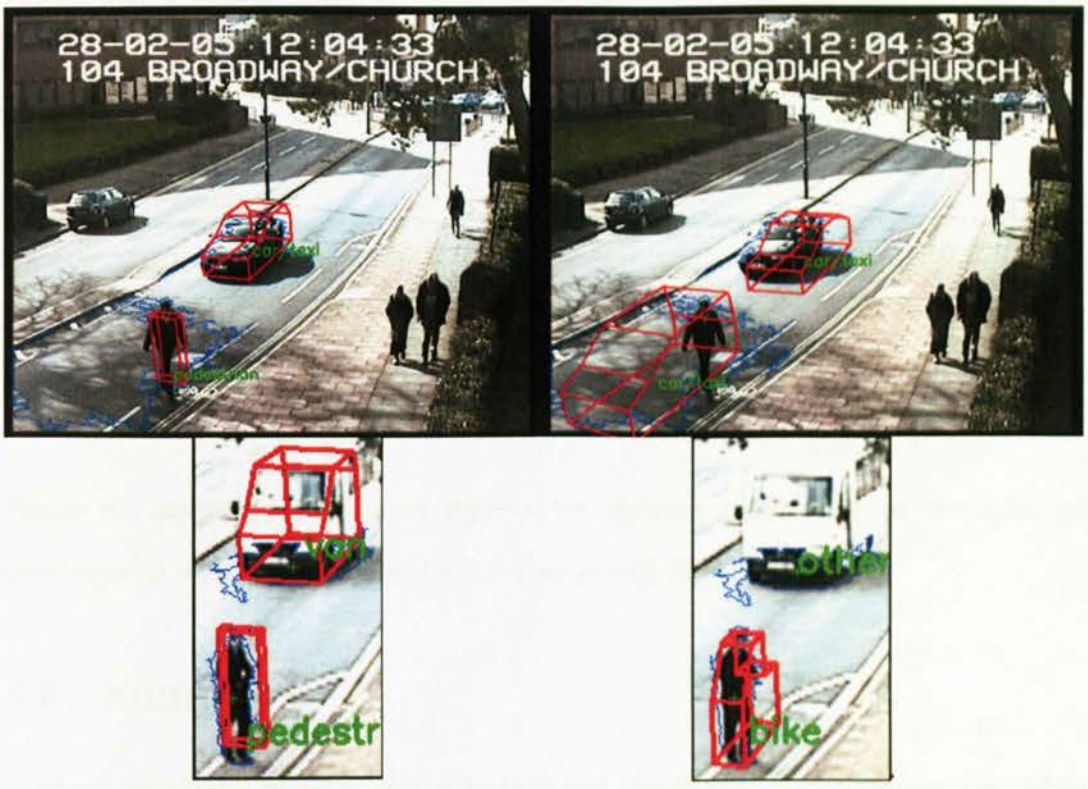


Figure 57 Comparison of the 3DHOG classifier (left) with the motion silhouette classifier (right). The first image shows a position offset and wrong classification of the pedestrian of the silhouette classifier due to the tree shadow. In comparison, the 3DHOG classifier correctly identifies the pedestrian within the silhouette and aligns the car better. The bottom images show a missed vehicle of the silhouette classifier, because the silhouette is too small due to the overexposed camera view. The pedestrian is detected as bicycle due to similar size. Both problems are resolved with the 3DHOG classifier.

appendix C.2. The recall increases with larger patch size from 66.8% to 77.3% as the models become more discriminative. At the same time, the classification performance slightly degrades from 89.1% to 87.9% due to the larger number of detections, which represent harder cases, which were rejected for the small patch size previously. The overlap and therefore the location performance improves slightly from 0.68 to 0.69. Increasing the patch size further could mean that patches represent global rather than local appearance.

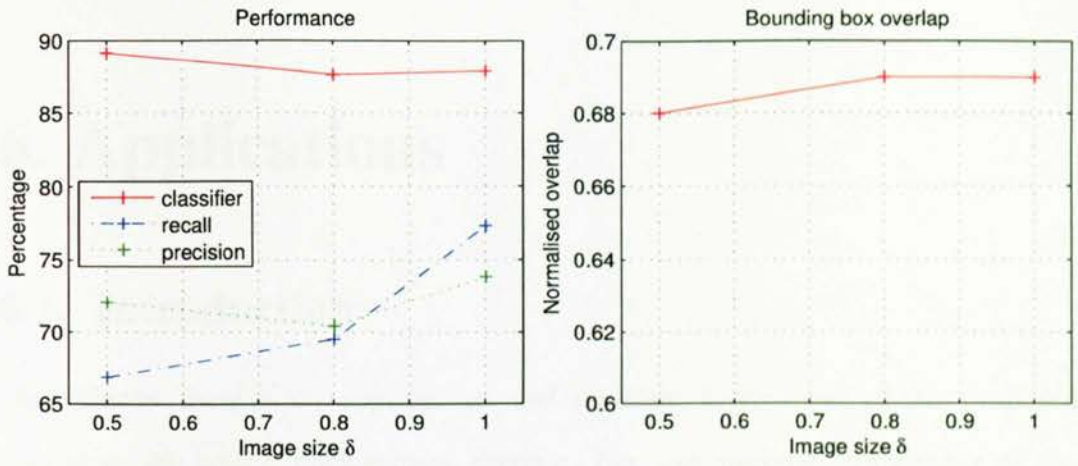


Figure 58 3DHOG performance figures for varying patch size to $\delta = 0.8\,\mathrm{m}$ at resolution $\rho = 20\,\mathrm{Pixel/m}$ and $\delta = 0.5\,\mathrm{m}$ at $\rho = 16\,\mathrm{Pixel/m}$

5.8. Summary

A novel algorithm, 3DHOG, for detection and classification of road users in urban scenes was presented in this chapter. This is an extension to the HOG feature extraction by applying 3D spatial modelling to operate on still images and thus overcoming the reliability limitations of motion silhouettes. This single solution handles variable viewpoints for rigid vehicles as well as pedestrians. A training framework has been proposed generating weights for learned interest points for classification. Three algorithms for features based on HOG, FFT and simple histograms have been evaluated. The 3DHOG shows comparable performance to a baseline approach using motion silhouettes. The classifier sweeps the hypotheses space to find the best match between images (observation) and 3D models based on the average match measure between interest points and the training data.

The next chapter will demonstrate applications of this classifier and show operation for occluded vehicles. The portability of the algorithm is evaluated by using the training data generated from the i-LIDS data set for classification of videos recorded with a high definition camera.

6. Applications

6.1. Introduction

This chapter focuses on applications and possible deployment of the proposed system in the traffic management domain. For this purpose, robustness of the proposed classifiers has been evaluated and extensions for tracking lead to a demonstrator for behaviour analysis. Firstly, section 6.2 considers portability between different cameras and operation under occlusions. To be of practical use, training data has to be usable between cameras to avoid lengthy re- training for every camera of a large network. Classification performance is evaluated on new data and directly compared to an industrial classifier. Following on, section 6.3 introduces a tracker which is integrated with the framework. It also provides comparative evaluation with a state of the art blob tracker. The tracking increases the temporal context of the vehicle classifications to enable better conceptual analysis *e.g.* illegal turns, bus lane intrusion or vehicle interactions. A discussion and demonstrator for this behaviour analysis is given in section 6.4 with a summary in section 6.5.

6.2. Occlusion and Portability

This section demonstrates the portability of the proposed detector and classifiers between different camera views and with different resolutions. Due to the nature of both vehicle classifiers proposed, the only scene dependent information is the camera calibration. Re- training of an appearance classifier for every camera view does not scale well and is not feasible for large networks, like the network operated by TfL for traffic operations. 3DHOG generates image patches with a normalised

scale using the camera calibration. Those patches can then be used across cameras, which allows a trained classifier to be applied to novel camera views. A digital camera with progressive scanning and a high resolution of 1360x1024 is used for recording of the test video. This represents effectively four times as many lines compared to the interlaced video used for training in chapter 5. Vehicles are also closer to the camera in this scene which results in an order of magnitude size difference between vehicles from training and classification. In addition to this new size, the test data also contains occluded and partially visible vehicles to demonstrate the effectiveness under those conditions. The 3DHOG framework operates with visible interest points, whereas the number of those points can vary to allow for occlusions. The system performance is compared to an industrial state of the art vehicle classifier. This classifier however, is limited to a single view and single size of vehicles. All systems are tested on the same video data recorded from a two lane road with traffic lights. Each of the following videos comprises of about 500 frames:

- AVIFile_2009_05_16_08_59_32.avi
- AVIFile_2009_05_16_09_01_29.avi
- AVIFile_2009_05_16_09_02_40.avi
- AVIFile_2009_05_16_09_03_57.avi
- AVIFile_2009_05_16_09_05_49.avi

Some frames contain artefacts from the wireless video transmission during recording. This can be noticed by a mixture of old and new images for one frame (*e.g.* Figure 59) and can be dealt with by the 3DHOG classifier. The industrial classifier does not detect vehicles in such damaged frames. Evaluation is provided for an industrial classifier as baseline in section 6.2.1, for the 3DHOG classifier in section 6.2.2 and for the motion silhouette classifier section 6.2.3.

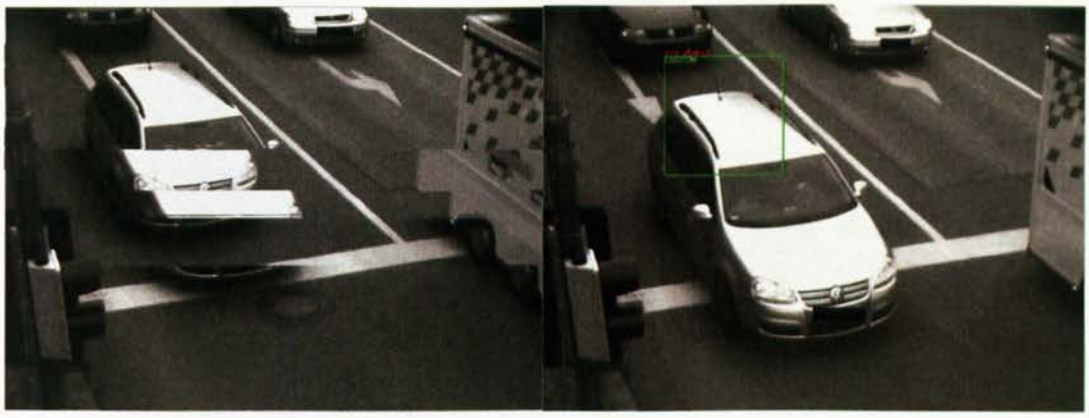


Figure 59 Example images from the new data set containing a transmission artefact on the left. The right image shows the classified car, which is typically fully visible. Partly occluded cars (at the camera's edge) are mostly ignored.

6.2.1. Industrial Classifier Results

The industrial classifier used as baseline combines a support vector machine (SVM) with a windowing approach of the input image. A sliding window is moved across an input frame to detect and classify vehicles. This global appearance based approach is a good baseline for the local appearances used in 3DHOG. The manually verified results from this classifier are used as ground truth to compare with the two proposed classifiers (3DHOG, motion silhouette). The ground truth data comprises of the class label and the top left corner of the bounding box of vehicles. The width and height of the bounding boxes is fixed with 300 pixels, because no size information is available from the classifier. Refer to Figure 59 for example views. Partly occluded cars are ignored by the baseline, which requires a small region of interest for the classifiers described in this thesis (*i.e.* motion silhouette and 3DHOG). By considering the full frame, many of those partly occluded vehicles would be detected by the author's classifiers and reported as false positives (see Figure 61). In addition, some frames with transmission artefacts are not detected by the industrial baseline and are also reported as false positives of the author's methods. Full quantitative results are provided in Table 19 and Table 20.

ground truth / *detection*	bike	car/taxi	van	bus/lorry
bike	.00	.00	.00	.00
car/taxi	.00	.92	.02	.00
van	.00	.07	.11	.00
bus/lorry	.00	.01	.87	.00
count	0	1744	290	0

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.00	.00	.00	.00	.00
car/taxi	.00	.92	.02	.00	.00
van	.00	.07	.11	.00	.00
bus/lorry	.00	.01	.87	.00	.00
FN	.00	.00	.00	.00	.00
count	0	1744	290	0	

Table 19 Industrial classifier confusion matrix and full system (detection and classification) confusion matrix. Both matrixes are identical, as the system output for detection is used as ground truth. The classifier shows a strong misclassification of vans as lorries.

Symbol	*Value*
Recall R	80.3%
Precision P	80.3%
Classifier P_C	80.3%
Detector R_D	100.0%
Detector P_D	100.0%
GT Overlap	NaN

	bike	car/taxi	van	bus/lorry
R_j	0.0%	91.9%	11.0%	0.0%
P_j	0.0%	99.7%	21.5%	0.0%
R_{Cj}	0.0%	91.9%	11.0%	0.0%
P_{Cj}	0.0%	99.7%	21.5%	0.0%
R_{Dj}	0.0%	100.0%	100.0%	0.0%
P_{Dj}	0.0%	100.0%	100.0%	100.0%

Table 20 Industrial classifier total system performance and class wise evaluation

The detection performance for the industrial classifier is 100%, because its results were used as ground truth. The classification performance, however, show a strong confusion between vans and lorries.

6.2.2. 3DHOG Results

The results in this section are generated with the 3DHOG classifier from chapter 5. The training data is also taken from that chapter and only the camera was newly calibrated. There is a significant change in vehicle size and viewing angle between the data sets. The part of the i-LIDS data set used for training does not contain a view of the front and right corner of vehicles being visible as it is the case in this new data set. Example classification results are provided in Figure 60 and Figure 61. The map for camera calibration was manually generated and is displayed with the result figures. The good match between wire frames and vehicles indicates that the calibration is sufficient for this application. Full performance results are given in Table 21 and Table 22, using the result of the previous section as ground truth. The 3DHOG classifier (91.1%) outperforms the industrial baseline (80.3%) in total classification precision. This measure only considers detected vehicles, whereas some stationary vehicles were not considered (*i.e.* missed) by 3DHOG, leading to a low detection recall of 45%. This is due to the fact that 3DHOG takes vehicle hypotheses from motion estimation, which does no detect objects which are stationary for a longer time period. In contrast, some vehicles with transmission artefacts are correctly classified but deemed false positives, because of their absence in the ground truth. Qualitative evaluation of this is given in Figure 59 and Figure 62.

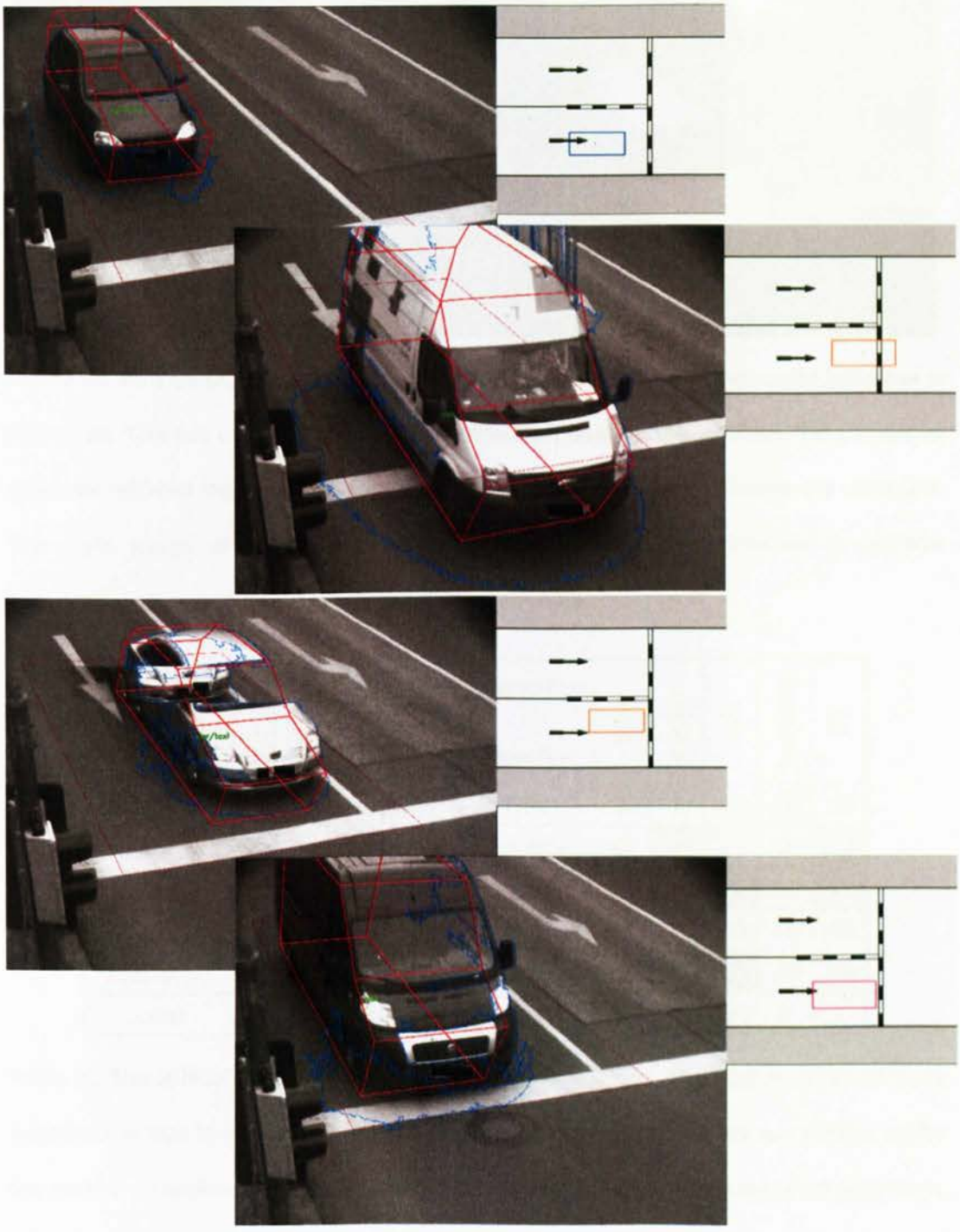


Figure 60 3DHOG classification results of 4 separate frames. The blue outline shows the estimated motion foreground. The 3DHOG classifier produces the wire frame and the respective 3D location of the vehicle on the road map. Good localisation performance is demonstrated even for the third example containing a transmission artefact.

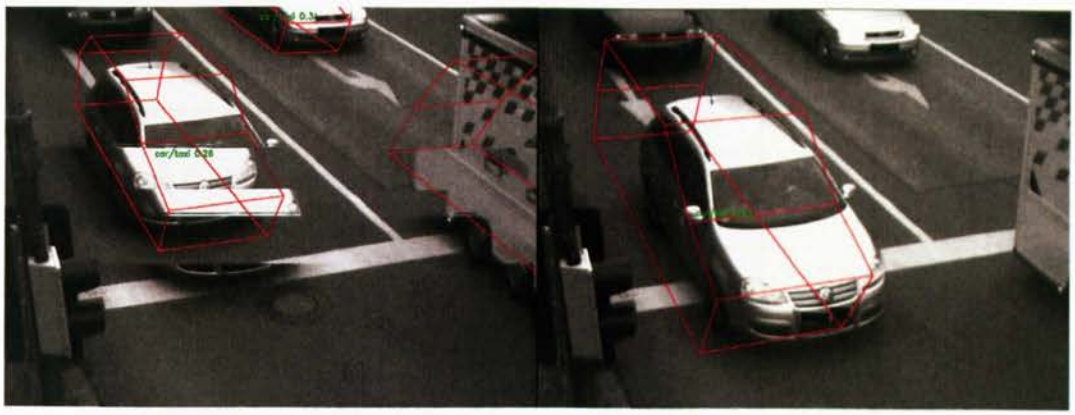


Figure 61 Two examples comparing the 3DHOG results with the industrial baseline in Figure 59. The left image shows correct detection despite the artefact. This example operates without region of interest, which is why the occluded vehicles are detected. The right image shows a later frame with active region of interest to remove occluded vehicles to avoid excessive false positives.

ground truth / *detection*	bike	car/taxi	van	bus/lorry
bike	.00	.00	.00	.00
car/taxi	.00	.94	.25	.00
van	.00	.06	.75	.00
bus/lorry	.00	.00	.00	.00
count	0	760	146	0

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.00	.00	.00	.00	.00
car/taxi	.00	.41	.13	.00	.23
van	.00	.03	.38	.00	.16
bus/lorry	.00	.00	.00	.00	.00
FN	.00	.56	.50	.00	.00
count	0	1739	290	0	

Table 21 The 3DHOG classifier exhibits good performance. The high number of false negatives is due to stationary objects at the traffic lights, which are not picked up by the motion detection. Some detections reported as false positives were actually cars, but were not picked up by the industrial classifier used as baseline. Examples of both of those issues are illustrated in Figure 59.

Symbol	*Value*
Recall R	40.7%
Precision P	60.7%
Classifier P_C	91.1%
Detector R_D	44.7%
Detector P_D	66.7%
GT Overlap	0.24

	bike	car/taxi	van	bus/lorry
R_j	0.0%	41.2%	37.6%	0.0%
P_j	0.0%	61.7%	55.1%	0.0%
R_{Cj}	0.0%	94.2%	74.7%	0.0%
P_{Cj}	0.0%	95.1%	71.2%	0.0%
R_{Dj}	0.0%	43.7%	50.3%	0.0%
P_{Dj}	0.0%	64.9%	77.3%	0.0%

Table 22 Total system performance for the 3DHOG classifier. The classification outperforms the industrial baseline.

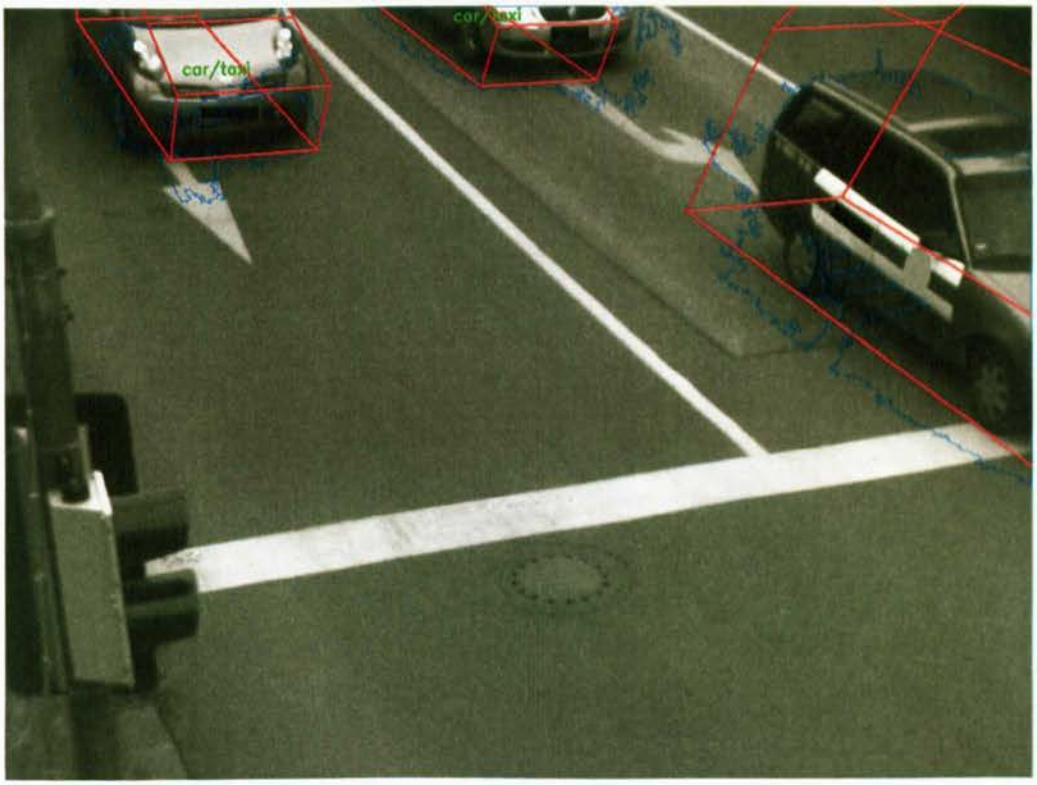


Figure 62 Example for partially occluded vehicles. This image illustrates that very limited visibility of vehicles is sufficient for detection and potentially classification. The algorithm for dealing with incomplete representations of objects is a core part of the 3DHOG classifier framework. Occlusion is resolved seamlessly in the same way as variable visibility of the 3D models depending on the camera view and vehicle orientation.

6.2.3. Motion Silhouette Results

This section provides results for the motion silhouette classifier to compare to the two appearance based classifiers earlier. The parameters for the algorithms were unchanged from the previous setup in chapter 3 and the same camera calibration as for 3DHOG was used. A reduced region of interest was defined in the view, to approximately reflect the detection region of the ground truth reference. Quantitative results are provided in Table 23 and Table 24. The classifier precision with 77.3% is slightly lower than the industrial classifier (80.3%) and significantly lower than 3DHOG (91.1%). The appearance based systems seem to be able to exploit the higher resolution image compared to i-LIDS better than the motion based system. In addition, the location performance with overlap 0.22 is slightly lower than 3DHOG (0.24), which can be seen by slightly offset vehicle wire frames in the results (Figure 63 and Figure 64).

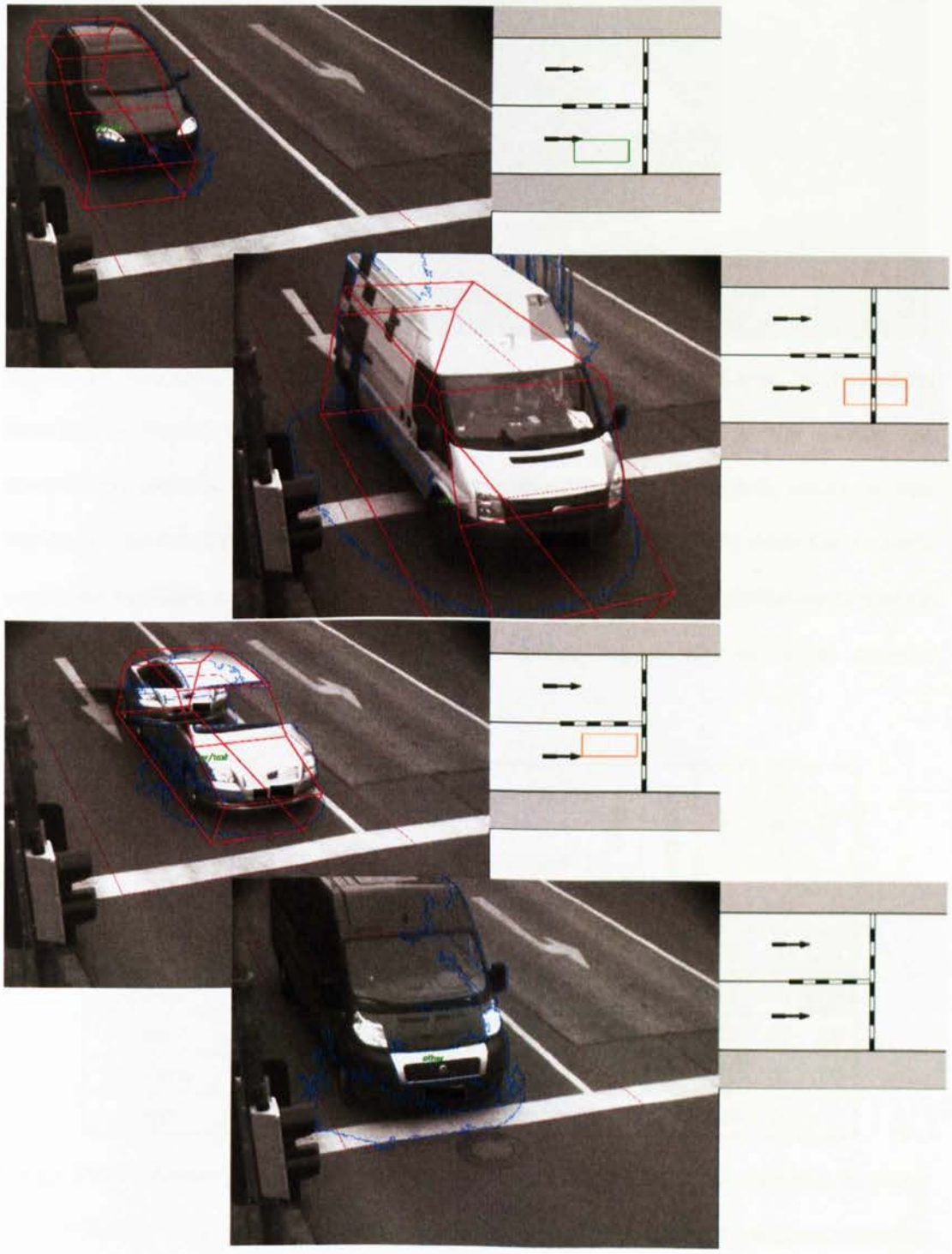


Figure 63 The motion silhouette classifier produces the wire frame and the respective 3D location of the vehicle on the road map. The wire frames are slightly offset and the last stopped van was missed due to ambiguous silhouette shape merging into the background, but correctly classified by 3DHOG (Figure 60).

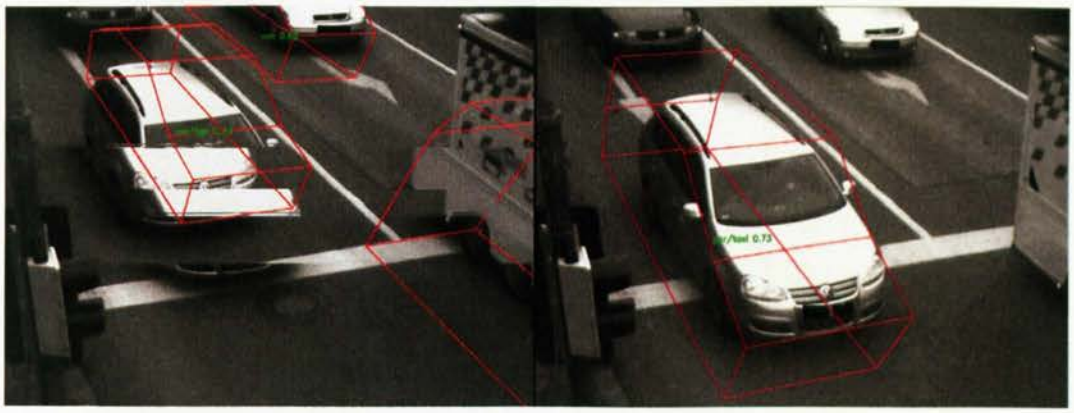


Figure 64 Two examples comparing the motion silhouette results with the industrial baseline in Figure 59. The left image shows correct detection of the central car despite the artefact. This example operates without region of interest, which is why the occluded vehicles are detected. The silhouette based classifier does not classify occluded vehicles as well as 3DHOG, especially the location performance is worse. The right image shows a later frame with active region of interest to remove occluded vehicles to avoid false positives.

ground truth / *detection*	bike	car/taxi	van	bus/lorry
bike	.00	.00	.00	.00
car/taxi	.00	.76	.11	.00
van	.00	.24	.89	.00
bus/lorry	.00	.00	.00	.00
count	0	592	55	0

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.00	.00	.00	.00	.00
car/taxi	.00	.26	.02	.00	.11
van	.00	.08	.17	.00	.47
bus/lorry	.00	.00	.00	.00	.00
FN	.00	.66	.81	.00	.00
count	0	1739	290	0	

Table 23 Confusion matrix for the silhouette classifier. The detection rate is lower than 3DHOG. This is due to ambiguous motion silhouettes for stopped vehicles merging with the background. Those silhouettes do not match a model well, but are sufficient for the 3DHOG classifier to start a search.

Symbol	Value
Recall R	24.6%
Precision P	51.4%
Classifier P_C	77.3%
Detector R_D	31.9%
Detector P_D	66.6%
GT Overlap	0.22

	bike	car/taxi	van	bus/lorry
R_j	0.0%	25.9%	16.9%	0.0%
P_j	0.0%	69.7%	15.1%	0.0%
R_{Cj}	0.0%	76.2%	89.1%	0.0%
P_{Cj}	0.0%	98.7%	25.8%	0.0%
R_{Dj}	0.0%	34.0%	19.0%	0.0%
P_{Dj}	0.0%	70.6%	58.5%	0.0%

Table 24 Classification and full system performance for silhouette classifier.

6.2.4. Comparison

The best detection is provided by the industrial classifier, because a vehicle search is performed on every single frame independently of motion. For classification precision of detected vehicles however, 3DHOG (91.1%) outperforms the industrial baseline (80.3%) and motion silhouettes (77.3%). For the 3DHOG classifier, good operation on occluded vehicles and damaged frames is demonstrated, which are not considered in the baseline. There is strong confusion between vans and lorries for this baseline resulting in a classifier recall of 11% for this class. Considering the better class of cars, the classifier recall of 91.9% is still inferior to the 3DHOG classifier precision of 94.2%

6.3. Tracking

Following on from portability between cameras without needed retraining, this section will show the integration of a variable sample rate Kalman filter with the classification framework for tracking. Consistent location information over time can then be used for behaviour analysis, which the next section will look into. Tracking performance has been evaluated using the framework of (Yin *et al.*, 2007) and compared to a state of the art OpenCV blob tracker (OpenCV, nd) operating on the

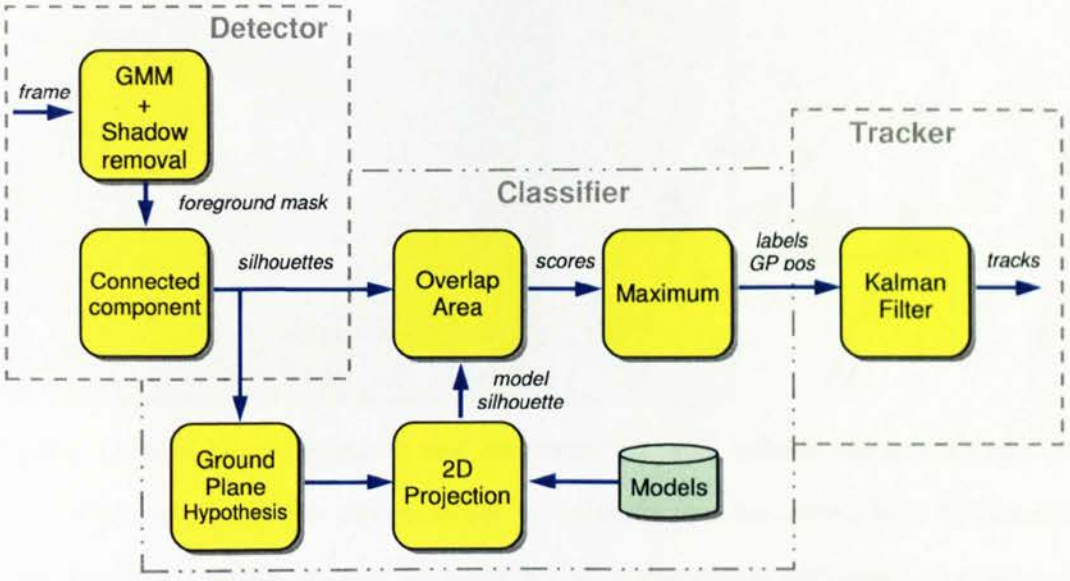


Figure 65 Block diagram of detector with 3D classifier and subsequent tracker.

same video data. Previously, vehicle tracking in urban environments has been performed in (Song and Nevatia, 2007). However, only a single 3D model for cars is used to estimate a vehicle constellation per frame with optimisation solved with a Markov Chain Monte Carlo (MCMC) algorithm. The paper of (Morris and Trivedi, 2006a) presents a combined tracking and classification approach for side views of highways which is an extension to (Morris and Trivedi, 2006b). Classification and tracking accuracy was increased by combining tracking and classification. A Kalman filter is used to track the foreground regions based on the centroids in the image plane only. The OpenCV blob tracker (OpenCV, nd) used as baseline here works in a similar fashion.

The framework of detector and classifier as introduced in chapters 3 and 5 is extended by a tracker illustrated in Figure 65. The ground plane positions and labels of classified vehicles are the input to a Kalman filter to provide temporal consistency and smoothing to the movements. An example result on footage from Transport for London is shown in Figure 66. The next sections will provide details on the Kalman filter and the evaluation framework before presenting the results.

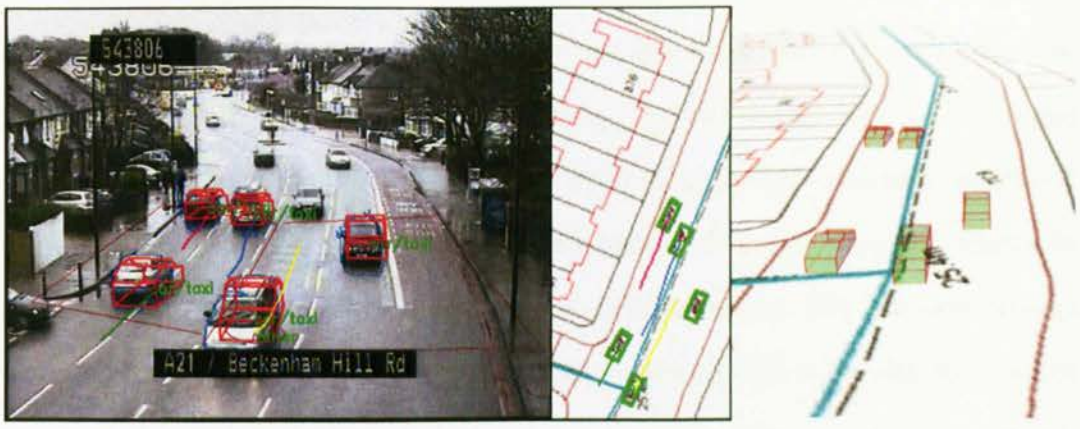


Figure 66 Example of detection and classification with ground plane tracking. The wire frame projection in red is used to estimate the bounding box for tracked vehicles. The information can be used for an anonymised animation to overcome privacy limitations of video data.

6.3.1. Kalman Filter

Tracking introduces temporal consistency to the detection and classification result of the previous section. The classifier is extended by a Kalman filter with variable sample rate. The detector with joint classifier operating on single frames may reject valid vehicles in some frames due to noise, which requires the Kalman filter to operate on variable (longer) time intervals when no observation is available. Alternatively, the update step of the Kalman filter could be modified when no observation is available by setting the measurement noise to infinity (or a very large value), which in practice has a similar effect. The author chose to use variable time intervals to avoid numerical problems when inverting large noise matrices.

Tracking is performed on the ground plane of the scene, which simplifies behaviour analysis like bus lane monitoring. The standard formulation of the Kalman filter for a constant velocity model of vehicles is used

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k + \mathbf{w}_k \qquad \mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \qquad \text{with } \mathbf{u}_k = \mathbf{0} \tag{46}$$

with state vector $\mathbf{x}_k = (v_x, x, v_y, y)^T$ and the measurement vector $\mathbf{z}_k = (x, y)^T$. The transition matrix $\mathbf{F}$ propagates the old state $\mathbf{x}_{k-1}$ to the current state $\mathbf{x}_k$. The input vector $\mathbf{u}_k = 0$, which means that there is no input to the system. The input matrix $\mathbf{B}$ specifies how the input influences the state update. The measurement matrix $\mathbf{H}$ transforms the state $\mathbf{x}_k$ to the output $\mathbf{z}_k$ (*i.e.* measurement). The constant process noise $\mathbf{w}_k$ and the measurement noise $\mathbf{v}_k$ are assumed Gaussian and have to be specified. All time and speed related constants for the filter are based on seconds rather than the sample rate or frame rate. The ground plane coordinates are in metres, all noise and position estimates are in metres or metres per second. The integration constant T_0 from speed to position in the transition matrix $\mathbf{F}$ is defined in seconds.

$$\mathbf{F} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ T_0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & T_0 & 1 \end{pmatrix}. \tag{47}$$

The only requirement to operate the Kalman filter at variable sample rate is to update T_0 in the transition matrix $\mathbf{F}$ constantly. For prediction steps, T_0 is the time between the last update step of the filter and the current time. The state prediction $\hat{\mathbf{x}}_{k|k-1}$ and the error covariance prediction $\mathbf{P}_{k|k-1}$ is therefore estimated for the correct time. If a measurement is available, the update step is performed with the same transition matrix $\mathbf{F}$. If no measurement is available, no update is performed. Future prediction steps will be performed with increasing time T_0 until an update takes place. Tracks can be discarded if the predicted error covariance $\mathbf{P}_{k|k-1}$ grows beyond a threshold.

The parameters for the filter are as follows: The process noise $\mathbf{w}$ is set to 1.1 m/s for velocity and 0.7 m for position. Those values can be derived from the expected acceleration of vehicles. The measurement noise is $\mathbf{v} = 2$ m corresponding to the detection grid. The initial error covariance $\mathbf{P}$ is set to 3 m/s for velocity and

1m for position. The initial position state corresponds to the detection position with zero velocity. The velocity is updated during the second detection using the first motion vector. Observations $\mathbf{m}_{i,k}$ are associated with tracks based on the distance $d_{i,j}$ in equation (48) between the observation $\mathbf{m}_{i,k}$ and the prediction $\hat{\mathbf{x}}_{j,k|k-1}$ normalised by the diagonal elements of the predicted error covariance $\mathbf{P}_{k|k-1}$ of track j. A distance $d_{i,j}$ is calculated between prediction $\hat{\mathbf{x}}_{j,k|k-1}$ of track j and every observation $\mathbf{m}_{i,k}$. The observation with the smallest distance is then associated with track j. Changes in the model id (*i.e.* classification result) between the last observation of a track id_j and the current observation id_i are penalised, as the difference between ids increases the distance $d_{i,j}$ in equation (48). The total number of model ids is 10. This approach is possible because the system performs classification before the tracking.

$$d_{i,j} = \sqrt{\left[\left(x_i - x_j\right)P_x^{-1}\right]^2 + \left[\left(y_i - y_j\right)P_y^{-1}\right]^2 + \tfrac{1}{10}\left|id_i - id_j\right|} \tag{48}$$

6.3.2. Evaluation Framework

The object tracking performance is evaluated by comparing the tracker with a baseline tracker (OpenCV blob tracker)). The OpenCV tracker uses an adaptive mixture of Gaussians for background estimation, connected component analysis for data association and Kalman filtering for tracking blob position and size. The i-LIDS benchmarking video data set is used for evaluation. Performance is evaluated on a subset (due to limited ground truth for tracking) of the previous chapter's data (PVTRA10xxxx): 1a03, 1a07, 1a13, 1a19, 1a20, 2a05, 2a10 and 2a11. Those videos contain overcast, sunny, changing weather conditions and camera saturation. The ground truth used for evaluation is provided with the i-LIDS data set. It is of limited duration within the videos and does not include pedestrians on the road. The evaluation was constrained to the two regions of interest on the road (dark red boxes in Figure 67) for both trackers.

Performance evaluation has played an important role in developing, assessing and comparing object tracking algorithms. (Lazarevic-McManus *et al.*, 2007) evaluated performance of motion detection based on ROC-like curves and the F-measure. The latter allows comparison using a single value domain, but is mainly designed to operate on motion detection rather than tracking. There is a significant body of work dealing with evaluation of both motion detection and tracking. (Needham and Boyle, 2003) proposed a set of metrics and statistics for comparing trajectories to account for detection lag, or constant spatial shift. However, taking only the trajectory (a set of points over time) as the input of evaluation may not give sufficient information about how precise the tracks are, since the size of the object is not considered. (Bashir and Porikli, 2006) use the spatial overlap of ground truth and system bounding boxes which is unbiased towards large objects. However they are counted per frame, which is justified when the objective is object detection. In object tracking, counting true positive (TP), false positive (FP) and false negative (FN) *tracks* is a more natural choice which is consistent with the expectations of video analytics end-users. Brown *et al.*, 2005 suggest a framework for matching of system track centroids and an enlarged ground truth bounding box which favours tracks of large objects.

The performance evaluation framework and implementation of Yin *et al.*, 2007 is used here. A rich set of metrics is proposed, such as Correct Detected Tracks, False Detected Tracks and Track Detection Failure to provide a general overview of the system's performance. Track Fragmentation shows whether the temporal and spatial coherence of tracks is established. ID Change is useful to test the data association module of the system. Latency indicates how quick the system can respond to an object entering the camera view, and Track Completeness how complete the object has been tracked. Metrics such as Track Distance Error and

Closeness of Tracks indicate the accuracy of estimating the position, the spatial and the temporal extent of the objects respectively.

6.3.3. Results

First, illustrative examples (Figure 67 to Figure 69) compare the tracker with the OpenCV baseline before providing the evaluation metrics. The full results in Table 25 indicate that the proposed system outperforms the OpenCV tracker on high level metrics such as correct detected tracks, track detection failure, false detected tracks and track fragmentation. This can mainly be attributed to the additional prior information from using 3D models to classify the content of the input video.

For metrics that evaluate the motion segmentation such as track closeness and distance error, both trackers have similar performance, which can be explained by the similar background estimation method. The track closeness of the proposed system is better than the baseline due to 3D models which are more robust against shadows, which can be observed for the bus in Figure 67 and the occluded car in Figure 68. The extent of the projected wire frame model is used as bounding box for the proposed system. The false detected tracks of the OpenCV tracker are high due to systematic detection of pedestrians, which cannot be classified. Both classifiers in chapters 3 and 5 exhibited low precision for bicycles when no pedestrian model was used due to the same reasons. Refer to Figure 69 for an example. The proposed system detected 94% of the ground truth tracks compared to 88% of the base line. It also has half of the track detection failures compared to the base line. The higher detection rate can be explained by a more sensitive background estimation setting, which produces more complete and additional noisy detections. However, the classification stage rejects many ambiguous detections. Id change can occur if a track of an object leaving is continued for a new object. This is worse for the proposed system compared to the OpenCV tracker, because the tracker is more

Figure 67 Correct detected tracks inside the active regions of interest (dark red boxes). Left: the proposed system with corresponding ground plane tracks. Right: OpenCV tracker result. Note the spatially fragmented tracks for the baseline in the first row and the correct number for tracks for the proposed tracker.

persistent, occasionally wrongly continuing a track but therefore generating much less track fragmentations. The path information in image and ground plane generated by the tracker can be used for high level behaviour analysis. The next section will introduce possible applications.

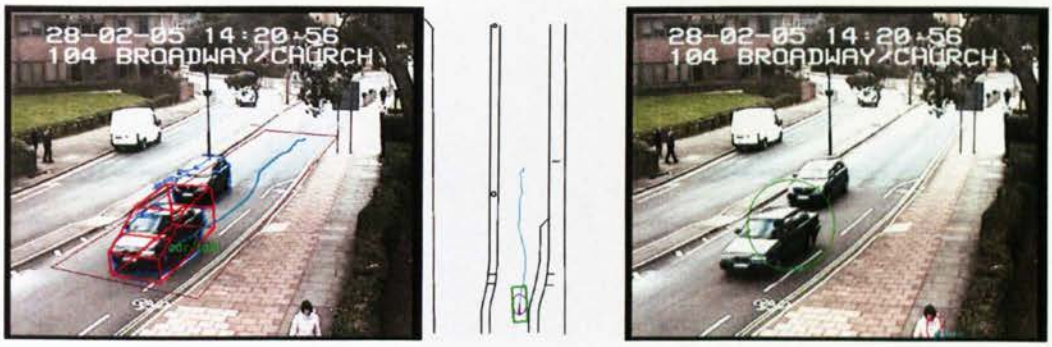


Figure 68 In this frame, the second car is missed due to occlusion between the vehicles. The proposed tracker on the left correctly locates the first car. The OpenCV tracker merged both cars with a large bounding box at a central position.

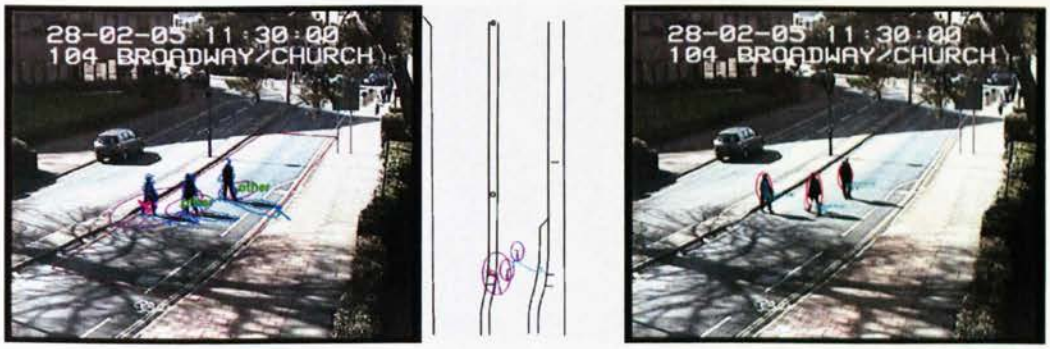


Figure 69 Pedestrians are correctly rejected as "other" class by the proposed tracker and detected by the OpenCV tracker.

Metrics	proposed Tracker	Object Kalman Tracker
Number of Ground truth tracks	100	100
Number of system tracks	144	203
Correct detected tracks	94	[illegible]
Track detection failure	6	12
False detected tracks	27	98
Latency (frames)	5	5
Track fragmentation	8	18
Average track Completeness (time)	64%	55%
ID change	10	5
Average track closeness (bbox overlap)	54%	45%
Standard Deviation of closeness	20%	11%
Average distance error (pixels)	22	21
Standard Deviation of distance error	19	15

Table 25 Tracking results

6.4. Behaviour Analysis

Behaviour analysis is the final step in most surveillance tasks and requires the highest level of conceptual understanding. For identifying incidents and interactions, a system requires information such as type of vehicles/people and their trajectories. The classifiers proposed in previous chapters with the tracking extension here can provide this information.

There are many possible scenarios in traffic management which could benefit from this automated analysis. Congestion monitoring is a relevant topic in London and has been rolled out to over 100 cameras in the last year. The most level of complexity is classified counts of vehicles. More complex in terms of interactions of road users is bus lane intrusion of non authorised vehicles, stopping in box junctions when the exit is blocked and banned turns at intersections. All these applications can be tackled by applying alarm rules to observed trajectories of an automated system.

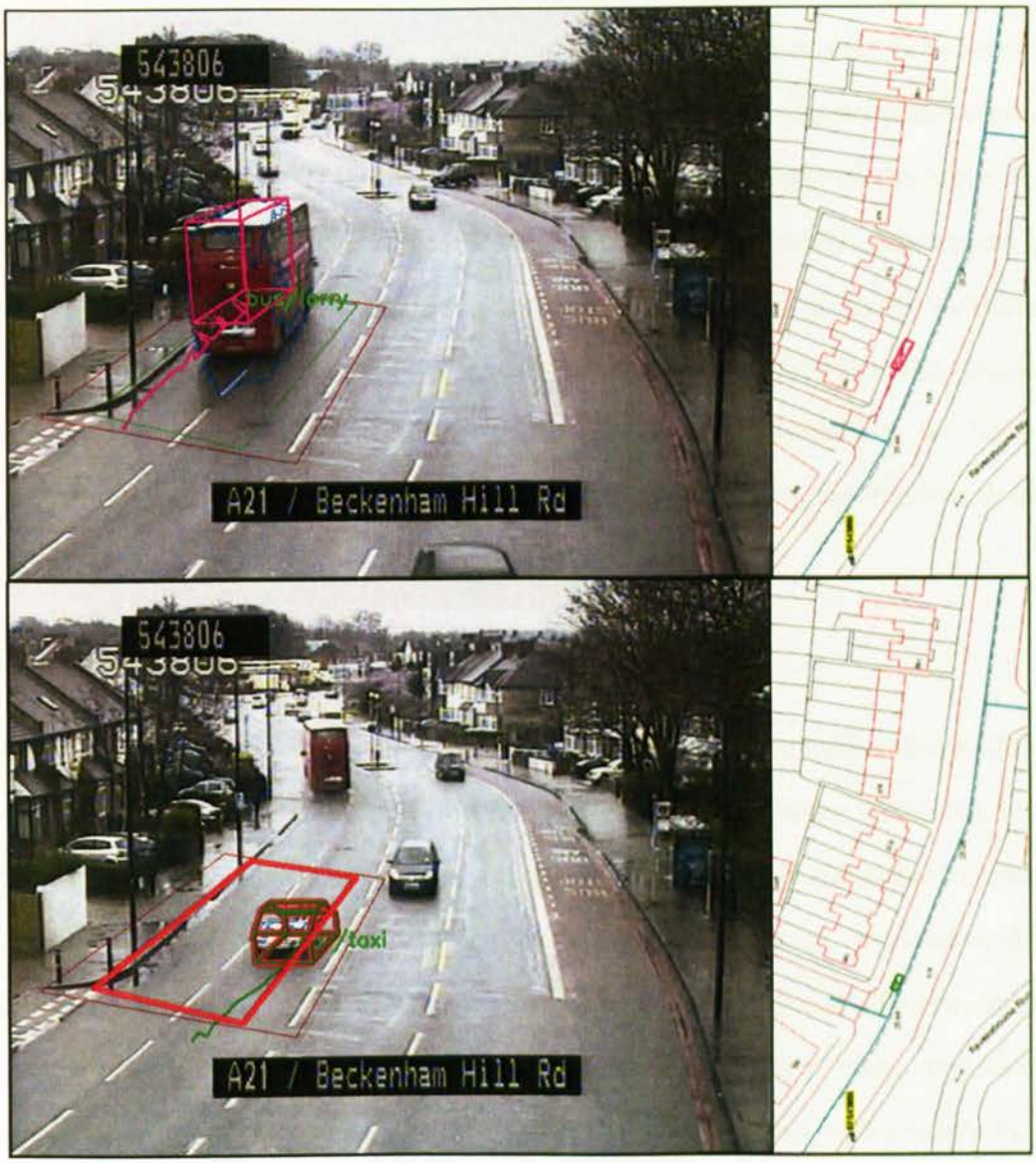


Figure 70 Bus lane intrusion detection example. The restricted zone is marked as green region permitting only buses. Vehicles of a restricted class with tracks inside this region trigger alarms shown in red in the lower image.

As a demonstrator for behaviour analysis, a bus lane intrusion monitor has been implemented. It is integrated as an output filter to the overall system and requires the setting of a monitoring region (see example in Figure 70 shown in green). This region is defined in the same way as the region of interest for the detector. In addition, a list of permitted classes is required to avoid alarms on buses, pedestrians or bikes. The green restricted region is applied to the ground plane

centre of vehicles rather than whole motion silhouettes. This is very beneficial in low angle urban camera views, where motion centroids and ground plane centres may differ significantly. If the region is defined along the edges of the lanes of a road, the majority of a high vehicle's silhouette might be outside this region as shown in the example. The ground plane centre however is correctly estimated within the road region according to the prior knowledge of the 3D scene. Correct operation for a bus and car is shown in Figure 70. Depending on further applications, many more monitoring tasks can be integrated into the framework due to a simple plug-in structure of the software (appendix A).

6.5. Summary

This chapter introduced application related issues for traffic surveillance systems in respect to the proposed system. Good portability between camera views has been demonstrated for a data set which was not part of the training. Superior performance compared to an industrial state of the art vehicle classifier is observed with a classifier precision of 95.1% for the car class. The data set and classification results were provided by the same company. Due to the zoomed view, significant occlusions occurred for vehicles. The classifier implicitly deals with occlusions in the same way as with visibility changes due to orientation. This enabled correct operation under those conditions.

Following on from the robustness considerations, a tracking extension based on Kalman filters was proposed. The performance was evaluated with the comprehensive framework of (Yin *et al.*, 2007). A baseline blob tracker was outperformed for most metrics demonstrating 94% correct detected tracks. The tracking information was then used to produce a behaviour analysis demonstrator. Examples for bus lane intrusions are shown. In addition, the framework allows integration of plug-ins for other monitoring objectives.

7. Conclusions

7.1. Summary

This thesis addressed computer vision algorithms for road user detection and classification. The aim was to provide more information to traffic managers, who in turn can improve the car travel experience in real-time or with strategic planning. Information from existing CCTV infrastructure is frequently lost due to a limited number of operators compared to cameras. This project was sponsored by Transport for London to investigate robust computer vision algorithm to support CCTV operators in the urban environments. For ease of reading, a short summary of the work is provided here. The next section will critically discuss the work and an outlook for further work will be provided in section 7.3.

Based on the literature, a novel framework for road user classification using 3D models has been introduced. Camera calibration makes this framework portable between cameras without the requirement for retraining the classifier. Good performance is demonstrated on the i-LIDS data set provided by the UK Home Office. The system is limited by the quality of the extracted motion silhouette. To mitigate the strong dependence on the motion silhouette, the use of local image features was investigated. As a test case, a texture saliency classifier has been implemented, which uses features derived from the fast Fourier transform (FFT) from local image patches. The classifier operates on single frames to detect salient objects, which in the evaluation are people approaching a fence. Good real-time performance figures are shown for the 24 hour sterile zone test of i-LIDS. The local features proposed are an effective way of exploiting appearance for visual surveillance.

This local feature concept was integrated into the earlier 3D framework to form the novel 3DHOG classifier. This algorithm integrates the advantages from both ideas by modelling road user appearance with local patches and spatially constraining them in real world space. The proposed framework allows for a variable number of features to be used, so that a single model for any viewing angle of a road user is sufficient. Depending on camera view and road user orientation, visible feature points are extracted and normalised using the 3D model and used for classification. In this way, training data is fully portable between camera views (scale, angle) and resolutions. Those capabilities have been demonstrated by evaluation on a new data set and comparison with an industrial single view appearance classifier trained on the new data set. The proposed system trained on i-LIDS outperforms the industrial classifier, both tested on the new data set. Finally, a tracking extension with Kalman filters has been introduced. This enabled the implementation of a behaviour demonstrator, which uses trajectory and class information of road users to generate alarms for a bus lane monitor.

7.2. Discussion

The work started based on motion estimation, which is the state of the art in visual surveillance. A static camera assumption is essential for such a system. Following on from this initial approach, the use of local features incorporates appearance into the object models. This in turn removes the strong reliance on motion estimation to weaken the static camera requirements. The novel 3DHOG classifier can be applied to still images in a dense search or in a targeted search based on an initial hypothesis. This hypothesis can be generated from motion or by prediction of a tracker. In this way, the system can incorporate historical information during detection. This structure was identified in section 2.6 to be an essential element for traffic surveillance systems to increase robustness.

The use of local features increased the computational complexity of the classifier. This is due to the image patch warping and feature extraction during the model matching process. This task however, is highly parallel in nature and hardware accelerators (GPU) are designed to perform such operations in parallel. For a smaller number of image patches used for intrusion detection in chapter 4, real time performance on a standard PC is possible. Optimisation for specific scenarios or hardware configuration (*e.g.* GPU) is desirable for a final product based on the proposed system.

Initial detection and hypothesis generation is based on motion silhouettes with the assumption that every silhouette contains one object. This is an essential assumption for the motion silhouette based classifier. The appearance based classification of 3DHOG would be able to detect several objects in a single silhouette. This could either be implemented through a multiple object search for every silhouette or by adding tracking predictions to the hypothesis list of a new frame. In this way, tracks will be continued and the hypotheses from motion silhouettes can start new tracks.

Evaluation of the proposed algorithms is somewhat limited, which is typical for surveillance applications. Longer and more comprehensive video data for testing is desirable, but comes with the cost of ground truth generation. Ground truth, which is part of the i-LIDS data set, has been used and extended to match the requirements of the evaluation. In addition, a new industrial data set is used. A baseline industrial classifier was used to generate ground truth, which was then manually checked for classification accuracy. Overall, a good attempt has been made on evaluation, especially for the human intrusion with full 24 hours of video data; however, more data would always be desirable.

An interesting consideration is the comparison of human observers and computer vision systems. It is clear, that even the latest technology as described in

this thesis is far inferior to human perception for object recognition, identification and scene understanding. Nevertheless, there is space for automatic video analysis for supporting human operators in specific tasks. As outlined in the introduction, the continuous observation of a camera can be performed automatically, but tires humans very quickly. The work presented in this thesis is therefore targeted at specific continuous observation tasks which are currently achievable with technology, which then alerts human operators to make a judgement or follow up incidents further. This aspect makes the technology particularly valuable for the human intrusion detection case presented in chapter 4.

7.3. Future Work

There is significant potential to improve and extend the algorithms presented in this thesis. For intrusion detection with local features, extensions for features and clustering can be considered. The current classifier uses a scalar feature value derived from the image spectrum to distinguish between foreground and background. A multi dimensional feature could improve performance, but would trade off computational speed at the same time. The current clustering of individual patches to objects can also be improved. Incorporating time and clustering in the spatio-temporal domain would reduce the effect of single frame noise. Motion estimation could also be included directly into the clustering: it is currently fused with the clustering result. The clustering itself can be improved by employing a spatial- temporal Markov random field (ST-MRF). This could incorporate smoothing and consistency constraints.

For road user detection and classification with local features, there may be many ways of extending and improving the current systems. Firstly, the image patches could be extracted at several scales and the feature extraction could consider this whole image pyramid. The selection and weighting of interest points

could be performed by an Adaboost framework. A dense grid of interest points can represent the weak hypotheses. The final strong classifier would use the best subset of those points. This training methodology could benefit from additional negative training samples or could use the proposed concept of positive samples with position offsets to generate sharp location performance.

Training data itself can be extended to incorporate more diverse conditions including night time. This might result in the need for modelling the appearance vectors with multiple Gaussians (GMM) rather than the current single Gaussian model. For computational efficiency, the spatial 3D models can be re-used between classes. In this way, every extracted feature patch would be compared with the trained appearance of several classes (*e.g.* bus, lorry) to generate a match measure for all those classes from only a single image feature extraction step.

The search strategy for new object instances may be improved. The current grid approach can be replaced by mean shift or similar methods to provide faster convergence. Tracking information is very useful for this task to be incorporated as parameters into the search strategy. More advanced tracking techniques like particle filters can be integrated with the search framework to allow for multiple hypotheses tracking. Finally, the 3DHOG classifier could be applied to moving cameras for driver assistance inside vehicles. The homography of a vehicle mounted camera to the road surface remains approximately constant apart from short intervals for example when ascending onto a speed bump. A fixed window in front of the car could be searched in every frame to detect and classify road users.

The assumptions required for surveillance systems using background estimation can limit the applications and robustness significantly. Background models cannot be generated from moving cameras. Surveillance systems in general could benefit from moving away from the use of background and motion estimation to more generic approaches. Full scene understanding and object recognition

without strong constraints are undoubtedly harder tasks, but emerging technology mastering this will be able to enable more applications, which so far can only be solved by humans.

7.4. Publications

Norbert Buch, James Orwell and Sergio A. Velastin. **Urban Road User Detection and Classification using 3D Wire Frame Models.** *IET Computer Vision Journal* 2010, Vol. 4, Issue 2, pages 105-116.

Norbert Buch, James Orwell and Sergio A. Velastin. **3D Extended Histogram of Oriented Gradients (3DHOG) for Classification of Vehicles and Pedestrians in Urban Scenes.** In *British Machine Vision Conference BMVC 2009*, London, September 2009.

Norbert Buch, Fei Yin, James Orwell, Dimitrios Makris and Sergio A. Velastin. **Urban Vehicle Tracking using a Combined 3D Model Detector and Classifier.** In *13th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems KES 2009*, Part I, LNCS 5711, pages 169–176, Santiago, Chile, September 2009

Norbert Buch, Mark Cracknell, James Orwell and Sergio A. Velastin. **Vehicle Localisation and Classification in Urban CCTV Streams.** In *Intelligent Transport Systems World Congress 2009*, Stockholm, Sweden, September 2009

Norbert Buch and Sergio A. Velastin. **Human intrusion detection using texture classification in real- time.** In *First International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences THEMIS 2008 co-hosted with BMVC2008*, pages 1–6, Leeds, September 2008.

Norbert Buch, James Orwell, and Sergio A. Velastin. **Detection and classification of vehicles for urban traffic scenes.** In *International Conference on Visual Information Engineering VIE08*, pages 182–187. The Institution of Engineering and Technology, Xi'an, China, July 2008.

7.4.1. Accepted conditional to changes

Norbert Buch and Sergio A. Velastin. **Local Feature Saliency Classifier for Real-Time Intrusion Monitoring.** *Pattern Recognition Letters*

Norbert Buch, James Orwell and Sergio A. Velastin. **A Review of Computer Vision Techniques for the Analysis of Urban Traffic.** *IEEE Transaction on Intelligent Transportation Systems*

7.4.2. In preparation for journal

Norbert Buch, James Orwell and Sergio A. Velastin. **3D Extended Histogram of Oriented Gradients (3DHOG) for Classification of Road Users.**

7.4.3. Presentations

Derek Renaud and Norbert Buch, **Data to Decision to Action in London's Traffic Systems.** *British Computer Society Evening Lecture*, BCS, Southampton Street, London, March 2010

Norbert Buch, James Orwell, and Sergio A. Velastin. **Classifying and Tracking Vehicles in Urban CCTV Streams.** *BMVA Symposium on Vision for Automotive Applications*, London, May 2009.

Norbert Buch. **Classification of Vehicles and their Behaviour for Urban Traffic Scenes.** *Doctoral Consortium*, British Computer Society, London, May 2009. (best presentation award)

Norbert Buch and Mark Cracknell, **The Future of "Smart" CCTV - Classification of Vehicles and their Behaviour for Urban Traffic Scenes**, Invited Talk at Directorate for Traffic Operations, London, April 2008.

7.5. Personal Statement

The work on this thesis and the exploration of the field of computer vision was enjoyable. It was very satisfying to be involved with practical issues of Transport for London and finding novel solutions for computer vision and transport problems. I would never want to do without my time here at Kingston University.

„Wer fertig ist, dem ist nichts recht zu machen,
Ein Werdender wird immer dankbar sein."

- Johann Wolfgang von Goethe, Faust I

"A mind once formed finds naught made right thereafter;
A growing mind will thank you evermore."

- translation G. M. Priest

A. Mathematical Proofs

A.1. Sigmoid Parameters

This section provides the proof for the parameter calculation of section 5.5.3.2 on page 143. A logistic sigmoid function is estimated to transform a given Mahalanobis distance measure d_k into a match measure m_k in the interval $[0,1]$:

$$m_k = \mathrm{s}(d_k) \tag{49}$$

The sigmoid function $\mathrm{s}(d)$ is defined as

$$\mathrm{s}(d) = \frac{1}{1+e^{a(b-d)}} \tag{50}$$

and uses two parameters a and b. The parameters can be estimated from the distance surface $\bar{\mathbf{D}}_M$ for every model. The proposed parameterisation places the centre point of the distance surface d_C at the middle of the sigmoid function (Figure 71) with a match measure of $\mathrm{s}(d_C) = 0.5$. This results in

$$a = \frac{2}{d_C - \bar{d}} \tag{51}$$

$$b = d_C, \tag{52}$$

where d_C is the centre point value of the distance surface $\bar{\mathbf{D}}_M$ and $\bar{d}$ in the mean value of the surface $\bar{d} = \mathrm{mean}(\bar{\mathbf{D}}_M)$. See Figure 49 on page 143 for an example surface of an interest point and Figure 71 for the resulting sigmoid function.

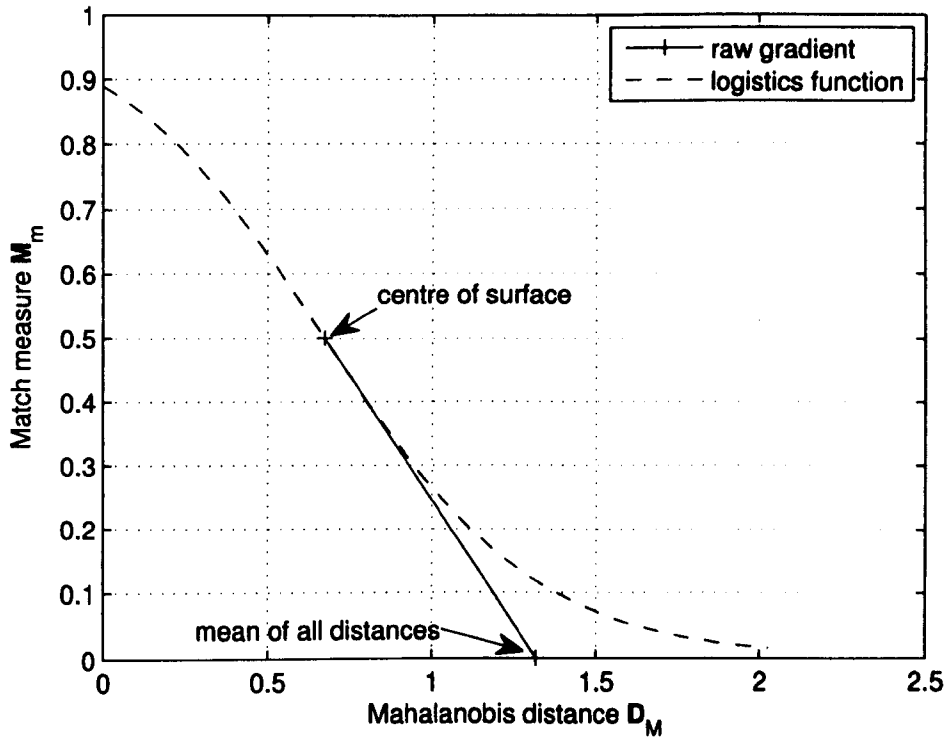


Figure 71 Estimated sigmoid function shown as a dashed line. The continuous line is the gradient of the sigmoid function defined by the centre value d_C of the distance surface $\bar{D}_M$ and the mean distance $\bar{d}$ of all values.

Proof

This proof shows how equations (51) and (52) provide the properties of $s(d)$ as described above. The first constraint is $s(d_C) = 0.5$ to centre the sigmoid function, which enforces

$$s(d_C) = \frac{1}{1+e^{a(b-d_C)}} = \frac{1}{2} \tag{53}$$

The above function is smooth and strictly monotonic for d_C and also for variations of parameter b. A valid solution for b will therefore also be a unique solution. Equation (52) is the unique solution to equation (53) which is shown by inserting (52) in (53):

$$s(d_C) = \frac{1}{1+e^{a(d_C-d_C)}} = \frac{1}{1+e^{0}} = \frac{1}{2} \qquad \text{q.e.d.} \tag{54}$$

The gradient of the sigmoid function at point $s(d_C)=0.5$ (like before) defines the other parameter a. The gradient should be equal to the gradient of a line of the training data between the centre distance value d_C at match measure $m_k=0.5$ and the mean of all distance values $\bar{d}$ at match measure $m_k=0$. This is illustrated as a continuous blue line in Figure 71. The gradient g of this line is

$$g=\frac{0.5}{d_C-\bar{d}}, \tag{55}$$

which will be made equal to the gradient of the sigmoid function $s(d)$. The gradient of the sigmoid function is calculated as first derivative of $s(d)$ with respect to the distance d

$$\frac{ds(d)}{dd}=\frac{-1}{\left(1+e^{a(b-d)}\right)^2}e^{a(b-d)}(-a). \tag{56}$$

Considering the gradient at the point d_C and using the existing parameter $b=d_C$, the gradient can be expressed as

$$\frac{ds(d_C)}{dd_C}=\frac{-1}{\left(1+e^{a(d_C-d_C)}\right)^2}e^{a(d_C-d_C)}(-a)=\frac{-1}{\left(1+e^{0}\right)^2}e^{0}(-a)=\frac{a}{4}. \tag{57}$$

The above gradient is then made equal to the line gradient g from equation (55):

$$\frac{a}{4}=g. \tag{58}$$

Using equation (55) and (58), the parameter a is given by

$$a=\frac{2}{d_C-\bar{d}} \qquad \text{q.e.d.} \tag{59}$$

B. Plug-in Hierarchy and Parameters

B.1. Overview of Modules

Application Core application for detection and classification of vehicles

- **-videoFilter** Filters the input video before detector
  - **InterlaceFilter** Interlace filter for input video
- **-detector** Takes frame images and generates contour list
  - **DetectorStd** Detector using standard OpenCV foreground estimation
    - **-dt-fgEstimator** Generates a foreground mask for frame image
      - **GMMHSV** GMM with HSV shadow removal
- **-classifier** Takes contour list and tracks to generate object positions
  - **ClassifierStd** Class for standard classifiers
    - **-cl-positionSearch** Module to generate 3D hypothesis and find maximum score
      - **PositionSearchGrid** Grid position search module
      - **PositionSearchTracking** Tracker grid position search module
    - **-cl-matchMeasure** Module to generate match measure for hypothesis based on models
      - **MatchLocalFeature** Local feature score module
        - **-mm-featureExtraction** Module to extract feature vector from image patches
          - **FftFeature** FFT feature extraction module
          - **HOGFeature** HOG feature extraction module
          - **HistFeature** Histogram feature extraction module
      - **MatchSilhouette** Silhouette overlap score module

-tracker	Takes object positions of one or more classifiers to generate tracks
TrackerBasic	Class for basic tracker
TrackerDummy	Class for empty tracker
TrackerFrame	Class for frame tracker
-tr-filterGenerator	Module to handle filter for tracking
KalmanFilterGenerator	Class for Kalman filter handling
-overlayWriter	Auxiliary module for CV modules to save overlay images for objects
OverlayWriter	Class to generate overlay output
-resultWriter	Takes track list and generates output from it
BusLane	Class for bus lane intrusion detection
GPview	Class for ground plane result writer
Video3Dview	Class for 3D video result writer
ViperWriter	Class for generating viper output

B.2. Parameter List

Application Core application for detection and classification of vehicles

Parameter	*Type*	*Short description*
-videoFilter	string list	Video filters used
-detector	string	Detector to be used
-classifier	string	Classifier to be used
-tracker	string	Tracker to be used
-resultWriter	string list	Track result writer to be used
-video	file name	Input video file
-framedivider	number	Divider of the input framerate for processing
-screen-on	boolean	Show application windows
-wait-time	number	Wait time in (0,50) ms
-output	file name	Output video file
-fourcc	string	4 letter identifier for output codec
-frames	file name	Frames of interest file
-start-frame	number	Start frame in video
-stop-frame	number	Stop frame in video

-time-log	boolean	Activate time logging

-videoFilter Filters the input video before detector

InterlaceFilter Interlace filter for input video

Parameter	*Type*	*Short description*
-vf-algorithm	number	De- interlacing algorithm
-vf-output	file name	Output video file
-vf-fourcc	string	4 letter identifier for output codec
-vf-screen-on	boolean	Show detector windows
-time-log	boolean	Activate time logging

-detector Takes frame images and generates contour list

DetectorStd Detector using standard OpenCV foreground estimation

Parameter	*Type*	*Short description*
-dt-output	file name	Output video file
-dt-fourcc	string	4 letter identifier for output codec
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml
-dt-roi	string	x3d file with ROI as line set
-dt-min-roi-overlap	number	Minimum overlap to trigger silhouette
-dt-screen-on	boolean	Show detector windows
-dt-silhouette	boolean	Show silhouette in output image
-dt-display-roi	boolean	Display region of interest in output
-dt-fgEstimator	string	Name of foreground estimator to use
-dt-segm-file	file name	File with segmentation information
-dt-segm-generate	boolean	Flag to generate segmentation file
-dt-min-length	number	Minimum length of contour
-time-log	boolean	Activate time logging

-dt-fgEstimator Generates a foreground mask for frame image

GMMHSV GMM with HSV shadow removal

Parameter	*Type*	*Short description*
-dt-remove-shadow	boolean	Activate shadow removal module
-dt-min-value-sim	number	Minimum vaule similarity for shadow

-classifier Takes contour list and tracks to generate object positions

ClassifierStd Class for standard classifiers

Parameter	*Type*	*Short description*
-cl-output	file name	Output video file
-cl-fourcc	string	4 letter identifier for output codec
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml

-cl-screen-on	boolean	Show classifier windows
-output-dir	directory	Path to save output results
-cl-output-match	boolean	Flag to save match numbers
-cl-positionSearch	string	Module to generate 3D hypothesis and find maximum score
-cl-matchMeasure	string	Module to generate match measure for hypothesis based on models
-cl-calc-bbox	boolean	Flag to calculate 2D bounding box
-cl-wireframe	boolean	Display wire frame of models
-cl-label	boolean	Display class label for objects
-cl-use-prediction	boolean	Use prediction to initialise classifier
-cl-event-loop	boolean	Run event loop in classifier
-time-log	boolean	Activate time logging

-cl-positionSearch Module to generate 3D hypothesis and find maximum score

PositionSearchGrid Grid position search module

Parameter	*Type*	*Short description*
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml
-ps-rotation	number	Angle of fixed model orientation
-ps-grid-width	number	Width of position grid (m)
-ps-grid-rows	number	Number of rows in grid
-ps-min-score	number	Minimum score for object detection (0..1)

PositionSearchTracking Tracker grid position search module

Parameter	*Type*	*Short description*
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml
-ps-rotation	number	Angle of fixed model orientation
-ps-grid-width	number	Width of position grid (m)
-ps-grid-rows	number	Number of rows in grid
-ps-min-score	number	Minimum score for object detection (0..1)

-cl-matchMeasure Module to generate match measure for hypothesis based on models

MatchLocalFeature Local feature score module

Parameter	*Type*	*Short description*
-mm-featureExtraction	string	Module for feature vector extraction
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml
-mm-model-dir	directory	Directory with models
-mm-patch-size	number	Size of patches in metres
-mm-patch-resolution	number	Resolution of patches in pixels / metre
-mm-dir-threshold	number	Minimal angle similarity of IP and camera (0..90)

-mm-patches-dir	directory	Directory to save X3D files of extraced patches
-mm-normalise-patch	boolean	Flag to normalise input image patch
-mm-min-ips	number	Minimum interest points required
-mm-features-dir	directory	Directory to save X3D files of extraced features
-mm-x3d-template	file name	Template for saving patches -mm-patches-dir
-frame-save-data	number	Frame to save any extra data for
-mm-training-data	file name	File containing training data
-mm-detailed-score	directory	Directory to save detailed score
-mm-use-weights	boolean	Activate usage of IP weights
-mm-descriptor-patch-scale	number	Scale for interest point patches

-mm-featureExtraction Module to extract feature vector from image patches

FftFeature FFT feature extraction module

Parameter	*Type*	*Short description*
-save-images	boolean	Save temporary images
-frequency-bins	number	Bin in frequency histogram
-angle-bins	number	Bin in angle histogram

HOGFeature HOG feature extraction module

Parameter	*Type*	*Short description*
-angle-bins	number	Number of angle bins used
-save-images	boolean	Save temporary images

HistFeature Histogram feature extraction module

Parameter	*Type*	*Short description*
-angle-bins	number	Number of angle bins used
-save-images	boolean	Save temporary images

MatchSilhouette Silhouette overlap score module

Parameter	*Type*	*Short description*
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml
-mm-model-dir	directory	Directory with models

-tracker Takes object positions of one or more classifiers to generate tracks

TrackerBasic Class for basic tracker

Parameter	*Type*	*Short description*

-tr-output	file name	Output video file
-tr-fourcc	string	4 letter identifier for output codec
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml
-tr-screen-on	boolean	Show tracker windows
-time-log	boolean	Activate time logging
-tr-max-predictions	number	Maximum number of position predictions
-tr-distance-limit	number	Maximum track/object distance for assignment

TrackerDummy Class for empty tracker

Parameter	*Type*	*Short description*
-tr-output	file name	Output video file
-tr-fourcc	string	4 letter identifier for output codec
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml
-tr-screen-on	boolean	Show tracker windows
-time-log	boolean	Activate time logging

TrackerFrame Class for frame tracker

Parameter	*Type*	*Short description*
-tr-output	file name	Output video file
-tr-fourcc	string	4 letter identifier for output codec
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml
-tr-screen-on	boolean	Show tracker windows
-time-log	boolean	Activate time logging
-tr-max-predictions	number	Maximum number of position predictions
-tr-max-uncertainty	number	Maximum position uncertainty
-tr-min-observations	number	Minimum observations for valid track
-tr-filterGenerator	string	Module to handle filter for tracking

-tr-filterGenerator Module to handle filter for tracking

KalmanFilterGenerator Class for Kalman filter handling

Parameter	*Type*	*Short description*
-time-log	boolean	Activate time logging
-ft-process-noise-location	number	Process noise for location states
-ft-measurement-noise-location	number	Measurement noise for location
-ft-initial-error-location	number	Initial error estimate for location
-ft-process-noise-speed	number	Process noise for speed states
-ft-initial-error-speed	number	Initial error estimate for speed
-ft-xy-distortion	number	Relative importance of y to x
-ft-catch-area	number	Multiple of covariance for catch
-ft-process-noise-angle	number	Process noise for angle state
-ft-measurement-noise-angle	number	Measurement noise for angle

-ft-initial-error-angle	number	Initial error estimate for angle
-ft-process-noise-angle-speed	number	Process noise for angle speed state
-ft-initial-error-angle-speed	number	Initial error estimate for angle speed

-overlayWriter Auxiliary module for CV modules to save overlay images for objects

OverlayWriter Class to generate overlay output

Parameter	*Type*	*Short description*
-overlay-dir	directory	Directory for overlay images
-output-pic	string	Extension for saved images
-output-dir	directory	Path to save output results

-resultWriter Takes track list and generates output from it

BusLane Class for bus lane intrusion detection

Parameter	*Type*	*Short description*
-rw-output	file name	Output video file
-rw-fourcc	string	4 letter identifier for output codec
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml
-rw-screen-on	boolean	Show result writer windows
-time-log	boolean	Activate time logging
-rw-map	file name	Map file as background
-rw-permitted-class	string list	Permitted class in the observation region
-rw-observation-region	file name	x3d file with observation region

GPview Class for ground plane result writer

Parameter	*Type*	*Short description*
-rw-output	file name	Output video file
-rw-fourcc	string	4 letter identifier for output codec
-pixelspermetre	number	Scale of map image
-calibration	file name	Camera calibration as xml
-rw-screen-on	boolean	Show result writer windows
-time-log	boolean	Activate time logging
-rw-map	file name	Map file as background
-rw-show-predicted	boolean	To active output of predicted positions
-rw-show-catch	boolean	To active output of catch area
-rw-show-confidence	boolean	To active output on map
-rw-valid-only	boolean	Only show valid tracks
-rw-redraw-wireframe	boolean	Redraw wire frame

Video3Dview Class for 3D video result writer

Parameter	*Type*	*Short description*
-time-log	boolean	Activate time logging

-rw-template	file name	Template X3D to expand
-rw-x3d-output	file name	Output X3D file
-rw-include-other	boolean	Include detection of 'other' in output
-rw-timer-interval	number	Interval for base timer in X3D

ViperWriter Class for generating viper output

Parameter	*Type*	*Short description*
-time-log	boolean	Activate time logging
-rw-viper-config	file name	Config Viper file to expand
-rw-viper-output	file name	Output Viper file
-rw-viper-save-invalid	boolean	Save invalid tracks as well

B.3. Setup GUI

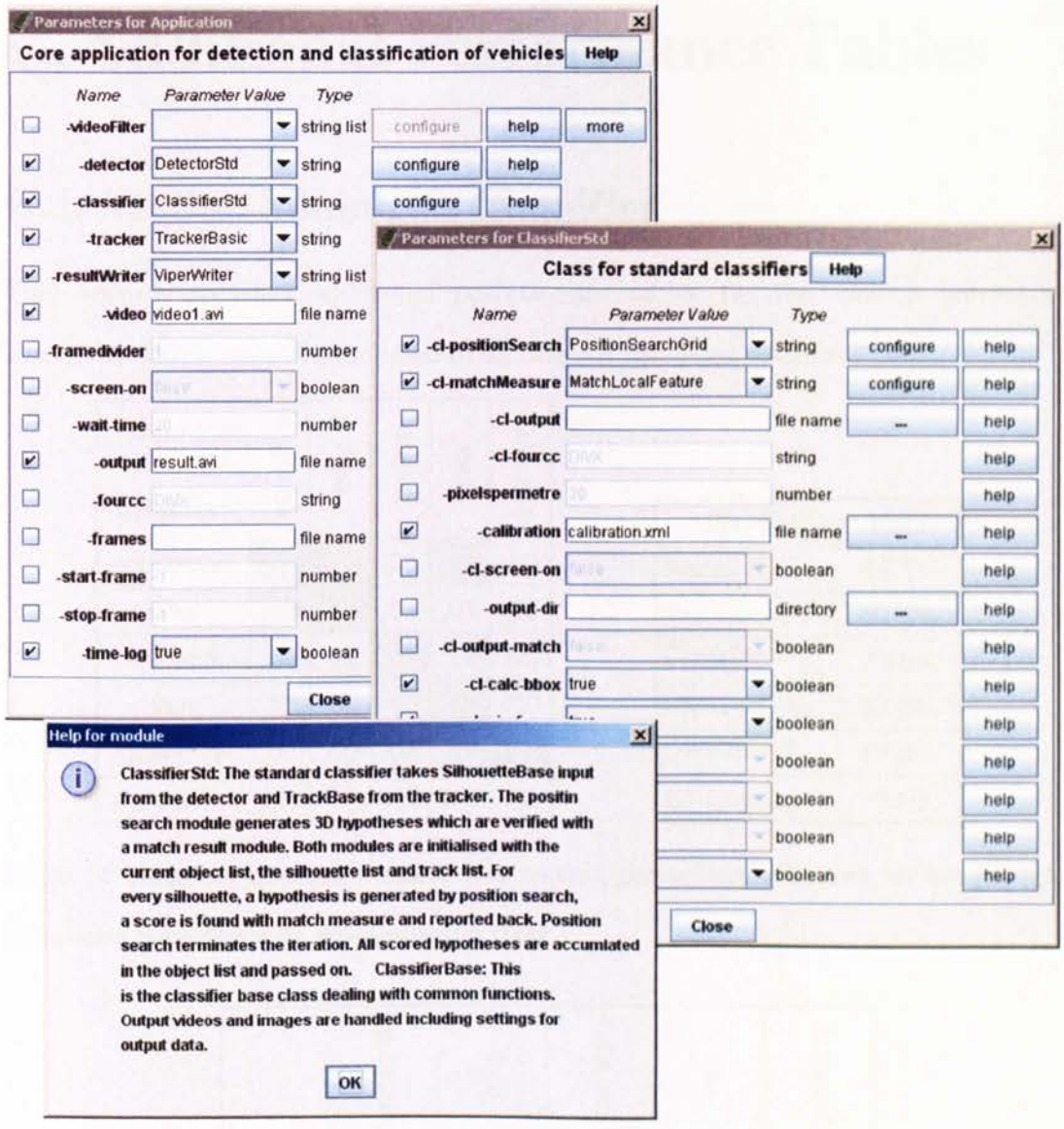


Figure 72 Snapshot of graphical user interface for parameter selection. Default values for parameters are displayed, selected modules can be configured and help is available for every element.

C. Additional Performance Tables

C.1. Motion Silhouette Classifier

This section provides additional performance tables for the motion silhouette classifier, which were not included in section 3.4 for space reasons.

ground truth / *detection*	pedestrian	bike	car/taxi	van	bus/lorry
pedestrian	.81	.54	.04	.31	.00
bike	.15	.44	.01	.01	.00
car/taxi	.02	.02	.78	.09	.00
van	.02	.00	.10	.45	.00
bus/lorry	.01	.00	.08	.14	1.00
count	188	41	342	109	52

Symbol	*Value*
Recall R	64.5%
Precision P	64.5%
Classifier P_C	73.6%
Detector R_D	87.6%
Detector P_D	87.6%
GT Overlap	0.56

Table 26 Classifier confusion matrix and overall performance figures for the motion silhouette classifier with de-interlacing filter

	pedestrian	bike	car/taxi	van	bus/lorry
R_j	63.2%	35.3%	69.9%	45.0%	100.0%
P_j	54.4%	34.0%	86.1%	52.7%	52.5%
R_{Cj}	81.4%	43.9%	78.1%	45.0%	100.0%
P_{Cj}	68.9%	36.7%	95.0%	57.0%	55.3%
R_{Dj}	77.7%	80.4%	89.5%	100.0%	100.0%
P_{Dj}	79.0%	92.5%	90.6%	92.5%	94.9%

Table 27 Class wise performance figures for the motion silhouette classifier with de-interlacing filter

ground truth / *detection*	pedestrian	bike	car/taxi	van	bus/lorry
pedestrian	.86	.59	.00	.08	.00
bike	.10	.39	.01	.00	.00
car/taxi	.03	.02	.83	.11	.00
van	.01	.00	.10	.63	.00
bus/lorry	.01	.00	.06	.19	1.00
count	152	41	322	80	53

Symbol	*Value*
Recall R	67.8%
Precision P	73.4%
Classifier P_C	79.8%
Detector R_D	85.0%
Detector P_D	92.0%
GT Overlap	0.56

Table 28 Classifier confusion matrix and overall performance figures for the motion silhouette classifier without additional filter

	pedestrian	bike	car/taxi	van	bus/lorry
R_j	63.0%	31.4%	72.2%	62.5%	100.0%
P_j	74.0%	44.4%	88.4%	53.2%	55.8%
R_{Cj}	86.2%	39.0%	82.9%	62.5%	100.0%
P_{Cj}	81.4%	48.5%	95.0%	59.5%	59.6%
R_{Dj}	73.1%	80.4%	87.0%	100.0%	100.0%
P_{Dj}	91.0%	91.7%	93.0%	89.4%	93.7%

Table 29 Class wise performance figures for the motion silhouette classifier without additional filter

ground truth / *detection*	bike	car/taxi	van	bus/lorry
bike	1.00	.00	.00	.00
car/taxi	.00	.98	.00	.00
van	.00	.02	.95	.00
bus/lorry	.00	.00	.05	1.00
count	12	130	20	9

Symbol	*Value*
Recall R	92.8%
Precision P	91.3%
Classifier P_C	98.2%
Detector R_D	94.5%
Detector P_D	92.9%
GT Overlap	0.65

Table 30 Classifier confusion matrix and overall performance figures for the motion silhouette classifier under sunny conditions

	bike	car/taxi	van	bus/lorry
R_j	70.6%	94.8%	95.0%	100.0%
P_j	80.0%	96.2%	73.1%	90.0%
R_{Cj}	100.0%	98.5%	95.0%	100.0%
P_{Cj}	100.0%	100.0%	90.5%	90.0%
R_{Dj}	70.6%	96.3%	100.0%	100.0%
P_{Dj}	80.0%	96.2%	80.8%	100.0%

Table 31 Class wise performance figures for the motion silhouette classifier under sunny conditions

ground truth / *detection*	bike	car/taxi	van	bus/lorry
bike	1.00	.04	.00	.00
car/taxi	.00	.87	.05	.00
van	.00	.03	.91	.03
bus/lorry	.00	.07	.05	.97
count	27	107	22	33

Symbol	*Value*
Recall R	85.1%
Precision P	82.3%
Classifier P_C	91.0%
Detector R_D	93.6%
Detector P_D	90.4%
GT Overlap	0.69

Table 32 Classifier confusion matrix and overall performance figures for the motion silhouette classifier under overcast conditions

	bike	car/taxi	van	bus/lorry
R_j	96.4%	78.2%	90.9%	97.0%
P_j	64.3%	93.0%	80.0%	76.2%
R_{Cj}	100.0%	86.9%	90.9%	97.0%
P_{Cj}	87.1%	98.9%	83.3%	80.0%
R_{Dj}	96.4%	89.9%	100.0%	100.0%
P_{Dj}	73.8%	94.0%	96.0%	95.2%

Table 33 Class wise performance figures for the motion silhouette classifier under overcast conditions

ground truth / *detection*	bike	car/taxi	van	bus/lorry
bike	.00	.06	.05	.00
car/taxi	.00	.91	.25	.00
van	.00	.02	.70	.00
bus/lorry	.00	.02	.00	1.00
count	0	106	20	20

Symbol	*Value*
Recall R	82.8%
Precision P	82.8%
Classifier P_C	89.0%
Detector R_D	93.0%
Detector P_D	93.0%
GT Overlap	0.67

Table 34 Classifier confusion matrix and overall performance figures for the motion silhouette classifier under changing weather conditions

	bike	car/taxi	van	bus/lorry
R_j	0.0%	82.8%	66.7%	100.0%
P_j	0.0%	88.1%	87.5%	87.0%
R_{Cj}	0.0%	90.6%	70.0%	100.0%
P_{Cj}	0.0%	95.0%	87.5%	90.9%
R_{Dj}	0.0%	91.4%	95.2%	100.0%
P_{Dj}	77.8%	92.7%	100.0%	95.7%

Table 35 Class wise performance figures for the motion silhouette classifier under changing weather conditions

C.2. 3DHOG Classifier

This section provides additional performance tables for the 3DHOG classifier, which were not included in section 5.7.3 for space reasons.

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.78	.02	.04	.06	1.04
car/taxi	.00	.74	.27	.29	.11
van	.00	.00	.56	.34	.06
bus/lorry	.00	.00	.00	.31	.03
FN	.22	.24	.13	.00	.00
count	50	356	48	35	
overlap	.66	.68	.76	.83	

Symbol	*Value*
Recall R	69.5%
Precision P	70.4%
Classifier P_C	87.6%
Detector R_D	79.3%
Detector P_D	80.3%
GT Overlap	0.69

Table 36 Confusion matrix and system performance for the 3DHOG classifier with patch size of $\delta = 0.8\,\mathrm{m}$ at resolution $\rho = 20\,\mathrm{Pixel/m}$

ground truth / *detection*	bike	car/taxi	van	bus/lorry
bike	1.00	.03	.05	.06
car/taxi	.00	.97	.31	.29
van	.00	.00	.64	.34
bus/lorry	.00	.00	.00	.31
count	39	272	42	35

	bike	car/taxi	van	bus/lorry
R_j	78.0%	73.9%	56.3%	31.4%
P_j	37.9%	80.9%	62.8%	91.7%
R_{Cj}	100.0%	96.7%	64.3%	31.4%
P_{Cj}	76.5%	92.0%	67.5%	100.0%
R_{Dj}	78.0%	76.4%	87.5%	100.0%
P_{Dj}	49.5%	88.0%	93.0%	91.7%

Table 37 Classifier confusion matrix and class wise performance for the 3DHOG classifier with patch size of $\delta = 0.8\,\mathrm{m}$ at resolution $\rho = 20\,\mathrm{Pixel/m}$

ground truth / *detection*	bike	car/taxi	van	bus/lorry	FP
bike	.91	.02	.02	.11	.98
car/taxi	.00	.66	.20	.44	.10
van	.00	.02	.66	.11	.05
bus/lorry	.00	.00	.00	.33	.06
FN	.09	.30	.11	.00	.00
count	43	323	44	18	
overlap	.67	.67	.76	.85	

Symbol	*Value*
Recall R	66.8%
Precision P	72.0%
Classifier P_C	89.1%
Detector R_D	75.0%
Detector P_D	80.9%
GT Overlap	0.68

Table 38 Confusion matrix and system performance for the 3DHOG classifier with patch size of $\delta = 0.5\,\mathrm{m}$ at resolution $\rho = 16\,\mathrm{Pixel/m}$

ground truth / *detection*	bike	car/taxi	van	bus/lorry
bike	1.00	.03	.03	.11
car/taxi	.00	.94	.23	.44
van	.00	.03	.74	.11
bus/lorry	.00	.00	.00	.33
count	39	225	39	18

	bike	car/taxi	van	bus/lorry
R_j	90.7%	65.6%	65.9%	33.3%
P_j	42.9%	81.5%	74.4%	85.7%
R_{Cj}	100.0%	94.2%	74.4%	33.3%
P_{Cj}	79.6%	92.6%	78.4%	100.0%
R_{Dj}	90.7%	69.7%	88.6%	100.0%
P_{Dj}	53.8%	88.1%	94.9%	85.7%

Table 39 Classifier confusion matrix and class wise performance for the 3DHOG classifier with patch size of $\delta = 0.5\,\mathrm{m}$ at resolution $\rho = 16\,\mathrm{Pixel/m}$

References

(Acunzo *et al.*, 2007) Acunzo, D., Zhu, Y., Xie, B., and Baratoff, G. (2007). Context-adaptive approach for vehicle detection under varying lighting conditions. In *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, pages 654–660.

(Agarwal *et al.*, 2004) Agarwal, S., Awan, A., and Roth, D. (2004). Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1475–1490.

(Alonso *et al.*, 2007) Alonso, D., Salgado, L., and Nieto, M. (2007). Robust vehicle detection through multidimensional classification for on board video based systems. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, volume 4, pages 321–324.

(Atev *et al.*, 2005) Atev, S., Arumugam, H., Masoud, O., Janardan, R., and Papanikolopoulos, N. (2005). A vision-based approach to collision prediction at traffic intersections. *Intelligent Transportation Systems, IEEE Transactions on*, 6(4):416–423.

(Atev and Papanikolopoulos, 2008) Atev, S. and Papanikolopoulos, N. (2008). Multi-view 3d vehicle tracking with a constrained filter. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 2277–2282.

(Autoscope, nd) Autoscope (n.d.). http://www.autoscope.com (accessed 8th December 2009).

(Bardet and Chateau, 2008) Bardet, F. and Chateau, T. (2008). Mcmc particle filter for real-time visual tracking of vehicles. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pages 539–544.

(Bashir and Porikli, 2006) Bashir, F. and Porikli, F. (2006). Performance evaluation of object detection and tracking systems. In *IEEE Int. W. on Performance Evaluation of Tracking and Surveillance, PETS'06*, pages 7–14.

(Bay *et al.*, 2006) Bay, H., Tuytelaars, T., and Gool, L. V. (2006). Surf: Speeded up robust features. In *European Conference on Computer Vision (ECCV)*, volume 3951 of *LNCS*, pages 404–17.

(Bloisi and Iocchi, 2009) Bloisi, D. and Iocchi, L. (2009). Argos–a video surveillance system for boat traffic monitoring in venice. *To appear in International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI). 2009.*

(Boninsegna and Bozzoli, 2000) Boninsegna, M. and Bozzoli, A. (2000). A tunable algorithm to update a reference image. *Signal Processing: Image Communication*, 16(4):353 – 365.

(Boykov and Veksler, 2005) Boykov, Y. and Veksler, O. (2005). *Mathematical Models in Computer Vision: The Handbook*, chapter Graph Cuts in Vision and Graphics: Theories and Applications, pages 100–119. Springer.

(Brown *et al.*, 2005) Brown, L. M., Senior, A. W., li Tian, Y., Connell, J., Hampapur, A., Shu, C.-F., Merkl, H., and Lu, M. (2005). Performance evaluation of surveillance systems under varying conditions. In *IEEE Int'l Workshop on Performance Evaluation of Tracking and Surveillance*, pages 1–8, Colorado.

(Chen and Zhang, 2007) Chen, X. and Zhang, C. C. (2007). Vehicle classification from traffic surveillance videos at a finer granularity. In *Advances In Multimedia Modeling*, volume 4351 of *Lecture Notes in Computer Science*, pages 772–781. Springer.

(Chen *et al.*, 2007) Chen, Y.-T., Chen, C.-S., Huang, C.-R., and Hung, Y.-P. (2007). Efficient hierarchical method for background subtraction. *Pattern Recognition*, 40(10):2706 – 2715.

(Ciciora *et al.*, 2004) Ciciora, W., Farmer, J., and Adams, M. (2004). *Modern cable television technology: video, voice, and data communications*. Morgan Kaufmann.

(Citilog, nd) Citilog (n.d.). http://www.citilog.com (accessed 8th December 2009).

(CLEAR, 2007) CLEAR (2007). Clear classification of events, activities and relationships evaluation and workshop. http://isl.ira.uka.de/clear07/ (accessed 16th November 2009).

(Colombo *et al.*, 2007) Colombo, A., Leung, V., Orwell, J., and Velastin, S. (2007). Markov models of periodically varying backgrounds for change detection. In *Visual Information Engineering 2007*, London. IET.

(Cornelis *et al.*, 2008) Cornelis, N., Leibe, B., Cornelis, K., and Gool, L. V. (2008). 3d urban scene modeling integrating recognition and reconstruction. *International Journal of Computer Vision*, 78(2-3):121–141.

(Cracknell, 2007)Cracknell, M. (2007). Image detection in the real world – interactive session. In *Intelligent Transportation Systems ITS '07 Aalborg*.

(Cracknell, 2008)Cracknell, M. (2008). Image detection in the real world – a progress update. In *Intelligent Transportation Systems World Congress ITS WC 2008 – New York*.

(Crandall *et al.*, 2005) Crandall, D., Felzenszwalb, P., and Huttenlocher, D. (2005). Spatial priors for part-based recognition using statistical models. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 10–17.

(Creusen *et al.*, 2009) Creusen, I., Wijnhoven, R., and de With, P. (2009). Applying feature selection techniques for visual dictionary creation in object classification. In *Proc. Int. Conf. on Image Processing, Computer Vision and Pattern Recognition (IPCV)*, pages 722–727.

(CRS, nd) CRS (n.d.). Computer recognition systems. http://www.crs-traffic.co.uk/ (accessed 8th December 2009).

(Cucchiara *et al.*, 2001) Cucchiara, R., Grana, C., Piccardi, M., Prati, A., and Sirotti, S. (2001). Improving shadow suppression in moving object detection with hsv color information. In *Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE*, pages 334–339.

(Culibrk *et al.*, 2009) Culibrk, D., Antic, B., and Crnojevic, V. (2009). Real-time stable texture regions extraction for motion-based object segmentation. In *British Machine Vision Conference (BMVC)*, London.

(Dahlkamp *et al.*, 2006) Dahlkamp, H., Ottlik, A., and Nagel, H.-H. (2006). Comparison of edge-driven algorithms for model-based motion estimation. In *First International Workshop on Spatial Coherence for Visual Motion Analysis SCVMA*, volume 3667 of *Lecture Notes in Computer Science*, pages 38–50.

(Dahlkamp *et al.*, 2004) Dahlkamp, H., Pece, A. E., Ottlik, A., and Nagel, H.-H. (2004). Differential analysis of two model-based vehicle tracking approaches. In *Pattern Recognition*, volume 3175 of *Lecture Notes in Computer Science*, pages 71–78.

(Dalal and Triggs, 2005) Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893.

(Dalal *et al.*, 2006) Dalal, N., Triggs, B., and Schmid, C. (2006). Human detection using oriented histograms of flow and appearance. In *ECCV 2006*, pages 428–441.

(Davies and Lienhart, 2006) Davies, B. and Lienhart, R. (2006). Using cart to segment road images. In *Proceedings of SPIE Multimedia Content Analysis, Management and Retrival.*

(Doucet and Johansen, 2009) Doucet, A. and Johansen, A. (2009). A tutorial on particle filtering and smoothing: Fifteen years later. *The Oxford Handbook of Nonlinear Filtering, Oxford University Press. To appear.*

(Dunn and Parberry, 2002) Dunn, F. and Parberry, I. (2002). *3D math primer for graphics and game development*. Wordware Publishing.

(Everingham *et al.*, 2009) Everingham, M., Gool, L. V., Williams, C. K. I., Winn, J., and Zisserman, A. (2009). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision (in press)*.

(Freund, 1995) Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Information and Computing*, 121(2):256–285.

(Freund and Schapire, 1999) Freund, Y. and Schapire, R. E. (1999). A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780.

(Gandhi and Trivedi, 2007) Gandhi, T. and Trivedi, M. M. (2007). Video based surround vehicle detection, classification and logging from moving platforms: Issues and approaches. In *Intelligent Vehicles Symposium, 2007 IEEE*, pages 1067–1071.

(Gao *et al.*, 2009a) Gao, T., Liu, Z., Gao, W., and Zhang, J. (2009a). Moving vehicle tracking based on sift active particle choosing. In *Advances in Neuro-Information Processing in LNCS*, volume 5507 of *Lecture Notes in Computer Science*, pages 695–702. Springer.

(Gao *et al.*, 2009b) Gao, T., Liu, Z., Gao, W., and Zhang, J. (2009b). A robust technique for background subtraction in traffic video. In *Advances in Neuro-Information Processing*, volume 5507 of *Lecture Notes in Computer Science*, pages 736–744. Springer.

(Gordon *et al.*, 1993) Gordon, N., Salmond, D., and Smith, A. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107–113.

(Guha *et al.*, 2006) Guha, P., Mukerjee, A., and Venkatesh, K. (2006). Appearance based multiple agent tracking under complex occlusions. In *PRICAI 2006: Trends in Artificial Intelligence*, volume 4099 of *Lecture Notes in Computer Science*, pages 593–602. Springer.

(Guo *et al.*, 2008) Guo, Y., Rao, C., Samarasekera, S., Kim, J., Kumar, R., and Sawhney, H. (2008). Matching vehicles under large pose transformations using approximate 3d models and piecewise mrf model. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8.

(Gupte *et al.*, 2002) Gupte, S., Masoud, O., Martin, R., and Papanikolopoulos, N. (2002). Detection and classification of vehicles. *Intelligent Transportation Systems, IEEE Transactions on*, 3(1):37–47.

(Heikkila and Pietikainen, 2006) Heikkila, M. and Pietikainen, M. (2006). A texture-based method for modeling the background and detecting moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):657–662.

(Hsieh *et al.*, 2006) Hsieh, J. W., Yu, S. H., Chen, Y. S., and Hu, W. F. (2006). Automatic traffic surveillance system for vehicle tracking and classification. *IEEE Transactions On Intelligent Transportation Systems*, 7(2):175–187.

(Hu *et al.*, 2004) Hu, W., Xiao, X., Xie, D., Tan, T., and Maybank, S. (2004). Traffic accident prediction using 3-d model-based vehicle tracking. *Vehicular Technology, IEEE Transactions on*, 53(3):677–694.

(Huang and Liao, 2004) Huang, C. L. and Liao, W. C. (2004). A vision-based vehicle identification system. In *Proceedings Of The 17th International Conference On Pattern Recognition, Vol 4*, pages 364–367.

(iLIDS, nd)Home Office Scientific Development Branch, (n.d.). Imagery library for intelligent detection systems i-lids. http://www.ilids.co.uk/ (accessed 4 September 2009).

(Ipsotek, nd) Ipsotek (n.d.). http://www.ipsotek.com/ (accessed 8th December 2009).

(Isard and Blake, 1998)Isard, M. and Blake, A. (1998). Condensation—conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28.

(Johansson *et al.*, 2009) Johansson, B., Wiklund, J., Forssén, P.-E., and Granlund, G. (2009). Combining shadow detection and simulation for estimation of vehicle size and position. *Pattern Recognition Letters*, 30(8):751 – 759.

(Jones and Snow, 2008) Jones, M. and Snow, D. (2008). Pedestrian detection using boosted features over many frames. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4.

(KadewTraKuPong and Bowden, 2001) KadewTraKuPong, P. and Bowden, R. (2001). An improved adaptive background mixture model for real-time tracking with shadow detection. In *Proceedings of 2nd European Workshop on Advanced Video-Based Surveillance Systems*.

(Kalman, 1960) Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45.

(Kamijo *et al.*, 2004) Kamijo, S., Harada, M., and Sakauchi, M. (2004). Incident detection based on semantic hierarchy composed of the spatio-temporal mrf model and statistical reasoning. In *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, volume 1, pages 415–421.

(Kamijo *et al.*, 2001a) Kamijo, S., Ikeuchi, K., and Sakauchi, M. (2001a). Event recognitions from traffic images based on spatio-temporal markov random field model. In *8th World Congress on Inteligent Transportation Systems*.

(Kamijo *et al.*, 2001b) Kamijo, S., Ikeuchi, K., and Sakauchi, M. (2001b). Vehicle tracking in low-angle and front-view images based on spatio-temporal markov random field model. In *8th World Congress on ITS*.

(Kamijo *et al.*, 2000) Kamijo, S., Matsushita, Y., Ikeuchi, K., and Sakauchi, M. (2000). Traffic monitoring and accident detection at intersections. *Intelligent Transportation Systems, IEEE Transactions on*, 1(2):108–118.

(Kamijo and Sakauchi, 2002)Kamijo, S. and Sakauchi, M. (2002). Classification of traffic events based on the spatio-temporal mrfmodel and the bayesian network. In *9th World Congress on ITS*.

(Kanhere, 2008) Kanhere, N. (2008). *Vision-based Detection, Tracking and Classification of Vehicles using Stable Features with Automatic Camera Calibration*. PhD thesis, Clemson University, USA.

(Kanhere and Birchfield, 2008) Kanhere, N. and Birchfield, S. (2008). Real-time incremental segmentation and tracking of vehicles at low camera angles using stable features. *Intelligent Transportation Systems, IEEE Transactions on*, 9(1):148–160.

(Kanhere *et al.*, 2005) Kanhere, N., Pundlik, S., and Birchfield, S. (2005). Vehicle segmentation and tracking from a low-angle off-axis camera. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 1152–1157.

(Kastrinaki *et al.*, 2003) Kastrinaki, V., Zervakis, M., and Kalaitzakis, K. (2003). A survey of video processing techniques for traffic applications. *Image and Vision Computing*, 21(4):359 – 381.

(Khammari *et al.*, 2005) Khammari, A., Nashashibi, F., Abramson, Y., and Laurgeau, C. (2005). Vehicle detection combining gradient analysis and adaboost classification. In *Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE*, pages 66–71.

(Kim and Malik, 2003) Kim, Z. and Malik, J. (2003). Fast vehicle detection with probabilistic feature grouping and its application to vehicle tracking. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, volume 1, pages 524–531.

(Kläser *et al.*, 2008) Kläser, A., Marszalek, M., and Schmid, C. (2008). A spatio-temporal descriptor based on 3d-gradients. In *British Computer Vision Conference BMVC 2008*, volume 2, pages 995 – 1004.

(Kumar *et al.*, 2008) Kumar, P., Mittal, A., and Kumar, P. (2008). Study of robust and intelligent surveillance in visible and multi-modal framework. *Informatica*, 32:63–77.

(Kumar *et al.*, 2003) Kumar, P., Ranganath, S., and Huang, W. M. (2003). Bayesian network based computer vision algorithm for traffic monitoring using video. *IEEE Intelligent Transportation Systems Proceedings, Vols. 1 & 2*, pages 897–902.

(Lazarevic-McManus *et al.*, 2007) Lazarevic-McManus, N., Renno, J., Makris, D., and Jones, G. (2007). An object-based comparative methodology for motion detection based on the f-measure. *Computer Vision and Image Understanding, Sp. Is. on Intelligent Visual Surveillance*, pages 74–85.

(Leibe *et al.*, 2007) Leibe, B., Cornelis, N., Cornelis, K., and Van Gool, L. (2007). Dynamic 3d scene analysis from a moving vehicle. In *Computer Vision and Pattern Recognition. CVPR '07. IEEE Conference on*, pages 1–8.

(Leibe *et al.*, 2004) Leibe, B., Leonardis, A., and Schiele, B. (2004). Combined object categorization and segmentation with an implicit shape model. In *ECCV'04 Workshop on Statistical Learning in Computer Vision*, pages 17–32.

(Leibe *et al.*, 2008a) Leibe, B., Leonardis, A., and Schiele, B. (2008a). Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision Special Issue on Learning for Recognition and Recognition for Learning*, 77(1-3):259–289.

(Leibe *et al.*, 2008b) Leibe, B., Schindler, K., Cornelis, N., and Van Gool, L. (2008b). Coupled object detection and tracking from static cameras and moving vehicles. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(10):1683–1698.

(Leibe *et al.*, 2005) Leibe, B., Seemann, E., and Schiele, B. (2005). Pedestrian detection in crowded scenes. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 878–885.

(Liebelt *et al.*, 2008)Liebelt, J., Schmid, C., and Schertler, K. (2008). Viewpoint-independent object class detection using 3d feature maps. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8.

(Lou *et al.*, 2005)Lou, J., Tan, T., Hu, W., Yang, H., and Maybank, S. J. (2005). 3-d model-based vehicle tracking. *Image Processing, IEEE Transactions on*, 14(10):1561–1569.

(Lowe, 1999) Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *International Conference on Computer Vision 1999, ICCV 1999*, volume 02, page 1150–1157, Los Alamitos, CA, USA. IEEE Computer Society.

(Ma and Grimson, 2005) Ma, X. and Grimson, W. (2005). Edge-based rich representation for vehicle classification. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1185–1192.

(MacKay, 2003) MacKay, D. J. (2003). *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, version 7.2 (fourth printing) march 28, 2005 edition.

(Martel-Brisson and Zaccarin, 2007) Martel-Brisson, N. and Zaccarin, A. (2007). Learning and removing cast shadows through a multidistribution approach. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(7):1133–1146.

(Masoud and Papanikolopoulos, 2007) Masoud, O. and Papanikolopoulos, N. P. (2007). Using geometric primitives to calibrate traffic scenes. *Transportation Research Part C: Emerging Technologies*, 15(6):361 – 379.

(Mauthner *et al.*, 2008) Mauthner, T., Donoser, M., and Bischof, H. (2008). Robust tracking of spatial related components. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4.

(Messelodi *et al.*, 2005a) Messelodi, S., Modena, C. M., Segata, N., and Zanin, M. (2005a). A kalman filter based background updating algorithm robust to sharp illumination changes. In Roli, F. and Vitulano, S., editors, *Lecture Notes in Computer Science*, volume 3617, pages 163–170. Springer Berlin / Heidelberg. Proceedings of the 13th International Conference on Image Analysis and Processing.

(Messelodi *et al.*, 2005b) Messelodi, S., Modena, C. M., and Zanin, M. (2005b). A computer vision system for the detection and classification of vehicles at urban road intersections. *Pattern Analysis & Applications*, 8(1-2):17–31.

(Mikolajczyk and Schmid, 2005) Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630.

(Monnet *et al.*, 2003) Monnet, A., Mittal, A., Paragios, N., and Ramesh, V. (2003). Background modeling and subtraction of dynamic scenes. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1305–1312.

(Morris and Trivedi, 2006a) Morris, B. and Trivedi, M. (2006a). Improved vehicle classification in long traffic video by cooperating tracker and classifier modules. In *AVSS '06: Proceedings of the IEEE International Conference on Video and Signal Based Surveillance*, page 9, Washington, DC, USA. IEEE Computer Society.

(Morris and Trivedi, 2006b) Morris, B. and Trivedi, M. (2006b). Robust classification and tracking of vehicles in traffic video streams. In *Intelligent Transportation Systems Conference. ITSC '06. IEEE*, pages 1078–1083.

(Muller *et al.*, 2001) Muller, K.-R., Mika, S., Ratsch, G., Tsuda, K., and Scholkopf, B. (2001). An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–202.

(Nagel, nd) Nagel, H.-H. (n.d.). Image sequence server. http://i21www.ira.uka.de/image_sequences/ (accessed 13th November 2009).

(Needham and Boyle, 2003) Needham, C. and Boyle, R. (2003). Performance evaluation metrics and statistics for positional tracker evaluation. In *International Conference on Computer Vision Systems, ICVS'03*, pages 278–289, Graz, Austria.

(Nguyen and Le, 2008) Nguyen, P. and Le, H. (2008). A multi-modal particle filter based motorcycle tracking system. In *PRICAI 2008: Trends in Artificial Intelligence in LNCS*, volume 5351, pages 819–828. Springer.

(Nummiaro *et al.*, 2003) Nummiaro, K., Koller-Meier, E., and Gool, L. V. (2003). An adaptive color-based particle filter. *Image and Vision Computing*, 21(1):99 – 110.

(NXP, nd) NXP (n.d.). Trimedia media processor. www.nxp.com/pip/PNX17XX_SER_N_1.html (accessed 4th September 2009).

(Opelt, 2006) Opelt, A. (2006). *Generic Object Recognition*. PhD thesis, Graz University of Technology, Austria.

(Opelt *et al.*, 2006a) Opelt, A., Pinz, A., Fussenegger, M., and Auer, P. (2006a). Generic object recognition with boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3):416–431.

(Opelt *et al.*, 2006b) Opelt, A., Pinz, A., and Zisserman, A. (2006b). A boundary-fragment-model for object detection. In *Proceedings of the European Conference on Computer Vision*, pages 575–588. Springer-Verlag Berlin Heidelberg.

(Opelt *et al.*, 2006c) Opelt, A., Pinz, A., and Zisserman, A. (2006c). Incremental learning of object detectors using a visual shape alphabet. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 3–10, Los Alamitos, CA, USA. IEEE Computer Society.

(OpenCV, nd) OpenCV (n.d.). Open source computer vision library. http://sourceforge.net/projects/opencvlibrary (accessed 19 December 2008).

(Ottlik and Nagel, 2008) Ottlik, A. and Nagel, H. (2008). Initialization of model-based vehicle tracking in video sequences of inner-city intersections. *International Journal of Computer Vision*, 80(2):211–225.

(Park *et al.*, 2007) Park, K., Lee, D., and Park, Y. (2007). Video-based detection of street-parking violation. In *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition CVPR 2007.*

(Pingkun *et al.*, 2007) Pingkun, Y., Khan, S., and Shah, M. (2007). 3d model based object class detection in an arbitrary view. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–6.

(Pinz, 2005) Pinz, A. (2005). *Foundations and Trends in Computer Graphics and Vision*, volume 1, chapter Object Categorization, pages 255–353. nowpublishers.com.

(Power and Schoonees, 2002)Power, P. W. and Schoonees, J. A. (2002). Understanding background mixture models for foreground segmentation. In *Proceedings Image and Vision Computing New Zealand 2002*, pages 267–271.

(Prati *et al.*, 2003) Prati, A., Miki, I., Cucchiara, R., and Trivedi, M. M. (2003). Comparative evaluation of moving shadow detection algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:918–923.

(project PASCAL, nd) project PASCAL (n.d.). The pascal visual object classes homepage. http://pascallin.ecs.soton.ac.uk/challenges/VOC/ (accessed 3rd November 2009).

(Rad and Jamzad, 2005) Rad, R. and Jamzad, M. (2005). Real time classification and tracking of multiple vehicles in highways. *Pattern Recognition Letters*, 26(10):1597–1607.

(Remagnino *et al.*, 1998) Remagnino, P., Maybank, S., Fraile, R., Baker, K., and Morris, R. (1998). *'Advanced Video-based Surveillance Systems'*, chapter Automatic Visual Surveillance of Vehicles and People, pages 97–107. Hingham, MA., USA.

(Robert, 2009a) Robert, K. (2009a). Night-time traffic surveillance: A robust framework for multi-vehicle detection, classification and tracking. In *Advanced Video and Signal Based Surveillance, IEEE Conference on*, pages 1–6, Los Alamitos, CA, USA. IEEE Computer Society.

(Robert, 2009b) Robert, K. (2009b). Video-based traffic monitoring at day and night time. In *12th International IEEE Conference on Intelligent Transportation Systems.*

(Saunier and Sayed, 2006)Saunier, N. and Sayed, T. (2006). A feature-based tracking algorithm for vehicles in intersections. In *Computer and Robot Vision, 2006. The 3rd Canadian Conference on*, page 59.

(Serre *et al.*, 2007) Serre, T., Wolf, L., Bileschi, S., Riesenhuber, M., and Poggio, T. (2007). Robust object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):411–426.

(Shafique and Shah, 2005) Shafique, K. and Shah, M. (2005). A noniterative greedy algorithm for multiframe point correspondence. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(1):51–65.

(Sheikh and Shah, 2005) Sheikh, Y. and Shah, M. (2005). Bayesian modeling of dynamic scenes for object detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(11):1778–1792.

(Shotton *et al.*, 2009) Shotton, J., Winn, J., Rother, C., and Criminisi, A. (2009). Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *International Journal of Computer Vision*, 81(1):2–23.

(Song and Tai, 2008) Song, K.-T. and Tai, J.-C. (2008). Real-time background estimation of traffic imagery using group-based histogram. *Journal of Information Science and Engineering*, 24(2):411–423.

(Song and Nevatia, 2007) Song, X. and Nevatia, R. (2007). Detection and tracking of moving vehicles in crowded scenes. In *Motion and Video Computing. WMVC '07. IEEE W. on*, page 4.

(Starck and Hilton, 2005) Starck, J. and Hilton, A. (2005). Spherical matching for temporal correspondence of non-rigid surfaces. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1387–1394.

(Stauffer and Grimson, 1999) Stauffer, C. and Grimson, W. (1999). Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, pages 246–252.

(Stauffer and Grimson, 2000) Stauffer, C. and Grimson, W. (2000). Learning patterns of activity using real-time tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):747–757.

(Sturgess *et al.*, 2009) Sturgess, P., Alahari, K., Ladicky, L., and Torr, P. H. (2009). Combining appearance and structure from motion features for road scene understanding. In *British Machine Vision Conference*.

(Su *et al.*, 2007) Su, X., Khoshgoftaar, T., Zhu, X., and Folleco, A. (2007). Rule-based multiple object tracking for traffic surveillance using collaborative background extraction. In *Advances in Visual Computing*, volume 4842 of *Lecture Notes in Computer Science*, pages 469–478. Springer.

(Sullivan *et al.*, 1996) Sullivan, G. D., Baker, K. D., Worrall, A. D., Attwood, C. I., and Remagnino, P. R. (1996). Model-based vehicle detection and classification using orthographic approximations. In *Proceedings of 7th British Machine Vision Conference*, volume 2, pages 695–704.

(Sun *et al.*, 2004) Sun, Z., Bebis, G., and Miller, R. (2004). On-road vehicle detection using optical sensors: a review. In *Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on*, pages 585–590.

(Sun *et al.*, 2006) Sun, Z., Bebis, G., and Miller, R. (2006). On-road vehicle detection: a review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(5):694–711.

(Taj *et al.*, 2008) Taj, M., Maggio, E., and Cavallaro, A. (2008). Objective evaluation of pedestrian and vehicle tracking on the clear surveillance dataset. *Lecture Notes In Computer Science*, 4625:160–173.

(Takala and Pietikainen, 2007) Takala, V. and Pietikainen, M. (2007). Multi-object tracking using color, texture and motion. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–7.

(Tan *et al.*, 1998) Tan, T., Sullivan, G., and Baker, K. (1998). Model-based localisation and recognition of road vehicles. *International Journal of Computer Vision*, 27(1):5–25.

(Tanaka *et al.*, 2007) Tanaka, T., Shimada, A., Arita, D., and ichiro Taniguchi, R. (2007). A fast algorithm for adaptive background model construction using parzen density estimation. In *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on*, pages 528–533.

(Thi *et al.*, 2008) Thi, T. H., Robert, K., Lu, S., and Zhang, J. (2008). Vehicle classification at nighttime using eigenspaces and support vector machine. In *Image and Signal Processing, 2008. CISP '08. Congress on*, volume 2, pages 422–426.

(Torr, 2007) Torr, P. H. S. (2007). Graph cuts and their use in computer vision. In *International Computer Vision Summer School 2007.*

(Traficon, nd) Traficon (n.d.). http://www.traficon.com (accessed 8th December 2009).

(Tsai, 1986) Tsai, R. Y. (1986). An efficient and accurate camera calibration technique for 3d machine vision. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition (CVPR), (1986)*, pages 364–374.

(Ullman, 2007) Ullman, S. (2007). Object recognition and segmentation by a fragment-based hierarchy. *Trends in Cognitive Sciences*, 11(2):58–64.

(Valera and Velastin, 2005) Valera, M. and Velastin, S. (2005). Intelligent distributed surveillance systems: a review. *Vision, Image and Signal Processing, IEE Proceedings* -, 152(2):192–204.

(van der Maaten *et al.*, 2009) van der Maaten, L., Postma, E., and van den Herik, H. (2009). Dimensionality reduction: A comparative review. *Submitted to Journal of Machine Learning Research.*

(Veeraraghavan *et al.*, 2002) Veeraraghavan, H., Masoud, O., and Papanikolopoulos, N. (2002). Vision-based monitoring of intersections. In *IEEE 5th International Conference On Intelligent Transportation Systems, Proceedings*, pages 7–12.

(Viola and Jones, 2004) Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International Journal Of Computer Vision*, 57(2):137–154.

(Viper,) Viper. ground truth schema. http://viper-toolkit.sourceforge.net/ (accessed 19 December 2008).

(Virage, nd) Virage (n.d.). http://www.virage.com/ (accessed 8th December 2009).

(VISOR, nd) VISOR (n.d.). Video surveillance online repository. http://www.openvisor.org/ (accessed 25th November 2009).

(Wang *et al.*, 2009a)Wang, H., Ullah, M. M., Klaser, A., Laptev, I., and Schmid, C. (2009a). Evaluation of local spatio-temporal features for action recognition. In *British Machine Vision Conference*, London.

(Wang *et al.*, 2006) Wang, J., Bebis, G., and Miller, R. (2006). Robust video-based surveillance by integrating target detection with tracking. In *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW '06. Conference on*, pages 137–137.

(Wang *et al.*, 2004) Wang, J., Chung, Y., Lin, S., Chang, S., Cherng, S., and Chen, S. (2004). Vision-based traffic measurement system. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 4, pages 360–363.

(Wang *et al.*, 2009b) Wang, J., Ma, Y., Li, C., Wang, H., and Liu, J. (2009b). An efficient multi-object tracking method using multiple particle filters. In *Computer Science and Information Engineering, 2009 WRI World Congress on*, volume 6, pages 568–572.

(Ward, 1963) Ward, J. H. J. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244.

(Wijnhoven *et al.*, 2008) Wijnhoven, R., de With, P., and Creusen, I. (2008). Efficient template generation for object classification in video surveillance. In *Proc. of 29th Symposium on Information Theory in the Benelux*, page 255–262.

(Wijnhoven and de With, 2007) Wijnhoven, R. G. J. and de With, P. H. N. (2007). Experiments with patch-based object classification. In *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on*, pages 105–110.

(Wijnhoven and de With, 2009) Wijnhoven, R. G. J. and de With, P. H. N. (2009). Comparing feature matching for object categorization in video surveillance. In *Advanced Concepts for Intelligent Vision Systems*, volume 5807 of *Lecture Notes in Computer Science*, pages 410–421.

(Woodford *et al.*, 2009) Woodford, O. J., Rother, C., and Kolmogorov, V. (2009). A global perspective on map inference for low-level vision. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2319–2326.

(Wu and Nevatia, 2005) Wu, B. and Nevatia, R. (2005). Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 90–97 Vol. 1.

(Yin *et al.*, 2007) Yin, F., Makris, D., and Velastin, S. A. (2007). Performance evaluation of object tracking algorithms. In *10th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, PETS'07*, Rio de Janeiro.

(Yoneyama *et al.*, 2005) Yoneyama, A., Yeh, C.-H., and Kuo, C.-C. J. (2005). Robust vehicle and traffic information extraction for highway surveillance. *EURASIP J. Appl. Signal Process.*, 2005:2305–2321.

(Zhang *et al.*, 2008a) Zhang, D., Qu, S., and Liu, Z. (2008a). Robust classification of vehicle based on fusion of tsrp and wavelet fractal signature. In *Networking, Sensing and Control, 2008. ICNSC 2008. IEEE International Conference on*, pages 1788–1793.

(Zhang *et al.*, 2007a) Zhang, G., Avery, R. P., and Wang, Y. (2007a). Video-based vehicle detection and classification system for real-time traffic data collection using uncalibrated video cameras. *Transportation Research Record*, 1993:138–147.

(Zhang *et al.*, 2007b) Zhang, J., Marszalek, M., Lazebnik, S., and Schmid, C. (2007b). Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision*, 73(2):213–238.

(Zhang and Tan, 2002) Zhang, J. and Tan, T. (2002). Brief review of invariant texture analysis methods. *Pattern Recognition*, 35(3):735–747.

(Zhang *et al.*, 2008b) Zhang, W., Wu, Q., Yang, X., and Fang, X. (2008b). Multilevel framework to detect and handle vehicle occlusion. *Intelligent Transportation Systems, IEEE Transactions on*, 9(1):161–174.

(Zhang *et al.*, 2005) Zhang, W., Yu, B., Zelinsky, G., and Samaras, D. (2005). Object class recognition using multiple layer boosting with heterogeneous features. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 323–330.

(Zhang *et al.*, 2008c) Zhang, Z., Li, M., Huang, K., and Tan, T. (2008c). Boosting local feature descriptors for automatic objects classification in traffic scene surveillance. *International Conference on Pattern Recognition (ICPR) 2008.*

(Zheng *et al.*, 2005) Zheng, J., Wang, Y., Nihan, N. L., and Hallenbeck, M. E. (2005). Extracting roadway background image: Mode-based approach. *Transportation Research Record*, (1944):82–88.

Published Papers

All papers published during the project are appended in chronological order. A list of references is provided in section 7.4 on page 187.

The Published Papers after page 227 have not been digitised at the request of the university.