Qualitative and Structural Analysis of Video Sequences

by

Alessio Brits

Submitted in Fulfilment of the

Academic Requirements for the degree of

Master of Science

in the

School of Computer Science

University of KwaZulu-Natal

Westville Campus, South Africa

May 2011

As the candidate's supervisor, I have/have not approved this Thesis for Submission

Signed:	Name:	Date:

Abstract

This thesis analyses videos in two distinct ways so as to improve both human understanding and the computer description of events that unfold in video sequences. Qualitative analysis can be used to understand a scene in which many details are not needed. However, for there to be an accurate interpretation of a scene, a computer system has to first evaluate – discretely– the events in a scene. Such a method must involve structural features and the shapes of the objects in the scene.

In this thesis we perform qualitative analysis on a road scene and generate terms that can be understood by humans and that describe the status of the traffic and its congestion. Areas in the video that contain vehicles are identified regardless of scale. The movement of the vehicles is further identified and a rule-based technique is used to accurately determine the status of the traffic and its congestion.

Occlusion is a common problem in scene analysis tracking. A novel technique is developed to vertically separate groups of people in video sequences. A histogram is generated based on the shape of a group of people and its valleys are identified. A vertical seam for each valley is then detected using the intensity of the edges. This is then used as the separation boundary between the different individuals. This could definitely improve the tracking of people in a crowd.

Both techniques achieve good results, with the qualitative analysis accurately describing the status and congestion of a traffic scene, while the structural analysis can separate a group of people into distinctly separate persons.

Keywords

Video surveillance, Background subtraction, Occlusion detection, Segmentation, Scene analysis, Traffic monitoring

Preface

The experimental work described in this dissertation was carried out at the School of Computer Science at the University of KwaZulu-Natal, Westville Campus, from February 2008 to April 2011, under the supervision of Professor J.R. Tapamo.

These studies represent original work by the author and have not been otherwise submitted for any degree of diploma to any tertiary institution. When use has been made of the work of others, this is duly acknowledged within the text.

Declaration 1 - Plagiarism

I, Alessio Brits, declare that

1. The research reported in this dissertation, except where otherwise indicated, is my original research.
2. This dissertation has not been submitted for any degree or examination at any other university.
3. This dissertation does not contain another person's data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.
4. This dissertation does not contain another person's writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
   (a) Their words have been re-written but the general information attributed to them has been referenced.
   (b) Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.
5. This dissertation does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the dissertation and in the References section.

Signed: ....

Declaration 2 - Publications

THE DETAILS OF CONTRIBUTION TO PUBLICATIONS that form part and/or include research presented in this dissertation.

A. M. Brits and J. R. Tapamo, A shape and energy based approach to vertical people separation in video surveillance. G. Bebis et al. (Eds), ISVC 2009, Part II, LCNS 5,878, Springer-Verlag, pp. 345-356, 2009.

The authors' contributions to the publication are as follows:

First author: literature review; design and implementation of algorithms; exploration and testing of possible solutions; experimentation; data acquisition and analysis; and writing manuscripts.

Second author: critical feedback and discussion on ideas and concepts; explanation of mathematical models and theory; general guidance of first author's research effort; and editing of manuscripts.

Signed: ....
Name: Alessio M Brits
Date: ....

Acknowledgments

I would like to thank the following people for all their help with my research and my thesis:

First and foremost I would like to thank my supervisor, Professor Jules Tapamo, who provided me with excellent guidance; without him this thesis would be non-existent. His patience and dedication throughout the years showed me how to better myself as a researcher.

I would also like to thank the CSIR Defence Peace Safety and Security, as well as Armscor, whose funding helped this research become what it is.

I would like to thank my friends and family for their continued support throughout the entire time I was researching and writing this thesis.

Finally, I would like to thank Zane Mayo, who helped me considerably in proofreading my research paper. He also helped me refine my research ideas and showed me how to improve on them.

Contents

List of Figures

List of Tables

List of Abbreviations

CCTV	Closed-circuit television
CGI	Computer-generated imagery
EM Algorithm	Expectation-maximization algorithm
PCA	Principal component analysis

Chapter 1

Introduction

1.1 Context and Motivation

In today's technologically advanced society the impact of computers is so great that their use has become mandatory in almost every area of life. The incorporation of computers into our daily lives has happened at such a rapid rate that we have been brought to the point where their use can become overwhelming, and even inefficient, if one does not have a proper understanding as to what their use is and what they are supposed to accomplish.

Visual surveillance has become a virtual requirement for industrialised cities and even businesses. Many cities have closed-circuit television (CCTV) cameras located in numerous places throughout their quarters in order to prevent crime. The manner in which CCTV cameras help prevent crime is that they first of all serve as a deterrence (as they gather evidence that can aid the authorities in preventing further crimes and then they also aid in actively preventing crimes because the crimes are observed in real time (i.e. as they are happening) and can then be responded to by the authorities. Furthermore, it is not only the municipalities of cities that are introducing CCTV cameras into their work environment, but also various businesses, from banks to small grocery stores.

Crime prevention is not, however, the only reason for surveillance systems. Traffic cameras, which were first introduced to detect traffic violators, are increasingly being used to identify and monitor the status of traffic in real time. Video surveillance has become a widely used tool and it generates a considerable amount of data. With the popularity of cameras increasing, surveillance operators have more cameras to monitor at the same time, and the operators tend to have long shifts. Of importance to this study is the fact that the operators of these systems are being overwhelmed by their task. A single person working an eight-hour monitoring shift, which involves monitoring ten cameras simultaneously, has in essence eighty hours of footage to view. Moreover, almost all of these eighty hours of footage will be mundane and uneventful. The probability, therefore, that the person monitoring a set of security cameras will actually witness a crime decreases with time.

The effectiveness of surveillance systems therefore needs to be enhanced. We not only need computers to control these systems, but there is a large amount of data that needs to be processed as well. Computer systems thus need to incorporate automated and intelligent systems in order to interpret the phenomena contained within the data.

1.2 Problem Statement

This thesis discusses methods of performing scene analysis on video sequences. Scene analysis is a technique that involves extracting information from any given scene in a video sequence. Its function ranges from counting important objects to extracting and separating specific objects to even describing the entire scene as a whole. The domain of video sequences varies from simple CGI simulated videos to homemade videos and surveillance systems We will be focusing on a more functional domain, namely that of video surveillance. Video surveillance is commonly used to help better people's lives by working to monitor and prevent crime, and performing scene analysis on surveillance systems will therefore potentially help many people.

The different types of scene analysis that can be applied to surveillance systems can vary vastly from one another, each one performing their own specific task. In this thesis we will study qualitative and structural analysis of a scene.

Qualitative analysis is the process of extracting information from a given scene and processing it into a description that is easy for a person to comprehend. Instead of extracting a multitude of statistics that only an expert in the domain can understand, qualitative analysis supplies the casual observer with an understanding of a scene. Although qualitative analyses can be applied to any type of surveillance system, a casual observer does not need to know what is happening in many of the scenes as they are usually meant solely for safety operators. However, there is one particular domain where a casual observer could gather useful information from this type of analysis, and that is in the domain of traffic surveillance.

In traffic surveillance a camera is positioned in such a way as to monitor a stretch of road. The information gathered by this camera can be used to send vital information to first responders in case of an accident. The same camera can also provide the drivers on that particular stretch of road with helpful information. Modern highways, as well as some smaller roads, are increasingly being monitored by cameras, and many roads are also equipped with electronic notice boards. These electronic notice boards are used to inform drivers about the condition of that road. The condition of the road could refer to its traffic status (e.g. heavy and congested) or information about a particular event (e.g. an accident has occurred there). An automated computer system is needed, however, in order to decrease the delay between an incident occurring and it being reported on the electronic notice board. This system would need to collect relevant data from a traffic scene and process it in such a way that the necessary information can then be communicated to the drivers on that route (which would in turn enable drivers to make informed decisions about their trips). The messages relayed on the notice boards need to be simple, but at the same time express the status of the traffic in its entirety. Traffic needs to be identified as congested or otherwise. Furthermore, the data collected from a qualitative analysis could also be used to highlight certain areas of interest for those safety operators monitoring the same system. All this needs to be performed automatically, without any input from an operator, as that operator could well be monitoring multiple systems at once.

This thesis focuses on structural analysis of a scene. Structural analysis looks at the shape and texture of objects in the scene in order to extract detailed information about each object's structure. The result of the structural analysis that is done in our work will mainly be used in another system to help explain the observed scene. Structural analysis can be used in many systems, ranging from analysing the shape of a vehicle to comparing it to a database for identification purposes to understanding the shape of a person in order to segment them from other objects in the scene. In this thesis the structural analysis will be applied in the domain of indoor scenes that contain small groups of people in order to segment and separate each distinct person from those around them.

The indoor surveillance of people is used in many public and private venues, like shopping malls and government facilities. Such surveillance can be used to prevent crime or to observe areas so as to detect any safety violations. However, since people are dynamic and unpredictable, it is hard for a system to track a single person amongst a group of people. A person may move in an unpredictable manner or be occluded amongst other people and objects. The latter problem is a major concern and it is this problem that the present investigation tries to solve. Previous research has also tried to solve this problem. The most common method for solving this problem is to extract the features of each person and use them to identify that person when he/she has become occluded. The problem occurs when a person enters a scene while being occluded. This generally arises when a person is walking in the midst of a group of people and his/her unique features cannot be extracted in order to separate the individuals of the group. This is where a structural analysis can take place. By understanding the structure of a person and their texture we are able to find a method that can separate the people within a group. This structural analysis is integral because it does not require previous knowledge of each person, just an understanding of the average person's structure. With this understanding in place, further algorithms can use it and apply their feature extraction program in order extract unique features about each person, since they are already segmented.

The third problem this study attempts to address is the choice of background subtraction techniques. Before any type of analysis can take place of scenes that contain many objects (most of which are not relevant to the system), one must find a way of separating the useful objects from the rest of those in the scene. The most common method for accomplishing this is background subtraction. Background subtraction is a method by which the background of a scene is removed, thereby leaving only the relevant foreground objects. As this is integral in almost all video applications, various methods have been proposed. Each of the methods has advantages and disadvantages. The background subtraction algorithm needs to be accurate whilst also not taking a long time to process, especially when it is being used in real-time applications.

1.3 Contribution

The contribution of this thesis and its research is as follows:

- Algorithm design and modeling.
  - An extensive study was undertaken for the testing and evaluation of several background subtraction algorithms and techniques. The study yielded information as to which background subtraction algorithm was best suited for the two-scene analysis. All but one of the background subtraction algorithms were implemented in C++ using the OpenCV library. The excluded algorithm was instead implemented in Mathematica.
  - An adaptation of a qualitative analysis algorithm was designed and developed in order to facilitate a qualitative analysis of local traffic scenes. The qualitative analysis algorithm was fully implemented in C++ using

the OpenCV library. The qualitative analysis algorithm was then tested on and evaluated with regard to the footage of traffic scenes located on the N1(a national highway).

  - The design and development of a novel structural analysis algorithm was done in order to help solve the problem of occlusion in indoor surveillance. The structural analysis algorithm was fully implemented in C++ and made use of the OpenCV library. The algorithm was designed to be easily used in conjunction with other algorithms. The testing and evaluation of this algorithm was done on footage from the common data set CAVIAR.

- Publications. The work done towards this thesis has led to the following publication:

  - A. M. Brits and J. R. Tapamo.“A Shape and Energy Based Approach to Vertical People Separation in Video Surveillance,” in Proc. of the 5th Int. Symposium on Advances in Visual Computing: Part II (Las Vegas, USA, November 30 – December 2, 2009). [2]

1.4 Thesis Outline

.

This thesis will be structured as follows:

- Chapter 1: introduces the thesis and the problem to be addressed.
- Chapter 2 provides a literature review. This chapter is segmented into three distinct categories, which help to isolate the different areas of research that were undertaken. Them being: Background Subtraction, Occlusion Detection and Video Analysis.
- Chapter 3 addresses the problem of background subtraction. This chapter is separated into seven different sections, five of which provide a description and

explanation of different types of background subtraction, while one deals with their common problems and the last provides a comparison of the problems.

- Chapter 4 describes the types of scene analysis that was done. This chapter is separated into two distinct areas, namely qualitative and structural analyses. Their respective problems and solutions are described in detail.
- Chapter 5 offers the researchers' final conclusions with regard to the thesis, and then suggests future work that could complement this research.
- Appendix A provides further examples of the techniques discussed throughout the thesis.

Chapter 2

Literature Review

Computer vision is currently one of the fastest growing fields of research in computer science. The analysis of videos in particular has only recently become more common with the increase in processing power and the decrease in the price of cameras. This state of affairs has led to the design of many different techniques and methods for analysing and understanding scenes. Background subtraction is one of the key steps in video analysis, and this is introduced in the next section.

2.1 Background Subtraction

The identification and extraction of foreground regions in a video sequence is a key component in most computer vision applications. Detecting objects relevant to the phenomenon to be identified is fundamental in scene analysis. This problem has generated a considerable amount of research, and it is background modelling that has received most of the attention. What follows is a discussion of some of the techniques that have thus far been proposed in the literature.

2.1.1 Gaussian Modelling

One approach to background modelling is the use of statistical models to represent the behaviour of the grey level of a pixel. Wren et al. [29] model each pixel over time with a single Gaussian, as it has been hypothesised that, in any given scene, a single pixel will belong to the background much more often than it will belong to the foreground. The model for each pixel is updated at each frame. If the current pixel intensity value does not match that of its model which needs to be within two to three standard deviations), then that pixel can be classified as a foreground pixel. This model was found to be suitable for indoor scenes, but it has to have a decent initialisation period in order to produce accurate results. The model is continuously being updated so as to cater for slight and gradual lighting changes and subtle camera noise. However, this model performs poorly in more dynamic outdoor scenes or in general scenes that can have fast or sudden lighting changes as well as moving background objects (like trees blowing in the wind). This model can also underperform in scenes where the initial frames contain foreground objects, as the model will identify them as background. The clear disadvantage to using a single Gaussian model for each pixel is that the standard deviation can be large when one is comparing a pixel where no foreground object is detected with a pixel where some foreground is detected. Such a situation can cause multiple erroneous background classifications.

The background of a traffic scene uses three Gaussians in the Friedman and Russell model [7]. Friedman and Russell hypothesise that the majority of pixels in a traffic scene belong to one of the following three categories: a road, a shadow or an actual vehicle. Each Gaussian then represents one of those three categories. The mixture model is updated using the EM algorithm [3] and, once the model is sufficiently constructed, every following pixel can be classified according to the class that corresponds to the Gaussian that fits the pixel. This system is much more reliable than one using a single Gaussian for outdoor traffic scenes, but since it has been designed to only function in a traffic scene, it does not perform well in almost all other types of scenes, and the effectiveness of this system is thus reduced.

Stauffer and Grimson [28] have developed an adaptive method to model the background. This was done so as to ensure that errors in the background model do not accumulate over time. Like Friedman and Russell, Stauffer and Grimson have modelled each pixel with a mixture of Gaussians. Their model determines whether or not each pixel's intensity fits one of the Gaussians designated to be the background; if it does not, then that pixel is marked as foreground and a new Gaussian is created. Instead of specifying which Gaussians do or do not belong to the background, Stauffer and Grimson use a heuristic method for determining which Gaussians in the mixture do in fact belong to the background. The mixture of Gaussians approach is much more robust and can cater for more types of scenes than can the single Gaussian. For example, areas of an outdoor scene with moving trees can be accurately marked as background. In addition, lighting changes, such as those that are formed by moving clouds in an outdoor scene, can also be modelled accurately.

KaewTraKulPong and Bowden [11] have furthered the work done by Stauffer and Grimson by enhancing the online update algorithm as well as by adding a shadow suppression module to perform the background subtraction. KaewTraKulPong and Bowden's model ensures that the problems encountered by Stauffer and Grimson are reduced. The first problem that is corrected by their model is that of foreground objects that appear at the start of a scene. This problem has the effect of creating one Gaussian that incorrectly represents the background, and this error can only be corrected after a significant amount of time. This situation is worsened in crowded scenes. This difficulty can be overcome by using the online updating algorithm so as to use an estimate of the mixture of Gaussians before a length of time in the scene. The estimate reduces many of the errors that might occur at the beginning, while it is also able to reduce compound errors that occur after the initial phase. The second problem can be corrected by removing the shadows. Shadows in background subtraction can often be detected as foreground objects, thereby greatly increasing the area of the foreground objects. The existence of shadows can increase the number of Gaussians in the mixture. This can have the effect of wasting an entire Gaussian in the instances where a fixed number of Gaussians are reused in the mixture. Therefore, the removal of shadows can improve the foreground detection in outdoor scenes where shadows are more prominent and distinct.

There are many adaptations and improvements that can be applied to the mixture of Gaussians method of background subtraction. Traditionally, the maximum number of components in the mixture of Gaussians per pixel is usually given as a static value; however, in some cases, several pixels might require more Gaussians for an accurate representation, while other pixels require less. Zivkovic [31] presents an adaptive method for calculating the number of components in the mixture of Gaussians per pixel. In this method each pixel is represented more accurately. The results show improvement not only in accuracy, but in running time as well. The improvement in processing time comes from the reasoning that the optimum number of components in the mixture of Gaussians for most pixels in any given scene is relatively low, thereby reducing the total number of Gaussians that need to be calculated.

Further improvements to the EM algorithm used in the mixture of Gaussians can be seen in [16]. Lee applies an incremental EM algorithm that has the advantage of having a fast convergence whilst still managing to maintain the stability of the statistical model. Traditional incremental variants of the EM algorithm [22] could not be efficiently used for background subtraction in this case. Lee solves this problem by applying an adaptive learning rate for each Gaussian per pixel for every frame. His results show an increase in the convergence rate while maintaining an accurate classification between the background and foreground. A fast convergence of the EM algorithm not only reduces errors while the system is not yet initialised, but it also reduces the errors when new foreground objects are introduced into the scene.

2.1.2 Non-Parametric Background Subtractions

One disadvantage to the mixture of Gaussians method with regard to background subtraction is that it always requires a trade-off between a low learning rate, which can cause the changes in the background to be incorrectly detected as foreground, and a high learning rate, which can learn slow-moving foreground objects into the background.

If memory capacity and processing power are not primary concerns, then this problem of the trade-off can be overcome by way of several different techniques. One such technique [20, 21] uses statistical prediction models in order to predict what each frame is going to be before it appears. The difference between the currently observed frame and the predicted frame is considered to be the foreground. By using the auto-regressive model for prediction, one can accurately predict the next frame within reasonable error bounds. However, the amount of data that is contained in a single frame is too much for the prediction model to handle and it also exceeds the physical memory constraints. Therefore, in order to reduce the dimensionality of the data, principle component analysis is first applied on each frame in order to extract only the more relevant information. The results of the principle component analysis seem to be promising, especially as no trade-off is required between learning rates, as they are bound to the actual prediction model. However, even with the reduction in the dimensionality of the input data that comes with principle component analysis, the entire system still uses a great deal of computational power and memory to extract a feasible background model.

Another method that makes use of principle component analysis (PCA) in background subtraction is presented in [12]. This method uses PCA to detect novel events in a sequence of images, and these events are usually described as being a foreground region. However, the difference is that certain foreground regions, such as ocean waves, are sometimes unwanted. The use of PCA is to extract a small set of principle vectors that can describe the history of that particular image sequence. Similar to other examples that use PCA, the methods presented in [12] reduce the entire image sequence to a series of windows and apply PCA to each separate window.

One method that is derived from the mixture of Gaussians method and that eliminates the parameterised trade-off between learning rates is the kernel density estimate [5], which was developed by Elgammal et al. The kernel density estimate method adapts both the background modelling and the subtraction of the background by using a generalisation of the mixture of Gaussians method. The method creates a single distribution that can represent the history of the pixel over the entire scene instead of having to create a separate Gaussian for the different intensities of a single pixel. One core aspect of their model is that it can model very recent events; this is done so as to ensure that quick changes in the background model are quickly updated without there being any corruption of the previous instances of the background. The results achieved show that the model has a higher sensitivity towards foreground detection than does the mixture of Gaussians model.

2.1.3 Selective Updating

Selective updating is used in many background subtraction techniques. Selective updating only updates the background model with predicted background pixels. Shoushtarian and Bez [26] make use of this technique in relation to three different background models. The first technique models the background with a simple mean of the previous set of frames and whenever a pixel's intensity deviates from the mean by a set threshold, it is marked as foreground and the mean for that pixel is not updated with this value. Similarly, Shoushtarian and Bez's second technique models the background with the median of the previous set of frames instead of its mean and if any pixel's intensity deviates from this median, it is marked as foreground and is not taken into account in the calculation of the median. In the third technique, the background model per pixel is basically the previous intensity in the scene that was marked as a background. All three of these techniques provide an acceptable result for background subtraction; however, use of these techniques alone does not produce the accuracy that can be obtained through the use of several other background subtraction techniques. One of the main problems is ghosting, which is where an area of background is misclassified as a foreground object that was previously present. Many attempts have been made to use this technique in conjunction with other techniques in order to solve this problem; these techniques include tracking and Kalman filtering.

These techniques are so as to have additional information with regard to the foreground objects so that they can be identified as marked if such ghosting does occur.

There are also several methods that make use of the selective updating principle. For example, one component of the kernel density estimate by Elgammal et al. [5] uses the selective update to suppress the foreground in its update of the background model. This is important as this method only uses a recent history of the background to create the model; a foreground object can therefore pollute the entire background model and thereby reduce the accuracy of the model. However, the problems associated with selective updating – ghosting being the major one – are also taken care of by modelling two versions of the background, one with selective updating and the other without. The final background model is the intersection of the two, with slightly less priority being given to the selective updating background model when neighbouring pixels are taken into account.

Another technique that uses a similar design to selective updating is presented in [24]. In this work, Ridder et al. perform a background subtraction with the use of a Kalman filter. A Kalman filter [13] is a time series process that develops a prediction model from a set of observed points. As each point is observed, the entire prediction model is updated according to the newly observed point. Ridder et al. use this process as a prediction mechanism for each pixel. Therefore, if a new pixel is encountered and it deviates from the predicted model, it is then considered as foreground, otherwise it is considered to be background. However, the Kalman filter is supposed to be updated with each newly observed point, but in the case of foreground objects, the entire filter could be corrupted by the new value. In order to account for this problem, Ridder et al. apply an approach similar to selective updating, in which the Kalman filter is not fully updated with the emergence of a foreground object, while it is normally updated when a background pixel is located.

2.2 Video Analysis

Finding methods and techniques to interpret and analyse a video scene has received considerable attention in recent years. In [19], crowd analysis is performed by estimating the number of people in a scene containing large crowds. This is done by combining several existing image processing and machine learning techniques. With the use of a support vector machine the system is trained to recognise the shape and contours of a person's head. The features used are extracted by applying statistical methods on the Haar wavelet transform of the original image. This process is only accurate, however, when the people in a crowd are on the same horizontal plane. This problem is then solved by applying a perspective transform to the images before any feature is calculated. By calculating the vanishing point in the scene, people's heads are then extracted on any horizontal plane in any given image. Once the heads in the scene have been identified, they are grouped. By splitting each group into intervals and making the assumption that the spacing between heads is constant, an estimate of the size of the crowd is calculated. The authors were then able to establish that by combining the shape and contour of a human head, an accurate estimate of the crowd density can be obtained, even without prior knowledge of the scene.

Some of the methods used for crowd analysis involve identifying the contour of a person in order to determine their actions and intentions. One such method is presented by Yokoyama and Poggio [30]. In their method, Yokoyama and Poggio first apply an adapted form of optical flow with an edge detector. Then, by thresholding edges with little motion and by removing the edges that were present in the background model, a set of edges for each foreground object are obtained. Using the nearest neighbour technique, these sets are grouped into distinct objects and then snakes [14] are used to find the contour of each separate object. The above process does not cater for occluded objects, which are only taken care of when the tracking starts. As each object is tracked, several states are detected in order to identify whether an object is occluded, reappeared, merged or separated.

Lara and Hirata [15] discuss a method for contour extraction in an image sequence. They apply the external morphological gradient in order to first calculate the edges of an entire frame. Then a rough contour of the target object is obtained by modelling the background from these edges using the median of a set of previous frames, and then calculating the difference between the edges of the background and the current frame. The contour is refined by applying several other morphological steps, which include noise filtering. As Lara and Hirata primarily make use of mathematical morphological operators, the computational time complexity of this technique is reasonable considering the results that are achieved. Further minor improvements to this technique would allow for its use in real-time applications. This method does not cater for occlusion and further improvements are needed to correct occluded objects.

Most of the methods that analyse video scenes require analysis of the objects represented in the scene. Several key problems need to be addressed when attempting to identify and calculate feature descriptors for each object. Motion detection is required to locate areas of interest in a given video scene; this is often solved by applying a background subtraction technique. The blobs detected by the background subtraction technique need to be separated and identified. A connected component algorithm, as used in [29], is commonly applied to extract, separate and identify these blobs.

The tracking of objects in a scene can provide valuable information with regard to video analysis. Many methods for tracking that make use of shapes have been proposed in the literature. Haritaoglu, Harwood and Davis [9] have developed a system that uses the shape of an object's silhouette to identify the number of people present in that object. This is done by locating areas within the silhouette that resemble the head, arms and legs of a person. Then, applying the assumption that the head is always above the torso, each person's head is located and counted. The area of the silhouette to which each person belongs is then extrapolated by evaluating the normalised distance from the head to the torso, and so on. Once this is finalised the tracking of each person proceeds based on their detected heads and matching them is based on the appearance model that is stored for each person. The position of each object is also predicted. This is to ensure that a completely occluded object is still tracked. Haritaoglu, Harwood and Davis's results seem promising; however, most of the segmentation boundaries that separate people are straight lines.

2.3 Occlusion Detection

In crowd analysis, most of the methods emphasise locating the object for which feature descriptors are required. To ensure that feature extraction techniques are applied to single objects, instead of being applied to the combination of two or more objects which are in close proximity, one of the classic occlusion detection techniques, discussed in [8], may be applied. One common approach to solving this problem of occlusion is described by Elgammal and Davis [4]. In Elgammal and Davis's technique, a person is split into three distinct areas (i.e. head, torso and legs), and the colour and spatial features of each of these areas are then calculated. Using a technique to maximise the likelihood that a pixel belongs to one object, the object can then be accurately tracked over a video sequence. Occlusion is modelled using two methods. The first is to reason whether one object is in front of or behind another object. By using elliptical regions to label each object, the relative depth of each object can then be evaluated. Labelling a pixel as one object or another and determining which instance maintains the least error calculation will allow one to determine which object is being occluded. The second occlusion modelling method uses the relative depth of an object; in other words it evaluates the probability that a ray from a given pixel would pass through one object before another. Once it is determined that an object is occluded by another object, the occluded object is then correctly re-detected by comparing its current features to the features that were extracted earlier on in the process. The results obtained by using this method are shown to be reasonably good at solving the occlusion problem while tracking people. The main problem with this technique is that it requires the objects to be separated before occlusion takes place.

Chapter 3

Background Subtraction

The detection of moving objects is paramount to the understanding of video sequences. One method of detecting moving objects is background subtraction, which is the separation of the background from what is considered to be the foreground. This separation is required so that future analysis on the videos can be performed on only the foreground objects.

3.1 Frame Differencing

The most common aspect of background subtraction can be associated with motion detection. In any given scene, the objects that do not represent the background are most likely moving. Therefore, one method of detecting foreground objects is to use frame differencing. This method is based on the principle that each pixel (x, y) in the foreground binary mask $F(t)$ at frame t is considered foreground if the difference between the grey levels of the pixel at frame t and frame $t-1$ is greater than a preset threshold T. Otherwise the pixel is considered background. The foreground mask will be calculated as follows:

$$F(t)_{x,y} = \begin{cases} 1 & \text{if } \mid I(t)_{x,y} - I(t-1)_{x,y} \mid \geq T \\ 0 & \text{if } \mid I(t)_{x,y} - I(t-1)_{x,y} \mid < T \end{cases} \tag{3.1}$$

where $F(t)_{x,y}$ is the value of the foreground mask at the position (x, y), $I(t)_{x,y}$ the grey value at the frame t and pixel x, y.

This implies that the background model is the frame just before the current frame. Fast-moving objects or video sequences with a low rate of frames per second actually perform well considering the simplicity of the algorithm.

3.2 Selective Updating

Background modelling is the process of calculating what the background should be according to a previous set of observed frames. Most methods that calculate the background model do not just calculate it once off, but instead they continuously update it. This is to ensure that any changes to the background are applied to the background model as well. For example, if there is a global lighting change over an entire scene, then the background model must incorporate it in order to accurately model the background.

Problems occur when a foreground object is detected and the pixel information of that object is used in the updating of the background model. This pollutes the background model as it starts to contain the information of several foreground objects within it. This can cause errors in background subtraction, especially when it is a slow-moving foreground object. The object then becomes learned into the background model over time and eventually the system starts believing that the object is part of the background.

This problem can be countered by the process of selective updating. Selective updating is when the information contained in a foreground object is, once it is detected, ignored in the calculation of the background model. Several methods of selective updating are studied in [26]. The difference between those methods is the actual modelling of the background, for example, modelling the background on the temporal mean or on the temporal median.

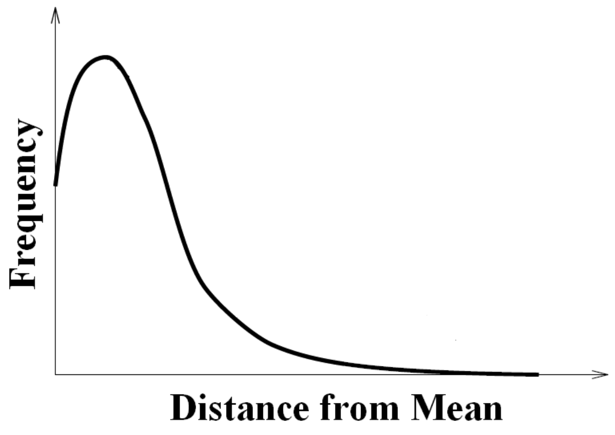


Figure 3.1: Unimodal Distribution

3.2.1 Automatic Threshold Calculation

In many segmentation techniques, all the calculations are done in order to make one simple decision: is the current pixel part of the segmented object or not? Some methods find a statistical model for each pixel and then decide how far away from this model a pixel must be in order for it not to belong to that object. Unfortunately, the notion of 'far away' can vary depending on what type of data is being used. For example, in one case the difference between a background pixel, whose statistical model is calculated over time, and a foreground pixel might be slight, while another time it might be great. Therefore an automated way of evaluating the threshold is highly desirable. In selective updating one is dealing with the distance from the mean, therefore most of the data will lie in or around the mean (Chebyshev's inequality). This will most often result in a unimodal distribution of difference in pixel intensities from the mean (see figure 3.1).

According to Rosin [25], the threshold in a unimodal distribution can be found by extending a line from the highest peak of the histogram to the end of it. Calculating the largest perpendicular distance from this line to the histogram will indicate the point from where the threshold should be chosen (see figure 3.2).

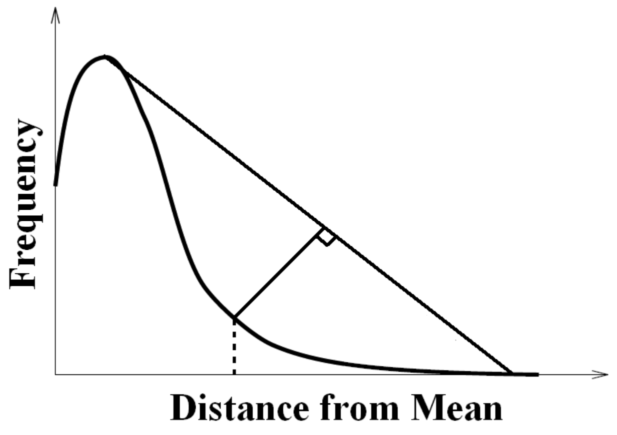


Figure 3.2: Threshold Calculation [25]

As stated before, an optimum distance threshold varies according to the actual data used. Therefore, to manually select the appropriate threshold, several tests need to be performed on each set of data. In addition, three distance thresholds would need to be calculated for colour image sequences (i.e. one for each colour band). To manually select these values would thus take some time, especially when it comes to finding the correct combination for the three-colour bands. In the results that were obtained, it was found that the values, from using the technique in [25], were close (i.e. within 10 intensity values) to the optimum results that were obtained manually.

In conclusion, this technique not only reduces the number of tests that need to be performed to obtain good results, but it also allows the system to automatically choose these threshold values, regardless of what data is being used, thereby making the choice of distance thresholding more generic.

3.2.2 Selective Updating Methods

The actual method of selective updating is made easier with the use of an automatic thresholding method, as is described in [25]. The two selective updating methods that are tested in this thesis are: selective update using temporal averaging and selective update using temporal median. These methods are similar in that they both model the background and never incorporate the foreground objects into their model. The only difference lies in how the background is modelled. In the former method a simple mean of the previous background observations is used as the background model. Any pixel that deviates away from that background model by a threshold, calculated automatically, is then considered foreground. This foreground pixel's information is then excluded from the updating of the background model. This leaves the background model unpolluted from the foreground objects. The threshold is adjusted with the automatic method after each frame and if any previous foreground pixel falls back within the threshold then it is once again flagged as the background and incorporated into the background model.

3.3 Single Gaussian

Using a statistical distribution to calculate the probability of a pixel belonging to the background is the basis behind this type of background subtraction. Assuming a Gaussian distribution of background pixels versus foreground, this method predicts whether or not a pixel with certain intensities belongs to the background. Each pixel in the video sequence generates its own Gaussian distribution and, as the system starts, it extracts each pixel's intensity and inputs it into its Gaussian distribution. Over a short period of time a decent model of the background is generated per pixel. Therefore if any further pixel that is tested against this model falls outside the mean by a number of standard deviations, that pixel is considered a foreground pixel.

Formally given a random variable X of a Gaussian distribution such that:

$$X \sim N(\mu, \sigma^2) \tag{3.2}$$

then the probability of pixel x belonging to the background is:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp[\frac{-(x-\mu)^2}{2\sigma^2}] \tag{3.3}$$

Therefore the foreground mask $F(t)$ at time t at position x, y is calculated as follows:

$$F(t)_{x,y} = \begin{cases} 1 & \text{if } \mu - k\sigma < I(t)_{x,y} < \mu + k\sigma \\ 0 & \text{if } I(t)_{x,y} < \mu - k\sigma \text{ or } I(t)_{x,y} > \mu + k\sigma \end{cases} \tag{3.4}$$

Where k is the number of standard deviations away from the mean and $I(t)_{x,y}$ is the grey value at time t and position x, y. Usually $k = 2.5$. A larger k generates less noise, while a smaller k has a greater chance of extracting a foreground object that is similar to the background model. More information on this method can be found in [29].

3.4 Mixture of Gaussians

3.4.1 Concept

A mixture of Gaussians [7] uses the same strength as does the single Gaussian in modelling the background, namely that of using a statistical distribution to do so. However, using a single Gaussian to represent all background and foreground observations leads to problems when the scene becomes more complex. Simple changes to the lighting conditions and the motion of the background require a more complex approach. Therefore by taking the strengths of a single Gaussian, a mixture of Gaussians uses multiple Gaussians per pixel instead of one. This makes it easier to model the complex conditions observed in the scene.

The concept behind using a mixture of Gaussians can be broken down into three steps. As there are multiple Gaussians for each pixel in the scene, the first step is to calculate which Gaussian the pixel belongs to in each frame. In a single Gaussian method all pixels – whether they are background or foreground – join the same Gaussian. This can cause the Gaussian to over-extend itself. That single Gaussian can contain multiple background scenarios. Like a leaf that is moving in front of a blue sky, both the sky and the leaf are considered background. The single Gaussian might also contain foreground information if a foreground object moves in the area of that pixel. This is the main reason why the single Gaussian is inferior to a mixture of Gaussians. In a mixture of Gaussians, there can be a whole Gaussian for each of those backgrounds as well as one for the foreground.

Having more than one Gaussian does, however, make the job of updating the background model that much harder. The mean and standard deviation of each Gaussian needs to be updated. This is done so as to ensure that if it was originally chosen to belong to the incorrect Gaussian, it will at least not cause a large problem. This update depends on the probability that it belongs to each Gaussian. If the probability is low, then its mean and standard deviation updates are not great. Similarly, if it is high or is identified as most likely belonging to a Gaussian, then that Gaussian's mean and standard deviation updates are much greater.

As the method is using the theory of probability, the total probability of the pixel belonging to the scene should be equal to one. Therefore a weight is used in the method in order to reduce each output of the Gaussians so that their sum equals one. The weight of each Gaussian is increased if the current pixel is found to most likely belong to that Gaussian, but the weight is decreased if otherwise.

Once everything is updated, the final problem that one faces is to determine which Gaussians belong to the background and which belong to the foreground. This is calculated by looking at the weights of each Gaussian. If the weights are high, it means that there were many observations belonging to that Gaussian and it must therefore represent a background. A very low weight will represent a foreground object.

3.4.2 Method

The method for calculating the mixture of Gaussians tested in this thesis is based on the work presented by Stauffer and Grimson [28]. In this work there are several parameters that need to be input to use the mixture of Gaussians. The three main parameters are:

- The number of Gaussians per pixel required K: Several researchers have developed adaptive methods to calculate this number [31]. The common consensus is to have a number greater than three, but not too high a number at the same time. The upper bound is determined by the type of background expected. A busier background (e.g. many trees, clouds or birds) needs more than does a less busy background.

- The learning rate α: This is used to calculate how fast an object is learnt into the background. The faster rate is more adaptive to changing background conditions (such as variable lighting conditions), however, it will also make it easier for the system to incorrectly learn a foreground object into the background.

- The estimated threshold of background T: This threshold is used for estimating how much of the scene should consist of background. A higher number indicates that the scene is expected to be empty, while a lower number indicates that the scene is expected to be quite busy.

Therefore, for every pixel location in the scene, if the set of pixels X is the group of the observations at that pixel location over time,

$$\{X_1, ..., X_t\} = \{I(x_0, y_0, i) : 1 \leq i \leq t\} \tag{3.5}$$

then the probability of observing X_t is

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{3.6}$$

where K is the number of Gaussians we want to use. $\omega_{i,t}$ is the weight of the i^{th} Gaussian at time t. η is the Gaussian probability density function with the mean (μ)and co-variance matrix (Σ) of the i^{th} Gaussian at time t.

$$\eta(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t-\mu_t)^T \Sigma^{-1}(X_t-\mu_t)} \tag{3.7}$$

It is assumed that the red, green and blue channels are independent of each other. Therefore the co-variance matrix is defined as:

$$\Sigma_{k,t} = \sigma_k^2 I \tag{3.8}$$

Once it has calculated which Gaussian X_t is the closest to, the system updates its three values over all the Gaussians. The weight $\omega_{k,t}$ of the k^{th}Gaussian at time t is updated as:

$$\omega_{k,t} = (1-\alpha)\omega_{k,t-1} + \alpha(M_{k,t}) \tag{3.9}$$

where α is the learning rate parameter and

$$M_{k,t} = \begin{cases} 1 & \text{if } X_t \text{is closest to this Gaussian } k \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

Then the mean (μ) and variance (σ^2) is updated as follows:

$$\mu_t = (1-\rho)\mu_{t-1} + \rho X_t \tag{3.11}$$

$$\sigma^2 = (1-\rho)\sigma_{t-1}^2 + \rho(X_t - \mu_t)^T(X_t - \mu_t) \tag{3.12}$$

where ρ is the learning rate based on α which is calculated as:

$$\rho = \alpha\eta(X_t \mid \mu_k, \sigma_k) \tag{3.13}$$

Each Gaussian is then classified as background depending on parameter T as:

$$B = \arg\min_b \left(\sum_{k=1}^{b} \omega_k > T\right) \tag{3.14}$$

3.5 Dynamic Background

Dynamic textures [20, 21] are a complex type of texture found in multiple video sequences. It is a texture that does not remain static over time, but instead consists of a periodic repetition of a type of pattern. Video sequences consisting of smoke, fire and moving bodies of water are considered to be dynamic textures as the process that governs their movement is a distinguishable, repeating pattern. However, these patterns are considered to be too complex to model by way of the previous techniques discussed. This is because those techniques try to model only the history of the pixel's intensity. These patterns could consist of many variations and a single pixel might not be able to accurately experience every instance of the pattern, therefore it would not be modelled. This will, in many cases, mark that pixel as foreground. The solution to this is to model the actual pattern and not just the pixel. Once a model of the dynamic texture is established, it can be used for predicting what the background of a video sequence is and can therefore be used as another background subtraction technique. The model behind the derivation of dynamic textures described in the following sections is based on work presented by Mittal et al. [20] and Monnet et al.[21].

3.5.1 Formal Mathematics behind the Model

Let $\{I(t)\}_{t=1...T}$ be an image sequence. The prediction of the next image in the sequence, based on the last observed k images, can then be represented as follows:

$$I_{pred}(t) = f(I(t-1), I(t-2), ..., I(t-k)) \tag{3.15}$$

where f is the prediction function that is to be determined.

However, as each image is described by a set of pixel intensities, these intensities do not describe the image as a whole. Therefore, a more accurate way of describing the image is to analyse the features obtained by way of a feature extraction method.

Let $\{\phi_i\}_{i=1}^n$ be a filter bank of operators ϕ calculated by any feature extraction method (e.g. Wavelet, Gabor, PCA).

Then for each operator in the filter bank we can calculate $s_i(t) = \phi_i(I(t))$ (the convolution of operator ϕ_i on image $I(t)$). This gives us a vector that can represent the current state of the system at time t:

$$\vec{s}^T(t) = [s_1(t), ..., s_n(t)] \tag{3.16}$$

So by applying equation 3.16 to equation 3.15 we arrive at:

$$\vec{s}_{pred}(t) = f(\vec{s}(t-1), \vec{s}(t-2), ..., \vec{s}(t-k)) \tag{3.17}$$

Once the prediction function f is calculated (see section 3.5.2), we will need a way to revert back to the image I from the state vector $\vec{s}(t)$. This can be done by a manipulation of the operator ϕ.

Suppose the ϕ operator is calculated and known, we can say: $\{\phi_i = b_i\}_{i=1}^n$ where b_i are the set of basis vectors of ϕ_i. Therefore the state vector can be evaluated as follows:

$$\begin{aligned}
\vec{s}(t) &= [s_1(t), ..., s_n(t)]^T \\
&= [\phi_1(I(t)), ..., \phi_n(I(t))]^T \\
&= [b_1^T \cdot I(t), ..., b_n^T \cdot I(t)]^T \\
&= B^T \cdot I(t) \qquad (3.18)
\end{aligned}$$

We can then say:

$$\vec{s}_{pred}(t) = B^T \cdot I_{pred}(t) \tag{3.19}$$

Therefore, using the pseudo-inverse of B^Twe arrive at:

$$I_{pred}(t) = pseudoinv(B^T) \cdot s_{pred}(t) \tag{3.20}$$

where

$$pseudoinv(X) = (X^T \cdot X)^{-1} \cdot X^T \tag{3.21}$$

if X is orthonormal then $X^T \cdot X = I$ and since B^T is the set of basis vectors then $B \cdot B^T = I$. We obtain:

$$pseudoinv(B^T) = B \tag{3.22}$$

we can then conclude from equation 3.20 that:

$$I_{pred}(t) = B \cdot s_{pred}(t) \tag{3.23}$$

This gives us a method for converting between the current state of the system and the image as long as we are able to calculate the basis vectors of the operator ϕ. This calculation is shown in section 3.5.3.

3.5.2 Auto Regressive Models

The prediction function f used in equation 3.17 can include a wide variety of the known prediction functions used in statistics. These functions can range from being linear functions to much more complex non-linear functions. However, a linear prediction model is used to ease the overall computational time. The one chosen in [20] and [21] is the auto-regressive model.

Auto-regressive models are used in time series for a prediction system that is considered to be stationary. Stationary systems are systems whose data has a constant

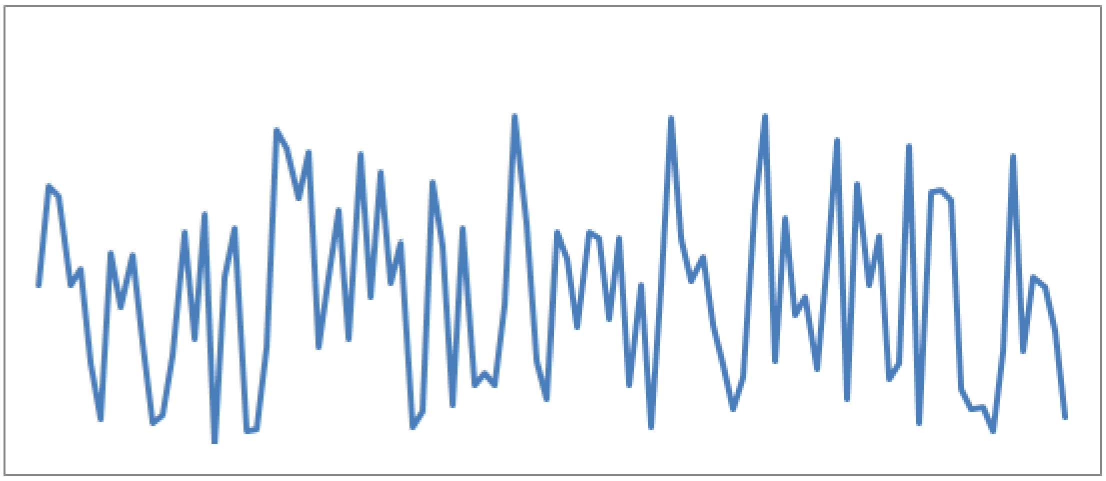


Figure 3.3: Graph showing a set of random data that has a constant probability distribution

probability distribution, regardless of the change in time and space. See figure 3.3.

There are many methods for taking unseen and unknown data sets and making them have a much higher chance of adhering to a stationary system. One such method that is used in calculating the state space is to calculate it on the mean subtracted image: $I(t) - \bar{I}$.

So the auto-regressive model of the system is:

$$\begin{aligned} s_{pred}(t) &= f(\vec{s}(t-1), \vec{s}(t-2), ..., \vec{s}(t-k)) \\ &= \sum_{i=1}^{k} A_i \vec{s}(t-i) \end{aligned} \tag{3.24}$$

Equation 3.24 is a k^{th} order auto-regressive model and A_i is the set of $n \times n$ prediction matrices that is needed for each k. n is the number of features in each feature vector $\vec{s}(t)$.

In equation 3.24 k can be seen as the maximum "lag" in the model. As $i \to k$, $\vec{s}_{pred}(t)$ is compared to a vector further away from it (hence "lag").

The previous histories of the state vectors are required in order to solve auto-regressive models. So for example, if we take $k = 3$, we want to solve A_1, A_2 and A_3. Therefore we will need to compare the set of state vector histories for three different "lag" values, by concatenating the set of state vectors into matrices S_A and S_B where the difference between them is the "lag" value. That is:

$$S_A = \{\vec{s}(a)\} \qquad where \quad a = 1 + i, ..., t \tag{3.25}$$

and

$$S_B = \{\vec{s}(b)\} \qquad where \quad b = 1, ..., t - i \tag{3.26}$$

Therefore for $i = 1$ we will need to compare and evaluate :

$$\begin{aligned} S_A &= [\vec{s}(2), \vec{s}(3), ..., \vec{s}(t)] \\ S_B &= [\vec{s}(1), \vec{s}(2), ..., \vec{s}(t-1)] \end{aligned} \tag{3.27}$$

Similarly for $i = 2$:

$$\begin{aligned} S_A &= [\vec{s}(3), \vec{s}(4), ..., \vec{s}(t)] \\ S_B &= [\vec{s}(1), \vec{s}(2), ..., \vec{s}(t-2)] \end{aligned} \tag{3.28}$$

and for $i = 3$

$$\begin{aligned} S_A &= [\vec{s}(4), \vec{s}(5), ..., \vec{s}(t)] \\ S_B &= [\vec{s}(1), \vec{s}(2), ..., \vec{s}(t-3)] \end{aligned} \tag{3.29}$$

Therefore for each i we have to solve for the unknown A in:

$$S_A = A \centerdot S_B \tag{3.30}$$

If we take a closer look at equation 3.30, by expanding it out and taking for example $i = 1$, $n = 2$ and say $t = 4$, we arrive at the following:

$$\begin{pmatrix} \vec{s}(2)_1 & \vec{s}(3)_1 & \vec{s}(4)_1 \\ \vec{s}(2)_2 & \vec{s}(3)_2 & \vec{s}(4)_2 \end{pmatrix} = \begin{pmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{pmatrix} \begin{pmatrix} \vec{s}(1)_1 & \vec{s}(2)_1 & \vec{s}(3)_1 \\ \vec{s}(1)_2 & \vec{s}(2)_2 & \vec{s}(3)_2 \end{pmatrix} \tag{3.31}$$

$$= \begin{pmatrix} x_{1,1}\vec{s}(1)_1 + x_{1,2}\vec{s}(1)_2 & x_{1,1}\vec{s}(2)_1 + x_{1,2}\vec{s}(2)_2 & x_{1,1}\vec{s}(3)_1 + x_{1,2}\vec{s}(3)_2 \\ x_{2,1}\vec{s}(1)_1 + x_{2,2}\vec{s}(1)_2 & x_{2,1}\vec{s}(2)_1 + x_{2,2}\vec{s}(2)_2 & x_{2,1}\vec{s}(3)_1 + x_{2,2}\vec{s}(3)_2 \end{pmatrix} \tag{3.32}$$

where $x_{a,b}$ for $a = 1..n$ and $b = 1..n$ are the unknowns.

This therefore creates an over-constrained set of linear equations. An over-constrained set of linear equations is a set of simultaneous equations where there are more equations than there are variables. In equations 3.31 and 3.32 there are four variables and six equations.
When this is expanded to the general case we get:

$$\begin{pmatrix} \vec{s}(1+i) \\ \vdots \\ \vec{s}(t) \end{pmatrix}^T = \begin{pmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,n} \end{pmatrix} \begin{pmatrix} \vec{s}(1) \\ \vdots \\ \vec{s}(t-i) \end{pmatrix}^T \tag{3.33}$$

It can be seen in equation 3.33 that the size of the matrices decreases as $i \to k$. For example the size of the matrices as i increases is as follows:

$$\begin{aligned} &\text{For} \quad i = 1 \quad :- \quad (n \times n) \centerdot (n \times (t-1)) \\ &\text{For} \quad i = 2 \quad :- \quad (n \times n) \centerdot (n \times (t-2)) \\ &\text{For} \quad i = 3 \quad :- \quad (n \times n) \centerdot (n \times (t-3)) \end{aligned}$$

Therefore, the number of columns decreases for every iteration of i. So in order for the system to stay as an over-constrained set of linear equations we need $t-k > n$ where n is the number of feature vectors in each feature vector $\vec{s}(t)$. In addition a "lag" k cannot exceed the total time passed therefore the creation of the auto-regressive model cannot begin until:

$$0 < n < t - k \tag{3.34}$$

There are several methods for solving an over-constrained set of linear equations, one of which is the method of least squares.
In equation 3.30 we can say that the best solution for A is:

$$\arg\min_{A} \|S_A - A \cdot S_B\| \tag{3.35}$$

which can be rewritten as:

$$A = S_A \cdot S_B^T \cdot (S_B \cdot S_B^T)^{-1} \tag{3.36}$$

Now that A_i is solved for all i, the prediction model is complete.

3.5.3 Evaluating the Basis Vectors

In order to calculate the orthogonal basis vectors that are needed in equation 3.23 as is shown in section 3.5.1, we first have to consider the set of images that will be used in prediction

$$\{I(t-k)\}_{k=0..m-1} \tag{3.37}$$

where m is the previous number of frames to be included in the prediction.

PCA is used on this set of images to extract orthogonal basis vectors. In order to use these sets of basis vectors in the auto-regressive model (section 3.5.2), the system must be considered to be stationary. One method that will further increase the chances that the system conforms to stationarity is to use the set of zero mean images

instead of just the images in the PCA. A zero mean image is just the original image subtracted by the mean value of the image.
Therefore,

$$\tilde{I}(t) = I(t) - \bar{I}(t) \tag{3.38}$$

where

$$\bar{I}(t) = \frac{1}{n} \sum_{i=t-m+1}^{t} I(t) \tag{3.39}$$

the mean of $\{I(t-k)\}_{k=0..m-1}$.

Each image $I(t)$ is of resolution $r \times c$, where r is the number of rows and c is the number of columns in the image. If we then concatenate the set of images into a column vector, where each element is in itself a row vector consisting of all the pixels in the image we can create a matrix $\tilde{I}_M$ with dimensionality $(r \times c) \times m$, such that

$$\tilde{I}_M = \begin{pmatrix} \tilde{I}(t)_{0,0} & \cdots & \tilde{I}(t)_{0,c} & \tilde{I}(t)_{1,0} & \cdots & \tilde{I}(t)_{r,c} \\ \tilde{I}(t-1)_{0,0} & \cdots & \tilde{I}(t-1)_{0,c} & \tilde{I}(t-1)_{1,0} & \cdots & \tilde{I}(t-1)_{r,c} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \tilde{I}(t-m+1)_{0,0} & \cdots & \tilde{I}(t-m+1)_{0,c} & \tilde{I}(t-m+1)_{1,0} & \cdots & \tilde{I}(t-m+1)_{r,c} \end{pmatrix} \tag{3.40}$$

In PCA the covariance of matrix $\tilde{I}_M$ is calculated in order extract the orthogonal basis vectors by extracting the eigenvectors from the covariance matrix.
For a given matrix X, the covariance (Σ) with itself is calculated by the expected value of that matrix multiplied by that matrix transposed

$$\Sigma_X = E[(X - \mu_X)(X - \mu_X)^T] \tag{3.41}$$

As $\tilde{I}_M$ is already the zero mean vector (see equation 3.38), then the covariance of $\tilde{I}_M$ can be calculated as follows:

$$\Sigma_{\tilde{I}_M} = E[(\tilde{I}_M)(\tilde{I}_M^T)] \tag{3.42}$$

According to Jolliffe [10] the orthogonal basis vectors can therefore be calculated by the eigenvectors of $\Sigma_{\tilde{I}_M}$.

According to Mittal et al [20] and Monnet et al [21], the calculation of the eigenvectors on the covariance matrix can be shortened by using an approximation of the covariance matrix.

$$\Sigma_{\tilde{I}_M} \approx (\tilde{I}_M)(\tilde{I}_M^T) \tag{3.43}$$

Then by using the singular value decomposition on $\tilde{I}_M$:

$$\tilde{I}_M = UDV^T \tag{3.44}$$

the set of eigenvectors, which will be used as the orthogonal set of basis vectors, will be stored in the matrix U.

Tests done in Mathematica show that using the approximation and extracting the eigenvectors by the singular value decomposition do not yield much difference. The difference is further negated by the discrete nature of the images that are used.

3.5.4 Results and Future Work in Dynamic Background Modelling

This technique for background subtraction was coded and tested using Mathematica. Although the tests show a positive outcome when modelling the background of a video sequence, the computational time required for the entire technique is extremely high. The tests could not be performed on video sequences larger than the test cases, which had a resolution of 32 by 32. Several methods are thus needed in order to be able to use real video sequences. Two techniques, both of which are mentioned by Mittal et al. [20] and Monnet et al. [21], are mentioned in order to improve the performance of the entire process. The first technique is to use an incremental version of the PCA, which both sets of researchers discuss in detail. The second technique is to window each frame and perform the incremental method on each window instead of on the entire image. Therefore, for each window, a prediction of what that window should be, is created It is used when it comes to determining if that entire window belongs to the background, including the dynamic texture, or the foreground. Even with this implemented, one can discern many future improvements that could greatly improve the overall use of the technique. As already mentioned, finding and implementing algorithms and techniques to improve the running time of this technique would be a great improvement in itself. Such improvements could include the use of high-performance computing techniques, such as the parallelising of the technique over multiple processes. It could also include finding alternative ways of calculating and solving several areas of the technique. Other improvements or variations could also be researched, such as using different feature extractions and comparing their outcomes. For example, one could probably find a way of using a different filter bank – such as Gabor filters or wavelets – instead of having to use PCA.

3.6 Common Problems Associated with Background Subtraction

A common problem in background subtraction is that foreground objects are sometimes classified as background. One way that this can happen is when the object falls inside the acceptable difference bounds of the background model. This could be because the acceptable bounds are too inclusive and therefore class vastly different objects as background. Alternatively, this could occur simply because the object in the foreground appears similar to the actual background.

Another common problem associated with many background subtraction problems is ghosting. Ghosting is when a foreground object is detected, when in fact no object is present in the actual footage. Ghosting usually occurs in the aftermath of when the system classifies a foreground object as background. In the initial stages of a background subtraction algorithm, a foreground object may be detected as the background. When the object that the system missed has moved off, the system then registers that area where the object was originally as being vastly different to the perceived background. It thus flags that area as foreground and in so doing creates a ghost. Usual background subtraction algorithms, such as the mixture of Gaussians, automatically correct this anomaly with time. This is done because the system learns what the intended background is and adjusts itself accordingly. But before these systems are able to correct themselves, the results of the background subtraction are usually unusable, depending on the size of the ghost.

A more frequent problem in background subtraction is the appearance of holes in the foreground object (that is to say, a foreground object that has been identified as foreground but is not perceived as a solid entity by the background subtraction algorithm). Fast-learning algorithms usually have this problem. The data of a slow-moving object in a scene, which incorporates a fast-learning background subtraction algorithm, might learn the object into the background. The front of the object might therefore get picked up as foreground while the rest of the object might not. The rear of the object is actually picked up in these cases because, as the object moves off the actual background is now present. The system then briefly notices the change and creates a small ghost at the rear of the object. Because the ghost is close to the actual moving object, it appears to actually belong to the object. This then gives the impression that the moving object has large holes in its structure.

3.7 Comparison of Techniques

The results and comparisons of background subtraction techniques are crucial when it comes to deciding which technique one should choose for future use.

3.7.1 Results of Frame Differencing and Single Gaussian Background Subtraction

Frame differencing is a rather simple method, which means it is the fastest algorithm to implement and run. Its simplicity is, however, offset by the results it obtains. Since frame differencing only measures a change in motion from one frame to another, video sequences with a high frame rate suffer in accuracy, as the total difference between one frame and the next is small. This method can capture fast, complex objects in a scene. It can capture fast objects because the difference between two consecutive frames would be high. It can also capture complex objects because the differences between one part of the object and the next are great.

The single Gaussian method is also quite simple when compared with the other types of algorithms for background subtraction. It does however use a statistical distribution, which makes up for its simplicity. By using a Gaussian distribution for each pixel, the system is able to differentiate between a foreground object and a background object more easily than are other methods that do not use a statistical distribution. However, this method fails whenever there is more than one change in the background. For example, a tree moving in the wind would normally be considered background, but because of its motion the moving tree is considered foreground. Although the method will, with time, learn the tree into the background, it causes another problem. This problem refers to the Gaussian distribution: if more diverse data is added to the distribution, the wider the Gaussian becomes. Conversely, if the background point does not change at all, the Gaussian distribution becomes really narrow. Therefore, in the case of a wide distribution, an actual foreground object has more chance of falling within the standard deviation of the distribution and consequently is not classified as a foreground object. In the case of a narrow distribution, slight camera noise or slight camera motion will result in a foreground object being detected, as the standard deviation of the distribution only covers a narrow selection of possible background values.

3.7.2 Results of Selective Updating

It was found that the methods that incorporate the selective updating strategy manage to keep the full shape of the foreground objects (i.e. show little to no holes) better than do the other methods. This was to be expected, as the foreground objects were not learned into the background model and therefore the full object stays visible. However, because the system does not take foreground objects into account, the problem of ghosting becomes a major concern. In other background subtraction scenarios, ghosts are slowly faded away as the system learns that that area is supposed to be background. In selective updating methods, however, these areas are always marked as foreground and are therefore never incorporated into the background model. But as these are ghosts, which means that these areas should be considered background, the real background is never learned into the model and therefore the ghosts persist throughout the scene.

It was found that between the two main selective updating methods described in [26] (i.e. selective update using temporal averaging and selective update using temporal median),it was selective update using temporal median that performed the best, but it did, however, have a much higher computational time. In many examples the increase in accuracy that selective update using temporal median has over the accuracy of selective update using temporal averaging was not offset by the increase in computational complexity.

Additional examples for the results of selective updating can be viewed in the appendix (see figure A.1). Notice the number of ghosts in the result. The ghosts will be harder to spot as they are not highlighted, but it shows how hard it is to use this type of technique.

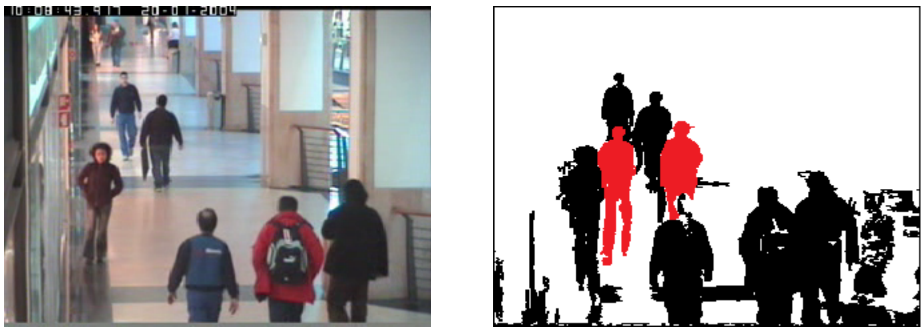


Figure 3.4: Example of the ghosting problem encountered when using selective updating. Left: original frame. Right: foreground mask with highlighted areas indicating ghosts

3.7.3 Results of Mixture of Gaussians

Using mixtures of Gaussians we were able to achieve good results. Single Gaussian failed when there were additional background variations, like a tree moving in the wind or slow lighting changes. But in the case of the mixture of Gaussians this was not the case, as a single Gaussian was created for each of these concerns and the one that belonged statistically to the foreground Gaussian was the only one displayed. This removed a great deal of noise in the extracted foreground region where the single Gaussian failed. In addition, because of imperfect equipment and video compression, a video sequence usually consisted of additional noise, which a single Gaussian would struggle to handle. Instead, in a mixture of Gaussians, the noise was considerably reduced. In fact, after applying simple morphological operators, the noise was, in many cases, reduced substantially. Figure 3.5 shows the results of applying a mixture of Gaussians on a video sequence. Morphological operators were also applied to reduce the remaining noise.

3.7.4 Cross Comparison

A comparison of techniques studied was performed. Several randomly selected frames were chosen from a set of videos and their relative backgrounds and foregrounds were extracted. Of the background subtraction techniques studied, frame differencing was

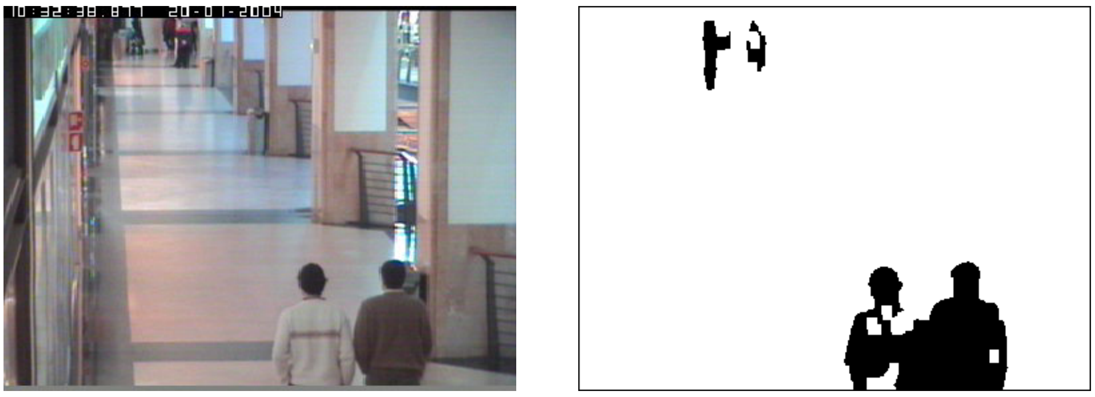


Figure 3.5: Results of the application of the mixture of Gaussains for background subtraction. The left image is the original image and the right one is extracted foreground regions.

used as it has the best computational time. As a mixture of Gaussians is an upgraded form of single Gaussian, only the former method was chosen. Dynamic background was eliminated because of its extreme computational cost. Selective updating using the mean was chosen amongst the selective updating methods as its computational time is not as complex as using the median.

Each frame was then analysed and its accuracy over each background subtraction method was calculated. Table 3.1 shows the overall accuracy for each of the three methods. Mixture of Gaussians had the best performance overall, while frame differencing performed the worst. Analysing each frame separately also shows why mixture of Gaussians performed better than the other two. The ground truth is obtained by performing a hand segmentation on the frame being analysed. Each one of the background subtraction techniques is then performed over the entire video. At the correct frame rate its relative foreground mask is extracted. This is then overlayed onto the hand segmented ground truth and areas are highlighted which show the difference between the two. Three examples of this process are shown.

The first example that will be analysed is in figure 3.6. Here there is a traffic scene in bad weather conditions. Due to the fast moving cars in the scene the frame differencing algorithm picks out the cars quite well, including the ones far in the distance.

Table 3.1: Average Sensitivity and Specificity of the results of Background Subtraction.

	Frame Differencing	Mixture of Gaussians	Selective Updating
Sensitivity	33.06 %	73.44 %	66.85 %
Specificity	98.39 %	97.77 %	96.44 %

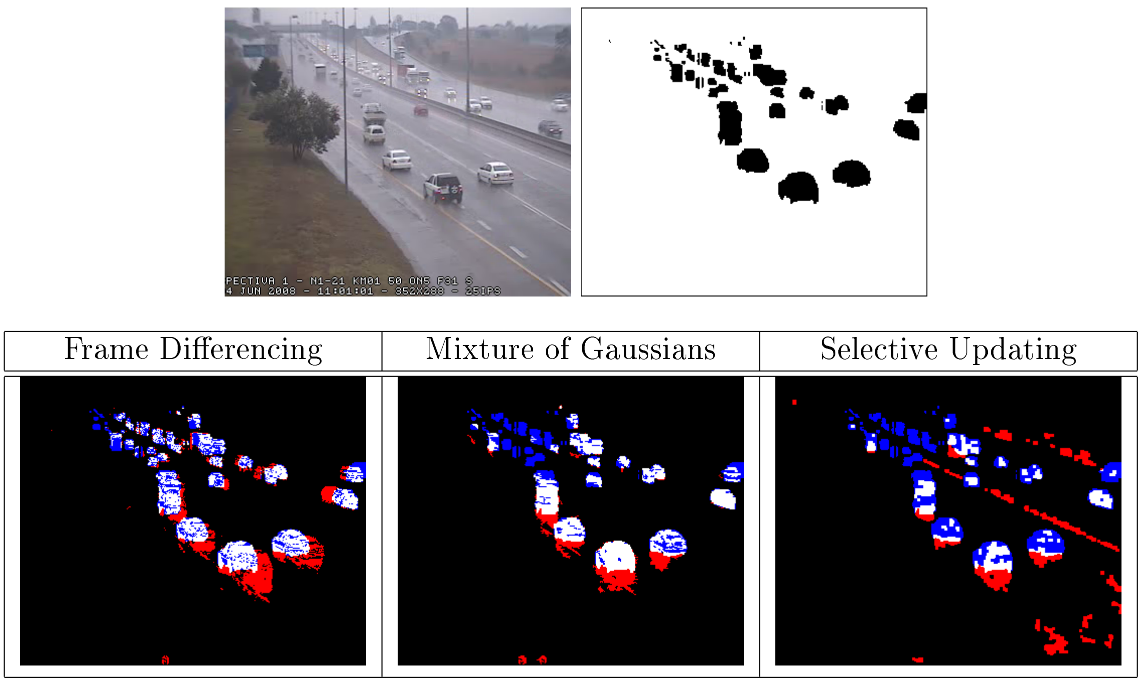


Figure 3.6: Results of Background Subtraction performed on a video of traffic. Above: Original frame with its hand segmented ground truth. Below: the results of each background subtraction overlayed onto the ground truth. White, indicates a True Positive response. Black, a True Negative response. Red, a False Positive response. Blue, a False Negative response.

However this method causes an increase in the number of false positives, especially for the foreground objects close to the camera. This is because the difference between two consecutive frames also includes where the object was previously. So although the speed of the cars helps in their detection with this technique, it also creates a slight ghosting effect behind them.

In the mixture of Gaussians most of the foreground objects gets detected, apart from those in the distance. This is because in the scene at that area there is a large amount of traffic. With a constant flow of traffic over a certain area the algorithm will not be able to learn what the correct background in that area should be. The objects in the foreground, close to the camera also contain some false positive responses. This is mainly a combination of the object's shadow and reflection, although it is worth noting that the volume of false positives is less than when using frame differencing.

The third background subtraction is Selective Updating. Here we can see it performs poorly. As with the mixture of Gaussians, the objects far in the distance are learned into the background as there is not enough information of what the background in that area actually should be. Additionally there is a lot of noise in this example. This noise is actually ghosting, this method learned several foreground objects into the background and because this method only updates the background model when it believes that area contains background, the actual background gets classified as foreground.

For this video sequence taking into account these results either frame differencing or mixture of Gaussians should be used. The choice comes down to what is actually more important, detecting the cars far off into the distance or getting a more accurate outline of the cars closer to the camera.

The next example is shown in figure 3.7. Here it shows that frame differencing, in contrast to the previous example, misses a lot of the foreground objects. This is because

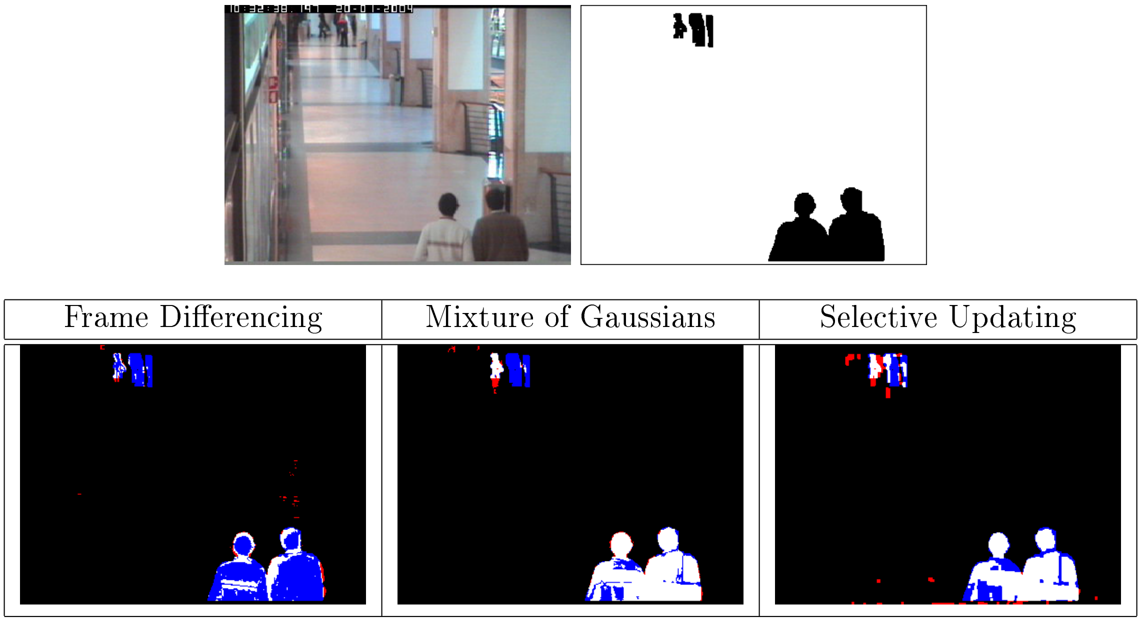


Figure 3.7: Results of Background Subtraction performed on a video of people. Above: Original frame with its hand segmented ground truth. Below: the results of each background subtraction overlayed onto the ground truth. White, indicates a True Positive response. Black, a True Negative response. Red, a False Positive response. Blue, a False Negative response. Here it can be clearly shown that Frame Differencing performed poorly, while Mixture of Gaussians and Selective Updating performing somewhat on par.

this scene is a lot slower than the previous traffic one. The change in between each consecutive frame is much less, therefore the difference between them is low. Only the edges differ which makes the foreground objects appear only as outlines. Although with the slower pace of this scene compared to the traffic scene, the percentage of false positives is much lower as ghosting is minimized. However this does not make up for the large loss in the interior of the foreground objects.

Mixture of Gaussians performs better. This shows that the algorithm learned the background of the scene quite well. There are several holes indicating that the algorithm, in some instances, does not see much of a difference between certain parts of the foreground object with the background. Similarly with the traffic scene the objects far in the distance are not correctly classified.

Selective updating has a similar result to the mixture of Gaussians, but with slightly extra noise. As this scene is much less crowded than the traffic scene there is a much smaller chance of ghosting. Additionally the initial starting conditions in this is a best case scenarios for a selective updating method. That is, in the initial learning phase the overall traffic was virtually non-existent, giving this method an unobstructed background to use.

The final example that will be discussed in detail is shown in figure 3.8. Here once again the slow moving people cause frame differencing to miss a large proportion of the foreground objects. The greatest difference between this example and the previous can be seen in the mixture of Gaussians and the selective updating method. In both, the reflections of the foreground objects off the floor are detected. So although frame differencing had a high false negative count, the other two have a high false positive count. In most cases however the actual full shape of the foreground object is more desirable than one that might include shadows or reflections.

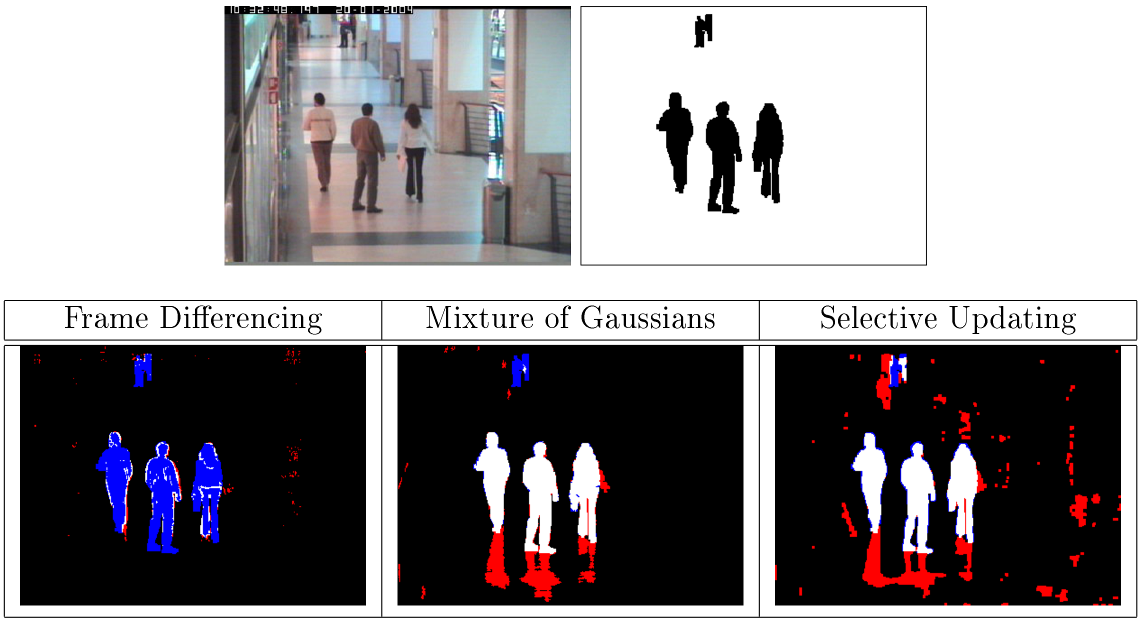


Figure 3.8: Results of Background Subtraction performed on a video of people. This scene shows three distinct persons in the foreground. Above: Original frame with its hand segmented ground truth. Below: the results of each background subtraction overlayed onto the ground truth. White, indicates a True Positive response. Black, a True Negative response. Red, a False Positive response. Blue, a False Negative response.

A proper comparison between the mixture of Gaussians and selective updating in this type of video sequence can be made. Previously there was not much difference, but here it can be clearly seen that selective updating has a lot more false positive responses. This shows that this method is sensitive to noise and camera artifacts. In mixture of Gaussians these camera artifacts and noise can get learned into the background model at their appropriate areas. However in selective updating once it detects noise it marks it as foreground and does not learn it into the background. Therefore over time the amount of noise will increase. We can conclude that for short periods selective updating performs well.

It can be clearly seen that mixture of Gaussians is better suited to these types of scenes. Frame differencing performs slightly better in fast scenes, like in traffic, but that is only because it can detect the vehicles in the far distance. If the requirement is that these vehicles are too small to extract any valuable data from, then there might be no point in using it. However if the requirement is that of a need to use a more accurate shape, then mixture of Gaussians is better. In slower moving scenes, it is clearly obvious that frame differencing should not be used. Selective updating is somewhat on par with mixture of Guassians if the initial starting conditions are suited for a best case scenario - that it is virtually empty at the start. Also over time the amount of noise increases when using a selective update method of background subtraction. Combining this analysis with the overall accuracy in table 3.1 a conclusion can be made in which background subtraction is best suited to our context.

3.8 Conclusions

Frame differencing, although fast and simple to implement, did not yield any type of results that could be used efficiently. The use of dynamic background methods would be too computationally intensive to be useful in real-time situations. The single Gaussian performed relatively well, considering its low computational cost. However, with just a slight increase in the computational cost, one could use the mixture of Gaussians, which has been shown to yield a higher accuracy. It was therefore decided that the use of the mixture of Gaussians was the best choice for the rest of this thesis.

Chapter 4

Scene Analysis

Scene analysis is the extraction of information from a particular scene for use by a person or for input into another program. In this chapter, two methods of scene analysis are studied. The first is qualitative analysis, which will extract information from a scene. The information extracted will be used by a human operator; that person can glance at the result and obtain an understanding of the entire scene. The second is a structural analysis, which will extract information from indoor surveillance footage. The primary purpose of the information extracted here is that it be used as a feed to a computer system for scene understanding.

4.1 Qualitative Analysis

4.1.1 Introduction

Automated road traffic analysis is the cornerstone for any modern transport system. It allows us to monitor the activities of different roads. Describing a scene in words that the average person will be able to understand is more complex than just assessing the values of a few parameters. A computer system, for example, could identify that there are a hundred cars on a road at any given time. This information may not, however, be sufficient, as more information will often be needed to assess what is happening on the road. However, giving a qualitative description, such as: "There is Heavy Traffic", could be enough information for a person to identify this phenomenon.

Many researchers are concentrating on applying a qualitative analysis of video scenes in order to give a more meaningful description. In a traffic scene, for example, a qualitative system would give an analysis, much like a traffic expert would give when viewing that scene.

There are several different approaches to qualitative analysis. The most common, and one of the easiest, is to map quantitative values to qualitative descriptions. For example, when the number of cars on a road is less than one hundred, one interprets that as ‘low traffic’. Anything higher than that is interpreted as ‘high traffic’. This system can work in many cases, but a problem becomes apparent when values on the borderline are reached. In connection to the above example, why would one consider ninety-nine cars as low traffic but not one hundred cars? Another approach called rule-based methods, tries to mitigate this problem. Lists of IF-THEN statements or finite state machines are used in order to incorporate several different values so as to obtain a broader understanding of a single scene. Again using the above example, one hundred cars on any given road could be considered to be low traffic if their movement is free flowing, while it could also be considered as heavy traffic if there is significant congestion. Stochastic methods are also used in qualitative analysis; by assigning and evaluating probabilities, it is possible to give a reasonable prediction of the current outcome as based on these probabilities. This method is best used when large volumes of data can be obtained and evaluated, as this helps to build up the statistical variations. After examining the advantages and disadvantages of each approach, a rule-based approach was used to determine the traffic status of a particular scene.

4.1.2 Rule-Based Analysis of a Traffic Scene

Fathy and Siyal [6, 27] both show that the identification of those areas in the traffic scene that contain vehicles is essential to the analysis of traffic. The motion of the vehicles is then evaluated; in other words, one verifies whether or not the vehicles

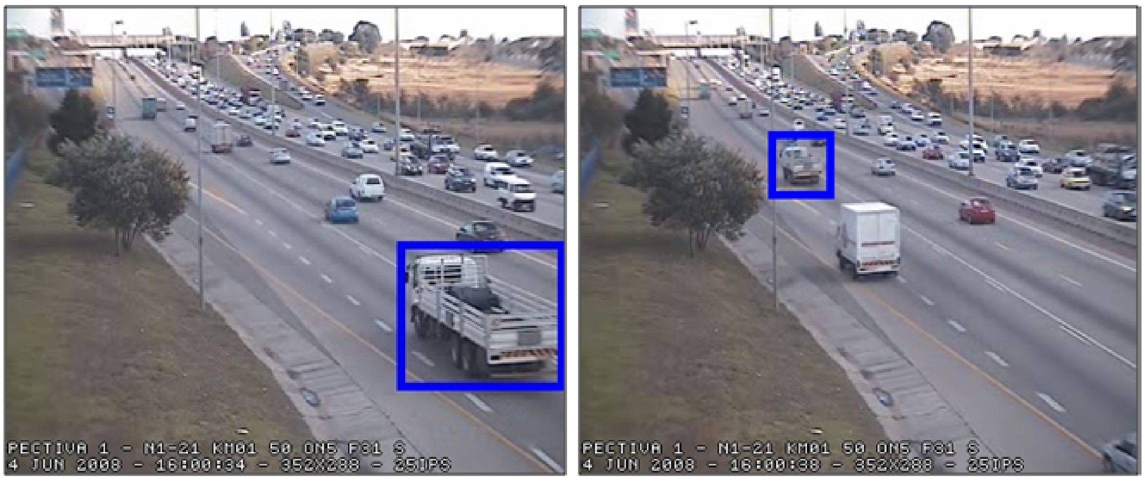

PECTIVA 1 - N1-21 KM01 50 ON5 F31 S
4 JUN 2008 - 16:00:34 - 352X288 - 25IPS
PECTIVA 1 - N1-21 KM01 50 ON5 F31 S
4 JUN 2008 - 16:00:38 - 352X288 - 25IPS

Figure 4.1: Same vehicle highlighted 23 frames apart

are flowing freely or are remaining relatively stationary. This assessment can change the status of the traffic, no matter how many cars there are in the scene. The latter process compares the vehicles detected in the present frame to those that were detected earlier. Then rules are applied to the information gathered so as to determine the status of the traffic.

4.1.2.1 Detecting Vehicles

To understand a traffic scene, the existence of vehicles needs to be determined in the areas that require analysis. However, certain problems must be overcome in order to accurately detect vehicles. Two of the problems encountered in video scene analysis are the scale invariant features of the vehicles and the variance in lighting conditions.

Scale Invariant Features: The problem with extracting features from moving objects is that the further away the object moves, the more distorted its features become. Traffic footage from the N1 highway in South Africa is taken at a low angle and away to the side of the highway. Therefore, as the objects move further away from the camera, their perspective changes and their features cannot be accurately mapped. For example, the same truck is highlighted 23 frames apart in figure 4.1.

To facilitate the use of features, such as the size and volume of the objects, objects

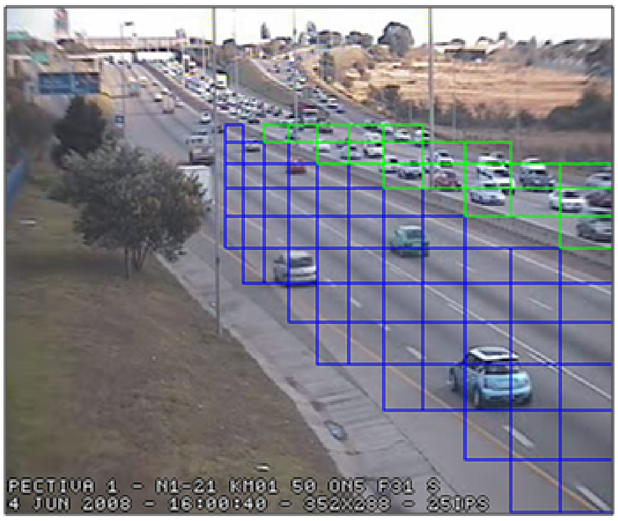

PECTIVA 1 - N1-21 KM01 50 ONS F31 S
4 JUN 2008 - 16:00:40 - 352X288 - 25IPS

Figure 4.2: The scene windowed and split into areas of interest, highlighted in blue and green. Notice how the windows decrease in size in both the vertical and horizontal direction as they represent objects further away from the camera's position.

closer to the camera must have a smaller weight relative to the entire scene than do those further away from the camera. One common method is to window the scene in such a way that all the windows have the same weight, but the size of a window decreases the further away the camera is from the scene.

The traffic scene is first windowed, with the windows' sizes decreasing linearly, both vertically and horizontally. This is because of the camera's position is to the side of the road rather than right above it. Finally, the scene is split into two areas of interest so as to represent the two carriageways (see figure 4.2).

Mitigation of the Dependency on Lighting Conditions: Due to the varying conditions of the weather and the position of the sun, the inconsistent lighting of the scene needs to be taken into account. In figure 4.3, the same scene is displayed at three different times during the same day. The differences can be quite substantial, from the low-contrast rainy conditions, to the sunny scene where the shadow of each vehicle becomes quite dominant.

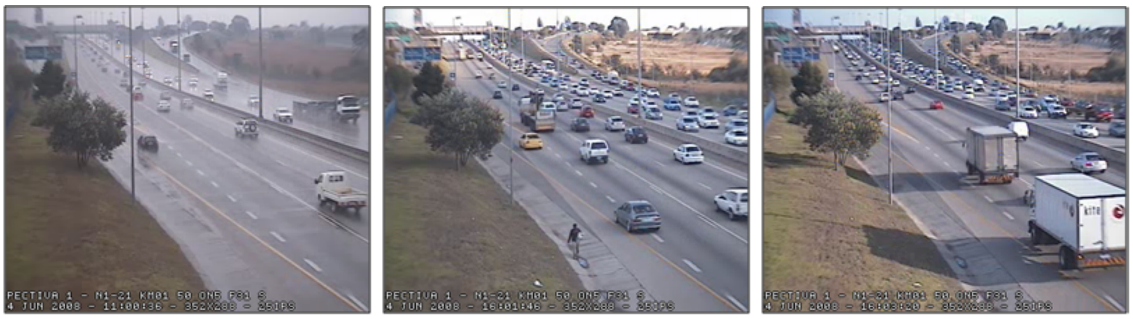

PECTIVA 1 - N1-21 KM01 50 ON5 F31 S
4 JUN 2008 - 11:00:36 - 352X288 - 25IPS
PECTIVA 1 - N1-21 KM01 50 ON5 F31 S
4 JUN 2008 - 16:01:46 - 352X288 - 25IPS
PECTIVA 1 - N1-21 KM01 50 ON5 F31 S
4 JUN 2008 - 16:03:20 - 352X288 - 25IPS

Figure 4.3: Differences in lighting conditions in a traffic scene (rainy, overcast, sunny).

As we are using the size and volume of objects to determine the existence of vehicles and are also faced with the added constraint of varying lighting conditions, Fathy and Siyal [6, 27] suggest the use of edge detection.

The information gathered from edge detection is dependent upon the edge detection algorithm that is used. If the direction of the edges is not taken into account, one can obtain all the relevant information needed when using the object's size and volume, while minimising the problems caused by lighting conditions. In the case of a low-contrast scene, the object's edges will still be detected if the edge detection algorithm is robust enough. In the case of extreme shadows caused by bright sunny days, the edges of the shadows are small as they contain relatively simple designs, while those of the vehicles remain a collage of different edges. The many edges of each vehicle stand out against the few edges of the shadow.

The edge detection algorithm that is used in this instant is the external morphological gradient, as it is independent of direction. The external morphological gradient is calculated with the use of the mathematical morphological operators. The edges are calculated as the difference between dilatation and the erosion of the image (see figure 4.4). In other words, given an image I, and a structuring element B, the edges of I based on B: $E_B(I)$ are calculated as follows:

$$E_B(I) = (I \oplus B) - (I \ominus B) \tag{4.1}$$

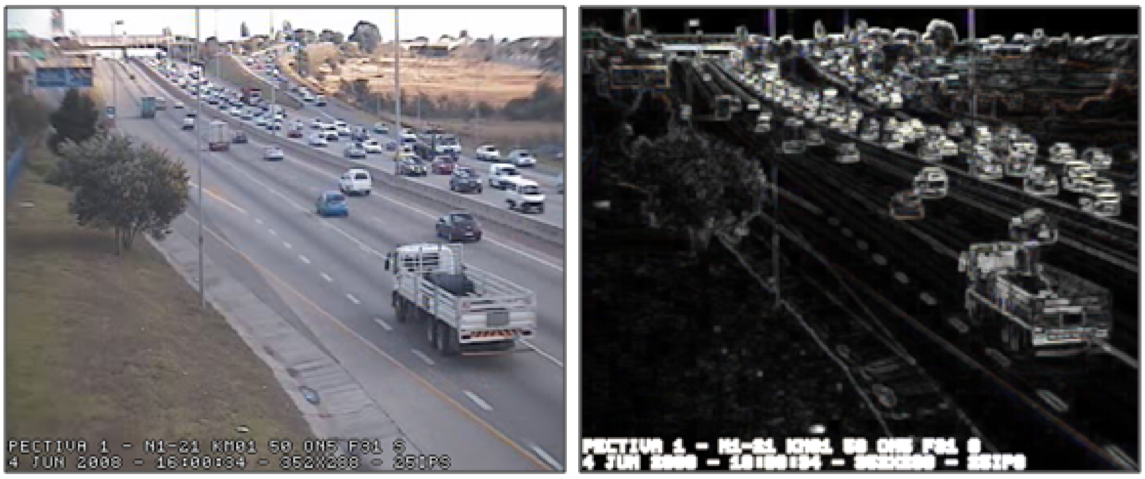

PECTIVA 1 - N1-21 KM01 50 ON5 F81 S
4 JUN 2008 - 16:00:34 - 352X288 - 25IPS
(a)
(b)

Figure 4.4: (a): Original image. (b): Edge detection by the external morphological gradient.

Background modelling: In the resultant image that contains the edges, there are still edges that should have little to no value in terms of determining the existence of vehicles, that is to say, the edges of the background. In order to remove these unwanted edges, the background of the scene is calculated and the edges are subtracted from the original edge map. This leaves only the edges of the foreground objects, which consist mainly of the actual vehicles in the scene. The calculation of the background is done by a mixture of Gaussians (section 3.4). Figure 4.5 shows what appears to be an empty scene, but is in fact the background model of the scene.

4.1.2.2 Detecting Movement

After detecting the existence of the vehicles, their actual motion is easily calculated. This is achieved by applying the same process of detecting vehicles as was previously ascribed to frames that have already passed. Finding the difference between the detected vehicles of the current frame and the detected vehicles of a previous frame will show areas with motion. This simple approach is useful not only because of its low computational cost but also because it does not require one to know whether or not any given vehicle is in motion, just whether or not the area itself contains motion.

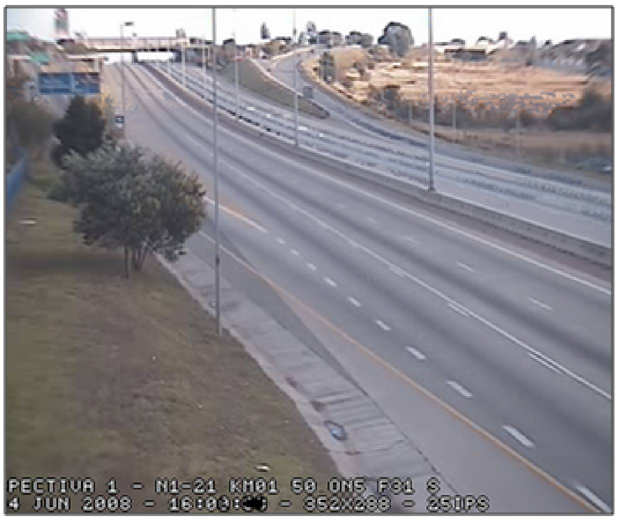

PECTIVA 1 - N1-21 KM01 50 ONS F31 S
4 JUN 2008 - 16:03 - 352X288 - 25IPS

Figure 4.5: The background of the traffic scene is calculated using a mixture of Gaussians. The edges calculated on this image are then removed from the original edge map, leaving only the vehicles.

4.1.2.3 Describing the Scene

Once both the existence of vehicles and their motion are determined, an actual description of the scene can take place. To determine the status of the traffic scene as a whole, each separate window must first be evaluated. The histograms of both the edge (from the calculation of the existence of vehicles) and the difference (from the calculation of motion) maps of each window are evaluated. Accumulating the pixels that are above a certain threshold brightness in both the edge and difference histogram, and accumulating it over the entire window's area, will give a percentage that shows whether or not the window contains a vehicle and whether or not there is motion in that window. A set of rules is then created whereby these results are taken into account in order to determine the status of the traffic. The total percentage of windows that contain vehicles is calculated in order to determine if the traffic scene contains many cars. The motion in each window is used to determine if there is congestion in that particular part of the traffic scene. Combining the two will yield the traffic status.

4.1.3 Results, Discussions and Future Work

The results of this qualitative system are shown in figure 4.6. The traffic is measured in three contextual and six graphical descriptions, which are calculated automatically by the system. Areas that contain congestion are also highlighted in the traffic scene. The three contextual definitions are as follows:

- Low: when few vehicles are detected. This is the case when a certain percentage of those windows that contain a high-edge response fall below a certain threshold.
- Normal: when many vehicles are detected. This is the case when a certain percentage of windows that contain a high-edge response fall above a certain threshold.
- Heavy: when many vehicles are detected and they are mostly congested. This occurs when a certain percentage of the windows that contain a high-edge response reach above a certain threshold and a large percentage of those windows also have a low-motion count.

Formally the traffic status at time t, S_t, is calculated as:

$$S_t = \begin{cases} \text{Low} & \text{if } \sum(E(w_i)) < T \\ \text{Normal} & \text{if } \sum(E(w_i)) > T \\ \text{Heavy} & \text{if } \sum(E(w_i)) > T \text{ and } \sum(M(w_i)) > Q \end{cases} \tag{4.2}$$

$$\text{where } E(w_i) = \begin{cases} 1 & \text{if window } w_i \text{contains a high number of edges} \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

$$\text{and where } M(w_i) = \begin{cases} 1 & \text{if window } w_i \text{has a low motion count} \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

and T and Q are the thresholds chosen as parameters.

This method that was used does not depend upon calculating the physical number

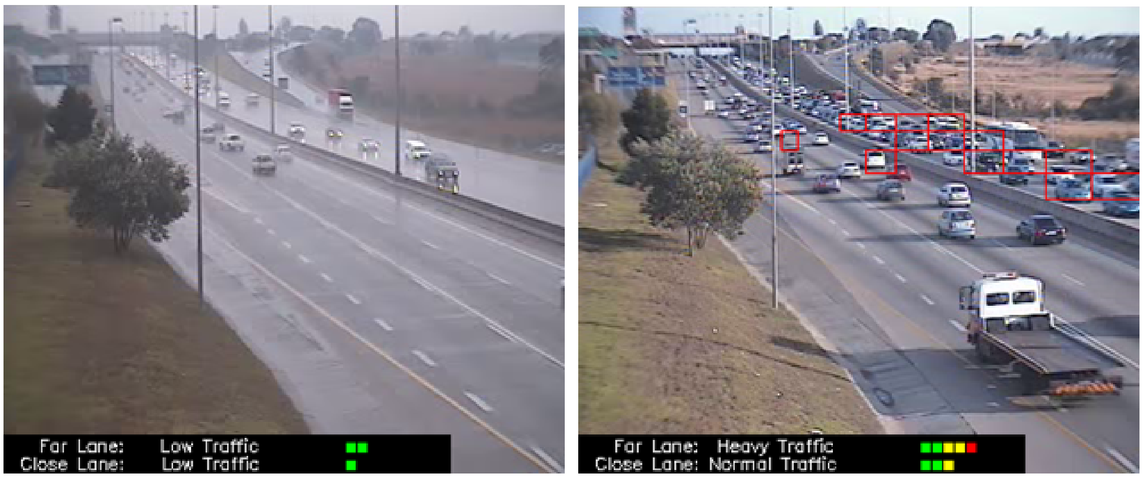


Figure 4.6: Two frames displaying the qualitative results for two scenes. The contextual description of the traffic status as well as a visual representation is shown in the image. Areas of congestion are highlighted on the actual frames. The scene on the left maintains a low traffic status on both carriageways. It also contains no areas of congestion. The scene on the right maintains a heavy traffic status on the far carriageway and a normal traffic status on the closer carriageway. Several areas of congestion are highlighted, especially in the far carriageway. (See Figure A.2 in the Appendix for additional results.)

of vehicles in the scene, but rather it relies on a calculation of the total number of windows that contain a vehicle over those that do not. This process somewhat mimics a human understanding of a traffic scene. Our eyes are able to glance at a particular scene and, without having to count the vehicles, are able to determine whether or not there are many vehicles.

This method also does not require knowledge of the trajectory of each and every vehicle in a scene. If many windows contain low motion in conjunction with detected vehicles, then the traffic scene will be considered as heavily congested. Once again, this mimics a human's comprehension of a traffic scene. We do not need to know where each and every vehicle is moving to, as a general glimpse of the scene is sufficient for us to notice whether or not the traffic is free flowing and this gives us an understanding of the scene.

Table 4.1: Qualitative Analysis Confusion Matrix

Low	Positive	Negative
True	31	25
False	3	1

Normal	Positive	Negative
True	8	44
False	2	6

Heavy	Positive	Negative
True	13	43
False	3	1

Table 4.2: Qualitative Analysis Sensitivity and Specificity

	Low	Normal	Heavy
Sensitivity	97%	57%	93%
Specificity	89%	96%	93%

Table 4.1 displays the results of the qualitative analysis. Different videos showing different weather and lighting conditions were taken into account. Of those videos sixty random timestamps were chosen, half on the far carriageway and half on the close. By viewing several frames before and after a ground truth of those timestamps was created. The ground truth was done by a human describing and writing down what he believed the traffic density was in each of those scenarios. These results were then compared to the results obtained by the qualitative analysis program. A binary classification was given for each of the three possible stages of traffic densities (low, normal and heavy). From this the true positive and negative (where the human and the computer agreed) and the false positive and negative (where the human and computer disagreed) were counted.

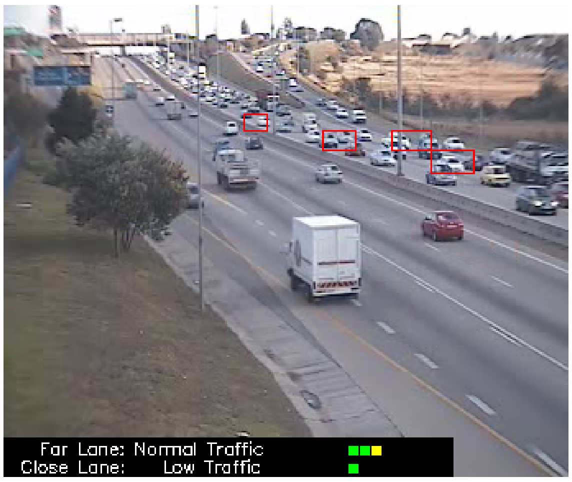
Table 4.2 displays the calculated sensitivity and specificity of the results displayed in table 4.1 for each classification. The results show a high classification for low and heavy traffic, being able to both identify which sets of traffic is low or high and which sets of traffic weren't. However it showed a relative low sensitivity in classifying normal traffic, but remained high with its specificity. This shows that the human ground truth and the computer results disagreed which sets of traffic contained a normal flow of traffic, but was able to mostly agree on which sets of traffic weren't normal flow. This shows one of the drawbacks of a qualitative system. Identifying what is a normal flow of traffic is very subjective between different people, so this can be expected as well between a human marked ground truth and a computer calculated one.

Figure 4.7 shows a few selected examples of the results obtained by the qualitative analysis program and their respective ground truth.

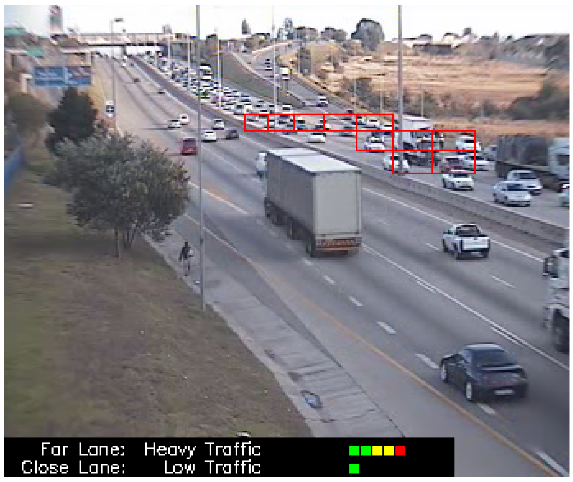
In Figure 4.7(a) the ground truth and qualitative results are the same showing a true positive response for normal traffic and low traffic on the far and close carriageways respectively.

Figure 4.7(b) shows a true positive response for low traffic on the close carriageway. However it shows the algorithm obtaining a false negative response for normal traffic on the far carriageway. This is where the human observer described the traffic status as normal, while the program described it as heavy. In the minimal cases where there is very heavy traffic in a scene, which is still flowing quite well, a human observer might, correctly, regard that as normal traffic. However the vast number of complex objects with their many edges might cause the program to detect more congestion than there actually is. This can be because the program might confuse the edges in one area from two different objects as the same object, thereby incorrectly classifying those areas as congestion thereby downgrading the traffic status from normal to heavy.

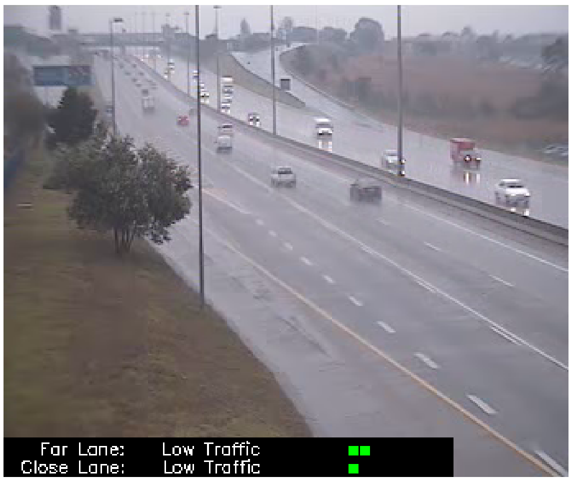
Figure 4.7(c) shows the algorithm working in bad weather and lighting conditions. The program mitigates the problem of bad visuals from bad weather by it's edge detection algorithm. The external morphological gradient is quite robust for edge de-



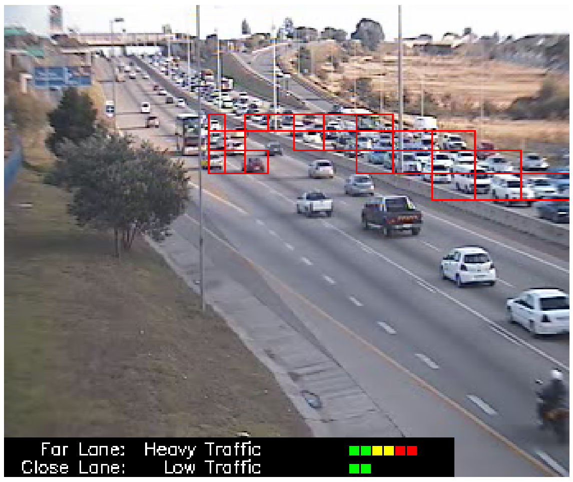
(a) Ground Truth Far: Normal
Ground Truth Close: Low



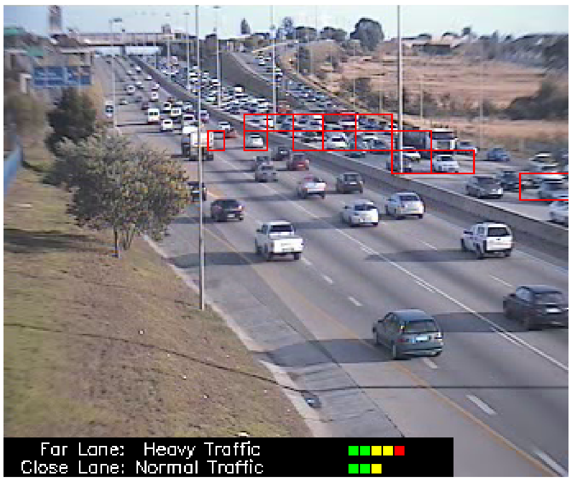
(b) Ground Truth Far: Normal
Ground Truth Close: Low



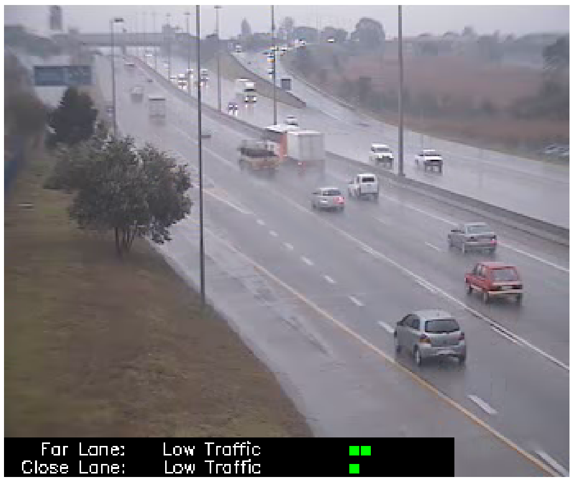
(c) Ground Truth Far: Low
Ground Truth Close: Low



(d) Ground Truth Far: Heavy
Ground Truth Close: Low



(e) Ground Truth Far: Heavy
Ground Truth Close: Normal



(f) Ground Truth Far: Low
Ground Truth Close: Normal

Figure 4.7: Selected Results: These are selected results showing the comparison between the computer output (shown in the figure), and human analysed ground truth for each frame (underneath the figure) for each carriageway.

tection. It is sometimes too sensitive for detecting edges, however combining it with a background subtraction technique helps in that regard. This gives this program the robustness it needs to overcome bad visuals without it detecting too many edges.

Figure 4.7(d) once again shows that the program and the ground truth agreeing on the traffic status. A true positive response for heavy traffic was detected in the far carriageway and a true positive response for low traffic in the close carriageway.

Figure 4.7(e) shows the algorithm working even when large shadows appear in the scene. The algorithm is robust when it comes to shadows as it uses volume of the edges of foreground regions instead of the volume of the entire object. A shadow is not a complex object so the amount of edges it produces from an edge detection algorithm is minimal compared to the number of edges in a complex object like a car.

Figure 4.7(f) shows a true positive response for low traffic on the far carriageway. However it shows the algorithm obtaining a false positive response for low traffic on the close carriageway. This could be due the human understanding of weather conditions and not the program. The program is designed to mitigate the effects of weather graphically. However the understanding of weather effects on traffic is not present. A human might see the same number of cars traveling at the same speed on two different scenes, one being bad weather the other good and give a different description of the traffic status of both.

There is room for this qualitative analysis to be improved upon in the future. For instance, one small improvement would be to make it work in a more generic system, such as automatically determining optimal window size and location, without needing to manually calibrate each camera, and automatically identify different regions of interest. A greater automatic understanding of the scene would be an even bigger improvement; for example, figuring out a way to make the analysis identify the causes of the congestion that was detected and highlight it in the traffic scenes. This would

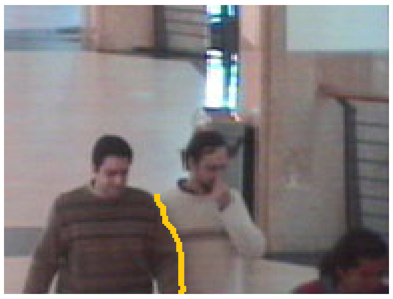


Figure 4.8: Example of two people that need to be separated vertically. The ideal separation boundary is highlighted.

allow for a better emergency response or a better performance in traffic simulations.

4.2 Segmentation of People for Video Surveillance

Occlusion is a major problem in object tracking. This is because very frequently the objects that are needed to be tracked are not isolated. In the case of people tracking, this could be because a group of people are walking together and the system is unable to distinguish them individually. What is needed to accurately track each person is a way of segmenting them. Many different techniques have been developed, such as the one by Elgammal [4], however Elgammal's work requires that the people enter the scene separately before they form a group. This is because his method calculates information prior to the occlusion in order to determine each person's characteristics. As this cannot always be the case, a method that does not have to rely on prior information would be a great advantage to occlusion handling.

A novel method for segmenting people in a crowd scene is proposed in this section. This method reduces the number of errors in tracking (see figure 4.8). It does not require any prior information about the objects, such as their features, and it thus decreases the number of possible errors that exist in those techniques that require initialisations, such as that in Elgammal's work.

4.2.1 Component Isolation

In order to analyse and detect how many people need to be separated and where to separate them, each connected region's shape is considered. All the connected regions are extracted by first running a background subtraction on the video, then performing connected component detection on the foreground mask.

With the use of the mixture of Gaussians (see section 3.4) and the application of dilations and erosions, the foreground region is extracted with several holes removed. (Experiments showed that a 'double closing' removed most of the holes whilst still maintaining the relative shape of each object). In order to isolate each component, a connected component algorithm is applied and all the components that are smaller than a pre-set threshold are removed. This is done by taking the foreground region that was extracted by the background subtraction and then finding separate groups of these regions. Each group is called a component and usually represents a single object in an image.

However, since more than one person can be moving in the same area, these components often represent one or more objects. One needs to locate those components that represent multiple objects and separate them into single objects.

4.2.2 Shape Analysis

To perform a vertical separation, the horizontal shape of each component needs to be considered. This is done by projecting the maximum height of each vertical column onto a histogram (see 4.9).

In other words, we must first find the bounding box $[x_b, y_b] \times [x_e, y_e]$ (see figure 4.10), around the connected component B to be split. Then let F be the binary foreground mask obtained from the background subtraction. We will then have:

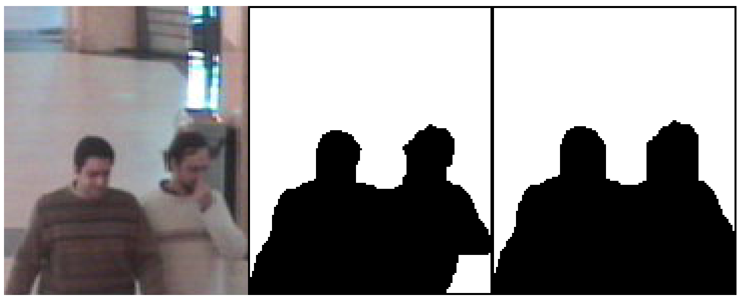


Figure 4.9: Example of converting an image to a horizontal height projection histogram. Left: original image. Middle: detection of significant blob. Right: horizontal height projection histogram.

$$B = \{(x, y) \in [x_b, y_b] \times [x_e, y_e],\ F(x, y) = 1\} \tag{4.5}$$

The horizontal height projection histogram $h(y)$ (see figure 4.10) is thus defined as follows:

$$h(y) = x_e - x_{min}(y) + 1 \quad \text{for } y \in [y_b, y_e] \tag{4.6}$$

where

$$x_{min}(y) = \min_{x \in [x_b, x_e]} \{x,\ F(x, y) = 1\} \tag{4.7}$$

As we are separating people, the distinct shape of a person can be easily described as a unimodal horizontal height projection histogram around each mode. This is true for the usual case of when a person's head appears above their shoulders. Therefore, when analysing the horizontal height projection histogram of any component, it can be concluded that the number of people within this component is at least the number of modes detected in the histogram. It is clearly shown in figure 4.9 that there are two modes in the histogram, and therefore two people.

In order to minimise the errors that could occur in the horizontal height projection histogram $h(y)$, a smoothing is performed on $h(y)$by adaptively calculating the standard deviation of the Gaussian and applying the Gaussian convolution on $h(y)$ as

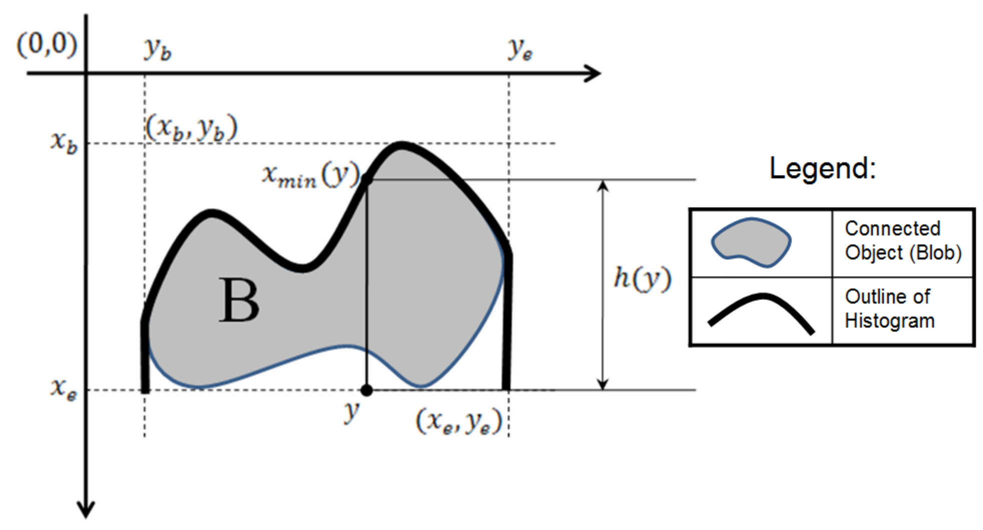


Figure 4.10: Graph describing the formulation of the horizontal height projection histogram [2].

described in [18].

$$H(y) = \int_{-\infty}^{\infty} h(\mu) \frac{1}{\sigma\sqrt{2\pi}} \exp[\frac{-(y-\mu)^2}{2\sigma^2}] d\mu \tag{4.8}$$

Then the first derivative of the Gaussian convolution with respect to $h(y)$:

$$H'(y) = \int_{-\infty}^{\infty} h(\mu) \frac{-(y-\mu)}{\sigma^3\sqrt{2\pi}} \exp[\frac{-(y-\mu)^2}{2\sigma^2}] d\mu \tag{4.9}$$

The standard deviation σ can be calculated using the following algorithm:

1. Initialize $\sigma_0 = 1$.
2. Count the number of zero crossings of $H'(y)$ at $\sigma = \sigma_i$ where $i = \{0, 1, ...\}$.
3. IF it is larger than the number of zero crossings of σ_{i-1} where $i = \{1, 2, ...\}$, or it is less than or equal to 1, THEN calculate $H(y)$ with $\sigma = \sigma_i$.
4. ELSE calculate $\sigma_{i+1} = \sqrt{d_{i+1}}$ where $d = \arg\max$ (distance of all pairs of neighbouring peaks).

Although the Gaussian smoothing does reduce errors when calculating the horizontal height projection histogram, a median filter is applied to the Gaussian smoothed

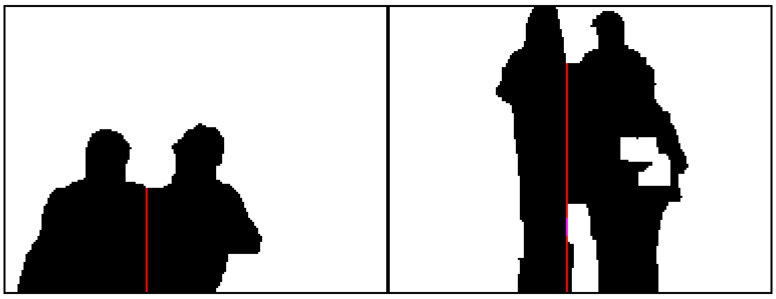


Figure 4.11: Separation by the simple method on two selected frames.

histogram to minimise them further. This removes any small changes in the gradient that might otherwise indicate a trough in the histogram.

Thereafter we count the number of times the gradient of the histogram crosses the zero mark. This number indicates the number of modes within the histogram and therefore the number of people into which the component should be split.

4.2.3 Mode Separation

Once the number of modes in the horizontal height projection histogram is calculated, the separation between the modes must be found in order to separate the people that the histogram represents. Two methods are described below.

4.2.3.1 Simple Method:

One method of separating the histogram into unimodal segments is to separate it whenever the gradient of the histogram crosses the zero mark. In figure 4.11 below this method has been used to split the component into separate unimodal segments for two different frames.

4.2.3.2 Otsu's Method:

A common method for separating a multi-modal histogram into separate unimodal parts is Otsu's method [23]. Otsu's method calculates the optimal thresholds $\{t_1, t_2, ..., t_M\}$, where M is the number of modes minus one, by calculating the maximum between-class variance σ^2_{BCV}.

$$\{t_1^*, t_2^*, ..., t_M^*\} = \arg \max_{(t_1, t_2, ..., t_M)} \{\sigma^2_{BCV}(t_1, t_2, ..., t_M)\} \tag{4.10}$$

where

$$\sigma^2_{BCV}(t_1, t_2, ..., t_M) = \sum_{i-1}^{M} w_i(\mu_i - \mu_T) \tag{4.11}$$

and

$$w_i = \sum_{k=t_{i-1}}^{t_i} \frac{f(k)}{N} \tag{4.12}$$

and

$$\mu = \sum_{k=t_{i-1}}^{t_i} \frac{k(\frac{f(k)}{N})}{w_i} \tag{4.13}$$

where $f(x)$ is the histogram and

$$N = \sum_{i=1}^{L} f(1) \tag{4.14}$$

and L is the length of the histogram.

The original version of this method can become computationally expensive for a large M, however a faster version of Otsu's was developed by Liao et al. [17].

In figure 4.12 Otsu's method has been used to split the component into unimodal segments for two different frames. Note how the simple method and Otsu's method give the same result in the first frame, as shown in figures 4.11 and 4.12, but that the results are quite different in the second selected frame.

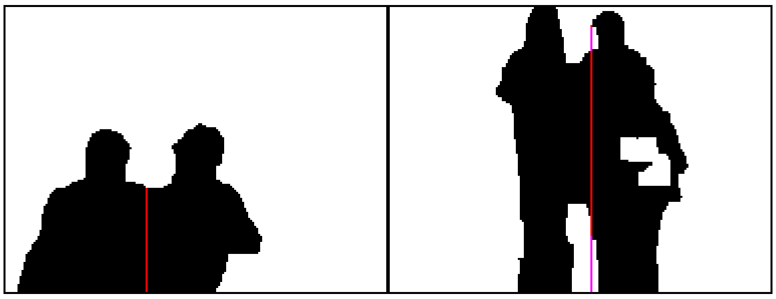


Figure 4.12: Separation by using Otsu's method on two selected frames.

4.2.4 Energy Based Segmentation

The separation based on the horizontal height projection histogram of a component into different objects only depends upon the first pixel. This is hardly accurate, especially as you get further away from the initial pixel. A method is therefore needed for analysing each pixel below the initial pixel in order to separate the component more accurately.

The method developed by Avidan and Shamir [1] reduces the horizontal size of an image by removing a vertical seam, which corresponds to the least energy, thereby not greatly reducing the overall quality of the image. Avidan and Shamir define a vertical seam as a set of 8-connected pixels from the top of the image to the bottom, with only one pixel in each row. (See figure 4.13).

What we want in the present study is to find a vertical seam that corresponds to the highest energy, which is to say we want to identify the pixels of an image that exhibit large changes in their neighbourhood (this is the converse of Avidan and Shamir's method). One method for computing the maximum deviation between two pixels is to calculate the edges of the image and use those edges in the energy calculation. There are many edge detection algorithms, but we will be comparing edges calculated from both the Sobel edge detection (only vertical separation) and the external morphological gradient. The latter is calculated as the difference between the dilation and the erosion of the image. The energy $E(i,j)$ at pixel (i,j) is the pixel intensity

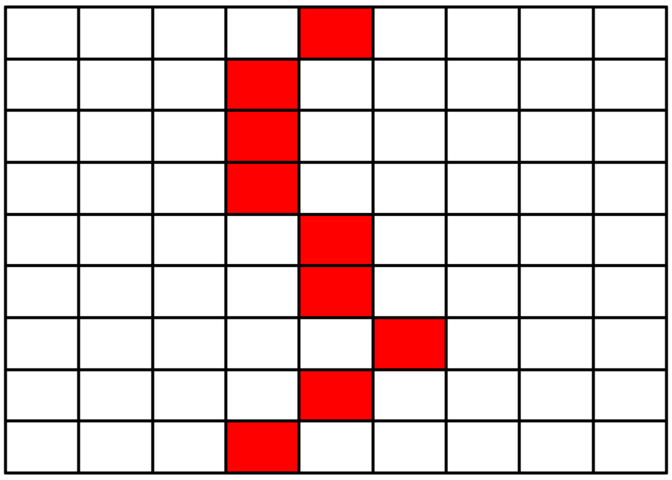


Figure 4.13: A simple example of what a vertical seam looks like on a 9x9 sized image. Note that there is one and only one highlighted pixel in each row. Also note that each highlighted pixel touches the highlighted one both above and below itself.

of the edge detection of image I at (i, j).

The calculation of the vertical seam needs to iterate through each pixel from the initial point to the end (the initial point is evaluated by separating the modes as seen in section 4.2.3) which is done by looking at each pixel below it and finding the one that corresponds to the maximum energy. (See figure 4.14.) This can be done using either a greedy or a dynamic programming algorithm.

4.2.4.1 Greedy Algorithm:

The vertical seam from the initial point derived from the separation of the horizontal height projection histogram to the bottom of the component can be calculated as follows:

1. Initialise the current pixel to the first pixel found by way of the splitting method.
2. Find all pixels in the next row that are 8-connected to the current pixel and calculate their energy.
3. Choose the pixel with the maximum energy and set it as the current pixel. Go

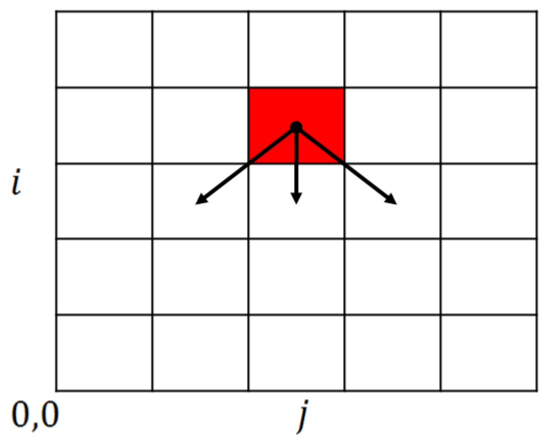


Figure 4.14: Calculating the vertical seam is done by starting at the initial pixel (highlighted red) and iterating through its bottom three neighbours to find the pixel that corresponds to the maximum energy.

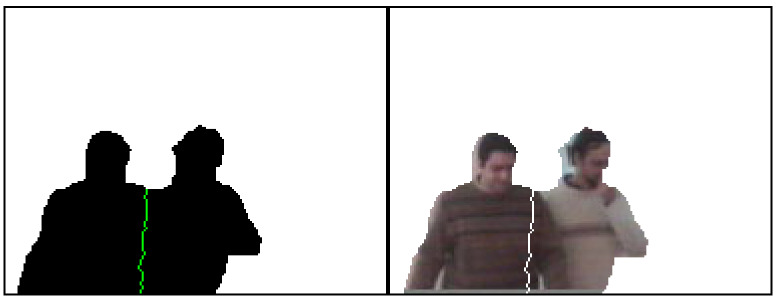


Figure 4.15: Results when using the greedy algorithm. Left: vertical seam calculated. Right: overlaid over original frame.

to step 2 and repeat until the entire seam is calculated.

Figure 4.15 shows the result of applying the greedy algorithm on a selected frame.

The greedy algorithm is a simple and computationally efficient way of calculating the vertical seam, however it will not detect the seam with the maximum energy, as choosing the local maximum at each iteration will not necessarily choose the global maximum.

4.2.4.2 Dynamic Programming Algorithm:

An alternative method for calculating the global maximum is one that uses dynamic programming. However, in some cases the global maximum seam jumps from one extreme to another between frames. This occurs when there are two seams with similar energy. This causes the segmentation method to become less focused on the initial starting point, which often results in it choosing regions on the outside of the object, and this causes the segmentation line to move erratically over the entire object between frames. To remove this effect a limiter is placed on horizontal movement. This limiter is applied by multiplying the energy by the Gaussian at that point. The energy used in the vertical seam is therefore calculated by:

$$E^*(i,j) = E(i,j)\frac{1}{\sigma\sqrt{2\pi}}\exp[\frac{-x^2}{2\sigma^2}] \tag{4.15}$$

where x is the horizontal distance from the initial point and σ the standard deviation calculated as a fraction of the maximum height of the vertical seam (height from initial point to the bottom of the object).

Thereafter to calculate the vertical seam M that has the global maximum energy using dynamic programming, a method similar to the one used in [1] is applied using the following formula:

$$M(i,j) = E^*(i,j) + \max(M(i-1,j-1), M(i-1,j), M(i-1,j+1)) \tag{4.16}$$

where $M(i,j)$ is the vertical seam starting from point (i,j) and moving down the image.

Figure 4.16 shows the results of applying the dynamic programming algorithm on a selected frame.

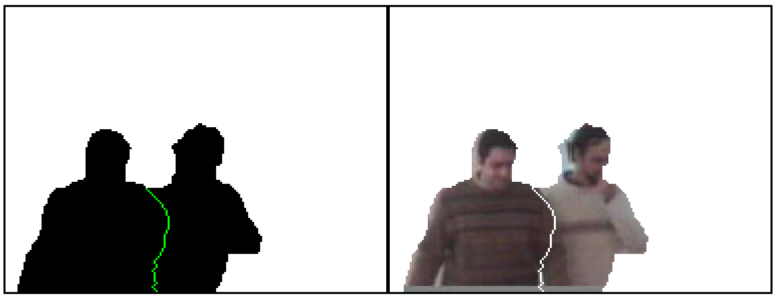


Figure 4.16: Results when using the dynamic programming algorithm. Left: vertical seam calculated. Right: overlaid over original frame.

4.2.5 Experimental Results and Discussions

Several experiments were conducted that included combinations that use methods for histogram separation. These experiments uses either the Sobel edge detection or the external morphological gradient for the energy calculation. They also used either the greedy or the dynamic programming algorithm to evaluate the vertical seam. This then gives eight different combinations to compare against, they include:

1. Simple Separation + Morphological Edge + Greedy Algorithm
2. Simple Separation + Morphological Edge + Dynamic Algorithm
3. Simple Separation + Sobel Edge Detector + Greedy Algorithm
4. Simple Separation + Sobel Edge Detector + Dynamic Algorithm
5. Otsu Separation + Morphological Edge + Greedy Algorithm
6. Otsu Separation + Morphological Edge + Dynamic Algorithm
7. Otsu Separation + Sobel Edge Detector + Greedy Algorithm
8. Otsu Separation + Sobel Edge Detector + Dynamic Algorithm

The results can be found in table 4.3. Accuracy was tested on selected key frames, and the percentage of pixels correctly segmented over the total number of pixels was recorded. In order to interpret the accuracy of the segmentation, each result was

Table 4.3: Table showing the results of the eight different methods for splitting people

Exp. No:	Mean Accuracy (%)	(%) excl. Incorrect Modes	Run Time (fps)
1	91.06	97.68	6.97
2	90.65	97.66	3.79
3	89.23	93.95	7.71
4	90.22	95.52	4.12
5	91.91	98.40	7.43
6	93.38	97.86	3.12
7	88.10	95.22	8.36
8	92.55	96.92	3.29

manually checked and any pixels belonging to another object were counted as incorrectly classified pixels. Because mode counting is not always accurate when it comes to evaluating the number of people in a group (usually because of background subtraction problems or because people are situated directly behind other people), there are several cases in our experiments where the number of separation boundaries did not equate to the number of people. Figure4.17b is an example of where the background subtraction creates large holes in the foreground objects, thereby distorting the shape of the object. Figure 4.17c is an example of where a person is walking directly behind another, thereby hiding her mode. This result will impact negatively on the accuracy of the results, and another set of results is therefore shown that only includes the accuracy when the number of modes was correctly calculated. The following equations explain the accuracy calculations formally:

$$\text{Accuracy} = \frac{\text{Number of Pixels Corrctly Classified}}{\text{Total Number of Pixels}} \tag{4.17}$$

$$\text{Mean Accuracy} = \frac{1}{N}\sum_{i=1}^{N}\text{Accuracy}_i \tag{4.18}$$

where Accuracy_i is the accuracy of the i^{th} frame, and N is the number of frames that were tested.

Table 4.3 shows that experiment 6 gave the highest accuracy all the frames, while experiment 5 gave the highest accuracy when the modes were calculated correctly. This shows that allowing a greater area for the vertical seam to exist when using the dynamic programming can reduce the errors that are obtained when the number of modes does not equate to the number of people. With experiment 6, 1,289 frames were further analysed and categorised into the following four sections: the first, containing 658 frames, was considered to have been perfectly segmented (see figure 4.17a); the second, with 430 frames, was considered to have been correctly segmented, but not having 100% accuracy (see figure 4.17 f); the third, which contained 200 frames, was incorrectly segmented(such as can be seen in figure 4.17b) because two people were segmented into three; and the fourth, which contained just one frame, had a false segmentation, where a single person was incorrectly segmented as multiple people.

Figures 4.17 and 4.18 show the difference between the results of the best and worst experiments (i.e. experiments 6 and7). Notice that experiment 7 used the greedy algorithm to calculate the vertical seam so that when the number of modes is calculated incorrectly the error is compounded (see figure 4.18b for an example). This is noticed when viewing the results that excluded the frames where the modes were incorrectly calculated; note that the difference between the experiments that use the greedy algorithm and the experiments that use dynamic programming do not vary as much as when those frames with incorrectly calculated modes are included.

4.2.6 Conclusions and Future Work

A method for separating groups of people walking together has been presented. This method could be used to facilitate the tracking of single people instead of entire groups. In cases where a person's head is not well separated, the number of modes was incorrectly detected, which reduced the accuracy of the overall result.

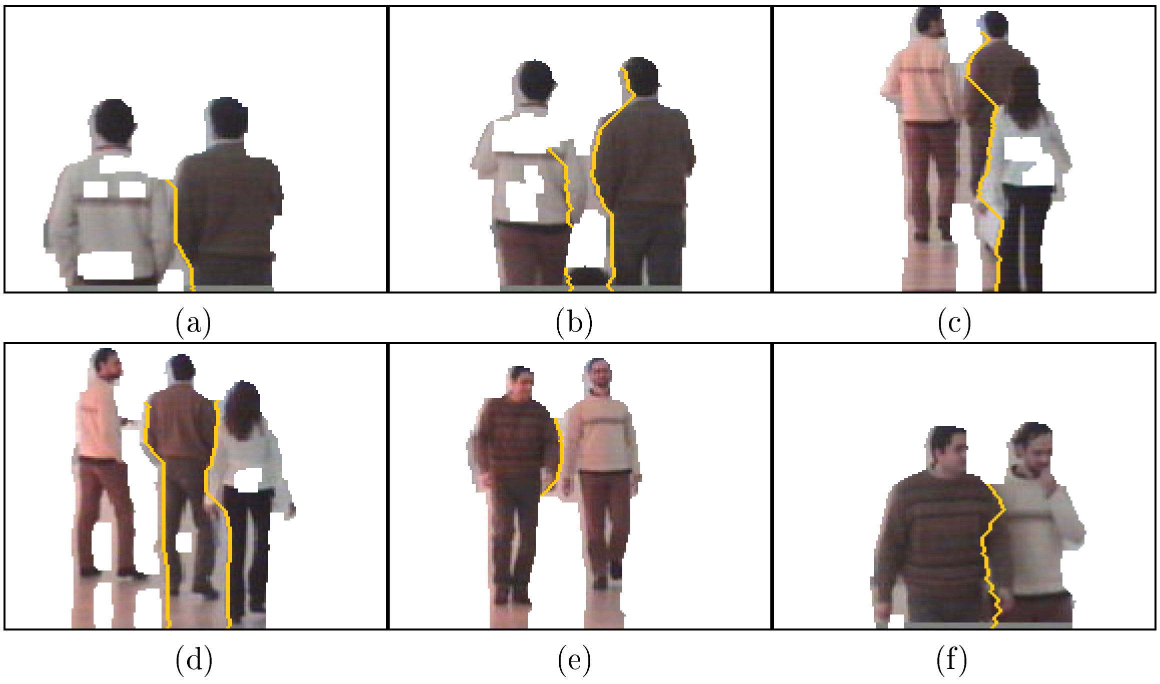


Figure 4.17: Overall best experiment: selected frames showing results (segmentation line highlighted in yellow) overlaid with original data using experiment no. 6 (best results).

It can be seen that this method can vertically separate people in a video sequence.

There are many ways to further extend this method: for instance, alternate techniques for mode detection and separations in the horizontal height projection histogram could be found. It may also be possible to devise different energy functions instead of using the intensity of the edges calculated with Sobel edge detection or the external morphological gradient. However, a more direct improvement to segmentation accuracy would be achieved by detecting more accurately the number of people in a single component.

A simple example of the type of future work that is possible can be seen in the appendix (see figure A.3).

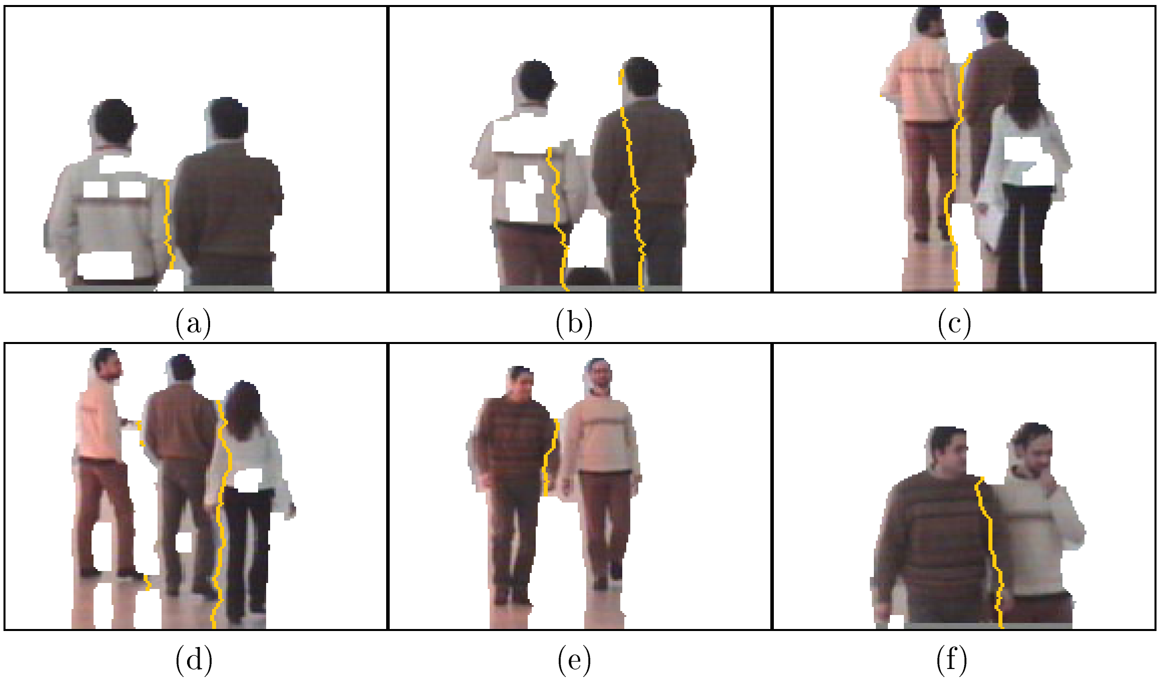

(a) (b) (c) (d) (e) (f)

Figure 4.18: Overall worst experiment: selected frames showing results (segmentation line highlighted in orange) overlaid with original data using experiment no. 7 (worst results).

Chapter 5

Conclusions

5.1 Conclusions

This thesis has provided a detailed description of the process of scene analysis in relation to video surveillance. It first incorporated a test of several background subtraction techniques and compared their advantages and disadvantages with respect to the different types of scene analysis. It was found that the mixture of Gaussians method of background subtraction was best suited to both qualitative and structural analysis. This method reflected the research that has gone into the mixture of Gaussians in that it is usually pretty good in quite a few different types of scene.

The thesis has also given a detailed description of how a qualitative analysis is performed on a traffic surveillance sequence. Many details are extracted from the scene and, accounting for variance in scale and lighting conditions, the qualitative scene analysis system compresses this information, by using several rules, into a description of the traffic scene automatically. The qualitative can tell any user the status of a traffic scene, whether it be low, medium or heavy. This is extremely useful as it can first of all compress all the statistical data extracted by the system into a single meaningful description and then secondly include a more detailed analysis for the sake of the operator of the system, which it does by highlighting the areas that contain the congestion. The description, easily understood by any driver, could be posted onto the electronic notice boards that are constructed across many roads.

This thesis has also shown how a structural scene analysis can be used to help solve the problem of occlusion detection. In the domain of indoor surveillance of people, the system can, without any unique knowledge of each person, separate a group of people into single persons. This is done by taking the general understanding of a shape of a person, combining it with the knowledge that at the exact point where two people meet the energy at each pixel is high, and then creating a vertical seam based on that information. This vertical seam is used as the boundary line between the two different people, thereby segmenting them. This segmentation can be used in two ways. The first is by using it solely to solve the occlusion problem. In many cases this will segment a group of people into their unique persons. The second way this segmentation can be used is in conjunction with those existing methods that solve the occlusion problem by extracting unique features and using them to compute which object is which. As was mentioned, existing methods can fail when the objects enter the scene already occluded, but the failure rate under such circumstances can be lowered with the use of the technique described in this thesis.

5.2 Limitations

There are several limitations that must be considered when these techniques are used. One major concern is the processing time. If these techniques are to be used in real time and in conjunction with other techniques, then we require an understanding that as a prototype these techniques might not adhere to this factor. Even with fast and accurate background subtraction techniques, such as the mixture of Gaussians method, combining them with other methods in both qualitative and structural analysis will increase the processing time. Without some time dedicated into optimising and a physical hardware review these methods will often exceed the time limits for real time analysis. As both the qualitative and structural analyses are prototypes, the code was written in un-optimised and non-parallelised code. Therefore with further research it may be possible to decrease the processing time of these algorithms.

Actual camera setup is also a limitation in these systems. In the qualitative analysis system the program is only designed for a camera that is placed at a low angle and to the side of the road. For another camera setup on another stretch of road it will require input into the actual program and an understanding of how the algorithm works. In the structural analysis the position of the camera must be at a downward angle and it must also face onto the section of road where you require surveillance.

5.3 Future Work

There are many more background subtraction techniques that were not tested as well as some new ones that are currently being developed. The use of the mixture of Gaussians method might not be the optimal method for these systems in the future. Methods that can increase the overall performance or reduce errors in the system, such as the elimination of holes for use in the structural analysis, would greatly improve results. With the elimination or reduction in the number holes in the foreground objects, an increase in accuracy for the structural analysis is almost guaranteed.

In terms of qualitative analysis, future work could be directed towards an automatic setup in order to facilitate the introduction of new cameras on different stretches of road. This automatic setup would remove the need for a person who is familiar with the integral algorithms used in the system whenever a new camera is installed. This automatic feature could include identifying each carriageway on a road, or even calculating the vanishing point from the camera's perspective in order to automatically size the windows accordingly.

Another area that could use further development is the understanding of the scene. Areas that are already highlighted as being congested can be further analysed and an understanding could be generated. This would help the average driver, as he/she would be able to read on electronic notice boards the reason for the road's congestion.

This development would also help the safety officer monitoring the system to locate actual hazards or accidents and dispatch emergency services accordingly.

Further work in structural analysis could take place in terms of either improving the system itself or incorporating it into other systems. Direct improvement to this system could be achieved by counting the number of modes more accurately and by recognising when the number is in fact incorrect. At present, the system can only calculate the lower bound of how many people could be in the group. The number usually does not exceed the lower bound by much, unless there is a heavily dense crowd. But by looking at table 4.3 and by comparing the accuracy between when the whole sample is used and the sample when the incorrect modes are excluded one can see that by solving this problem the accuracy is increased.

Appendix A

Additional Analysis

A.1 Background Subtraction

Figure A.1 shows additional examples of the background subtraction techniques described in Chapter 3.

A.2 Qualitative Scene Analysis

Figure A.2 provides some additional examples from the results of the qualitative analysis that were shown in Chapter 4 Section 4.1

A.3 Structural Scene Analysis

In figure A.3 is an example of a simple way to use the technique discussed in Chapter 4, section 4.2 as an extension for use in tracking applications. As is discussed in section 4.2.6, the technique can be used to further existing methods.

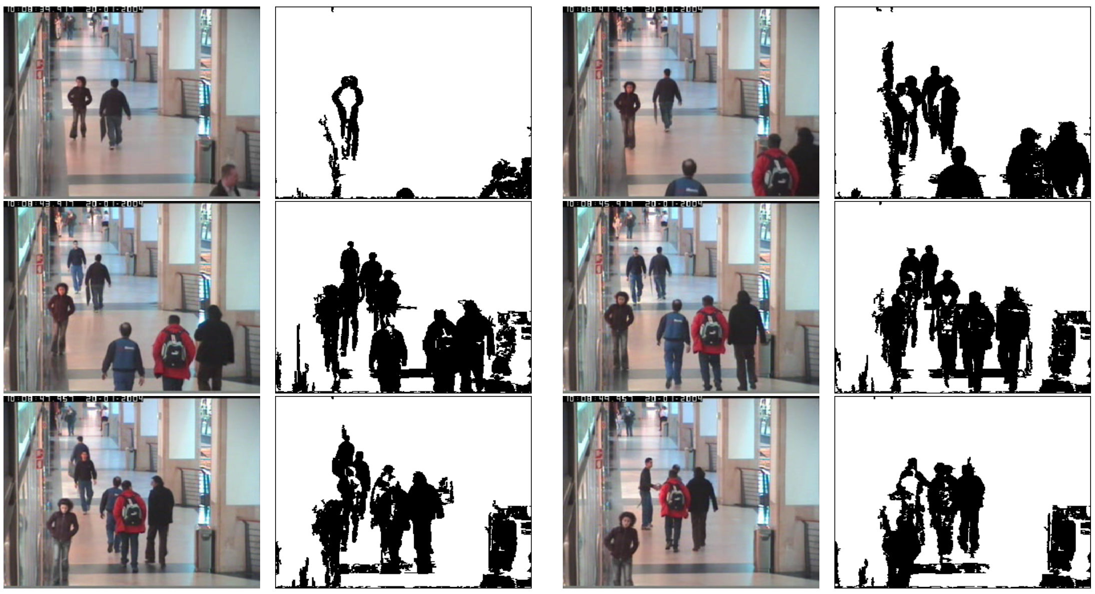


Figure A.1: Additional selective updating examples, as described in section 3.2 with the discussion of results in section 3.7.2. Left: Original Frames. Right: Foreground Mask

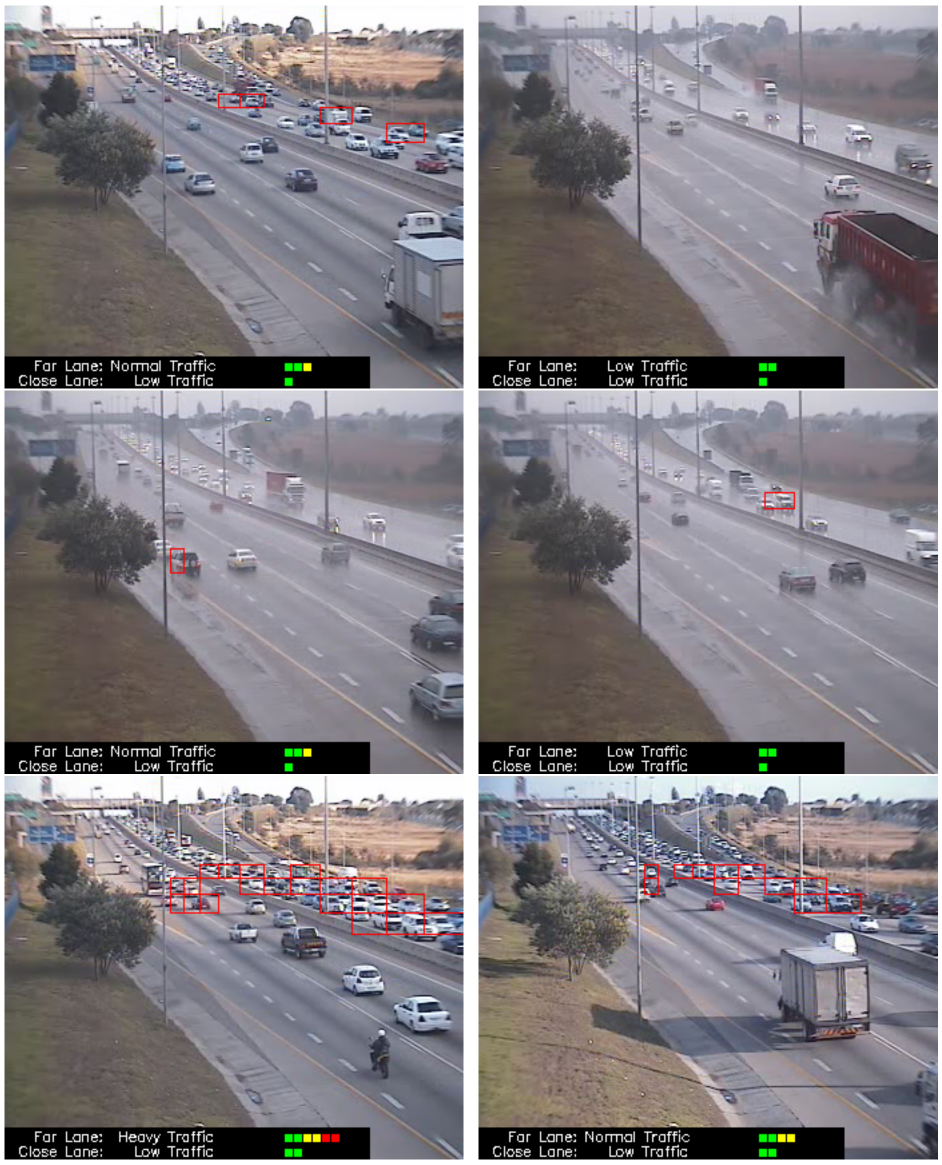


Figure A.2: Additional qualitative analysis examples. Results of the qualitative analysis are displayed in each image in the bottom left-hand corner. The results shown are a textual as well as graphical description of the volume of traffic generated automatically by the system. In addition, certain areas may be highlighted, which indicates the presence of congestion in that area.

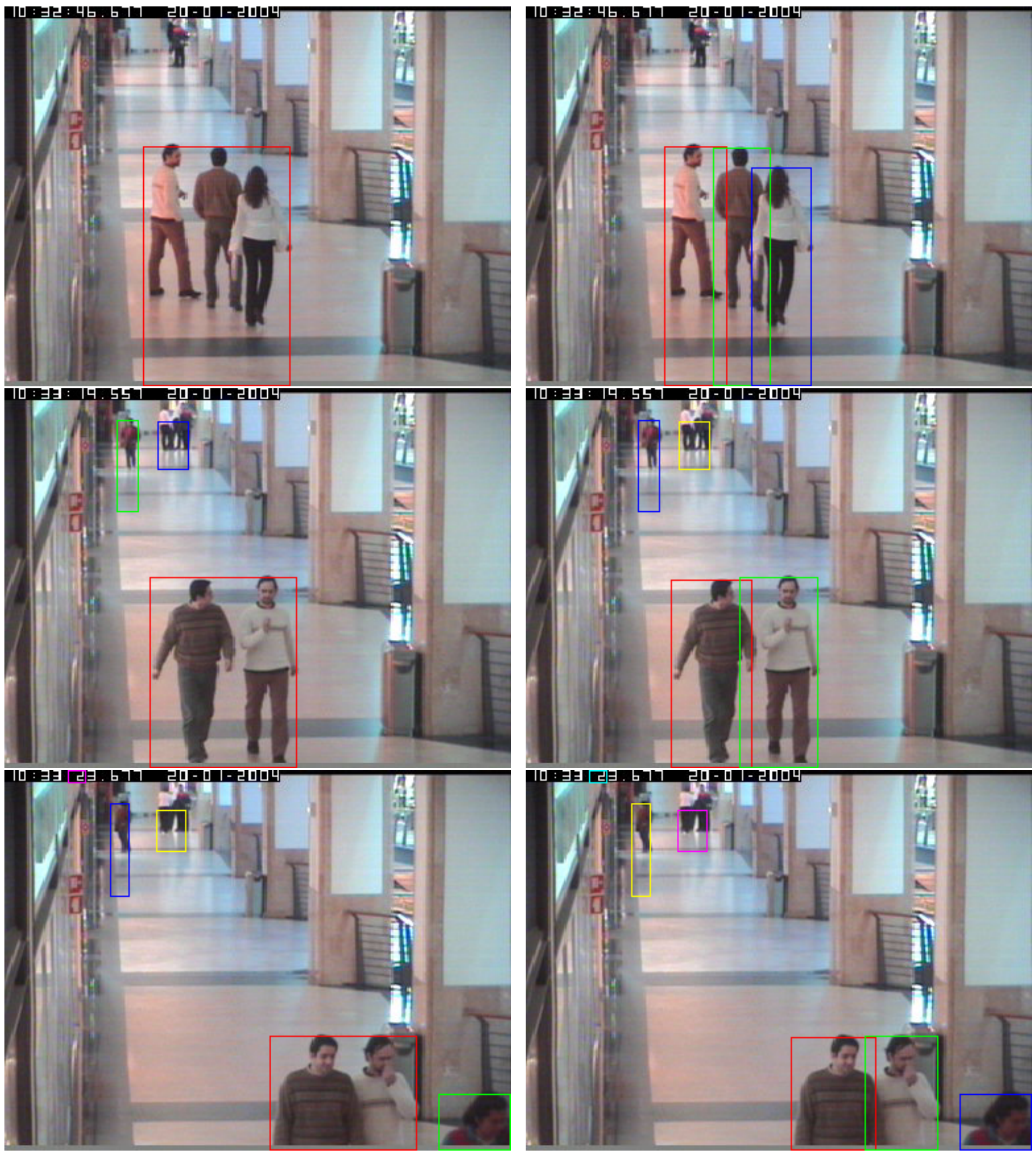


Figure A.3: Here is an example of the technique discussed in Chapter 4, section 4.2. On the left: a simple tracking algorithm is in place. And on each image's corresponding right-hand side one can see the results of same tracking algorithm with just the structural analysis technique in place. Notice how the objects are uniquely tracked on the right-hand side when compared with the left-hand side.

Bibliography

[1] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. *ACM Trans. Graph.*, vol. 26, no. 3, p.10, 2007.

[2] A.M. Brits and J.R. Tapamo. A shape and energy based approach to vertical people separation in video surveillance. In *Advances in Visual Computing, 5th International Symposium, ISVC 2009, Las Vegas, NV, USA, November 30 - December 2, 2009, Proceedings, Part II*, pages 345–356.

[3] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, vol. 39, no. 1, p. 1-38, 1977.

[4] A.M. Elgammal and L.S. Davis. Probabilistic framework for segmenting people under occlusion. In *In Proc. of IEEE 8th International Conference on Computer Vision*, pages 145–152, 2001.

[5] A.M. Elgammal, D. Harwood, and L.S. Davis. Non-parametric model for background subtraction. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part II*, pages 751–767, 2000.

[6] M. Fathy and M.Y. Siyal. An image detection technique based on morphological edge detection and background differencing for real-time traffic analysis. *Pattern Recognition Letters*, vol. 16, no. 12, p. 1321 - 1330, 1995.

[7] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. In *Thirteenth Conf. on Uncertainty in Artificial Intelligence*, pages 175–181, 1997.

[8] P.F. Gabriel, J.G. Verly, J.H. Piater, and A. Genon. The state of the art in multiple object tracking under occlusion in video sequences. In *In Advanced Concepts for Intelligent Vision Systems (ACIVS), 2003*, pages 166–173, 2003.

[9] I. Haritaoglu, D. Harwood, and L.S. Davis. Hydra: Multiple people detection and tracking using silhouettes. *Image Analysis and Processing, International Conference on*, page 280, 1999.

[10] I.T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 1986.

[11] P. Kaewtrakulpong and R. Bowden. An improved adaptive background mixture model for realtime tracking with shadow detection, 2001.

[12] F. Kahl, R.I. Hartley, and V. Hilsenstein. Novelty detection in image sequences with dynamic background. pages 117–128, 2004.

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

[14] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, (no. 4, 321-331), January 1988.

[15] A.C. Lara and R. Hirata, Jr. A morphological gradient-based method to motion segmentation. In *International Symposium on Mathematical Morphology, 8 (ISMM)*, volume 2, pages 71–72. Universidade de São Paulo (USP), 2007.

[16] D. Lee. Effective gaussian mixture learning for video background subtraction. *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 5, p. 827–832, 2005.

[17] P. Liao, T. Chen, and P. Chung. A fast algorithm for multilevel thresholding. *Journal of Information Science and Engineering*, vol. 17, p. 713-727, 2001.

[18] H. Lin, L. Wang, and S. Yang. Automatic determination of the spread parameter in gaussian smoothing. *Pattern Recogn. Lett.*, vol. 17, no. 12, p. 1247–1252, 1996.

[19] S. Lin, J. Chen, and H. Chao. Estimation of number of people in crowded scenes using perspective transformation. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, vol. 31, no. 6, p. 645-654, 2001.

[20] A. Mittal, A. Monnet, and N. Paragios. Scene modeling and change detection in dynamic scenes: A subspace approach. *Comput. Vis. Image Underst.*, vol. 113, no. 1, p. 63-79, 2009.

[21] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh. Background modeling and subtraction of dynamic scenes. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 1305. IEEE Computer Society, 2003.

[22] R M. Neal and G.E. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. pages 355–368, 1999.

[23] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, p. 62-66, January 1979.

[24] C. Ridder, O. Munkelt, and H. Kirchner. Adaptive background estimation and foreground detection using kalman-filtering. In *Proc. of the International Conference on Recent Advances in Mechatronics*, pages 193–199, 1995.

[25] P.L. Rosin. Unimodal thresholding. *Pattern Recognition*, vol. 34, no. 11, p. 2083-2096, November 2001.

[26] B. Shoushtarian and H.E. Bez. A practical adaptive approach for dynamic background subtraction using an invariant colour model and object tracking. *Pattern Recogn. Lett.*, vol. 26, no. 1, p. 5-26, 2005.

[27] M.Y. Siyal and M. Fathy. Image processing techniques for real-time qualitative road traffic data analysis. *Real-Time Imaging*, vol. 5, no. 4, p.271 - 278, 1999.

[28] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition.*, pages –252 Vol. 2, 1999.

[29] C.R. Wren, A. Azarbayejani, T. Darrell, and A.P. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, p. 780-785, 1997.

[30] M. Yokoyama and T. Poggio. A contour-based moving object detection and tracking. In *ICCCN '05: Proceedings of the 14th International Conference on Computer Communications and Networks*, pages 271–276, 2005.

[31] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, vol. 2, p. 23-31, 2004.