

SMART VIGILANCE

“An Algorithm for Multiple Objects Tracking and Velocity Estimation”

by

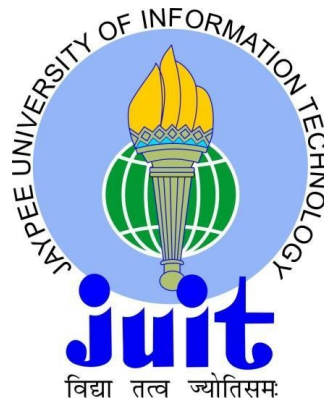
Name: PrateekVermani (101118)

Deepika Jindal (101130)

Himanshu Gupta (101133)

Under the supervision of

Dr. Davinder Singh Saini



May-2014

Dissertation submitted in partial fulfilment

Of the requirement for the degree of

BACHELOR OF TECHNOLOGY

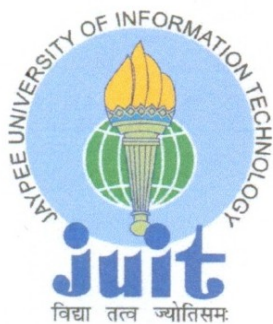
IN

ELECTRONICS & COMMUNICATION ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

WAKNAGHAT, SOLAN – 173234, INDIA



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

(Established under the Act 14 of Legislative Assembly of Himachal Pradesh)

Waknaghat, P.O. Domehar Bani. Teh. Kandaghat, Distt. Solan- 173234(H.P)

Phone: 01792-245367, 245368, 245369

CERTIFICATE

This is to certify that the work titled **SMART VIGILANCE** submitted by **Mr. Prateek Vermani, Ms. Deepika Jindal and Mr. Himanshu Gupta** in the partial fulfillment for the award of degree of Bachelor of Technology (ECE) of Jaypee University of Information Technology, Waknaghat has been carried out under my supervision. This work has not been submitted partially or wholly to any other university or institution for the award of this or any other degree or diploma.

Dr. Davinder Singh Saini

Assistant Professor(Grade-2)

Department of electronics and communication engineering

Jaypee University of Information Technology (JUIT)

Waknaghat, Solan – 173234, India

(Supervisor)



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

(Established under the Act 14 of Legislative Assembly of Himachal Pradesh)

Waknaghat, P.O. DomeharBani.Teh.Kandaghat, Distt. Solan- 173234(H.P)

Phone: 01792-245367, 245368,245369

Fax-01792-245362

DECLARATION

We hereby declare that the work reported in the B. Tech thesis entitled **SMART VIGILANCE** - "An Algorithm for multiple objects Tracking and Velocity estimation" submitted by "Mr.PrateekVermani, Ms. Deepika Jindal and Mr. Himanshu Gupta" at Jaypee University Of Information Technology, Waknaghat is an authentic record of our work carried out under the supervision of **Dr. Davinder Singh Saini**. This work has not been submitted partially or wholly to any other university or institution for the award of this or any other degree or diploma.

Mr. PrateekVermani (101118)

Ms.Deepika Jindal (101130)

Mr. Himanshu Gupta (101133)

Department of electronics and communication engineering

Jaypee University of Information Technology (JUIT)

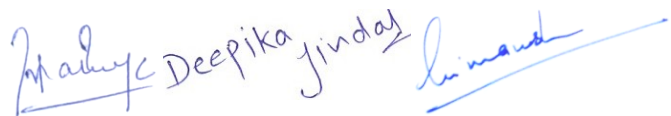
Waknaghat, Solan – 173234, India

©Copyright by
PrateekVermani, Deepika Jindal, Himanshu Gupta

2014

ACKNOWLEDGEMENT

It is divine, grace and blessing of god that today we have successfully reach milestone of our journey in this endless path of learning that has just begun. After the competitions of our project work, we feel to convey our indebttness to all those who helped us to reach our goal. We take this opportunity to express our profound gratitude and deep regards to our guide (Mentor Dr D.S.Saini) for his exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by him time to time shall carry us a long way in the journey of life on which we are about to embark. We are obliged to all our faculty members of JUIT, for the valuable information provided by them in their respective fields. We are grateful for their cooperation during the period of our project. Lastly, we thank almighty, our parents, brothers, sisters and friends for their constant encouragement without which this project would not be possible.



Prateek Vermani

Deepika Jindal

Himanshu Gupta

ABSTRACT

Today Security is of very much significance and a lot of electronic equipment is being used in security applications. Monitoring continuously the movements of human beings or vehicles and reporting when predefined events take place is a very common security application. A human observation based system for implementing this has several disadvantages. In the golden days people use to be employed for doing such observations. Decades before electronic cameras could solve the problem of man being physically present at such place. Instead man has to observe the camera's output on a TV and can detect when any expected events occur. The present day technology allows automatic detection based on predefined measures. This project deals with the tracking and following of Vehicles in a sequence of frames and its velocity determination. Algorithms are developed in MATLAB for improving the image quality, segmentation, feature extraction and for determining the velocity. In this foreground detection based moving object detection and vehicle tracking algorithm is implemented targeting a wide class of applications.

INDEX

Certificate	
Declaration	
Acknowledgement	
Abstract	
List of Figure	
1. Chapter 1 Introduction	
1.1 Introduction.....	1
1.2 Motivation.....	1
1.3 Objective	1
1.4 Software Used.....	2
2. Chapter 2 Theoretical Background	
2.1 Object Tracking.....	3
2.2 Different Aspects of Object Tracking.....	3
2.3 Background Initialization.....	19
2.4 Foreground Detection.....	23
2.5 Data Validation.....	28
3. Chapter 3 Proposed Algorithm	
3.1 Input Parameters for Mounted Camera.....	30
3.2 Reading AVI File.....	30
3.3 Frame Separation.....	31
3.4 Thresholding.....	31
3.5 Background Subtraction.....	31
3.6 Morphological Operations.....	32
3.7 Blob Analysis.....	33
3.8 Creating Bounding Box.....	33
3.9 Counting Number of Vehicles.....	33
3.10 Speed Detection.....	33
3.11 Speed Violation.....	34
4. Chapter 4 Functions Used	
4.1 Input Dialog.....	35
4.2 Vision Foreground Detector.....	36
4.3 String to Number.....	36
4.4 Vision Video File Reader.....	37

4.5 Step.....	38
4.6 Function Handle.....	38
4.7 Axes.....	39
4.8 Morphological Structuring Element.....	41
4.9 Vision Blob Analysis.....	45
4.10 Insert Shape.....	48
5. CHAPTER 5 Output.....	49
Conclusion	
References	

LIST OF FIGURES

Figure 1: Background Subtraction Modelling.....	4
Figure 2: Algorithm.....	30
Figure 3:Input Dialog Box.....	34
Figure 4: Foreground of an Input Frame.....	35
Figure 5: Original Video Frame.....	37
Figure 6: Handling Multiple Axis in a Single Window.....	38
Figure 7: Axes.....	39
Figure 8: Basic and Matrix.....	40
Figure 9: Diamond Shaped strel Matrix.....	41
Figure 10: Disk Shaped strel Matrix.....	42
Figure 11: Angle in strel Matrix.....	42
Figure 12: OFFSET in strel Matrix.....	43
Figure 13: Rectangle Shaped strel Matrix.....	43
Figure 14: Width determined Square strel Matrix.....	43
Figure 15: Foreground after Morphological Operations.....	44
Figure 16: Blob Analysis Block Diagram.....	44
Figure 17: Frame after Inserting Various Shapes.....	47
Figure 18: Initial Screen.....	48
Figure 19: Smart Vigilance is Selected.....	48
Figure 20: Output after Road Length is Entered.....	49
Figure 21: Vehicles Appear.....	49
Figure 22: Speed Limit is Violated.....	50
Figure 23: Snapshot of a Vehicle.....	50
Figure 24: Images saved to a Folder.....	51

CHAPTER 1

INTRODUCTION

1.1 Introduction

The Road & Traffic industry has many stakeholders and multiple departments including security, safety and management - each of which have their own requirements.

Smart Vigilance is a comprehensive portfolio of applications that can fulfil each department's specific requirements while operating as a single integrated system to ensure quick distribution of information to all concerned when any event occurs.

Smart Vigilance has a comprehensive monitoring system to monitor the entire city's traffic, from high speed freeways to low-speed areas, such as schools, shopping malls and hospitals. Many other monitoring systems, require induction loops, radar and other technology that require civil works. This system is highly accurate and the cost of the implementation is only a fraction of the cost of traditional systems.

1.2 Motivation

Today we can see video Surveillance cameras everywhere. They are prevalent in banks, stores, parking lots etc. but there the video data currently used only is 'after the fact'. Mounting Video camera is cheap but finding sufficient human resources to observe the output is expensive.

WHAT IS REQUIRED????

Continuous 24-hour monitoring of surveillance video to alert security officers to a burglary in progress, or to a suspicious individual loitering in the parking lot, while there is still time to prevent the crime.

So the actual motivation behind this project is to eradicate this problem and to develop software for real time in tracking, solving the problem of velocity estimation of vehicles seeing the major application of this in future in areas of security, surveillance and vision analysis.

1.3 Objective (Problem Statement)

In the recent times number of theoretical and/or simulation studies were done on the topic of object-tracking. While these studies are useful, they are too general and provide little guidance for the developing an object-tracking application and prescribes network configurations that work well with our algorithms. We implement our software using MATLAB software technology. The major issues addressed in this project are the evaluation and efficient use of a tracking algorithm in

tandem with our camera hardware product in a real-world application, and efficient ways to algorithmically analyze the velocity of vehicles and utilize this collected raw data further to solve security issues. It also provides be a great opportunity to explore the new area of security and surveillance.

1.4 Software used

We used MATLAB (Matrix Laboratory) 2013 for our network simulations. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. MATLAB is a numerical computing environment and fourth-generation programming language. Developed by Math Works, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, and Fortran.

Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the Mu PAD symbolic engine, allowing access to symbolic computing capabilities. An additional package, Simulink, adds graphical multi-domain simulation and Model-Based Design for dynamic and embedded systems.

In 2004, MATLAB had around one million users across industry and academic. MATLAB users come from various backgrounds of engineering, science, and economics. MATLAB is widely used in academic and research institutions as well as industrial enterprises. We simulated our network on MATLAB and compared our results with existing models.

CHAPTER 2

THEORETICAL BACKGROUND

2.1 Object Tracking

Object tracking is the process of locating and following the moving object in sequence of video frames. Smart cameras are used as input sensors to record the video. The recorded video may have some noise due to bad weather (light, wind, etc. or due to problems in sensors). Few algorithms are tested to improve the image quality, to detect moving object, calculation of distance and velocity of the moving object.

2.2 Different aspects of Object Tracking

The Problem of Object Tracking can be divided into two parts:

- Detecting moving objects in each frame.
- Associating the detection corresponding to same object over time.

Moving object detection provides a classification of the pixels in the video sequence into foreground and background. A common approach used to achieve such classification is background removal also referred to as background subtraction where each video frame is compared against a background or reference or background model. Pixels that deviate from background considered to be moving objects.

2.2.1 Features of a Robust Background Subtraction Algorithm

- A good Background Subtraction algorithm should handle the relocation of background objects .it should be adaptable to non-stationary background objects. For eg:-Waving Trees, Camera movement due to wind load.
- It should be adaptable to gradual illumination changes for eg:-Day to Evening or Evening to Night.
- It should be accurate enough to deal with the problems like shadows and inter-reflections.

Even though there exist numerous of background removal algorithms in the literature, most of them follow a simple flow diagram, passing through four major steps, which are (1) pre-

processing (simple image processing tasks that change the raw input video into a format that can be processed by subsequent steps), (2) background modelling (also known as background maintenance), (3) foreground detection (also known as background subtraction) and (4) data validation (also referred to as post-processing, used to eliminate those pixels that do not correspond to actual moving objects).

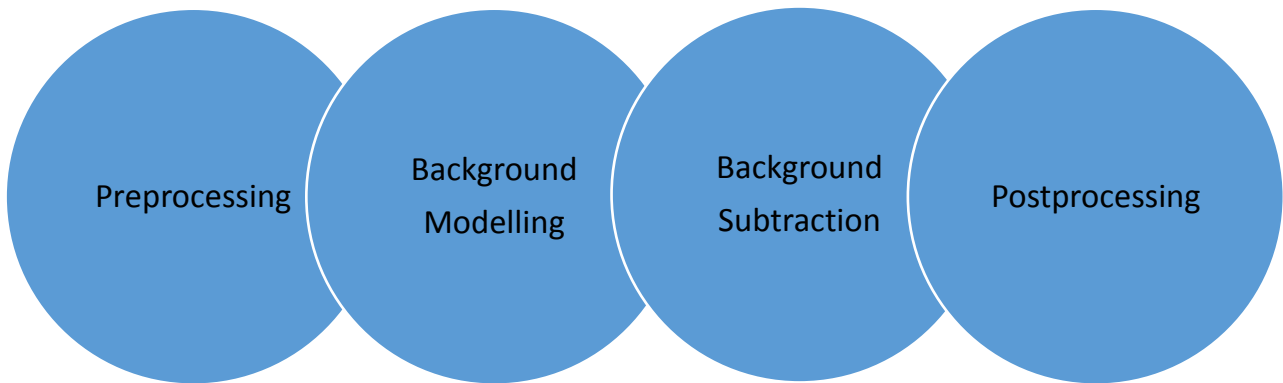


Figure 1: Background Subtraction Modelling

2.2.2 Pre-Processing

The goal of a moving object detection algorithm is to detect significant changes occurring throughout the video sequence while discarding trivial ones. The pre-processing steps described below are used to filter out common types of unwanted changes before making the object detection decision. These steps generally include geometric adjustments and intensity settings. Others involve using image derivatives or depth information as information source to the moving object detection algorithm. For real-time systems, frame-size and frame-rate reduction are commonly used to reduce the data processing rate. Simple temporal and/or spatial smoothing is often used in the early stage of pre-processing to reduce camera noise and to remove transient environmental noise such as rain and snow captured in outdoor applications.

2.2.2.1 Geometric adjustments

The intensity changes at a pixel resulting from camera movement alone are virtually never desired to be detected as real changes. A method of processing video images includes taking firstmost

photo with a camera having a first field of view. The capturing occurs at a first point in time. Commands are sent through MATLAB to the camera to make pan, tilt and zoom movements. A second image is captured with the camera at a second point in time. The second point in time is after the movements have commenced. A second field of view of the camera is calculated at the second point in time. The calculating is based upon the pan, tilt and zoom commands. The second image is processed based upon the first field of view and the calculated second field of view. The field of view of the camera may be calculated as a function of time. The calculated field of view may be output with a evaluation based upon a point in time associated with the calculated field of view. The processing may comprise determining a mask location or tracking an object of interest. Hence, frame registration is used to align several frames into the same coordinate frame. When the scenes of interest are mostly rigid in nature and the camera motion is small, registration can often be performed using low dimensional spatial transformations such as similarity, affine, or projective transformations. Excellent surveys and software implementations (e.g., the Insight toolkit) are available. switch automatically to higher-order transformations after being initialized with a low-order similarity transformation. In some scenarios a non-global transformation may need to be estimated to determine corresponding points between two frames.

2.2.2.2 Intensity adjustments

There are several techniques that attempt to make a solution for illumination variations between frames caused by changes in the strength or placement of light sources in the scene. Intensity normalization is used to deal with illumination changes, i.e. normalizing the pixel intensity values to have the same mean and variance as those in the estimated background. Alternatively, both the current frame and the background can be normalized to have zero mean and unit variance. This allows the use of decision thresholds that are independent of the original intensity values. Instead of using global statistics, the frames can be divided into corresponding disjoint blocks, and the normalization independently performed using the local statistics of each block. This can achieve better local performance at the cost of introducing blocking figures. However, algorithms which generally employ normalized colours typically work poorly in dark areas of the image. For scenes containing Lambertian surfaces, it is possible to extract the reflectance component by applying homomorphism filter to the input intensities. The reflectance component can be provided as input to the decision rule step of a moving object detection process (see, e.g. Modelling and compensating for local radiometric variation that deviates from the Lambertian assumption is necessary (e.g. underwater imagery. The principal component analysis (PCA) is also used to extract a set of basis images that represent the views of a scene under all possible lighting conditions. These

sophisticated models of illumination compensation are not commonly used in the context of foreground detection. Pre-processing may include feature extraction, i.e. transforming the input frame into the most appropriate feature space derived from the current frame only, i.e. it doesn't contain motion information. The feature space may represent the data format used by a particular background removal algorithm. Most of the algorithms handle luminance intensity, which is one scalar value per each pixel.

2.2.2.3 Depth Information

Using depth information presents fewer problems for segmentation, since depth information from stereo video is relatively unaffected by lighting conditions or extraneous motion. However, depth data does not produce valid results in scene locations with little visual texture, low contrast regions or that is not visible to all cameras, and tends to be noisy even where it is available. Hence, background removal methods based on depth alone produce unreliable answers in substantial portions of the scene, and often fail to find foreground objects in close proximity to the background, such as hands placed on walls or feet on a floor. A method using both depth and colour has been proposed, but it lacks time adaptivity and uses a relatively simple statistical model of the background. A significant advantage of the use of colour and depth space in the background estimation process is that, at pixels for which depth is usually valid, we can correctly estimate depth and colour of the background when the background is represented in only a minority of the frames. For pixels which have significant invalid range, we fall back to the same majority requirement as colour-only methods. In the general case where depth measurements at the pixel are largely valid, the background is simply represented by the mode which is farthest in depth and covers at least $T\%$ of the data temporally. It is better to work with disparities rather than depth because the error statistics are constant over the range of disparities.

2.2.3 Background Modelling

Background modelling, also referred to as background maintenance, is at the heart of any background removal algorithm. A set of principles has been proposed to which background modelling modules should adhere. The module performing background modelling should not attempt to extract the semantics of foreground objects on its own, since it is not an end by itself, larger systems use it as a component. One can evaluate background modelling by how closely it comes to finding all foreground pixels (as defined by the end task) when a foreground object *first* appears in the scene, while simultaneously ignoring all others. Backgrounds are not necessarily

defined by absence of motion, e.g. waving trees. No unimodal distribution of pixel values can adequately capture such a background, because these models implicitly assume that the background is static. An appropriate pixel-level stationary criterion should be defined. Pixels that satisfy this criterion are declared background and ignored. The background model must adapt to both sudden and gradual changes in the background. It is suggested that fast adaptation works quite well if the foreground consists of moving people. Some parts of a scene may remain in the foreground unnecessarily long if adaptation is slow, but other parts will disappear too rapidly into the background if adaptation is fast. Neither approach is inherently better than the other - a point that emphasizes the inadequacy of background modelling for all but the initialization of tracking. Most background modelling techniques operate at the pixel-level. Background models should take into account changes at differing spatial scales which are necessary to solve many of background modelling problems. Images are taken at the pixel, region, and frame levels, where the global illumination changes can be handled in the frame level. The issues characterizing a background modelling process are usually three; model representation, model initialization, and model adaptation. The first describes the kind of model used to represent the background; the second one regards the initialization of this model, and the third one relies to the mechanism used for adapting the model to the background changes (e.g. illumination changes).

2.2.3.1 Background Representation

The simplest background model is a known background. This occurs often in the entertainment or broadcast industry in which the environment can be engineered to simplify background removal algorithms. This includes the use of “blue screens”, backdrops with a constant colour which are designed to be easy to segment. The measurement used is the colour of a given pixel. If the camera is fixed and the background can be expected to stay relatively constant, we can model the background as a single static image that may be easily identified and ignored. The required measurement in this case is the intensity in case of gray level images and the colour components in case of colour images. If the background is not actually constant, then modelling both the mean intensity at a pixel and its variance gives an adaptive tolerance for some variation in the background. If a scene contains motion that should be considered part of the background, more tolerant models are required. One solution is to model measurements with a single multivariate Gaussian distribution. The parameters of this model are the mean and covariance matrix. When a single Gaussian is insufficient to model the distribution of pixel values, a finite mixture of Gaussians (MOG) may be used instead. Having a mixture model containing k Gaussians for some $k \in \mathbb{N}$. The parameters

of this model are then k mean values, k covariance matrices, and k scaling factors to weight the relevance importance of each Gaussian. The density function of pixel's distribution at any moment of time is estimated given only very recent history information hoping to obtain sensitive detection. The measurement of this model is a recent sample of intensity values for a pixel. Using this sample, the probability density function that this pixel will have a certain intensity value at time t can be non-parametrically estimated using the kernel estimator. A particular distribution of spatio-temporal image derivatives arises at points which always follow a constant optic flow. In this case, the image derivatives should fit the optic flow constraint equation: $I_x u + I_y v + I_t = 0$, for an optic flow vector (u, v) which remains constant through time for background pixels. Although motion-based approaches allows predicting the motion pattern of each pixel, this approach might fail when there is no obvious difference between the motion fields of the foreground and background. The fundamental background model is a one step Wiener filter which is linear predictor of the intensity at a pixel based upon the time history of intensity at that particular pixel. This can account for periodic variations of pixel intensity. The measurement includes two parts, the intensity at the current frame, and the recent time history of intensity values at a given pixel. Hidden Markov Models can also be used to represent the pixel process where its states can represent different states that might occur in the pixel process, such as background, foreground, shadows, day and night illumination. It can also be used to handle the sudden changes in illumination where the change from a status to another, such as the change from dark to light, day to night, indoor to outdoor, can be represented as the transition from state to state in the HMM. The pixel process with HMM, is using three-states HMM to represent background, shadows and foreground. Both background and shadows are modelled as single Gaussian distribution. The background can also be represented by a group of clusters which are ordered according to the likelihood that they model the background and are adapted to deal with background and lighting variations. Incoming pixels are matched against the corresponding cluster group and are classified according to whether the matching cluster is considered part of the background. Each pixel by a group of K clusters is modelled where each cluster consists of a weight w_k and an average pixel value or centroid. Quantize sample background values at each pixel into codebooks which represent a compressed form of background model for a long image sequence. A quantization/ clustering technique is generally adopted. Their method can handle scenes containing moving backgrounds or illumination variations, and it achieves robust detection for different types of videos. Mixed backgrounds can be modelled by multiple codewords. Unlike MOG, Kim do not assume that backgrounds are multimode Gaussians. Also, in contrast to Kernel, Kim does not store raw samples to maintain the background model.

2.2.3.2 Background Adaptation

The existing methods for background adaptation may be classified as either predictive or non-predictive. Predictive methods model the scene as a time series and develop a dynamical model to recover the current input based on past observations. The magnitude of the deviation between the predicted and actual observation can then be used as a measure of change, while non-predictive methods neglect the order of the input observations and build a probabilistic representation (pdf) of the observations at a particular pixel. Now the another way of classification is to divide background adaptation techniques into two broad categories - non-recursive and recursive. A non-recursive technique uses a sliding-window approach for background estimation. It stores a buffer of the previous L video frames, and estimates the background image based on the temporal variation of each pixel within the buffer. Non-recursive techniques are highly adaptive as they do not depend on the history beyond those frames stored in the buffer. On the other hand, the storage requirement can be significant if a large buffer is needed to cope with slow-moving objects. Given a fixed-size buffer, this problem can be partially alleviated by storing the video frames at a lower frame-rate. Recursive techniques do not maintain a buffer for background estimation. Instead, they recursively update either a single or multiple background model(s) based on each input frame. As a result, input frames from distant past could have an effect on the current background model. Compared with non-recursive techniques, recursive techniques require less storage, but any error in the background model can linger for a much longer period of time. Most schemes include exponential weighting to discount the past, and incorporate positive decision feedback to use only background pixels for updating.

2.2.3.2.1 Non Recursive Methods

Frame differencing, also known as temporal difference, uses the video frame at time $t-1$ as the background model for the frame at time t . This technique is sensitive to noise and variations in illumination, and does not consider local consistency properties of the change mask. This method also fails to segment the non-background objects if they stop moving. Since it uses only a single previous frame, frame differencing may not be able to identify the interior pixels of a large, uniformly-coloured moving object. This is commonly known as the aperture problem.

Average filter averages the images over time, creating a background approximation which is similar to the current static scene except where motion occurs. However, this is not robust to scenes with many moving objects particularly if they move slowly. It also cannot handle bimodal

backgrounds, recovers slowly when the background is uncovered, and has a single, predetermined threshold for the entire scene. The change of lighting condition can be handled using a moving-window average method, where exponential forgetting is used. An obvious problem with this technique is that all information coming from both background and foreground is used to update the background model. If some objects move slowly, these algorithms will fail. The solution to this problem is that only those pixels not identified as moving objects are used to update background model.

Median Filter

It defines the background to be the median at each pixel location of all the frames in the buffer. It assumes that the pixel stays in the background for more than half of the frames in the buffer. Median filtering has been extended to colour by replacing the median with the medoid. The complexity of computing the median is $O(L \log L)$ for each pixel.

Minimum-Maximum filter

Three values are estimated for each pixel using the training sequence without foreground objects: minimum intensity (Min), maximum intensity (Max), and the maximum intensity difference between consecutive frames. These values are estimated over several frames and are periodically updated for background regions. Two gray level background images B_1 , B_2 are used to cope with intensity variations due to noise or fluttering objects, moving in the scene.

Linear predictive filter

The current background is estimated by applying a linear predictive filter on the pixels in the buffer using a Wiener filter to predict a pixel's current value from a linear combination of its k previous values. Pixels whose prediction error is several times worse than the expected error are classified as foreground pixels. The filter coefficients are estimated at each frame time based on the sample covariance, making this technique difficult to apply in real-time. Linear prediction using the Kalman filter was also used before.

An autoregressive form is presented to predict the frame to be observed. Two different techniques are studied to maintain the model, one that updates the states in an incremental manner and one that replaces the modes of variation using the latest observation map. Other techniques can be considered to determine such prediction model. Principal component analysis refers to a linear transformation of variables that retains - for a given number n of operators - the largest amount of variation within the training data. Estimation of the basis vectors from the observed data set can be performed through singular value decomposition. Computing the basis components for large vectors

is a time consuming operation. Optimal algorithms for singular value decomposition of an $m \times n$ matrix take $O(m^2n + n^3)$ time. A simple way to deal with such complexity is by considering the process at a block level. To this end, the image is divided into blocks and run the algorithm independently on each block.

Non-parametric Modelling

It models the pixel as a random variable in a feature space with an associated probability density function (PDF). Nonparametric approaches estimate the density function directly from the data without any assumptions about the underlying distribution, avoiding having to choose a model and estimating its distribution parameters. Kernel density estimators asymptotically converge to any density function. In fact, all other nonparametric density estimation methods, e.g., histograms, can be shown to be asymptotically kernel methods. However, the major drawback with colour histograms is the lack of convergence to the right density function if the data set is small; also they are not suitable for higher dimensional features. Unlike histograms, even with a small number of samples, kernel density estimation leads to a smooth, continuous and differentiable density estimate. Kernel density estimation does not assume any specific underlying distribution and, theoretically, the estimate can converge to any density shape with enough samples. Therefore, this approach is suitable to model the colour distribution of regions with patterns and mixture of colours. Unlike parametric fitting of a mixture of Gaussians, kernel density estimation is a more general approach that does not require the selection of the number of Gaussians to be fitted; also the adaptation of the model is trivial and can be achieved by adding new samples. The advantage of using the full density function over a single estimate is the ability to handle multi-modal background distribution, i.e. pixels from a swinging tree or near high contrast edges where they flicker under small camera movement. One major issue that needs to be addressed when using kernel density estimation technique is the choice of suitable kernel bandwidth (scale). Theoretically, as the number of samples reaches infinity, the choice of the bandwidth is insignificant and the estimate will approach the actual density. Practically, since only a finite number of samples are used and the computation must be performed in real time, the choice of suitable bandwidth is essential. Too small a bandwidth will lead to a ragged density estimate, while too wide a bandwidth will lead to an over-smoothed density estimate. Since the expected variations in pixel intensity over time are different from one location to another in the image, a different kernel bandwidth is used for each pixel. Also, a different kernel bandwidth is used for each colour channel.

The median of the absolute differences is used between successive frames as the width of the kernel. Thus, the complexity of building the model is the same as median filtering. A variety of kernel functions with different properties have been used in the literature. Typically the Gaussian

kernel is used for its continuity, differentiability, and locality properties. Note that choosing the Gaussian as a kernel function is different from fitting the distribution to a Gaussian model. Here, the Gaussian is only used as a function to weight the data points. A good discussion of kernel estimation techniques can be found in [1]. The major drawback of using the nonparametric kernel density estimator is its computational cost. This becomes less of a problem as the available computational power increases and as efficient computational methods have become available recently. However, several pre-calculated lookup tables for the kernel function values can be used to reduce the burden of computation of this approach. Also, this method cannot resist the influence of foreground objects in the training stage (background initialization). In general, given N original data samples and M target points at which the density needs to be evaluated, the complexity is $O(NM)$ evaluations of the kernel function, multiplications and additions. A computational framework is presented for efficient density estimation, introducing the use of Fast Gaussian Transform (FGT) for efficient computation of colour densities, allowing the summation of a mixture of M Gaussians at N evaluation points in $O(M+N)$ time as opposed to $O(MN)$ time for a naive evaluation, and can be used to considerably speed up kernel density estimation. Given a new pixel sample, there are two alternative mechanisms to update the background; *selective update*: add the new sample to the model only if it is classified as a background sample and *blind update*: just add the new sample to the model. There are tradeoffs to these two approaches. The first enhances detection of the targets, since target pixels are not added to the model, however, any incorrect detection decision will result in persistent incorrect detection later, which is a deadlock situation. The second approach does not suffer from this deadlock situation since it does not involve any update decisions allowing intensity values that do not belong to the background to be added to the model. This leads to bad detection of the targets (more false negatives) as they erroneously become part of the model. This effect is reduced as we increase the time window over which the samples are taken as a smaller proportion of target pixels will be included in the sample. But as we increase the time window more false positives will occur because the adaptation to changes is slower and rare events are not as well represented in the sample. A way to combine the results is presented of two background models (a long term and a short term) in such a way to achieve better update decisions and avoid the tradeoffs discussed above. Short-term model is a very recent model of the scene. It adapts to changes quickly to allow very sensitive detection. The sample is updated using a selective-update mechanism, where the update decision is based on the final result of combining the two models. Long-term model captures a more stable representation of the scene background and adapts to changes slowly. The sample is updated using a blind-update mechanism. Computing the intersection of the two detection results will eliminate the persistent false positives from the short term model and will eliminate as well extra false positives that occur in the long term model results.

The only false positives that will remain will be rare events not represented in either model. If this rare event persists over time in the scene then the long term model will adapt to it, and it will be suppressed from the result later. Taking the intersection will, unfortunately, suppress true positives in the first model result that are false negatives in the second, because the long term model adapts to targets as well if they are stationary or moving slowly. To address this problem, all pixels detected by the short term model that are adjacent to pixels detected by the combination are included in the final result.

2.2.3.2.2 Recursive Methods

Approximated Median Filter

Due to the success of non-recursive median filtering, McFarlane and Schofield propose a simple recursive filter to estimate the median. This technique has also been used in background modelling for urban traffic monitoring where the running estimate of the median is incremented by one if the input pixel is larger than the estimate, and decreased by one if smaller. This estimate eventually converges to a value for which half of the input pixels are larger than and half are smaller than this value, that is, the median. The only drawback of the approximated median filter is that it adapts slowly toward a large change in background. It needs many frames to learn the new background region revealed by an object that moves away after being stationary for a long time.

Single Gaussian

One of the simplest background removal techniques is to calculate an average image of the scene, subtract each new frame from this image, and threshold the result. This basic Gaussian model can adapt to slow changes in the scene (for example, gradual illumination changes) by recursively updating the model using a simple adaptive filter. The main feature of modelling the probability distribution of the pixel intensity that differentiates it from other ways such as predictive filters is that it ignores the order in which observations are made and focuses on the distribution of the pixel intensities. Gordon models each pixel as an independent statistical process, recording the (R, G, B, Z) observations at each pixel over a sequence of frames in a multidimensional histogram (depth and colour information). Then they use a clustering method to fit the data with an approximation of a mixture of Gaussians. At each pixel, one of the clusters (Gaussians) is selected as the background process, the others are considered to be caused by foreground processes. They are working on extensions which will allow dynamic background estimation based on the previous N frames (allowing modelling slow changes in the background), they are also working on the estimation of multiple background processes at each pixel, similar to but using higher dimensional Gaussians.

Then the background is modelled in two distinct parts, the colour model and the edge model. For each colour channel, each pixel is represented by its mean and standard deviation throughout time. The edge model is built by applying the Sobel edge operator to each colour channel yielding a horizontal difference image and a vertical difference image. Weighted means and standard deviations are computed as in the colour model. This model is used to locate changes in the structure of the scene as edges appear, disappear, or change direction. However, their method cannot deal with sudden changes in illumination. Moreover, this algorithm doesn't present a solution to the relocation of background object problem.

Kalman filter

It is a widely-used recursive technique for tracking linear dynamical systems under Gaussian noise. It can be viewed as the simplest background model assuming that the intensity values of a pixel can be modelled by a Gaussian distribution $N(\mu, \sigma^2)$ where the mean and variance of the background are updated using simple adaptive filters to accommodate changes in lighting or objects that become part of the background. While this method does have a pixel-wise automatic threshold, it still recovers slowly and does not handle bimodal backgrounds well. This method has been successfully integrated into this method in an automatic traffic monitoring application. Many different versions of Kalman filter have been proposed for background modelling, differing mainly in the state spaces used for tracking. The simplest version uses only the luminance intensity. An algorithm has been proposed that explicitly models the dynamic, textured background via an Autoregressive Moving Average (ARMA) model. Although ARMA is a first-order linear model many dynamic textures can be well captured by it. A robust Kalman filter algorithm is used in estimating the intrinsic appearance of the dynamic texture. The foreground object regions are then obtained by thresholding the weighting function used in the robust Kalman filter. In their current implementation, the Kalman filter model is only trained using the empty scenes; however, they could use the Robust PCA method in to train the ARMA model on non-empty scenes in the future. After visual analysis of Kalman filter results, it is concluded that Kalman filter produces the worst foreground masks when compared with other schemes. Even with a large foreground threshold and slow adapting rates, the background model in Kalman filter is easily affected by the foreground pixels. As a result, it typically leaves a long trail after a moving object.

Mixture of Gaussians (MoG)

The background of the scene contains many non-static objects such as tree branches and bushes whose movement depends on the wind in the scene. This kind of background motion causes the pixel intensity values to vary significantly with time. So a single Gaussian assumption for the pdf of

the pixel intensity will not hold. Instead, a generalization based on a mixture of Gaussians has been used to model such variations. The pixel intensity was modelled by a mixture of K Gaussian distributions. A mixture of three Gaussian distributions was used to model the pixel value for traffic surveillance applications, corresponding to road, shadow, and vehicle distribution. Adaptation of the Gaussian mixture models can be achieved using an incremental version of the EM algorithm. Although, in this case, the pixel intensity is modelled with three distributions, still uni-modal distribution assumption is used for the scene background, i.e. the road distribution. Unlike Kalman filter which tracks the evolution of a single Gaussian, the MoG method tracks multiple Gaussian distributions simultaneously. MoG has enjoyed tremendous popularity since it was first proposed for background modelling in. The generalized mixture of Gaussians (MoG) has been used to model complex, non-static backgrounds. The background model is allowed to be a mixture of several Gaussians. Every pixel value is compared against the existing set of models at that location to find a match. The parameters for the matched model are updated based on a learning factor. If there is no match, the least-likely model is discarded and replaced by a new Gaussian with statistics initialized by the current pixel value. The models that account for some predefined fraction of the recent data are deemed “background” and the rest “foreground”. There are additional steps to cluster and classify foreground pixels into semantic objects and track the objects over time. A mixture of Gaussians method, slightly modified from the version presented by Stauffer and Grimson to perform background subtraction in the colour domain, where the covariance matrix is assumed to be diagonal to reduce the computational cost. A K-means approximation of the EM algorithm is used to update the mixture. A method for modelling has been proposed that uses per-pixel, time-adaptive, Gaussian mixtures in the combined input space of depth and luminance-invariant colour. They improve such combination by introducing the ideas of 1) modulating the background model learning rate based on scene activity, and 2) making colour-based segmentation criteria dependent on depth observations. The input to the algorithm is a time series of spatially registered, time-synchronized pairs of colour (YUV space) and depth images obtained by static cameras. “Background subtraction” is an old technique for finding moving objects in a video sequence---for example, cars driving on a freeway. The idea is that subtracting the current image from a time-averaged background image will leave only non-stationary objects. It is, however, a crude approximation to the task of classifying each pixel of the current image; it fails with slow-moving objects and does not distinguish shadows from moving objects. The basic idea of is that they can classify each pixel using a model of how that pixel looks when it is part of different classes. They learn a mixture-of-Gaussians classification model for each pixel using an unsupervised technique---an efficient, incremental version of EM. Unlike the standard image-averaging approach, this automatically updates the mixture component for each class according to likelihood of membership; hence slow-moving objects are handled perfectly.

Their approach also identifies and eliminates shadows much more effectively than other techniques such as thresholding. Application of this method as part of the Roadwatch traffic surveillance project is expected to result in significant improvements in vehicle identification and tracking. Earlier background statistics have been used to model disparity images derived from stereo video using a bimodal distribution, with a Gaussian representing good correlations between left and right images. Then an algorithm is proposed for updating background statistics on-the-fly called gated background adaptation, starting by assuming that the background will appear at some point during the video sequence.

Eveland's idea of the gate is to filter acceptable values for the update equations, where any value larger than $G = \mu_t + 3 \times 0.9$ above the current mean is rejected in subsequent applications of the update. The gate has the effect of excluding the outlying foreground readings, gradually reducing the estimated background statistics to their true value. To deal with changing background over time, i.e. background objects start to move, or some objects come and remain, the gate is relaxed to consider foreground objects to be background if they stay for a certain length of time. The background pixels are modelled as values as multi-dimensional Gaussian distributions in HSV colour space. The distribution for each background pixel is updated using the latest observation, in order to take into account changes in the scene background. They initialize the means using the values of the first frame, and set the standard deviations to zero. The actual distributions are learned as the incoming frames are processed. A Bayesian formulation of the background segmentation problem is presented at the pixel level based on Gaussian mixture modelling since its analytical form fits well into a statistical framework. However, the MoG has its own drawbacks. First, it is computationally intensive and its parameters require careful tuning. Second, it is very sensitive to sudden changes in global illumination. If a scene remains stationary for a long period of time, the variances of the background components may become very small. A sudden change in global illumination can then turn the entire frame into foreground. Backgrounds having fast variations cannot be modelled with just a few Gaussians accurately, so it fails to provide sensitive detection. However, when foreground objects are included in the training frames, MoG will misclassify. In addition, depending on the learning rate to adapt to background changes, MoG faces trade-off problems. For a low learning rate, it produces then a very wide and inaccurate model that will have low detection sensitivity, not able to detect a sudden change to the background. On the other hand, if the model adapts too quickly, slowly moving foregrounds will be absorbed into the background model, resulting in a high false negative rate. This is the foreground aperture problem. However, MoG maintains a density function for each pixel. Thus, it is capable of handling multimodal background distributions. On the other hand, since MoG is parametric, the model parameters can be easily updated without keeping a large buffer of video frames.

Clustering-Based

A moving object segmentation algorithm is proposed with a having the capability of processing 320 x 240 video in real-time on modest hardware. The premise of their algorithm is the more often a pixel takes a particular colour, the more likely that it belongs to the background. Therefore, this requires a technique for maintaining information regarding the history of pixel values. They model each pixel by a group of K clusters where each cluster consists of a weight and an average pixel value or centroid c_k . additionally; the algorithm assumes that the background region is stationary. Incoming pixels are compared against the corresponding cluster group. The matching cluster with the highest weight is sought and so the clusters are compared in order of decreasing weight. A matching cluster is defined to have a Manhattan distance (i.e. sum of absolute differences) between its centroid and the incoming pixel below a user prescribed threshold T . If no matching cluster is found, the cluster with the minimum weight is replaced by a new cluster having the incoming pixel as its centroid and a low initial weight. If a matching cluster is found, then the weights of all clusters in the group are adjusted. The centroid of the matching cluster must also be adjusted according to the incoming pixel. Previous approaches adjust the centroid based on a fraction of the difference between the centroid and the incoming pixel. However, doing so, results in fractional centroids and inefficient implementations. Butler chooses instead to accumulate the error between the incoming pixel and the centroid. After adaptation, the weights of all clusters in the group are normalised so that they sum up to one. Then normalised clusters are next sorted in order of decreasing weight to aid both the initial cluster comparisons and the final classification step. Quantize the background values of each pixel into group of code words constituting a codebook for each pixel, where each pixel might have a different codebook size than the other, according to the pixel variation throughout the time. In order to make their technique more practically useful in a visual surveillance system, they improve their basic algorithm by layered modelling/detection multiple background layers and adaptive codebook updating, to handle changing backgrounds.

Hidden Markov Models (HMM)

All of the previously mentioned models can adapt to gradual changes in illumination. On the other hand, a sudden change in illumination presents a challenge to such models. Another approach to model a wide range of variations in the pixel intensity is to represent these variations as discrete states corresponding to modes of the environment, e.g., lights on/off or cloudy/sunny skies. Hidden Markov models (HMMs) have been used for this purpose. A three-state HMM has been used to model the intensity of a pixel for a traffic-monitoring application where the three states correspond to the background, shadow, and foreground. The topology of the HMM representing global image

intensity is learned while learning the background. At each global intensity state, the pixel intensity is modelled using a single Gaussian. It was shown that the model is able to learn simple scenarios like switching the lights on and off.

Major issues in the use of HMMs in real world applications involve two points: real-time computation, and topology modification to address non-stationarity due to dynamically varying conditions. Automatic selection of HMM topology during batch-learning phase has been addressed in the literature. These methods use state-splitting or merging, criteria to iteratively estimate the optimal number of states. More recently, a maximum a posterior estimation scheme utilizing an entropic prior is presented wherein a redundant number of states is initially assumed and weak states satisfying a given probability threshold are eliminated iteratively. However, dynamic update of the model topology is not addressed in the work. Besides, state merging schemes are computationally intensive and online updates are not practical. Another interesting piece of work for real time background modelling based on Hidden Markov Models with fixed topology is recently introduced. Unlike the other methods where the model is fixed and its parameters are updated, a dynamic framework is presented (e.g. it can change naturally the topology over time) and can deal with sudden as well as gradual illumination changes. They describe a solution to the batch and online estimation of HMM topology. For computer vision problems where the online estimation is critical, they believe that state-splitting is the most reasonable approach since it is computationally efficient. Stenger compares several state-splitting criteria such as the chi-squared goodness-of-fit test, the cross-validation criterion and the MDL and AIC criteria. It is seen that the MDL criterion performed the best in terms of minimal variance on the number of states estimated. The topologies were compact and provided the right trade-off between approximation error and model simplicity. An online version of the HMM topology estimation algorithm is also presented. Both the online versions and offline versions are tested on real data. A system and method for coding text data wherein a first group of text data is coded using a Viterbi algorithm using a Hidden Markov model. The Hidden Markov Model computes a probable coding responsive to the first group of text data. A second group of text data is coded using the Viterbi algorithm using a corrected Hidden Markov Model. The Hidden Markov Model is based upon the coding of the first group of text data. Coding the first group of text data includes assigning word concepts to groups of at least one word in the first group of text data and assigning propositions to groups of the assigned word concepts. An offline Baum-Welch algorithm is used to learn the parameters of HMM, and use an online algorithm to adapt the parameters, assuming that no obvious sudden change of illumination takes place, the background is assumed to be approximately stationary (to avoid adding a module of recognition of background motion), and the speed of the moving objects doesn't vary greatly. These assumptions are the nature property of video sequence on freeway obtaining such video sequence is

very easy. Theoretical detail of online learning algorithm can be found as incremental version of the EM algorithm. A technique called exponential forgetting is also used where each pixel value's contribution is weighted so as to decrease exponentially as it recedes into the past.

2.3 Background Initialization

Recently, there has been a large amount of work addressing the issues of background model representation and adaptation (maintenance). However, a third problem which has received little attention is model initialization (also called bootstrapping). Actually, most of the background models are built on a set of initial parameters that comes out from a short sequence, in which no foreground objects are present. This is a too strong assumption, because in some situations it is difficult or impossible to control the area being monitored (e.g., public zones), which are characterized by a continuous presence of moving objects, or other disturbing effects. In such cases it may be necessary to train the model using a sequence which contains foreground objects. Several assumptions are necessary to make the task feasible. Each pixel in the image will reveal the background for at least a short interval of the sequence during the training phase to avoid randomly choosing background appearance. The background is approximately stationary; only small background motion may occur. A short processing delay is allowed subsequent to acquiring the training sequence. Other assumptions to the above mentioned ones; a foreground object can remain stationary for a short interval in the training sequence. However, the interval should be no longer than the interval from the revealed static background. The background scene remains relatively stable.

Median Filtration

Background initialization using the median intensity value for each pixel is used in a traffic monitoring system, relying on the assumption that the background at every pixel will be visible more than fifty percent of the time during the training sequence. However, this may not be always satisfied. A method of creating an image difference overlay comprises identifying a loop of reference images of a subject and identifying a loop of data images of the subject.

The loop of image data can be identified after an event, such as the administration of contrast agent to the subject. A reference loop image frame is compared to one or more data loop image frames and the reference loop frame is associated with a data loop image frame which closely resembles the data loop image frame. Each of the associated frames can then be processed and used to create an image difference overlay frame. The advantage of using the median rather than the mean is that it avoids blending pixel values. The mean of a pixel's intensity over time may not correspond to any of the

pixel's actual values during that time, in which case it is likely to be in error. Dawson-Howe proposed a similar technique called dynamic background subtraction (DBS) using a fixed window of three frames to recursively update the background, avoiding the rather expensive cost of computing the median.

Stable Intensity Extraction

An algorithm is proposed, called the adaptive smoothness method which also avoids the problem of blending, it finds intervals of stable intensity, and uses a heuristic which chooses the longest, most stable interval as the one most likely to represent the background. Their method performs well when all foreground objects are in motion throughout the sequence. However, for sequences in which foreground objects were stationary for a long period of time (sleeping person), many pixels are incorrectly classified. Like most other algorithms, adaptive smoothness makes the decision for each pixel independently, and does not utilize other information from the sequence.

One problem of this method is that when the data include multi-modal distributions (i.e., some modes from foreground objects and some modes from background), and when the modes from foreground objects tend to be relatively stable, this method cannot differentiate these modes from those from the background. The value which stays unchanged for a long period of time are found from the mixture of the stationary signal (the background) and the non-stationary signal caused by the moving objects such as cars. A technique from robust statistics is used by considering the non-stationary signal as outliers. The method works well if the background appears in the sequence more than 50% of the training period.

Relative Constant Intensity Extraction

Local ImageFlow algorithm is similar to adaptive smoothness in that they generate hypotheses by locating intervals of relatively constant intensity. To overcome the sleeping person problem, he considers the optical flow in the neighbourhood surrounding each pixel. If the direction of optical flow in the neighbourhood is toward the pixel, then there is likely to be a moving object approaching the pixel. However, if the majority of optical flow is directed away from the pixel, it is likely that the moving object is leaving the area. Only low-level motion information has been used to construct the background. Like the median filter and adaptive smoothness methods, it avoids the problem of blending pixel values present in many current methods. However, the heuristics used in his technique, based on local image flow, are stronger than those used by the median filter and adaptive smoothness. The main strength of the algorithm is that while the decision at each pixel is independent of its neighbours, it is not only based on past values observed at that pixel, but also local motion information. While using optical flow information potentially adds valuable

information, most optical flow computation methods themselves are computationally complex and very sensitive to noise.

Background with A Mixture of Gaussian Distribution

There are several methods available for building such a mixture model. A widely used algorithm is expectation maximization, which uses an iterative process to find the best-fitting mixture of Gaussians for a particular dataset. However, the parameters must be updated and calculated offline; therefore, expectation maximization cannot be used when online methods are required. One possible solution is to use the adaptive mixture method, which uses a data-driven approach to estimate the parameters of an underlying mixture model. .

Background Model Based on Kernel Density Estimation

The density function of this distribution is estimated at any moment of time given only very recent history information. Hoping to obtain sensitive detection, the probability density function that a pixel will have intensity value x_t at time t can be non-parametrically estimated using a kernel estimator. At the first glance, the kernel estimator function, K , to be a Normal function $N(0, \sigma^2)$, where σ^2 represents the kernel function bandwidth. Normal kernel function is a generalization of the Gaussian mixture model, where each single sample of the N samples is considered to be a Gaussian distribution $N(0, \sigma^2)$ by itself, enabling the model to quickly “forget” about the past and concentrate more on recent observation.

Hidden Markov Models (HMMs)

Numerous methods for estimation of the HMM model parameters exist in the literature. These methods can be classified into two major categories: batch and incremental. Some examples of the batch methods include: the EM algorithm (i.e. Baum-Welch) and segmental K-means algorithm. Offline methods guarantee that the parameter estimates correspond to local maxima, but are computationally expensive. The online methods cited attempt to incrementally update HMM parameters (without topology modifications) with initial starting points determined by batch estimation. These methods can deal with slowly varying drifts in model parameters with the risk of not being able to track the real parameters under sudden changes. An initialization algorithm is proposed which is able to bootstrap an integrated pixel- and region-based background modelling algorithm, where moving objects are present; the output is a pixel- and region-level statistical background model describing the static information of a scene. At the pixel level, multiple hypotheses of the background values are generated by modelling the intensity of each pixel with a Hidden Markov Model (HMM), also capturing the sequentiality of the different colour (or

gray-level) intensities. At the region level, the resulting HMMs are clustered with a novel similarity measure, able to remove moving objects from a sequence, and obtaining a segmented image of the observed scene, in which each region is characterized by a similar spatio-temporal evolution. The main drawback of this method is the strong computational effort, but an on-line computation for an initialization algorithm is indeed allowed. Nevertheless, a parallel computational architecture may solve this problem, permitting a very quickly and useful batch mode scheme. The algorithm is based on a heuristic colour model, which separates the brightness from the chromaticity component. The pixel is labelled as background, shadow or foregrounds by a decision procedure whose threshold values are used to determine the similarities of chromaticity and brightness between background image and the current observed image. If chromaticity difference is large, pixel is labelled as foreground, and then if brightness difference is large, pixel is labelled as shadow. The result is then used to initialize their HMM model. An image processing system useful for facial recognition and security identification obtains an array of observation vectors from a facial image to be identified. A Viterbi algorithm is applied to the observation vectors given the parameters of a hierarchical statistical model for each object, and a face is identified by finding a highest matching score between an observation sequence and the hierarchical statistical model.

Codebook-Based

The background values of each pixel are integrated into group of codewords constituting a codebook for each pixel. At the very beginning the codebook of a pixel is empty with no codewords, when a new sample for the pixel encountered (in the training period), if no codeword exists in the codebook, it is assumed to be a codeword, and its brightness value is used to estimate the measures used to represent the codeword, if there are codewords in the codebook, this new pixel sample is compared with each codeword in the codebook using colour distortion measure and brightness bound, if it is matched with a codeword, its brightness value is used to update the measures of this codeword, if there is no match, it is assumed to be a new codeword, and so on till the end of the training period. One way to improve the speed of the algorithm is to relocate the most recently updated codewords to the front of the codebook list. Since most of the time, the matched codeword was the first codeword thus relocated, making the matching step efficient. In the temporal filtering step, he refines the fat codebook by separating the codewords that might contain moving foreground objects from the true background code words, thus allowing moving foreground objects during the initial training period.

Statistical Approach

Initializing a background model might be approached statistically as the task should be robust against random occurrences of foreground objects, as well as against general image noise. The major advantage of this approach is that it can tolerate over 50% of noise in the data (including foreground pixels), in contrast with methods using the Median statistic which will break down totally when background constitutes less than 50% of the training data. A consensus-based robust method is used for background initialization to overcome the problems inherent in methods based on the median filtration, by employing a two-step framework; first all non-overlapping stable subsequences of pixel values are located; using a sliding window with a minimum length, if candidate subsequence with the predefined minimum length cannot be found, another minimum length is used, noting that even after this step, the chosen subsequence can contain pixels from foreground, background, shadows, highlights, etc. The second step is considered a crucial step where the most reliable subsequence is chosen, where the reliability definition is motivated, and use the mean value of either the grey-level intensities or the colour intensities over that subsequence as the model background value.

2.4 Foreground Detection

Foreground detection compares the input video frame with the background model, and identifies candidate foreground pixels from the input frame. To obtain this classification, the difference map is usually binarized by thresholding. The correct value of the threshold depends on the scene, on the camera noise, and on the illumination conditions. In the following subsections we will discuss first how to generate the difference map given the background model and the current frame, and then we will discuss the thresholding techniques to obtain foreground-background classification.

2.4.1 Foreground Detection Techniques

2.4.1.1 Difference-Based

The most trivial method to perform foreground detection is by taking the difference between two images. Locations of changes correspond to large values in the difference map which can be computed as the absolute values of the difference between corresponding pixels in the two images. Relative or normalized difference also can be used. Another approach to introduce spatial variability is to use two thresholds with hysteresis. The basic idea is to first identify "strong" foreground pixels whose absolute differences with the background estimates exceeded a large threshold. Then,

foreground regions are grown from strong foreground pixels by including neighbouring pixels with absolute differences larger than a smaller threshold. The region growing can be performed by using a two-pass, connected-component grouping algorithm. Keep track with the minimum and maximum values for each pixel throughout the past N frames, such algorithm uses two thresholds, T_L and T_H . The difference between each pixel and the closest background image is computed. If the difference exceeds a low threshold T_L , the pixel is considered as foreground. A target is a set of connected foreground pixels such that a subset of them exceeds the high threshold. The low and high thresholds as well as the background images are recursively updated in a fully automatic way for performing foreground detection by subtracting the colour channels and the edge channels separately from their corresponding model (mean and standard deviation images) and then combining their results.

Absolute Difference Edge-Based

Getting away from thresholds dilemma, Sobel edge detector is applied over the absolute difference image between the current frame and the reference frame, a Sobel edge extractor provides thick edges which are useful for application in foreground detection by allowing lightening the post-processing for filling the contours. In a monochrome image, an edge is defined as an intensity discontinuity. In case of colour images, the additional variation in colour may be considered in order to obtain more complete edge information. Monochrome edge detection, in fact, may not be sufficient for certain scenes. For instance, an object with different hue value from the background but equal intensity can be detected only by taking into account colour information. There are different possibilities to use colour for edge detection purposes. The most straightforward approaches to colour edge detection represent extensions from monochrome edge detection. These techniques are applied to three colour channels independently and then the results (an edge map for each colour channel) are combined by using a certain logical operator. Cavallaro adopts this approach which has the advantage of speeding up the computations if the different channels are processed in parallel. The edge information of the three channels is then fused by means of and or logical operator.

Relative Difference

Ideally, the threshold should be a function of the spatial location (x, y) . For example, the threshold should be smaller for regions with low contrast. They use the relative difference rather than absolute difference to emphasize the contrast in dark areas such as shadow. Nevertheless, this technique cannot be used to enhance contrast in bright images such as an outdoor scene under heavy fog.

Normalized Difference

Another popular foreground detection scheme is threshold based on the normalized statistics.

Predictive-Based

In case of predictive-based background modelling, differences in the state space between the prediction and the observation quantify the amount of change and are considered to perform detection. So a simple mechanism to perform detection is by comparing the prediction with the actual observation. Under the assumption that the autoregressive model is built using background samples, such technique will provide poor prediction for objects while being able to capture the background. Two types of changes in the signal may be considered for detection: (1) “structural” change in the appearance of pixel intensities in a given region, and (2) change in the motion characteristics of the signal. Measures are developed in order to detect each of these types of changes.

2.4.1.2 Statistical-Based

To calculate an adaptive and local threshold, a region-based statistical analysis can be used if the probability density function of the camera noise is known. The statistical analysis is based on modelling the intensity distribution of noise. Instead of thresholding the difference image, this approach compares the statistical behaviour of a small neighbourhood at each pixel position in the difference image to a model of the noise that could affect the difference image. The comparison is based on a significant test.

Single Gaussian-Based

Pixels in the current frame are compared with the background by measuring the log likelihood in colour space. If a small likelihood is computed, the pixel is classified as foreground. Otherwise, it is classified as background.

MOG-Based

The background pixel values are modelled as multi-dimensional Gaussian distributions in HSV colour space. When a new frame is processed, the value observed for each pixel is compared to the current corresponding distribution in order to decide whether the value is a measurement of the background or of an occluding element. A background segmentation problem at the pixel level based on Gaussian mixture modelling. From a Bayesian perspective, the foreground decision should be based on the posterior probability of the pixel being background $P(B|x)$ where x denotes the pixel observed in the frame at time t and B denotes the background class. Without giving a precise definition of foreground and background, which is most likely application dependent and requires

higher level semantics, Lee proceeds by considering them as two mutually exclusive classes as defined by some oracle. Considering the value observed at a pixel over time is usually resulted from different real world processes, a Gaussian mixture is appropriate to model the distribution, with each Gaussian representing an underlying process.

Kernel Density Estimation-Based

Using the probability estimate of the pixel, the pixel is considered a foreground pixel if $\Pr(x_t) < \theta$ where the threshold θ is a global threshold over all the image that can be adjusted to achieve a desired percentage of false positives.

MRF-Based

An approach has been developed to the process of foreground detection that exploits the spatial and temporal dependencies objects in motion impose on their images. This is achieved through the development and use of MRFs during the subtraction process. In total, three MRFs (M1, M2 and M3) are used, each with different properties. M1, being the simplest of the three, only encodes spatial relationships. M2, an extension of M1, takes advantage of temporal constraints by looking at past segmentations. M3, the only field to use a batch computation model, fully exploits the temporal constraint. Segmentations at each time t , S_t , are obtained by estimating the maximum a posteriori (MAP) configuration of the MRF. The MAP estimate is computed using the Gibbs sampler algorithm with a modified (linear) annealing schedule and vastly reduced numbers of iterations. This approach is not tied to any particular background model. In fact, this approach will work with any background model in which a cost function (d_s) can be defined for every pixel d_s . This implies (but does not necessitate) that the background models must be per-pixel based. This is not such a heavy constraint; however it is concluded that per-pixel models are sufficient to handle the complex motions present in real world environments.

2.4.1.3 Clustering-Based

The background values are quantized at each pixel into codebooks; they test the difference of the current image from the background model with respect to colour and brightness differences. If an incoming pixel meets two conditions, it is classified as background (1) the colour distortion to some codeword is less than the detection threshold, and (2) its brightness lies within the brightness range of that codeword. Otherwise, it is classified as foreground. The codebook method does not evaluate probabilities, which is very computationally expensive. Kim just calculates the distance from the cluster means. That makes the operations fast, pixels are classified by summing the weights of all

clusters used to model each pixel that are weighted higher than the matched cluster. The result, P is the total proportion of the background accounted for by the higher weighted clusters and is an estimate of the probability of the incoming pixel belonging to the foreground. Larger values of P are evidence the pixel belongs to the foreground and smaller values are evidence that it belongs to the background. This value can be thresholded to obtain a binary decision or can be scaled to produce a gray scale alpha map.

2.4.2 Thresholding Algorithms

Thresholding is a fundamental method to convert a grayscale image into a binary mask, so that the objects of interest are separated from the background. In the difference image, the gray levels of pixels belonging to the foreground object should be different from the pixels belonging to the background. Thus, finding an appropriate threshold will solve the localization of the moving object problem. The output of the thresholding operation will be a binary image whose gray level of 0 (black) will indicate a pixel belonging to the background and a gray level of 1 (white) will indicate the object. The efficiency of the foreground detection partially depends on the threshold selection, as clearly observed in the previous schemes. The threshold can be set empirically or computed adaptively. In the former case, the threshold is fixed for all pixels in the frame and all the frames in the sequence. The value is usually determined experimentally based on a large database. In the latter case, the threshold is adapted according to some rules.

Several experiments are conducted on many different criteria for choosing the threshold. Different methods have been discussed how the threshold can be chosen to achieve application-specific requirements for false alarms and misses (i.e. the choice of point on a receiver-operating characteristics curve). Different thresholding algorithms can be divided into 6 major groups. These algorithms can be distinguished based on the exploitation of (1) histogram entropy information, (2) histogram shape information, (3) image attribution information, (4) clustering gray level information, (5) local characteristics and (6) spatial information.

The entropy based methods result in different algorithms which use the entropy of the foreground-background regions or the cross-entropy between the original and binarized image, etc. Assuming that the histogram of an image gives some indication about this probabilistic behaviour, the entropy is tried to be maximized, since the maximization of the entropy of the thresholded image is interpreted as indicative of maximum information transfer. Histogram shape based methods analyze the peaks, valleys and curvatures of the image histogram and set the threshold according to these morphological parameters. Image attribute methods select the threshold by comparing the original image with its binarized version. The method looks for similarities like edges, curves, number of objects or more

complex fuzzy similarities. Iteratively searching for a threshold value that maximizes the matching between the edge map of the gray level and the boundaries of binarized images and penalizing the excess original edges can be a typical example of this approach. Clustering based algorithms initially divide the gray level data into two segments and apply the analysis afterwards. For example, the gray level distribution is initially modelled as a mixture of two Gaussian distributions representing the background and the foreground and the threshold is refined iteratively such that it maximizes the existence probability of these two Gaussian distributions. Locally adaptive methods simply determine thresholds for each pixel or a group of pixel, instead of finding a global threshold. The local characteristics of the pixels or pixel groups, such as local mean, variance, surface fitting parameters etc. is used to identify these thresholds. The spatial methods utilize the spatial information of the foreground and background pixels, such as context probabilities, correlation functions, co-occurrence probabilities, local linear dependence models of pixels etc. All these methods are tested on difference images of thermal camera sequences. Experiments showed that, the entropy-based approaches give best results for the tested dataset.

2.5 Data Validation

The output of a foreground detection algorithm where decisions are made independently at each pixel will generally be noisy, with isolated foreground pixels, holes in the middle of connected foreground components, and jagged boundaries. Data validation can be defined as the process of improving the candidate foreground mask based on information obtained from outside the background model. Data validation phase is sometimes referred to as the post-processing phase of the foreground mask (pixels). There are two kinds of misclassifications that may occur in segmentation results. False positives occur when background regions are incorrectly labelled as foreground. Conversely, false negatives occur when foreground regions are classified as background. Data validation aims to reduce the number of such misclassifications without an appreciable degradation in classification speed.

The simplest techniques simply post-process the foreground mask with standard binary image processing operations, such as median filters to remove small groups of pixels that differ from their neighbours' labels (salt and pepper noise) or morphological operations to smooth object boundaries. Post-processing can be applied either to the binary image representing the foreground map $F_t(x, y)$ resulted from the foreground detection phase only, or to both the binary image and the original frame. The former case has the advantage of reducing the false alarm probability at low computational cost. However, the a priori topological assumptions (compactness and regular contours) on which they are based may not always be valid. For this reason, these techniques often

result in blocky contours. To solve this problem, the original sequence may be used along with the detection result in the post-processing phase. Motion, colour and edge information are typical examples of features that are analyzed to improve the spatial accuracy of the detection result. All the background models discussed earlier have three main limitations: first, they ignore any correlation between neighbouring pixels; second, the rate of adaptation may not match the moving speed of the foreground objects; and third, non-stationary pixels from moving leaves or shadow cast by moving objects are easily mistaken as true foreground objects. The first problem typically results in small false-positive or false-negative regions distributed randomly across the candidate mask. False positives resemble pepper noise and are typically attributed to camera noise. That is, they are small (1-2 pixel), incorrectly classified regions surrounded by correctly classified background pixels. False negatives arise because of the existence of similarities between the colours of foreground objects and the background. They form holes in correctly classified foreground regions and can be quite large. Consequently, they are more difficult to remove than false positives. The most common approach is to combine morphological filtering and connected component grouping to eliminate these regions, whilst preserving the contours of correctly classified regions. Applying morphological filtering on foreground masks eliminates isolated foreground pixels and merges nearby disconnected foreground regions. Many applications assume that all moving objects of interest must be larger than a certain size. Connected-component grouping can then be used to identify all connected foreground regions, and eliminate those that are too small to correspond to real moving objects.

CHAPTER 3

PROPOSED ALGORITHM

Algorithm developed for the Smart Vigilance:

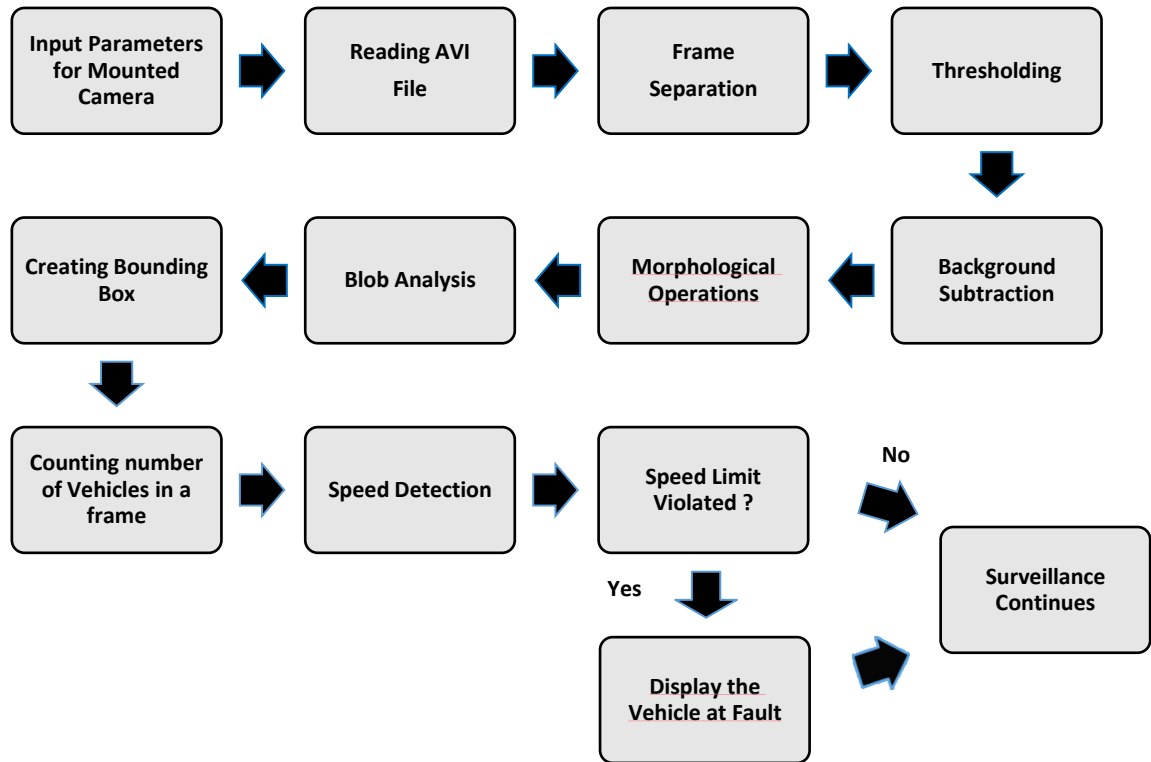


Figure 2: Algorithm

3.1 Input Parameters for Mounting Camera

The camera is placed at a certain height on the road. Camera is generally mounted at a good height so as to get a clear view for the surveillance. A dialog box is created for the taking the input parameter for the camera. Here, the parameter is road length covered in a camera frame from one end to another. Road length is needed for the speed calculation of the vehicles to be tracked.

3.2 Reading AVI File

AVI or Audio Video Interleave is a format brought by Microsoft in which audio and video are stored simultaneously. It was introduced in November 1992. Its extension is represented as “.avi”. It is derived from Resource Interchange File Format or RIFF which comprises of blocks which together forms the files data. An AVI file is converted into a single block in a RIFF format which comprises of two compulsory blocks and one optional block.

The first block contains width, height and frame rate of the video. The second block contains the data about audio and video .The third optional block contains the index numbers.

For the object detection, ‘.avi’ file is required to be used as an input to the algorithm. All the image processing techniques and further calculations are used on this video only.

Reading AVI file includes the extracting the properties of the same video i.e. the frame width, frame height, aspect ratio, frame rate and colour properties. These properties are required to fix the parameters for the further calculations.

3.3 Frame Separation

A video consists of many frames. Standard frame rate is 30 frames per second i.e. 30 frames run after one another in 1 second and so on which when seen by an individual appears to be a moving image or video. High Speed Camera captures 120 frames in a second which when seen appears to be a slow image playback.

To work on the video, we need to extract the original frames from the video which in this case is 30 frames in a second so that the further algorithms can be applied on the single frame.

3.4 Thresholding

Thresholding separates a frame into regions corresponding to objects. The regions are segmented on identifying common properties. The basic property that each pixel has is intensity. So separation of light and dark regions is done using thresholding. Thresholding creates binary images from grey-level ones by turning all pixels below some threshold to zero and all pixels about or above that threshold to one.

3.5 Background subtraction

Background Subtraction or Foreground Detection, is a technique of image processing wherein an image's foreground is extracted for further processing. Background Subtraction is a widely used approach for detecting moving objects in videos from static cameras. It detects moving objects from the difference between the current frame and a reference frame, often called "background image", or "background model". It is mostly done if the image in question is a part of a video stream.

Background Subtraction using Mixture of Gaussians Method

In the context of a traffic surveillance system, A model has been proposed where each background pixel using a mixture of three Gaussians corresponding to road, vehicle and shadows. This model is initialized using an EM algorithm. Then, the Gaussians are manually labelled in a heuristic manner

as follows: the darkest component is labelled as shadow; in the remaining two components, the one with the largest variance is labelled as vehicle and the other one as road. This remains fixed for all the process giving lack of adaptation to changes over time. For the foreground detection, each pixel is compared with each Gaussian and is classified according to its corresponding Gaussian. The maintenance is made using an incremental EM algorithm for real time consideration. This idea is generalized by modelling the recent history of the colour features of each pixel $X_1 \{ \dots, X_t \}$ by a mixture of K Gaussians. We remind below the algorithm.

3.6 Morphological Operations

Morphology is a broad set of image processing operations that process images based on shapes. Morphological operations apply a structuring element to an input image, creating an output image of the same size. The most basic morphological operations are dilation and erosion. In a morphological operation, the value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighbours. By choosing the size and shape of the neighbourhood, you can construct a morphological operation that is sensitive to specific shapes in the input image.

Dilation

Dilation is an operation that “grows” or “thickens” objects in a binary image. The specific manner and extent of this thickening is controlled by a shape referred to as a structuring element. In other words, the dilation operation usually uses a structuring element for probing and expanding the shapes contained in the input image. Dilation is commutative; that is $A+B = B+A$. It is a convention in image processing to let the first operand of $A+B$ be the image and the second operand be the structuring element, which usually is much smaller than the image.

Erosion

Erosion “shrinks” or “thins” objects in a binary image. As in dilation, the manner and extent of shrinking is controlled by a structuring element.

In practical image-processing applications, dilation and erosion are used most often in various combinations. An image will undergo a series of dilations and/or erosions using the same, or sometimes different, structuring elements.

3.7 Blob Analysis

The Purpose of Blob Analysis is to separate different objects in an image and then we have to evaluate the object which we are looking for i.e. Circles and Humans respectively. The former process is called **Blob Extraction** and the latter one is **Blob Classification**. Blob stands for binary large object and refers to a group of connecting pixels in binary image. The term 'Large' referring to objects of certain size is of interest. Smaller ones are not acceptable as they are generally noise.

BlobExtraction

The Purpose of Blob Extraction is to isolate the Blobs in a binary image. As mentioned above, a Blob consists of connected pixels. Whether or two pixels are connected is defined by connectivity, that is which pixels are neighbours and which are not. The two most often applied types of connectivity are 4-connectivity and 8-connectivity. A number of different algorithms exist for finding the blobs and such algorithms usually referred to as connected component analysis or connected component labelling.

Blob Classification

Basically in Blob Classification, Basically we evaluate the shapes of extracted objects i.e. Circles, Squares, Rectangles, Humans, Cars, Trees .etc and whatever shapes present in the picture at one particular instant in the video.

3.8 Creating Bounding Box

The bounding box also called smallest enclosing box is a term used in geometry. For a point Set(s) in N dimensions, it refers to the box with smallest measure (i.e. Area, value or hyper value in higher dimensions) within which all the points lay. When other kinds of measures are used, Bounding Box is usually called Minimum Parameter box.

3.9 Counting Number of Vehicles in a Frame

This step is needed to differentiate between two cars if a speed limit is violated by any of them. In this step, number of bounding box created through Blob Analysis is counted which in turn is equal to the number of vehicles in the frame.

3.10Speed Detection

Speed Detection is the major step in this algorithm as it fulfils the major objective of this algorithm. Speed Detection if violated marks the vehicle and takes a snapshot of the vehicle. Speed Detection is done through taking the centroid of the bounding box into consideration. Initial Coordinates for the centroid is taken and they are compared with the new coordinates for the centroid in the subsequent frame. Distance is calculated between the two coordinates and this distance is converted into the form of road length taken as input parameter. Now distance is converted into kilometres. As the input video is of 30 frames per second and two frames are considered, therefore time is calculated for two frames and by this way speed of a vehicle is determined.

3.11Speed Violation

A speed limit is set to mark as a parameter for the comparison. Speed detected for the vehicle is compared to the speed limit in each calculation. If the speed of the vehicle is found to be more than the speed limit then bounding box is converted into red and a snapshot is taken of the vehicle with a timestamp in the image.

CHAPTER 4

FUNCTIONS USED

List of functions used in the MATLAB Code.

1. Input Dialog
2. Vision ForegroundDetector

3. String to Number
4. Vision Video File Reader
5. Step
6. Function Handle
7. Axes
8. Morphological Structuring Element
9. Vision Blob Analysis
10. Insert Shape

4.1 Input Dialog

Input dialog is a predefined command in MATLAB which takes an input into a modal dialog box created by this command. This input is entered as a string and is stored into a variable. Format of this command is as follows:

Variable = inputdlg(input text)

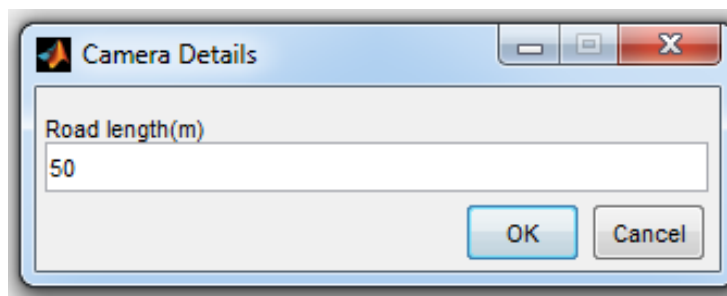


Figure 3: Input Dialogue Box

A modal dialog box stops the user from working on other windows before entering the text in it. It must contain the equal elements as input and all elements must be strings.

4.2 Vision Foreground Detector

The Foreground Detector System object takes a colour or gray scale video frame and compares it to a background model to check whether individual pixels are part of the background or the foreground. By using subtraction of background technique, we can analyze objects as foreground in an image taken from a still camera.

`ForegrndDetector = vision.ForegroundDetector(Name,Value)`

It returns a foreground detector System object, detector, with each specified property name set to the specified value. Properties which can be used in this system are as follows:

- Adapt learning rate
- Number of initial video frames for training background model
- Learning rate for parameter updates
- Threshold to determine background model
- Number of Gaussian modes in the mixture model
- Variance when initializing a new Gaussian model

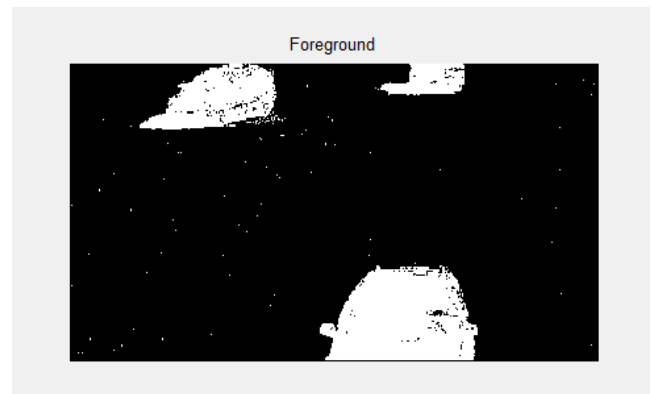
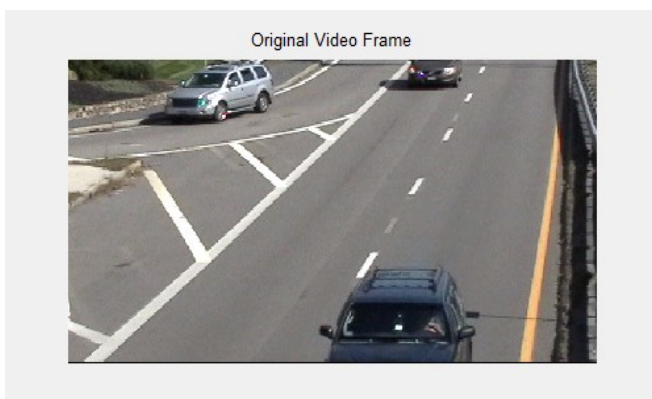


Figure 4: Foreground of an Input Frame

4.3 String to Number

It converts the string which is an ASCII character representation of a numeric value, to numeric value. `str2num` also converts string matrices to numeric matrices. If the input string does not represent a valid number or matrix, `str2num(str)` returns the empty matrix in variable.

`Variable = str2num(input)`

The input string can contain one or more numbers separated by spaces, commas, or semicolons, such as '5', '10,11,12'. In addition to numerical values and delimiters, the input string can also include a decimal point, leading + or - signs, the letter `e` or `d` preceding a power of 10 scale factor, or the letter `i` or `j` indicating a complex or imaginary number.

```
>>'50'
```

```
>>data = str2num(a1{:})
```

```
>>data =
```

```
50
```

4.4 Vision Video File Reader

The VideoReader object takes video frames, images, and audio samples as input from a video file. The object also enables the user to read image files.

`VideoReader = vision.VideoFileReader(FILENAME)`

It returns a video file reader System object, VideoReader. The object can simultaneously work on video frames and/or audio samples from the input video file, FILENAME. Every calling of the step method returns the next video frame.

`VideoReader = vision.VideoFileReader(Filename,Name,Value)`

It changes the properties of the video file reader, specified as numerous name-value pair arguments. Unspecified properties have default values.

`videoReader =`

System: `vision.VideoFileReader`

Properties:

Filename: [1x75 char]

PlayCount: Inf

ImageColorSpace: 'RGB'

VideoOutputDataType: 'single'

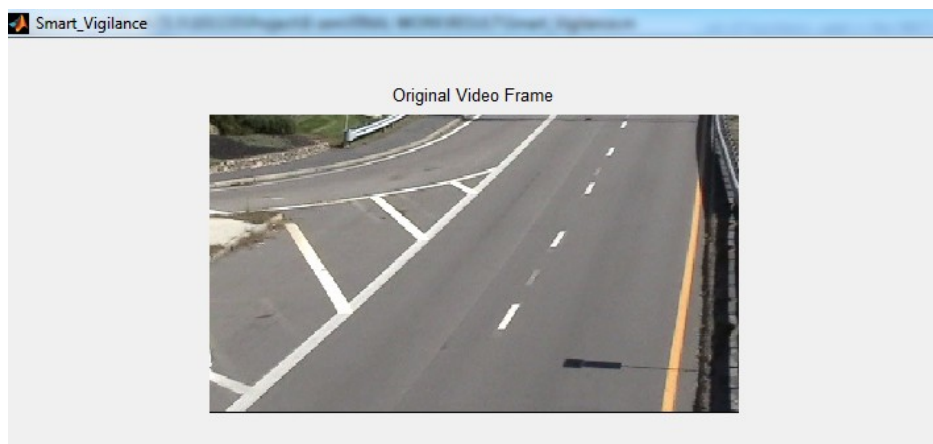


Figure 5: Original Video Frame

4.5 Step

Step function is used to play video or image sequence. Each call to the step method displays the next video frame.

`step(videoPlayer, I)`

It displays a grayscale or a RGB video frame, I, in the video player.

4.6 Function Handle

A function handle is a MATLAB value that acts like a means of calling a function through some other medium. We can use function handles in calling to other functions. We can also save function handles in arrays for later use.

At the time of creation of a function handle, the function we specify must be on the path provide by the MATLAB and in the code creating the handle. We can perform a local function from a separate (out-of-scope) file using a function handle. The requirement for that is the handle should be created by the local function.

It constructs an anonymous function and returns a handle to that function. The body of the function is a single MATLAB statement or command. Call the function by calling it by means of the function handle, handle.

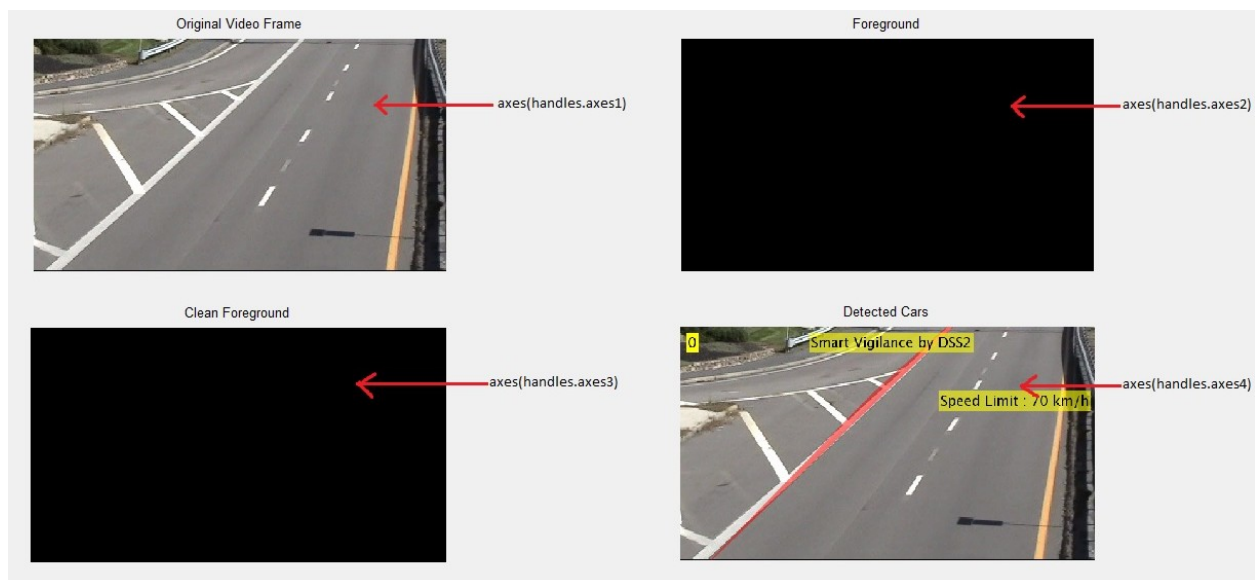


Figure 6: Handling Multiple Axes in a Single Window

4.7 Axes

Axes creates an axes graphics object in the current figure with the property values set to default. Axes are the basic function for developing axes graphics objects. MATLAB itself creates axes, if another one does not exist, when we give the command that creates a graph.

Axes('PropertyName',propertyvalue)

It makes an axes object with the property values as specified. The axes function takes property name/property value pairs, structure arrays, and cell arrays as input arguments. While the basic

purpose of an axes object is to provide a geometry system with coordinates for data plotted, axes properties provide a good control over the way of displaying data.

`axes(h)`

It makes h as the current axes and brings it into focus. It also makes h the first axes and sets the figure's `currentaxes` property to h. The current axes is used by the functions that draw image, line, patch, rectangle, surface, and text graphics objects.

If we want to make an axes the current axes without changing the properties of the figure, set the `CurrentAxes` property of the figure containing the axes:

`set(figure_handle, 'CurrentAxes', axes_handle)`

This command will make a figure to remain minimized or hidden below other figures, but want to specify the current axes.

`h = axes(...)`

It returns the handle of the created axes object.

Use the `set` function to change the properties of an axes or the `get` function to know the current values of axes. Use the `gca` command to get the handle of the current axes.

Default axes properties set on the figure and rootobject levels:

`set(0, 'DefaultAxesPropertyName', PropertyValue, ...)`

`set(gcf, 'DefaultAxesPropertyName', PropertyValue, ...)`

PropertyName is the name of the axes property and *PropertyValue* is the value we are specifying.

Use `set` and `get` to use axes properties.

`Axes(handles.axes)`

It handles the graphical output to the current axes being handled by the MATLAB code.

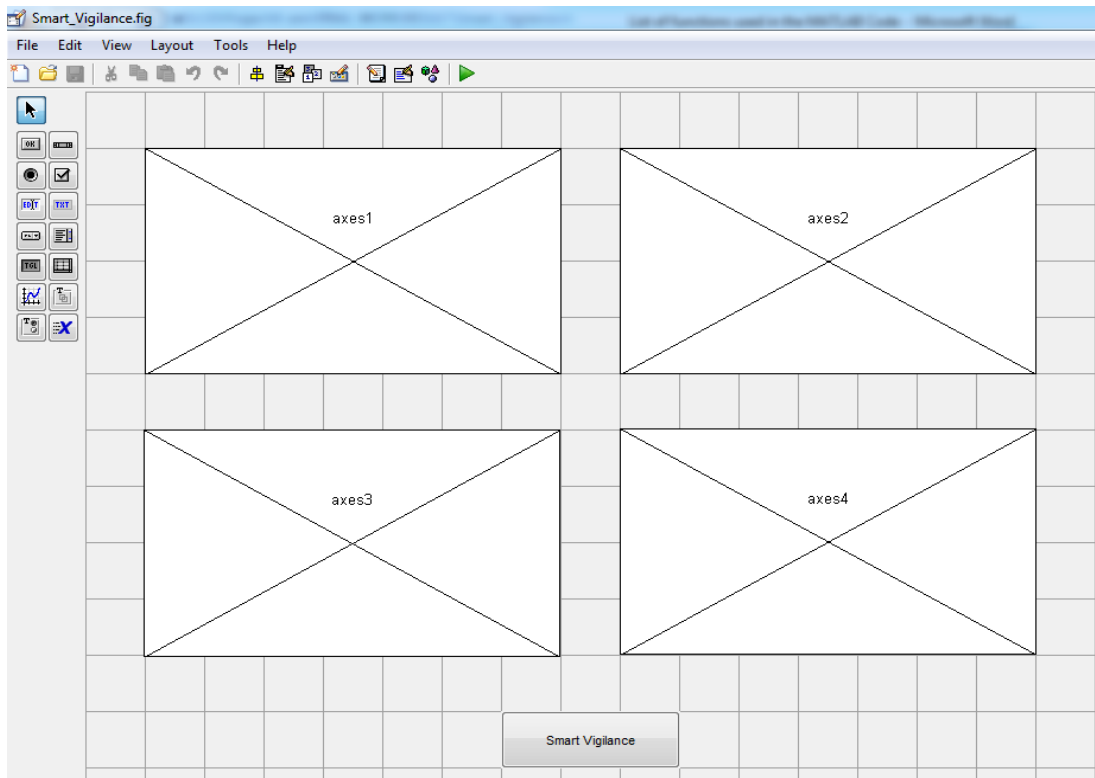


Figure 7: Axes

4.8 Morphological Structuring Elements

Structuring Elements command creates a structuring element, SE, of the type specified by shape. Depending on shape, it can take additional parameters.

Flat Structuring Elements

- Arbitrary
- Pair
- Diamond
- Periodicline
- Disk
- Rectangle
- Line
- Square
- Octagon

Non-flat Structuring Elements

- Arbitrary
- Ball


```
se = strel('arbitrary', NEIGHBORHOOD)
```

It creates a flat structuring element with neighbourhood. NEIGHBORHOOD is a matrix containing 1's and 0's; the location of the 1's are used for the morphological operations and defines the neighborhood. The *origin* of NEIGHBORHOOD is its center element, given by $\text{floor}((\text{size}(\text{NEIGHBORHOOD})+1)/2)$. We can just use

```
strel(NEIGHBORHOOD)
```

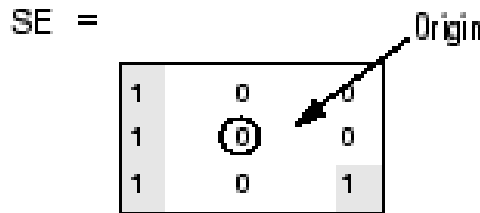


Figure 8: Basic strel Matrix

```
se = strel('arbitrary', NEIGHBORHOOD, HEIGHT)
```

It makes a non-flat structuring element. HEIGHT is a matrix with the same size as of NEIGHBORHOOD containing the height values associated with each nonzero element of NEIGHBORHOOD. You can use

```
strel(NEIGHBORHOOD,HEIGHT).
```

```
se = strel('ball', R, H, N)
```

It creates a non-flat, structuring element with a ball shape whose radius in the X-Y plane is R and whose height is H. When N is not less than or equal to 0, the ball-shaped structuring element is approximated by a sequence of N non-flat, line-shaped structuring elements. When N equals 0, no approximation is used, and the structuring element members contain all pixels whose centers are not greater than R away from the origin. The height values of the ellipsoid are determined by the formula specified by R and H. If N is not specified, the default value is 8.

```
se = strel('diamond', R)
```

It creates a flat structuring element with a diamond shape, where R marks the distance from the structuring element origin to the points of the diamond points in the matrix.

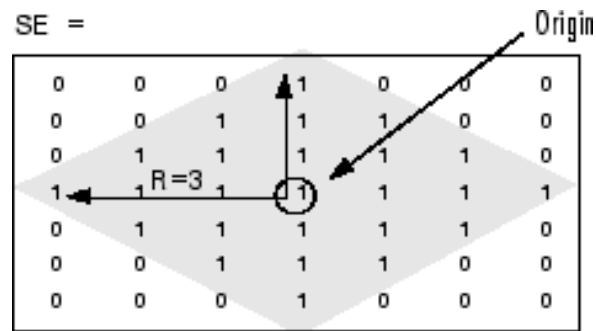


Figure 9: Diamond Shaped strel Matrix

`se = strel('disk', RADIUS, N)`

It creates a flat structuring element with a disk shape, where R specifies the radius. R must be a positive integer. N must be 0, 4, 6, or 8. When N is greater than 0, the disk-shaped structuring element is taken as a sequence of N periodic-line structuring elements. When N equals 0, nothing is approximated, and the structuring element members contain all pixels whose centers are not greater than R away from the origin. The default value is 4 if not specified.

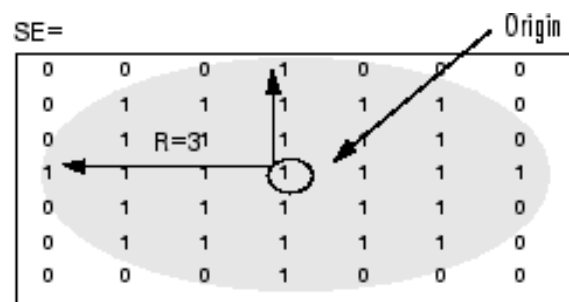


Figure 10: Disk Shaped strel Matrix

`se = strel('line', LENGTH, DEGREE)`

It creates a flat linear structuring element that is symmetric with respect to the neighborhood center. DEGREE specifies the angle of the line in a anti-clockwise direction from the horizontal axis. LENGTH is the distance between the centers of the structuring element members which are at the opposite ends of the line.

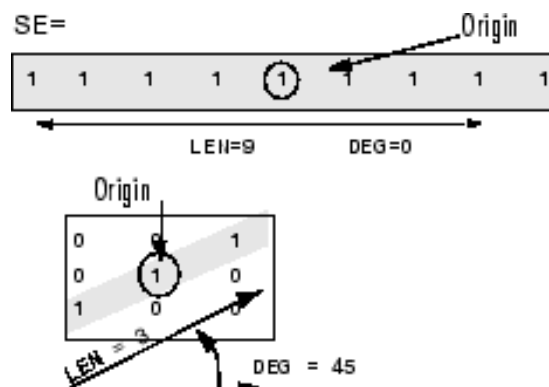


Figure 11: Angle in strel Matrix

`se = strel('octagon', RADIUS)`

It creates a flat octagonal structuring element, where RADIUS specifies the distance from the structuring element origin to the sides of the octagon.

`se = strel('pair', OFFSET)`

It creates a flat structuring element containing only two members and not more than that. First member is located at the origin. The second member's location is specified by the vector OFFSET.

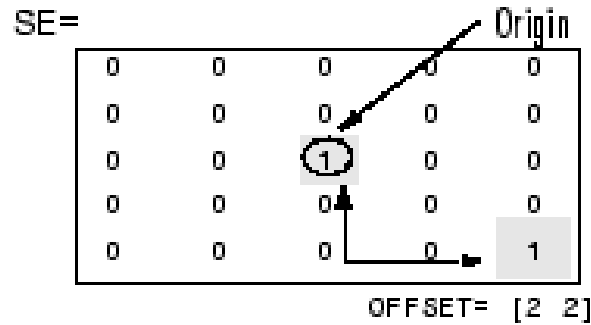


Figure 12: OFFSET in strel Matrix

`se = strel('rectangle', MN)`

It creates a flat structuring element with a rectangle shape, where MN specifies the size. The first element of MN is the number of rows in the structuring element and the second element is the number of columns.

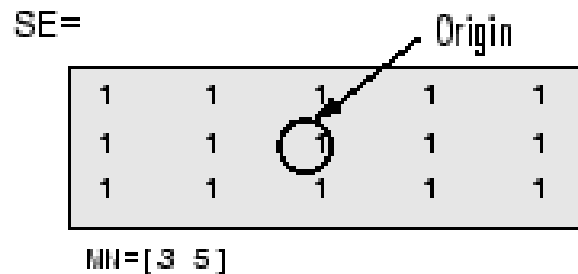


Figure 13: Rectangle Shaped strel Matrix

`se = strel('square', WIDTH)`

It creates a square structuring element whose width is WIDTH pixels.

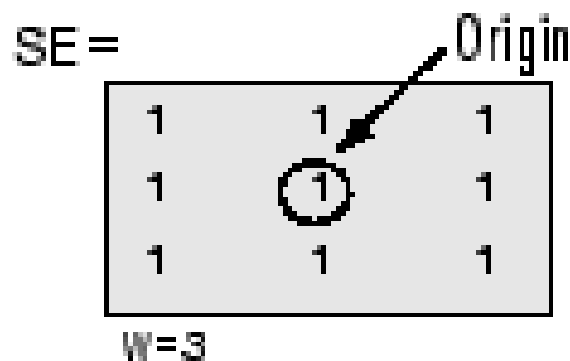


Figure 14: Width determined Square strel Matrix

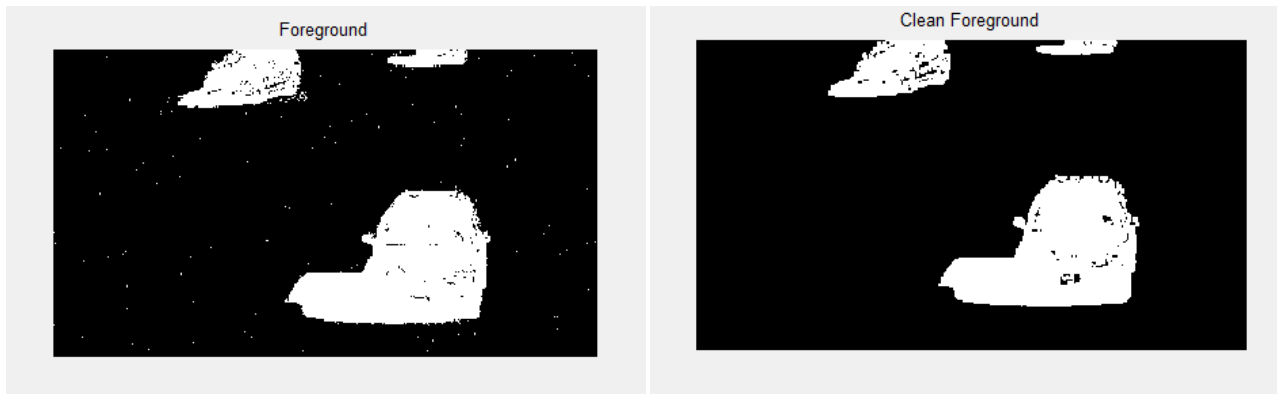


Figure 15: Foreground after Morphological Operations

4.9 Vision Blob Analysis

Blob Analysis block is used to calculate statistics for labelled regions in a binary image. The block returns values such as the centroid, bounding box, label matrix, and blob count. The Blob Analysis block has a support input and output variable signal size.

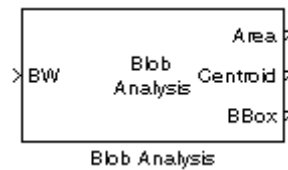


Figure 16: Blob Analysis Block Diagram

Area

This property is used to output a vector that represents the amount of pixels in labelled regions.

Centroid

This property is used to output an M -by-2 matrix of $[x \ y]$ centroid coordinates. The rows represent the coordinates of the centroid of each region in an image, where M represents the amount of blobs.

Example: Suppose there are two blobs, where coordinates of respective centroids are x_1 , y_1 and x_2 , y_2 . The block outputs at the Centroid field.

$$\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix}$$

Bounding box

This property is used to output an M -by-4 matrix of $[x \ y \ \text{width} \ \text{height}]$ bounding boxes. The rows represent the coordinates of each and every bounding box, where M represents the amount of blobs.

Example: Suppose there are two blobs, where x and y belongs to the location of the upper-left corner of the bounding box, and w , h denotes the width and height of the bounding box. The block outputs at the BBox field.

$$\begin{bmatrix} x_1 & y_1 & w_1 & h_1 \\ x_2 & y_2 & w_2 & h_2 \end{bmatrix}$$

Major axis length

This property is used to output a vector with the following characteristics:

- Lengths of the major axes of ellipses are represented
- Has the equal normalized second central moments as the regionslabelling

Minor axis length

This property is used to output a vector with the following characteristics:

- Lengths of the minor axes of ellipses are represented
- Has the same normalized second central moments as the regionslabelling

Orientation

This property is used to output a vector that represents the angles between the major axes of the ellipses and the x -axis. The angle values are measured in radians and range between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$.

Eccentricity

This property is used to output a vector that represents the eccentricities of ellipses that have the same moments as the region.

Equivalent diameter squared

This property is used to output a vector that represents the equivalent diameters squared.

Extent

This property is used to output a vector that represents the results of dividing the areas of the blobs by the bounding boxes area.

Perimeter

This property is used to output an N -by-1 vector of the perimeter lengths, in pixels, of each blob, where N is the amount of blobs.

Statistics output data type

The data type of the outputs at the Centroid, MajorAxis, MinorAxis, Orientation, Eccentricity, Equivalent diameter squared, and Extent. If we select Fixed-point, the block can't calculate the major axis, minor axis, orientation, or eccentricity and the associated properties become unavailable.

Connectivity

It defines which pixels connect to each other. If we want to connect pixels located on the top, bottom, left, and right, 4 is selected. If we want to connect pixels to the other pixels on the top, bottom, left, right, and diagonally, 8 is selected. The Connectivity parameter also affects how the block calculates the blob perimeter.

```
>>blobAnalysis =
```

```
    System: vision.BlobAnalysis
```

```
    Properties:
```

```
AreaOutputPort: false
```

```
CentroidOutputPort: false
```

```
BoundingBoxOutputPort: true
```

```
MajorAxisLengthOutputPort: false
```

```
MinorAxisLengthOutputPort: false
```

```
OrientationOutputPort: false
```

```
EccentricityOutputPort: false
```

```
EquivalentDiameterSquaredOutputPort: false
```

```
ExtentOutputPort: false
```

```
PerimeterOutputPort: false
```

```
OutputDataType: 'double'
```

```
    Connectivity: 8
```

```
LabelMatrixOutputPort: false
```

```
MaximumCount: 50
```

```
MinimumBlobArea: 150
```

```
MaximumBlobArea: 4294967295
```

```
ExcludeBorderBlobs: false
```

4.10 Insert Shape

It inserts shape in image or video.

```
RGB = insertShape(I,shape,position)
```

It returns a truecolor image with shape inserted. The input image, I, can be either a truecolor or grayscale image. You draw the shapes by overwriting pixel values.

RGB = insertShape(___,Name,Value)

It uses additional options specified by one or more Name,Value pair arguments.

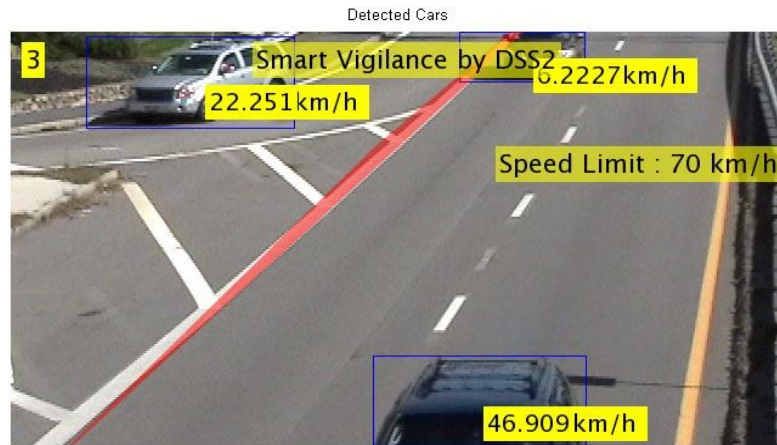


Figure 17: Frame after Inserting Various Shapes

CHAPTER 5

OUTPUT

Initial Screen

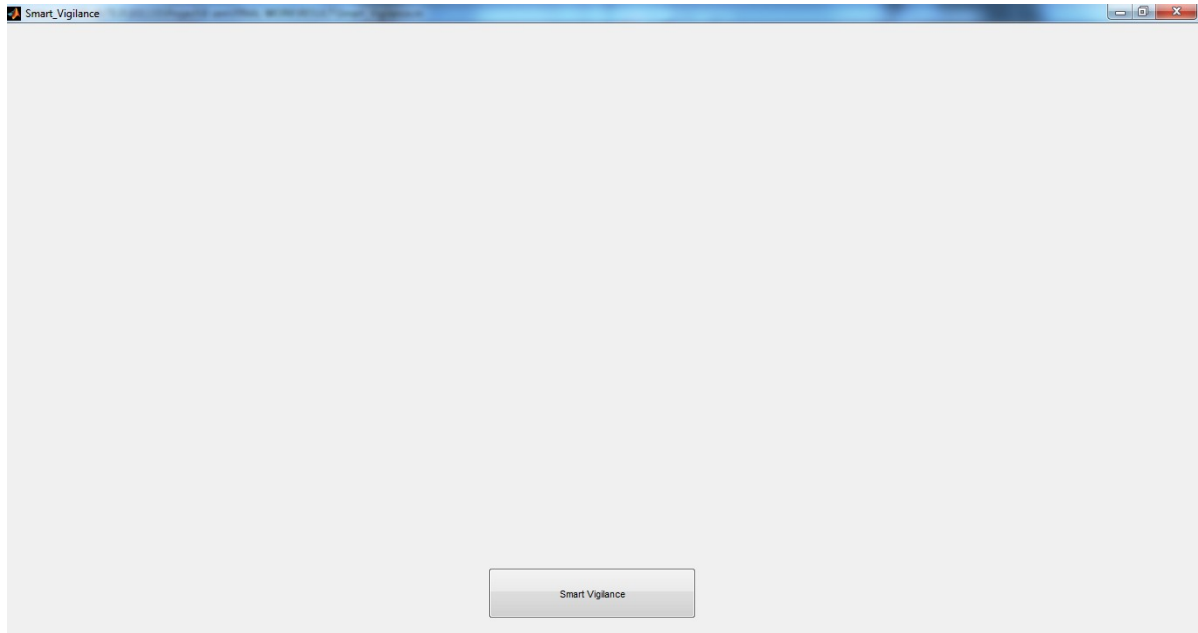


Figure 18: Initial Screen

Smart_Vigilance Button is Selected

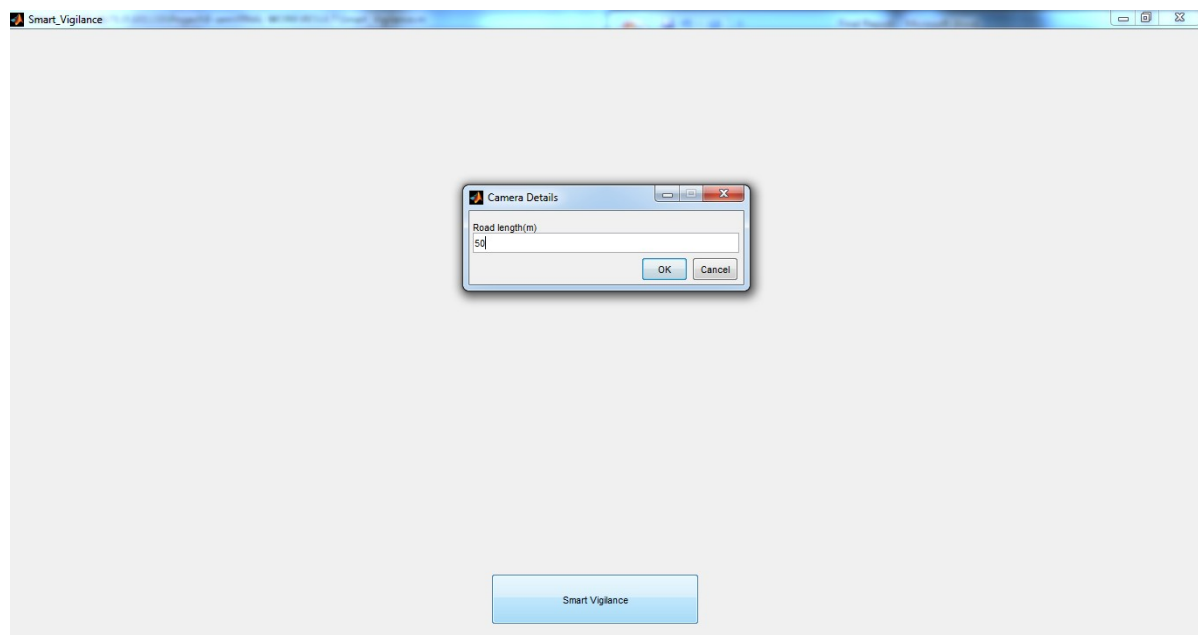


Figure 19: Smart_Vigilance Button is Selected

Road Length is Entered

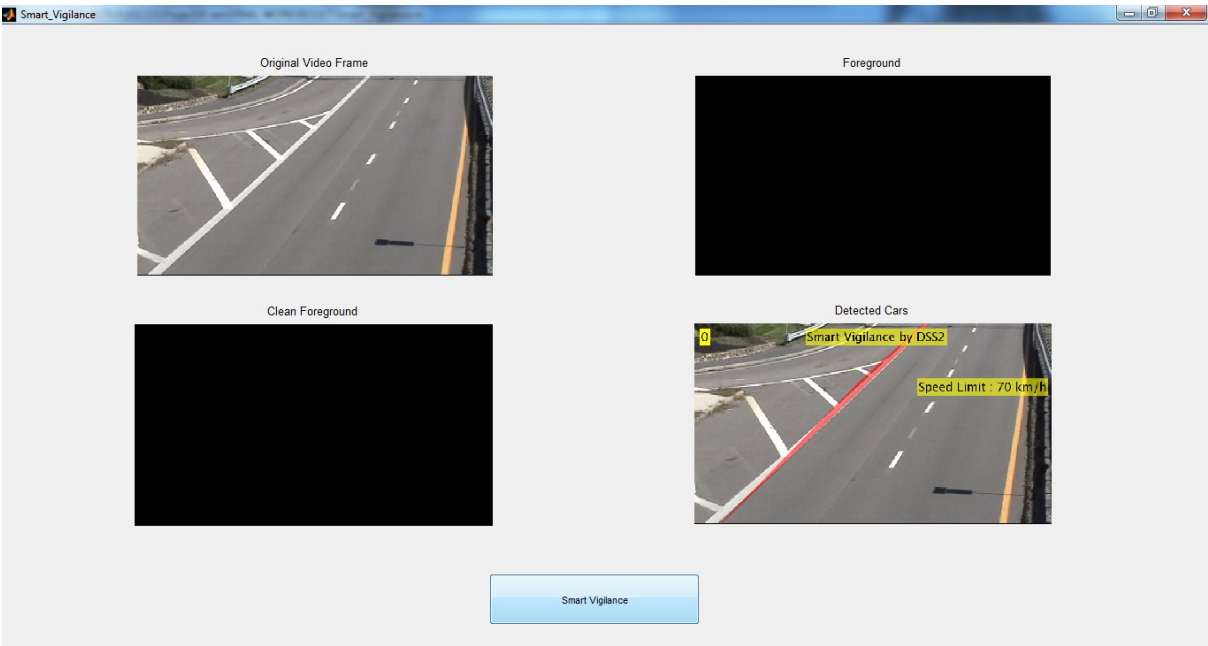


Figure 20: Output after Road Length is Entered

Vehicles Appear



Figure 21: Vehicles Appear

Speed Limit Violated by a Vehicle

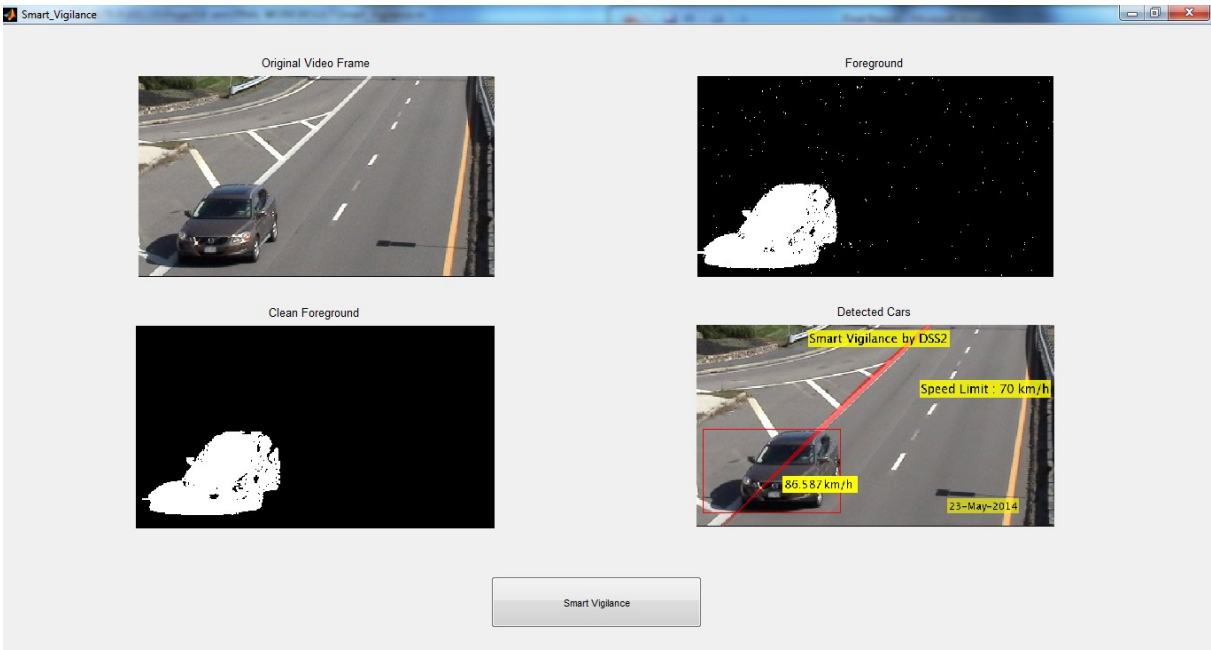


Figure 22: Speed Limit is Violated

Snapshot of a Vehicle

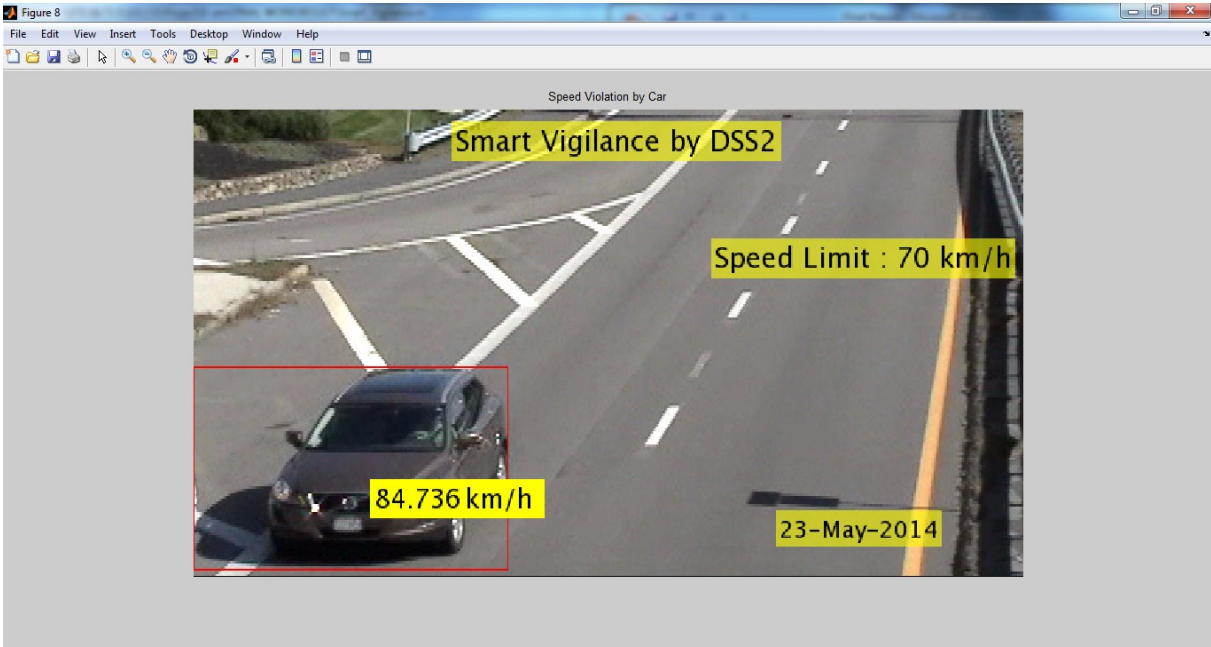


Figure 23: Snapshot of a Vehicle

All the Snapshots saved to a desired Folder

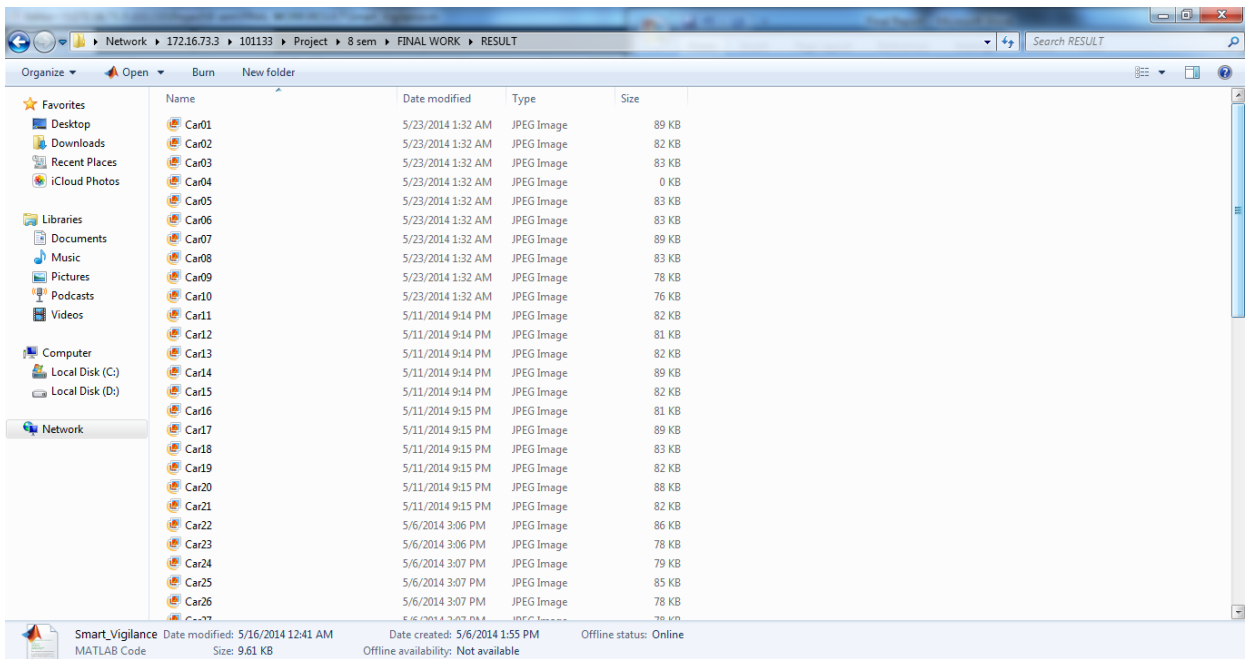


Figure 24: Images saved to a Folder

CONCLUSION

In this project, we have mainly focussed on Multiple Vehicle Tracking and then Velocity Estimation in Traffic Surveillance System. Firstly Different frames are separated in the input video and Vehicles lying in region of interest are detected and segmented. After that further tracking of Vehicles is done using Background Subtraction algorithm specifically through Mixture of Gaussians Method Technique which is best suited for surveillance systems as far as accuracy and precision is concerned and perform at an average level. We talk about issues like memory requirements and computation time, but since we are working on real time applications, we have worked on the improvisation of computation time using some Fusion Techniques. After Tracking of Vehicles, Number of Vehicles in each frame is calculated and Speed of each Vehicle is calculated and then according to Speed Violation Rules of different regions, calculated Data is implemented to capture the vehicle at fault.

In Future, we are willing to improvise our proposed algorithm by introducing following techniques:-

Lane Calibration: We will focus on identifying the vehicles in each lane separately, the lanes must be defined and the centre line of each Lane must be identified. User should be allowed to select the region of interest by selecting the number of lanes first and then starting and ending point on each lane to define the tracking region.

Compressive Sensing: Compressive Sensing is an emerging field that provides a framework for image work using sub-Nyquist Sampling Rates. The CS theory shows that a Signal can be constructed from a small set of random projections, provided that the signal is sparse in some basis. This approach is suitable for image coding in communication constrained problems by using data captured by multiple conventional cameras to provide 2D tracking and 3D shape reconstruction.

Lab View Software: Lab View is a System design platform and development environment for a VISUAL PROGRAMMING LANGUAGE from NATIONAL INSTRUMENTS. We are working on aspect of blending the good features of Lab View with MATLAB using API or DLL libraries because for practical real time hardware implementation Lab View constitutes some tremendous features.

REFERENCES

- [1] Gimp software. <http://www.gimp.org/downloads/>.
- [2] Image video machine software. http://www.download.com/Image-Video-Machine/3000-2186_4-10359035.html.
- [3] Ulead video studio software. <http://www.ulead.com/vs/>.
- [4] MATLAB/SIMULATION technical reference. Technical Report 508035-0001 Rev. A, November 2004.
- [5] C. Kamath A. Gyaourova and S.-C. Cheung. Block matching for object tracking. <https://computation.llnl.gov/casc/sapphire/pubs/UCRL-TR-200271.pdf>, October 2003.
- [6] David S. Bright. Removing shot noise from face image. <http://www.nist.gov/lispix/imlab/noise/shotfc.html>, May 25 2004.
- [7] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. ARTIFICIAL INTELLIGENCE, pages 389_407, 1992.
- [8] Jan F. Jørgensen. Mean filter in World Wide Web. http://www.imagemet.com/WebHelp/hid_filters_smoothing_mean.htm.
- [9] Martin Mangard. Motion detection on embedded smart cameras. Master's thesis, Technische Universität Graz, 2006.
- [10] Ashley Walker Robert Fisher, Simon Perkins and Erik Wolfart. Gaussian smoothing in world wide web. <http://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm>, 2003.
- [11] Ashley Walker Robert Fisher, Simon Perkins and Erik Wolfart. Mean filter in world wide web. <http://homepages.inf.ed.ac.uk/rbf/HIPR2/mean.htm>, 2003.
- [12] Dr. Mike Spann. Image segmentation in world wide web. <http://www.eee.bham.ac.uk/spannm/Teaching%20docs/Computer%20Vision%20Course/Image%20Segmentation.ppt>, Course resources for 2008.
- [13] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. ACM Compute. Serve., 38(4):13, 2006.
- [14] Y. T. Zhou, V. Venkateswar, and R. Chellappa. Edge detection and linear feature extraction using a 2-d random field model. IEEE Trans. Pattern Anal. Mach. Intel.,
- [15] Background Modeling using Mixture of Gaussians for Foreground Detection - A Survey T. Bouwmans, F. El Baf, B. Vachon *Laboratoire MIA, Université de La Rochelle, Avenue M. Crépeau, 17000 La Rochelle, France*

[16]Moving Object Detection in Spatial Domain using Background Removal

Techniques - State-of-ArtShireen Y. Elhabian*, Khaled M. El-Sayed* and Sumaya H. Ahmed*

[17]MathworksInstructions Software <http://www.mathworks.in/>