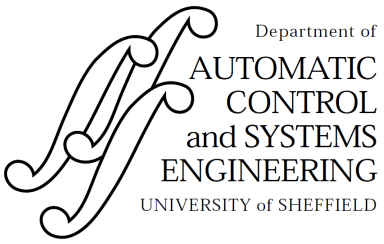
The University Of Sheffield.



Department of AUTOMATIC CONTROL and SYSTEMS ENGINEERING
UNIVERSITY of SHEFFIELD

Background Subtraction Techniques for Use in Drone Applications

by Fraz Anwar Makki

Registration Number 130135737

Supervised by Dr Lyudmila Mihaylova

Co-supervised by Dr Ata Ur-Rehman

A dissertation submitted in partial fulfillment of the requirements for the degree of Master of Science in Control Systems Engineering

Acknowledgments

I would like to give a special thanks to Dr Lyudmila Mihaylova for all her help and guidance she provided during the duration of this project. Secondly I would like to thank Dr Ata Ur-Rehman who helped me to understand the concepts of video processing and helping me resolve the many issues I encountered throughout my project. Lastly I would like to thank the Automatic Control and Systems Engineering department and also the University of Sheffield for the usage of the facilities throughout the practical work on the project.

Abstract

This project will be focused on the development of efficient methods and techniques for filtering and control of AR drones. The AR drone has cameras acquiring video data which can be noisy and subject to uncertainties. One direction of the work is development of image and video recognition techniques for important events detection.

The project work involves various background subtraction methods that can be further categorized as traditional methods which include frame differencing and running average as well as statistical methods which include the mixture of Gaussians method.

Furthermore performance evaluation for the various methods will be carried out and the different advantages and disadvantages of each method would be discussed using the acquired results. Performance will be then be evaluated in order to determine possible use in real-time applications.

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Description
AM	Approximate median
BS	Background subtraction
EASA	European aviation safety agency
FAA	Federal Aviation Administration
FAST	Features from Accelerated Segment Test
FD	Frame differencing
FN	False negative
FP	False positive
GMM	Gaussian mixture model
MOG	Mixture of Gaussians
RANSAC	RANdom SAmple Consensus
RTCA	Radio Technical Commission for Aeronautics
UAS	Unmanned aerial system
UAV	Unmanned aerial vehicle
ViBE	Visual Background Extractor

Contents

Introduction

1.1 Introduction to UAV Technology

A helicopter is a kind of flying machine that utilizes spinning rotors that cause air to be pushed downwards. Due to movement of air thrust is created that helps the helicopter to stay in the air and hover. Most helicopters incorporate two rotors. The first rotor is positioned as such so that upwards thrust is prioritized. This is the main rotor and can normally be found on top of the helicopter. The second rotor is usually found on the tail end of the helicopter. This rotor is normally much smaller than the main rotor and prioritizes sideways movement. The overall purpose to use these two rotors in tandem is to counteract the torque being produced by the main rotor. However, this design choice in order to control the angle of motion of the helicopter is complicated and although is the most common design choice, it does cause several design complexities and increased construction costs.

A quadrotor helicopter (quad-copter) as the name suggests consists of four equally spaced rotors. The rotors are normally seen at the four corners of the quad-copter body. The development of quad-copters never went in very advanced stages until recently, since these kind of aircraft required electronic assistance in order to be efficiently controlled in flight. The recent decreased costs for microprocessors in general has made electronic and in some cases completely autonomous control of quad-copters possible. Due to cost reduction, quad-copters are now being seen as a much more cost friendly and lower complexity aircraft that could be used in non-military applications such as domestic and hobbyist purposes.

Quad-copters are fundamentally complex. They consist of six degrees of freedom, of which three are translational and three are rotational and only four independent inputs exists that are the fan speeds for the four rotors. It is complicated due to the need to couple the rotational and translational

motion so that six degrees of freedom can be achieved. This results in highly non-linear dynamics. Also unlike ground vehicles, the quad-copter needs to have its own damping in order to remain stable due to very little friction that prevents its motion in the air. [1]

The Parrot AR Drone 2.0 is shown in figure 1.1. The quad-copter dynamics can be derived by

Figure 1.1: AR Drone 2 Frontal View

utilizing two frames in which the quad-copter is expected to operate in. The first frame is known as the body frame and second is known as the inertial frame. These are shown in figure 1.2. The

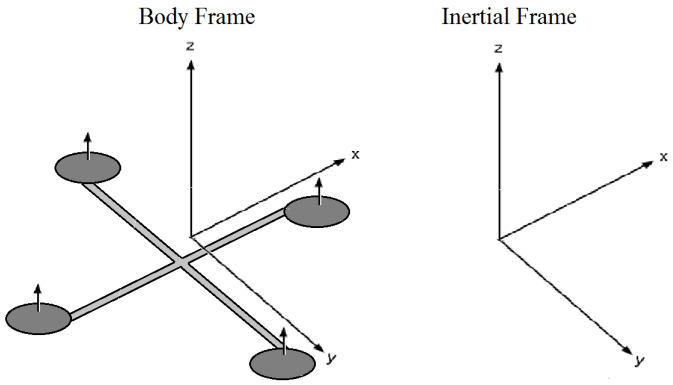


Figure 1.2: The body frame and the inertial frame

inertial frame is defined in relation with the ground, where the opposite direction to that of the z-axis is the direction of gravity. Likewise, the body frame is defined firstly by the orientation of the quad-copter body and secondly the upwards motion which is in line with the z-axis of the body frame and the inertial frame. x and y represent the directions of the quad-copter arms.

The position and velocity of the quad-copter are dependent on the inertial frame and are defined

as $x = (x, y, z)^T$ and $\dot{x} = (\dot{x}, \dot{y}, \dot{z})^T$ where x is the position and $\dot{x}$ is the velocity. The roll, pitch and yaw angles can be represented by $\dot{\phi}$, $\dot{\theta}$ and $\dot{\psi}$ respectively and hence the angular velocity of the quad-copter can be defined using $\dot{\theta} = (\dot{\phi}, \dot{\theta}, \dot{\psi})^T$. θ can also be described as the derivative with respect to time for yaw, pitch and roll. The angular velocity and the angular velocity vector, ω, are related as shown in equation 1.1,

$$\omega = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix} \dot{\theta} \tag{1.1}$$

The body of the quad-copter and the inertial frame can be related by a rotation matrix. The rotation matrix can be given by equation 1.2.

$$\text{Rotation Matrix} = \begin{bmatrix} c_\phi c_\psi - c_\theta s_\phi s_\psi & -c_\psi s_\phi - c_\phi c_\theta s_\psi & s_\theta s_\psi \\ c_\theta c_\phi s_\phi + c_\phi s_\psi & c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\psi s_\theta \\ s_\phi s_\theta & c_\phi s_\theta & c_\theta \end{bmatrix} \tag{1.2}$$

There are various *Unmanned Aerial Vehicle* (UAV) classifications. It can be seen that three main classifications have been agreed on by Aviation authorities such as the *European Aviation Safety Agency* (EASA) as well as national regulating authorities. Classifications are based mostly on the mass of the UAV in question. A summary of the properties for the three classifications of UAV are given in Table 1.1. Other relevant authorities include the *Radio Technical Commission for*

Civil category	**Mass/kg**	**Military version**	**Civil regulation**
Small unmanned aircraft	≤ 20	≤ 5 kg	National
Light UAS	$20 \geq x \leq 150$	≤ 30 kg	National
UAS	> 150	Tactical	EASA

Table 1.1: UAV Classifications

Aeronautics (RTCA) and the *Federal Aviation Administration* (FAA).

1.2 Application of UAV's

UAV's are used in many different applications in daily life. From the origination of UAV technology it was seen to be most frequently used by the military. But due to technology becoming more affordable and efficient, small UAV's caused an increase in UAV technology to be utilized in domestic

and commercial uses. Some of the uses in non military applications are described as follows.

1. *Filming and Photography*: One of the best examples of drones used in this way are in sports where they enable to get close to the actual play with minimal disruption to the players and allow the viewers to be closer with their sporting heroes [2]. Other examples in this category include the use of drones in film making. For example shooting scenes where high rise buildings need to be shown can easily be done using small UAV's with minimal cost and high quality video footage.

   However the use of UAV's in these applications is still in the grey area in terms of legality. There are no specific laws in disallowing UAV usage, however, it is still debated that UAV's are dangerous to be used in such a way and that UAV's licensed to be used in these application could be instead used for criminal activities. Nevertheless, UAV technology has a genuine application in entertainment [3].

2. *Policing*: Recently UAV's are being tested to be used in assisting the civilian law enforcement agencies like the Police. In fact the first recorded instance of using a UAV in order to assist the police to make an arrest was seen in January 2014 when a UAV was used in order to arrest a North Dakota man. UAV's may also be used by law enforcement agencies as a 'moving CCTV'. *CCTV* (closed circuit television) is often used by law enforcement agencies to detect and address potential crime scenes or even locate criminals after a crime. Drone technology is mobile, so if the law enforcement agencies desire to track potential suspects drone technology could let them do this with ease with minimal risk [4, 5].

3. *Oil, gas and mineral exploration*: This is another current UAV application. UAV's can be loaded with specific actuators and sensors in order to carry out tasks such as geographical surveys in order to help geophysicists to determine mineral locations and or terrain details. In the oil and gas industry UAV's are used for example to monitor the integrity of pipelines using digital monitoring cameras on the UAS [6].

4. *Commercial aerial surveillance*: UAV's can be used to monitor for example the roads or houses in a neighborhood or be used in agriculture for example to monitor livestock.

1.3 Problem Statement, Research Aims and Objectives

The aim of the project is to develop advanced video processing techniques that make use of background subtraction methods of various forms in order to detect objects of interest. Various algorithms will be implemented and tested for possible UAV application. The various algorithms that would be studied include:

1. Frame Differencing
2. Approximate Median
3. Adaptive Background Subtraction
4. Mixture of Gaussians
5. ViBE (Visual Background Extractor)

The purpose of evaluating these various background subtraction algorithms is to determine their real-time usage possibilities and then these methods may be then implemented in the AR Drone for use in related applications described in the previous section.

All these algorithms are various forms of *background subtraction* (BS) and are discussed in section 2 where the aspect of background subtraction is explained and the various algorithms are described.

Literature Review

2.1 Introduction to background subtraction

Before the concept of background subtraction is discussed it is important to define the various entities involved. There are two main entities that are involved, the *foreground* (FG) and *background* (BG). The *background* is known as the expected part of the image or video frame. It can be modelled in various ways. For example, at a traffic intersection it can be modelled using the image taken by the static camera when there is no traffic. The *foreground* is known as the unexpected part. It can be defined, for example, as the cars or pedestrians in the case of traffic surveillance.

Background subtraction (BS) is a method that is used to detect and localize moving objects that are being captured on a static camera. Background subtraction was first used in computer vision applications. Due to its nature it is generally agreed that moving object detection is the fundamental job of any background subtraction algorithm [7–10]. When designing algorithms for background subtraction there are two aspects which are taken into consideration, efficiency and simplicity. As mentioned earlier, objects that would be categorized as the foreground, would be entities that are not expected or more simply are not modelled as part of the background. Also in most cases the foreground object would be in motion. Hence in one way, it can be concluded that the pixel in an image at a certain time $t + n$ would be a moving object or a foreground object if its colour or texture is greatly varying from that of the background at a certain time t [11]. This assumption for a foreground entity works well if the camera in question has a low noise video with a fixed and static background [12] as is the case with most background subtraction algorithms.

Although it is possible to be able to use this definition of foreground in background subtraction algorithms, this method will yield several false negatives and false positives if the video has a poor signal-to-noise ratio, low resolution or utilizes compression codes (e.g. *H264 MP4*). False positives

for example could be caused by illumination changes in the background. False negatives for example could be caused if the moving object has similar colours to that of the background. Hence there are many challenges that involve background subtraction. The concept of false positives and negatives as well as most of the different challenges involving BS that are being researched on are described in the following section.

2.2 Key issues in background subtraction

The process of background subtraction has many challenges. One of the most important of which is to keep the number of false negatives and false positives to a minimum. A *false positive* (FP) is defined as the entity, after background subtraction, which is incorrectly identified as the foreground. A *false negative* (FN) on the other hand is known as the entity, after background subtraction, which is incorrectly identified as the background. There are many challenges in background subtraction such a low quality video, camera movement or environmental noise that cause an increase in false negatives and positives. The goal of designing a background subtraction algorithm is always to try and implement a background subtraction method that would reduce the number of incorrect detections. In order to do so it is necessary to design methods that would compensate for the various technical or environmental challenges. Some of these challenges are described in the following subsections. [13, 14]

2.2.1 Moving objects

One of the most basic challenges in background subtraction is the separation and identification of moving objects. This is normally done by modelling the background and then comparing the background with the newer images. If there exists a foreground, it will be separated using a background subtraction method for example frame differencing. One challenge involved is the possibility of an object being moved in the background. If this is the case, the background is now changed and the background model will have to be updated accordingly so that the moved object no longer is detected as a foreground object.

2.2.2 Gradual change in illumination

In the real world there is change in luminosity. As the day goes by the changes in light intensity will cause subtle changes in the background that may result in false negatives and positives to be

produced. Hence this is another challenge that needs to be taken into account of.

2.2.3 Sudden illumination

This problem is more challenging than that involving gradual illumination changes. This is due to the fact that a sudden illumination change will cause a rapid change in the background and this model will have to be updated accordingly in order to get the desired background subtraction results. During the period where the background model is not updated would result in several false positives and negatives.

2.2.4 Oscillations in background

The background may not be static as is mostly the case. This means that some entities in the background would consistently be categorized as the foreground. An example of the phenomenon is seen in moving tree branches due to wind. If the background contains trees, on a windy day the tree branches are bound to move in an oscillatory motion. This will cause a number of false positives to be produced. Hence this is another challenge involved in background subtraction techniques.

2.2.5 Camouflage

This issue is caused at a pixel level. The pixel properties of a foreground object may be detected as the background and vice-versa. This problem normally happens due to similar colours in the foreground entity and background entity.

2.2.6 Foreground aperture

If an object with a single colour tone moves, it may not be detected and subtracted correctly especially if the colour matches up with the colours in the background. For example, a person wearing a plain white shirt moving in a room whose background model includes a white wall, the white shirt may be incorrectly detected as the background. This is a greater problem, if the object is moving slowly.

2.2.7 Sleeping foreground

There may be cases where the foreground object is correctly detected and now the foreground object stops and stays in place. In this case the algorithm may incorrectly label this object as part

of the background. Instances like these have to modeled such that these objects are not made part of the background. This problem is in conjunction with the moving objects problem as for example in the moving object case, for example, includes a scene with a curtain or a chair. If someone comes in the room and opens or closes the curtains or moves the chair, and then leaves, then it would be required to update the background. This is similar to the sleeping foreground except in the sleeping foreground a previous actual foreground entity (a person for example) stops moving in the scene. Hence, in this case even though the person is still, it is best to develop some technique in order to continue detecting the person as a foreground entity and not a background entity.

2.2.8 Shadows and Reflections

Shadows are cast by objects. Some shadows are part of the background, however other shadows may be part of the moving foreground objects. It is necessary to model the algorithms as such so that the foreground shadows are detected correctly. Also this problem is also part of the time of day problem, since as the time of day changes so does the shadow distribution. Reflections are also part of a similar issue. There may be reflections in the background. For example, if a scene includes a lake, the sun may cast its reflection on the water, however this has to modelled as part of the background. Another instance is, for example, in a background model modelled after a room. The room may contain a mirror. If a foreground entity, such as a person, moves the background subtraction algorithm should detect the person and also any reflections in the mirror related to the person has to be detected. Hence reflections are also a challenging issue in background subtraction. Some reflections are background entities and some are foreground entities.

2.3 Background Subtraction Classifications

Background subtraction methods have a wide variety of techniques they use in order to implement them. For example the mixture of Gaussians is based on a statistical method of background subtraction whilst some forms of background subtraction use a form of robust learning. Background subtraction algorithms can be divided into two categories. The first category is known as the traditional methods of background subtraction. These methods are the classic implementation of the foreground detection problems. Likewise the more recent methods of background subtraction are known as the modern background subtraction methods. Some of these methods will be looked upon and a few will be implemented on MATLAB in order to determine realistic application

possibilities for the AR Drone.

2.3.1 Basic Background Modelling

Basic background modelling include techniques such as frame differencing and running average methods for background subtraction. These are the most simple kind of implementations available for background subtraction and hence are computationally efficient. Some these methods are described in the following subsections.

2.3.2 Frame Differencing

In the frame differencing approach, the background is estimated as the previous frame to current input frame. Hence the background subtraction equation becomes as shown in equation 2.1.

$$B(x, y, t) = I(x, y, t - 1) \tag{2.1}$$

Where $B(x, y, t)$ is the background at time t and $I(x, y, t-1)$ image at time $t-1$. Then the current and previous frames are subtracted from each other and the resulting matrix is compared with a threshold value in order to determine the foreground mask using equation 2.2.

$$|I(x, y, t) - I(x, y, t - 1)| > Th \tag{2.2}$$

where I is the current image frame, B is the background frame, Th is the threshold and t is time. If the value of $|I(x, y, t) - I(x, y, t - 1)|$ is greater than the threshold, then the pixel is classed as a foreground pixel, otherwise the pixel is classified as a background pixel. Depending on the structure, frame rate, threshold, conditions and speed of the object and environment this approach may or may not be successful. The frame differencing method was applied to a moving object scenario. Figure 2.1 shows the background, foreground, ground truth and actual result of method (foreground mask).

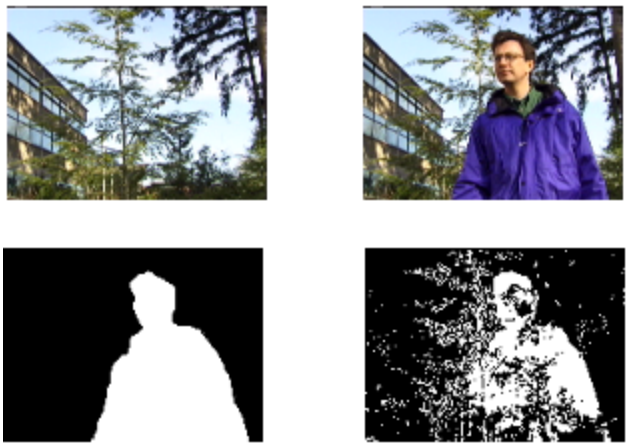


Figure 2.1: Background model, current frame, ground truth and actual result

2.3.3 Running Average

In the running average case of background subtraction, the difference between the frame differencing method and this methods exists in the way the background is modelled in order to perform the subtraction. In the running average method the background is modelled as shown in equation 2.3.

$$B(x, y, t) = (1 - \alpha)B(x, y, t - 1) + \alpha I(x, y, t) \tag{2.3}$$

Where α is known as the learning rate. The value of α can be varied between 0 and 1 and then the background model is accordingly altered by using the above formula. The value of the background as seen from the equation depends on the value of α. Once the background has been determined, the background model and current frame can be subtracted in order to obtain the difference result as in the frame differencing method. Similarly once this has been obtained, equation 2.4 can be used to compare the image matrix obtained with the threshold value defined.

$$|I(x, y, t) - I(x, y, t - 1)| > Th \tag{2.4}$$

If the result of the corresponding pixel is greater than the threshold, then the result is classified as a foreground pixel, otherwise the result is classified as a background pixel.

2.3.4 Approximate Median

In the approximate median approach, the background is estimated using the following criterion:

$$\text{if } I(x,y,t-1) > B(x,y,t-1) B(x,y,t) = B(x,y,t-1) + 1 \tag{2.5}$$

$$\text{if } I(x,y,t-1) < B(x,y,t-1) B(x,y,t) = B(x,y,t-1) - 1 \tag{2.6}$$

Using this method the background for each frame eventually converges to an estimate where half the input pixels have a value greater than that of the background and half the input pixels have the value less than the background pixels, hence approximately the median. The approximate median method was applied to a moving object scenario. Figure 2.2 shows the background, foreground, ground truth and actual result of method (foreground mask).

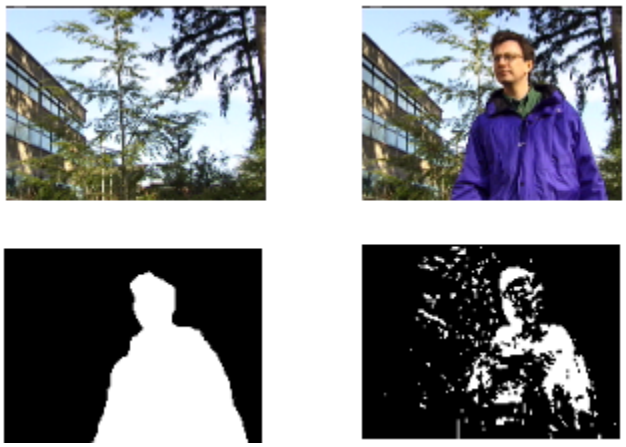


Figure 2.2: Background model, current frame, ground truth and actual result

2.3.5 Single Gaussian

Modelling the background as seen in the frame differencing method or approximate median method required a fixed low noise background model for best results. This requirement cannot always be satisfied in real-life. Hence, some authors have modelled the background using statistics. The way this is done in the single Gaussian case is by modelling each pixel of the background according to probability density function (PDF) which is learned over a series of video frames. Using this the traditional definition foreground changes from that of a purely texture and colour based categorization to a more probabilistic approach where the foreground is defined by comparing the pixels in the frame with the corresponding background pixel probability density function, and if

the probability of the existing pixel being part of the corresponding background is low, it would be categorized as the foreground. The single Gaussian method does utilize texture models in order to define its PDFs.

In the single Gaussian method, normally a texture method is used in order to determine the background. The points on the texture surface are associated with a mean colour value along with a distribution for that value about the mean. In the case of single Gaussian the colour is expressed in the YUV space. YUV works by separating the colour in an image into three distinct parts. Firstly, the Y is known as the greyscale version of the image. Then the colour information is separated into two channels. The information in these channels depends on the brightness. One channel is the blue information minus the brightness and one channel is the red information minus the brightness. These three channels combined will form the original image. Now for each pixel the colour distribution is modelled using the Gaussian described by a covariance matrix. The single Gaussian makes use of an adaptive filter as well due to which compensations can be made for common background subtraction issues such as change in illumination [15].

Wren *et al* [16] illustrated the usage of the single Gaussian. Every background pixel was modelled with a Gaussian distribution $N(\mu, \Sigma)$ where μ is known as the average background colour and Σ is the covariance matrix for a pixel p and time t. The covariance matrix Σ is by default a 3×3 matrix. It may be kept as a diagonal matrix in order to reduce cost and computational power. The mean and covariance can be updated iteratively using equations 2.7 and 2.8 [17].

$$\mu_{s,t+1} = (1 - \alpha)\mu_{s,t} + \alpha I_{s,t} \tag{2.7}$$

$$\Sigma_{s,t+1} = (1 - \alpha)\Sigma_{s,t} + \alpha(I_{s,t} - \mu_{s,t})(I_{s,t} - \mu_{s,t})^T \tag{2.8}$$

2.3.6 Gaussian Mixture Model (GMM)

Sometimes the backgrounds in the scene are complicated further due to animated backgrounds. For example at the beach, the background would contain the sea which has waves. Unlike oscillations from the tree the waves will require much stricter modelling of the background in order to compensate for it. Due to this BS problem, some authors suggested the use of multimodal PDFs.

Stauffer and Grimson method [16] uses the concept of multimodal PDFs to model the background. In their paper, they did this by modelling every pixel in the background model with a mixutre of K Gaussians. The probability of a certain colour to occur at a certain pixel is given as shown in equation 2.9.

$$P(I_{s,t}) = \sum_{i=1}^{K} \omega_{i,s,t} N(\mu_{i,s,t}, \Sigma_{i,s,t}) \tag{2.9}$$

where in the equation $\mathcal{N}(\mu, \Sigma)$ is the ith Gaussian distribution model and ω is the weight. In order to reduce the required memory and computational power, it was suggested similarly to the single Gaussian method for the covariance matrix to be a diagonal matrix. The difference between the single Gaussian and the GMM is the way the covariance is made diagonal. In single Gaussian the covariance matrix by definition is a 3×3 matrix, however in the GMM model, the covariance matrix depends upon the standard deviation σ, given by $\Sigma = I\sigma^2 d$. The GMM model is updated iteratively as shown in equations 2.10, 2.11 and 2.12.

$$\omega_{i,s,t} = (1 - \alpha)\omega_{i,s,t-1} + \alpha \tag{2.10}$$

$$\mu_{i,s,t} = (1 - \rho)\mu_{i,s,t-1} + \rho I_{s,t} \tag{2.11}$$

$$\sigma^2_{i,s,t} = (1 - \rho)\sigma^2_{i,s,t-1} + \rho d_2(I_{s,t}.\mu_{i,s,t}) \tag{2.12}$$

In the equations, α is the learning rate and ρ is a second learning rate that is determined by $\alpha\mathcal{N}(\mu_{i,s,t}, \Sigma_{i,s,t})$. In the mixture of Gaussians method, the thresholding is done by utilizing a number of distance functions. These are defined in equations 2.13, 2.14 and 2.15.

$$d_0 = |I_{s,t} - B_{s,t}| \tag{2.13}$$

$$d_1 = |I^R_{s,t} - B^R_{s,t}| + |I^G_{s,t} - B^G_{s,t}| + |I^B_{s,t} - B^B_{s,t}| \tag{2.14}$$

$$d_2 = (I^R_{s,t} - B^R_{s,t})^2 + (I^G_{s,t} - B^G_{s,t})^2 + (I^B_{s,t} - B^B_{s,t})^2 \tag{2.15}$$

Where, I is the current frame, B is the background model and B,G,R represent blue, green and red colour channels in an image. d_2 in equation 2.12 is obtained from equation 2.15.

2.3.7 Kernel Density Estimation

Kernel Density Estimation is another method that can be used for multimodal PDF modelling. Elgammal *et al* proposed [18] a function called a Parzen window estimate for each pixel of the background. The function is as given as follows:

$$P(I_{s,t}) = \frac{1}{N} \sum_{t=t-N}^{t-1} K(I_{s,t} - I_{s,t})$$

where K is defined as the Kernel which is normally a Gaussian and N is the number of previous frames that were used to estimate P(.) (PDF). In the case of the Kernel Density Estimation foreground detection will be done by comparing the pixel with the probability distribution, and if the corresponding value deems the current value to be unlikely of the distribution it would labelled as a foreground pixel (i.e. when the value of the probability is smaller than the set threshold). More advanced Kernel Density Estimation methods have be developed as seen in Mital *et al* [19].

2.3.8 Background Subtraction using Codebooks

Kim *et al* [20] proposed another method is order to deal with multimodal background modelling called background subtraction using codebooks. The way the codebook method works is by assigning each of the background model colour values for each pixel a series of colours which is stored on a codebook. The series of colours that are stored in the codebook are also known as codewords. The codeword may contain a single colour value information for example for a region in the background which stays the same, or, the codeword may contain several colour informations in the case of for example a tree oscillating or a wave motion of the sea.

From the paper [20] it was seen that this method performed well in removing false positives as well as coping well with illumination changes due to using a mathematical formula in order to do separate evaluations of colour distortions as well as brightness distortions. However, the number of false negatives being detected increased due to the colour based nature of the method. For example, grey cars in the background were incorrectly made and detected as shadows and hence resulted in a false negative.

2.3.9 Background Subtraction with modelling via Subspace learning

Many forms of subspace learning background modelling exist. The first case at subspace learning is based around Principal Component Analysis (PCA) and was first discussed by Oliver *et al* [21]. The way that the background model is constructed in SL-PCA (subspace learning using principal component analysis) is by applying PCA on a N number of images. The background model is represented by the mean obtained via PCA and the projection matrix consisting of the first p significant eigenvectors of PCA [21]. Hence in the case of SL-PCA the foreground segmentation is done by subtracting the current frame from the reconstructed PCA background model. Other forms of subspace learning exist as well. Many improvements to the SL-PCA model were implemented by Yamazaki *et al* and Tsai *et al* [22, 23] in which independent component analysis subspace learning (SL-ICA) was used in order to perform background subtraction tasks. There are two main kinds of subspace learning. Reconstructive subspace learning and Discriminative subspace learning. SL-PCA and SL-ICA are both types of reconstructive subspace learning. Reconstructive subspace learning methods are good because of good data approximation and provide incremental updating of data hence it is possible to use these methods in real-time such as that for the AR Drone. The second form of subspace learning known as discriminative subspace learning that consists of methods such as linear discriminant analysis (LDA) and canonical correlation analysis (CCA). Subspace learning using linear discriminant analysis (SL-LDA) was first proposed by Tang *et al* [24] and works by projecting the data into a lower dimensional vector space all while trying to achieve the maximal ratio between the within-class distance and the between-class distance. The within-class distance and between-class distance are represented by matrices [25]. The goal of SL-LDA is to achieve maximum amount of discrimination for best results. Canonical correlation analysis is a statistical method which studies the relationship between multiple dependant variables and independent variables. The way that subspace learning using canonical correlation analysis (SL-CCA) utilizes this statistical model is by using prediction algorithms in order to separate the dependant variables from the independent variables.

2.3.10 Fuzzy Background Modelling for Background Subtraction

As seen in the single Gaussian and mixture of Gaussian methods, the main challenge was to create a background subtraction algorithm that could model multimodal backgrounds that are dynamic. Single Gaussian was good at adapting for changes in illumination due to the statistical methods used

whereas the mixture of Gaussians method was introduced in order to improve the results for the single Gaussian in order to facilitate other background subtraction challenges such as waving trees and waves since the single Gaussian method was not able to perform as well for these challenges.

Similarly fuzzy background modelling is a method used for multimodal background modelling for background subtraction. Fuzzy background modelling utilizes the Gaussian Mixture Model. There are three aspects of background subtraction that may be modelled using Fuzzy methods. Fuzzy background modelling, fuzzy background maintenance and fuzzy foreground detection. Fuzzy background modelling is based around improving the multimodal background model. The mixture of Gaussians suffer problems from dynamic changes such as movement of the background. Also the Gaussian mixture model is started utilizing a training sequence which may either be insufficient or noisy and hence the background may not be modelled correctly. In order to improve the initial background modelling in the GMM case a Type-2 Fuzzy Gaussian Mixture Model was proposed by Bouwmans *et al* [26].

2.4 Comparison of the Various Algorithms

Table 2.1 shows the comparison of the various background subtraction methods used in the papers cited previously along with the key differences that are in the methods by which the actual background modelling is done.

Method	**Description**
Frame Differencing	Model is a greyscale image without moving objects and static
Running Average	Model is a greyscale image initially and then corresponding frames are used along with a learning rate in order to update the background
Approximate Median	Model is a greyscale image and the corresponding frames are used to update the background using a threshold comparison method
Single Gaussian	Each pixel in the background is modelled as a Gaussian distribution
Gaussian Mixture Model	Each pixel in the background is modelled as a mixture of Gaussians
Kernel Density Estimation	A non-parametric distribution is used to model each background pixel
Codebook	The background model is formed by colour coding each pixel as a series of colours and storing them in codebooks

Table 2.1: Comparison of the various background subtraction algorithms discussed

2.5 Methods to Evaluate Background Subtraction

It has been determined that the overall goal to perform BS is to reduce the number of false negatives and positives in order to get the best possible result in detecting objects that are in motion for computer vision applications. There are many ways to evaluate the results from background subtraction. Some researches have used descriptive methods in order to evaluate the results and performance of the various algorithms [27]. The problem with evaluating an algorithm using this method is the fact that it relies on judging visually if the result of the algorithm is better over the other. Whilst to the naked eye one result might seem superior over the other, it might not actually be the case. It is necessary therefore to create quantitative methods in order to evaluate the various BS algorithms.

Some examples of the use of quantitative methods to evaluate background subtraction algorithms were used by Panahi *et al* [28] and Toyama *et al* [29]. In both papers the background subtraction methods are first tested on various video sequences that had various different challenges (e.g. illumination changes, shadows) with a single set of parameters to compare the results for each video. The comparison was based around the number of false negatives and false positives that were produced in each of the video sequences after background background subtraction. However, even quantitative analysis is not prone to deficiencies. One problem with comparing background subtraction methods via the number of false negatives or positives is that if the number of false positives increase, the number of false negatives decrease and vice-versa. Thus due to this fact it cannot be said that for example if there is a method with high number of false positives and low number of false negatives would be better than say a method with low number of false positives and high number of false negatives. This is one limitation for using false negative/positive count to evaluate background subtraction algorithms. Another limitation is that in both the works cited, there is a fixed threshold that is being used to evaluate the algorithms. It is possible that a different value for threshold would improve the results from the background subtraction, as will be discussed in this thesis, hence using the number of false negatives and positives on its own in not enough to prove if one algorithm is superior to another or vice-versa. Another evaluation method used is known as a precision and recall method to evaluate background subtraction algorithms. This method was first used by Herero *et al* [27] and uses certain formula for both recall and precision in order to evaluate the performance of the algorithms. This method also has the same limitations as

seen with the false negative/positive count method in that a set threshold is being used to compare the different algorithms.

Other approaches to evaluate the various algorithms involves the use of blobs. This kind of approach was normally seen to be used in research that involved pedestrian modelling and detection or car modelling and detection [30, 31]. These algorithms focus on detecting certain objects in motion and grouping these foreground objects (their pixels) as moving together in blobs. The most glaring drawback in using this method is the inability for these algorithms to deal with occlusions. Since if two cars, lets say, are detected they would be both categorized as blobs. If the two cars are now in close proximity to each other they may be categorized as one blob during background subtraction. Normally various post-processing is required in order to separate the two objects at the end of the background subtraction. The actual problem with this merging of blobs is that is causes the number of false positives and negatives to increase. Hence, the algorithms may be incorrectly evaluated. The post-processing methods that are used may not be the same for all algorithms further reducing the quality of the evaluation.

O Barnich *et al* [32] in their paper used another kind of calculation that involved the use of false negatives and positives as well as true negatives and positives in order to evaluate their ViBE algorithm. A percentage of correct classification (PCC) value was calculated for the various values of threshold in order to evaluate the algorithm. The PCC was calculated using the formula below: In this thesis a combination of precision-recall and descriptive methods will be used to describe the various background subtraction algorithms and the results will be compared for different values of thresholds in order reduce the limitations due to quantitative evaluation.

Background Subtraction Methods Evaluation

Visually proving one method of background subtraction over the other is not a sufficient method for determining the effectiveness of the particular algorithm. It is necessary to define a mathematical method to define the performance evaluation of the results. Precision-Recall charts can be made for the algorithms. Precision-Recall calculations for the algorithms will be done on a pixel level. The foreground that is detected is segmented as a foreground mask and the matrix elements consisting of the foreground hold the value of 1 (white) and the background holds the value of 0 (black). An image is represented as a matrix. Hence it is possible to use the data from the matrix elements (1's and 0's) in order to perform pixel level calculations. Three various measures will be performed, the precision, recall and the f-measure.

The recall of the resulting background subtracted image can be found using equation 3.1.

$$\text{Recall} = \frac{\text{Number of correctly classified foreground pixels}}{\text{Foreground pixels in ground truth}} \tag{3.1}$$

The results are firstly compared with a ground truth. This is the defined as the ideal background subtracted result and is what is trying to be achieved via all the algorithms. The ultimate purpose is to have recall as close as possible to zero in order for the best results ideally. The number of correctly classified foreground pixels can be classified using image comparison methods to see which elements of the background subtracted result match with the ground truth elements. Once a comparison has been done then the total number of matching pixels can be counted. This pixel number is defined as the correctly classified foreground pixels.

The precision of the background subtraction algorithm can be defined as shown in equation 3.2.

$$\text{Precision} = \frac{\text{Correctly classified foreground pixels}}{\text{Pixels classified as foreground}} \tag{3.2}$$

The correctly classified foreground pixels are the same as for recall. However, the number of pixels classified as the foreground is actually the total number of pixels holding a '1' value in the background subtracted image. Hence it can be determined that the ideal value for precision is 1, since it is required that the actual number of correctly classified foreground pixels is the same of the pixels classified as the foreground. The total number of pixels classified as the foreground can be calculated by representing the background subtracted image as a matrix and evaluating the total number of non zero elements which is the same as the number of foreground pixels.

The f-measure (maximal average) of the algorithm can be calculated using the equation:

$$\text{F-measure} = 2\frac{(\text{Recall . Precision})}{(\text{Recall + Precision})} \tag{3.3}$$

This is another method for evaluating the efficiency of the algorithm.

The precision-recall charts will be produced in two forms. One form will show the precision-recall for varying threshold and another method will calculate the consistency of the precision-recall values over a certain number of frames in order to determine consistency. Various challenging circumstances such as illumination changes and moving objects will be researched.

3.1 Precision and Recall Code for MATLAB

In order to calculate the recall and precision for the background subtraction methods, it is first required to have the corresponding items:

1. Foreground mask after background subtraction
2. Ground truth for the corresponding frame

Due to the need of having a corresponding ground truth, the number of data sets that can be used to analyse the background subtraction algorithms became limited. The Wallflower [33] data set and the Stuttgart Artificial Background Subtraction (SABS) [34] data set were used. The benefit of the Wallflower data set is the lower resolution, so the algorithm will run faster. However lower resolution may hamper results. Also the ground truth for the case of the Wallflower data set is only available for a single frame. The SABS data set however has the ground truth for multiple frames and has a higher resolution. The disadvantage being that, firstly the resolution is very high and that

would result in different performance measures for the algorithms and also the data set is artificial and hence mostly ideal i.e. the data does not suffer from real life problems such as noise. Therefore when analysing the different background subtraction methods, in order to correctly compare one algorithm with the other it is necessary to use the same data set.

From the equations given in the equations in the previous section, it can be determined that the following data is required in order to calculate the precision and recall:

1. Number of correctly classified foreground pixels
2. Number of pixels classified as foreground
3. Number of foreground pixels in the ground truth

The number of correctly classified foreground pixels can be determined by first converting both the ground truth and the foreground mask to the same image format. Once done, it is known that a value of 0 represents a black pixel and a value of greater than zero (i.e. 255) represents a white (foreground) pixel. Hence a MATLAB code to compare the two image matrices (one matrix being for the ground truth image and one for the foreground mask) for values above 0. Once MATLAB is done comparing, it will create a new matrix where only values of the foreground which matches with those of the ground truth would have a value of 1 and the remainder of the values will be given a value of 0. Hence only the correctly classified foreground pixels will hold a value of 1. Once this matrix has been determined, it is possible to use the function nnz in MATLAB to count the number of non-zero elements in a matrix, in order words for this specific case the number of correctly classified foreground pixels.

The number of pixels classified as the foreground can be calculated by taking the image matrix of the foreground mask and simply using the MATLAB function nnz in order to determine the number of non-zero elements in the foreground which in turn is the total number of foreground classified pixels.

The number of foreground pixels in the ground truth can be acquired in a similar fashion. By taking the ground truth image matrix and determining the number of non-zero elements in the image it is possible to know the total number of foreground pixels in the ground truth.

Appendix 4 shows the code used to perform these calculations.

3.2 Frame Differencing Algorithm Evaluation

The threshold value that is used to run the algorithm was varied from 0 to 200 in steps of 10. The results for the recall, precision and the f-measure are given Table 3.1.

Figure 3.1 shows the results of the background subtraction, where each image is showing an increment of 10 for the value of threshold, starting from 0 and going till 200. The results in the form of threshold-recall, threshold-precision and precision-recall graphs are shown in figures 3.2, 3.3 and 3.4.

Threshold	**Recall**	**Precision**	**F-measure**
0	0.9937	0.325	1.3002
10	0.9189	0.5018	2.0071
20	0.8437	0.5793	2.3171
30	0.7682	0.6399	2.5598
40	0.6924	0.6839	2.7356
50	0.6285	0.7268	2.9074
60	0.5724	0.7624	3.0496
70	0.5261	0.796	3.1839
80	0.4848	0.8273	3.3093
90	0.4491	0.8561	3.4243
100	0.418	0.8854	3.5418
110	0.3868	0.9035	3.614
120	0.3618	0.9276	3.7105
130	0.3321	0.9421	3.7685
140	0.2979	0.9552	3.8212
150	0.2545	0.9639	3.8558
160	0.2173	0.9726	3.8904
170	0.1838	0.9774	3.9096
180	0.1529	0.9793	3.9172
190	0.1124	0.988	3.9522
200	0.0774	0.9934	3.9738

Table 3.1: Precision, Recall and F-Measure results for various thresholds for the frame differencing case

Figure 3.1: Result for frame differencing with the threshold varied from 0-200

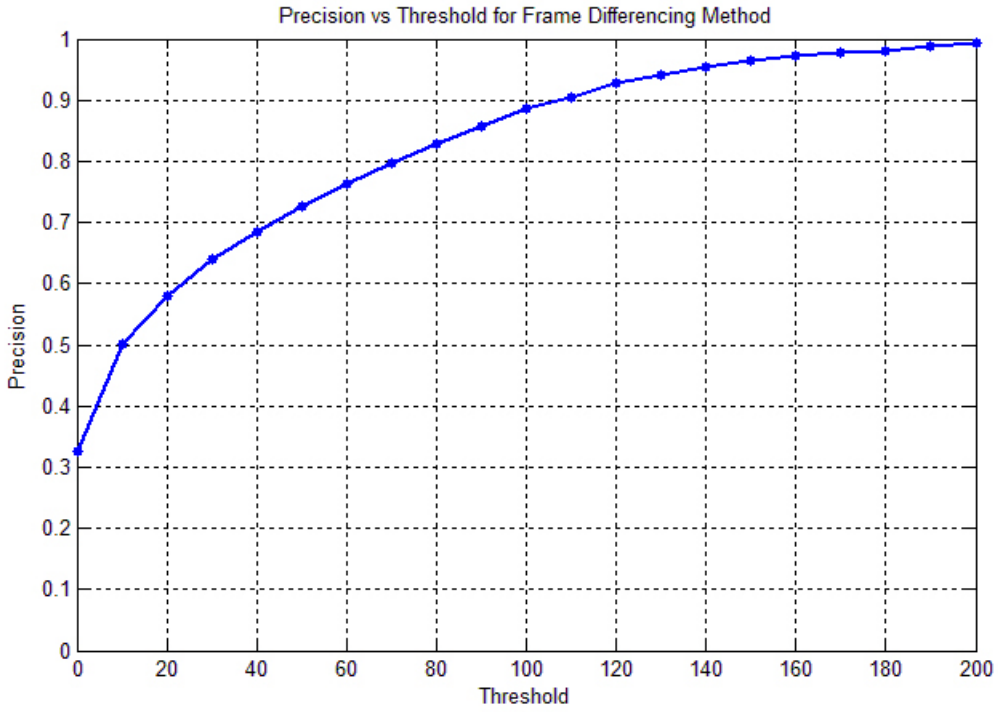


Figure 3.2: Frame Differencing Precision vs Threshold

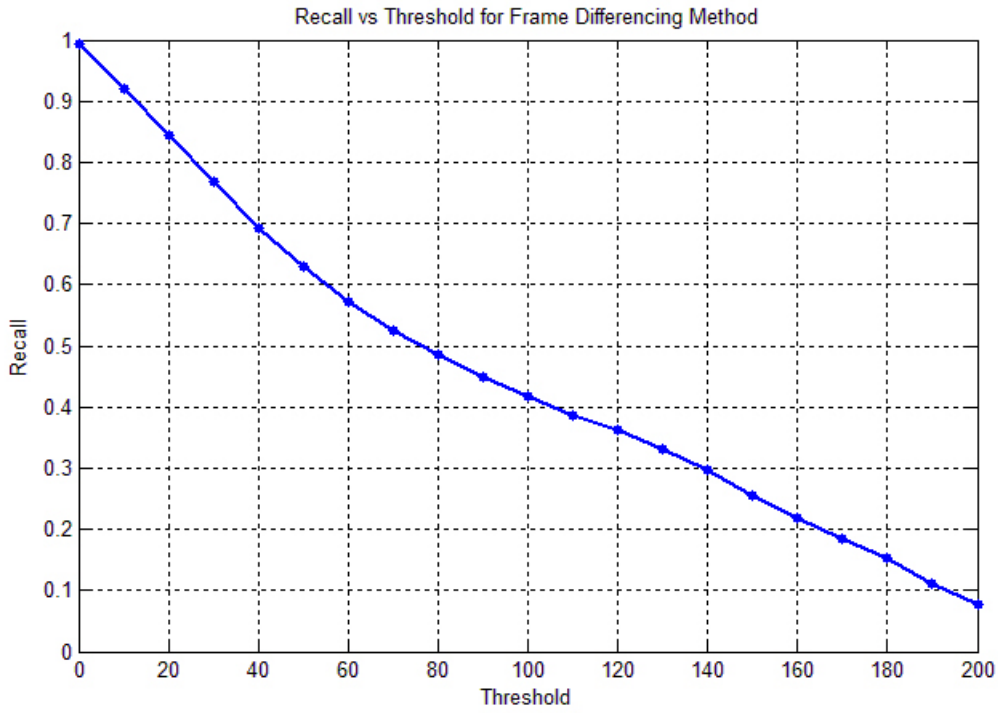


Figure 3.3: Frame Differencing Recall vs Threshold

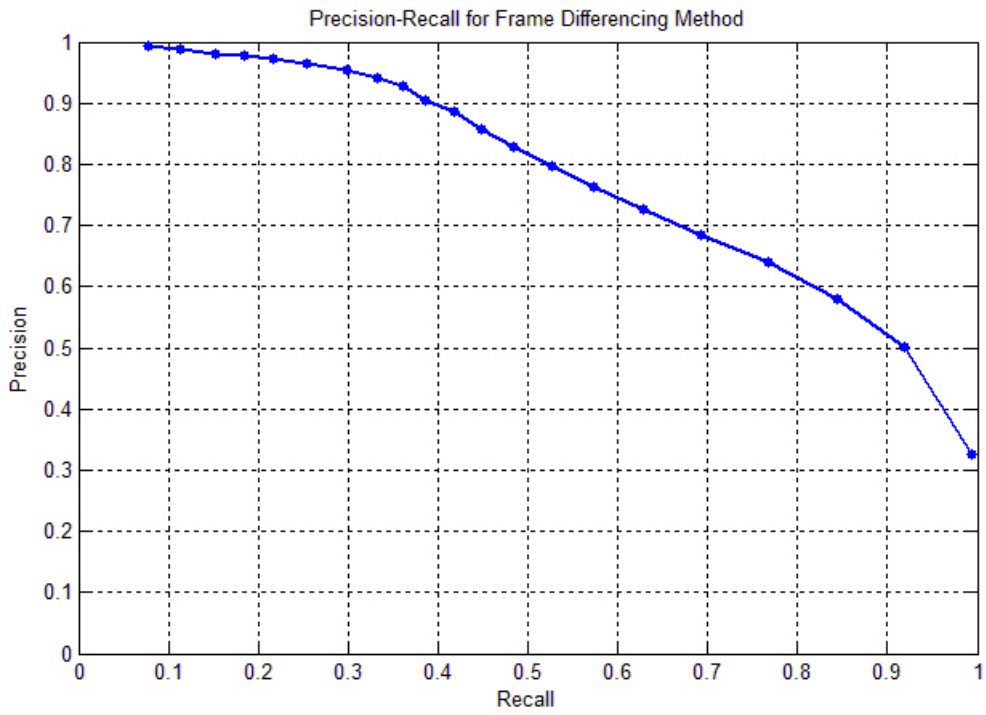


Figure 3.4: Frame Differencing Precision-Recall

3.3 Approximate Median Algorithm Evaluation

The value of threshold for the algorithm was varied from 0 to 100 in increments of 10. The results for the precision, recall and f-measure are given in Table 3.2.

The results for the foreground masks for each of the background subtracted images are shown in figure 3.5. The threshold-precision, threshold-recall and precision-recall graphs are given in figures 3.6, 3.7 and 3.8.

Threshold	Recall	Precision	F-measure
0	0.8849	0.4912	1.9647
10	0.6887	0.6366	2.5464
20	0.5543	0.7008	2.8031
30	0.4615	0.7541	3.0164
40	0.3851	0.8023	3.2094
50	0.3333	0.8383	3.3533
60	0.2804	0.8593	3.4374
70	0.2391	0.8843	3.5371
80	0.1967	0.8948	3.5793
90	0.1661	0.9088	3.6353
100	0.1313	0.9104	3.6415

Table 3.2: Precision, Recall and F-Measure results for the various thresholds for the approximate median case

Figure 3.5: Result for approximate median with the threshold varied from 0-200

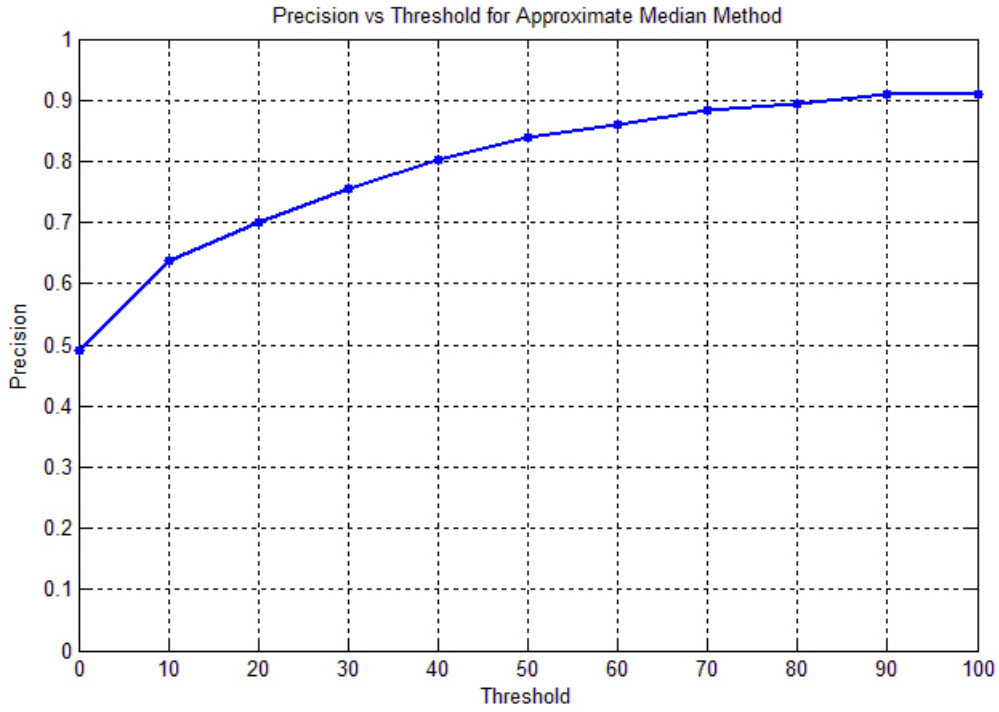


Figure 3.6: Approximate Median Precision vs Threshold

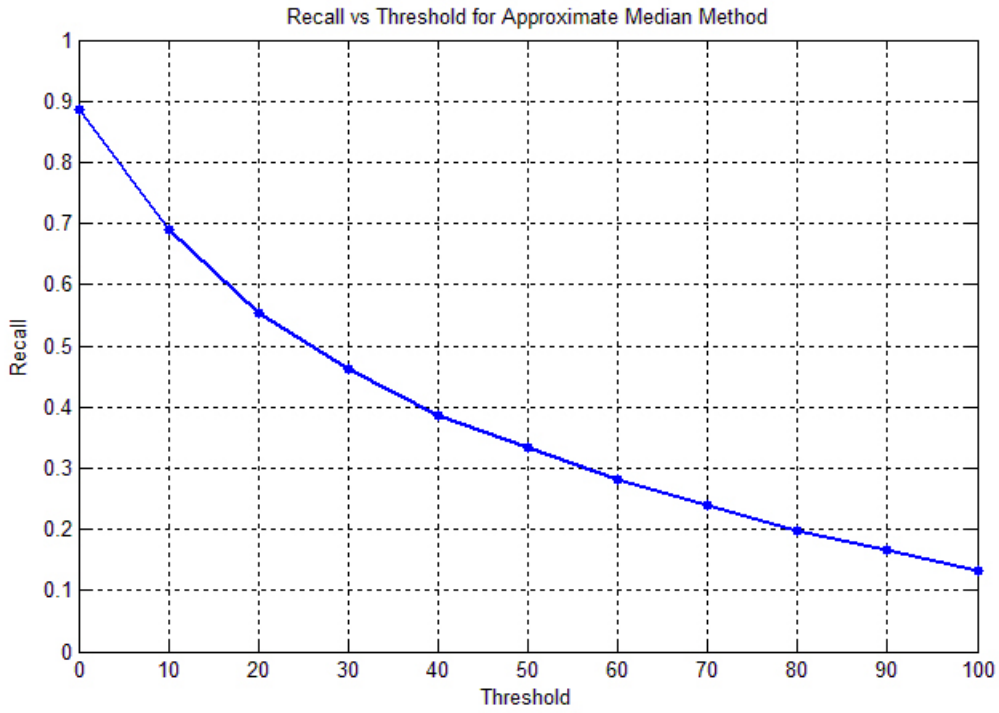


Figure 3.7: Approximate Median Recall vs Threshold

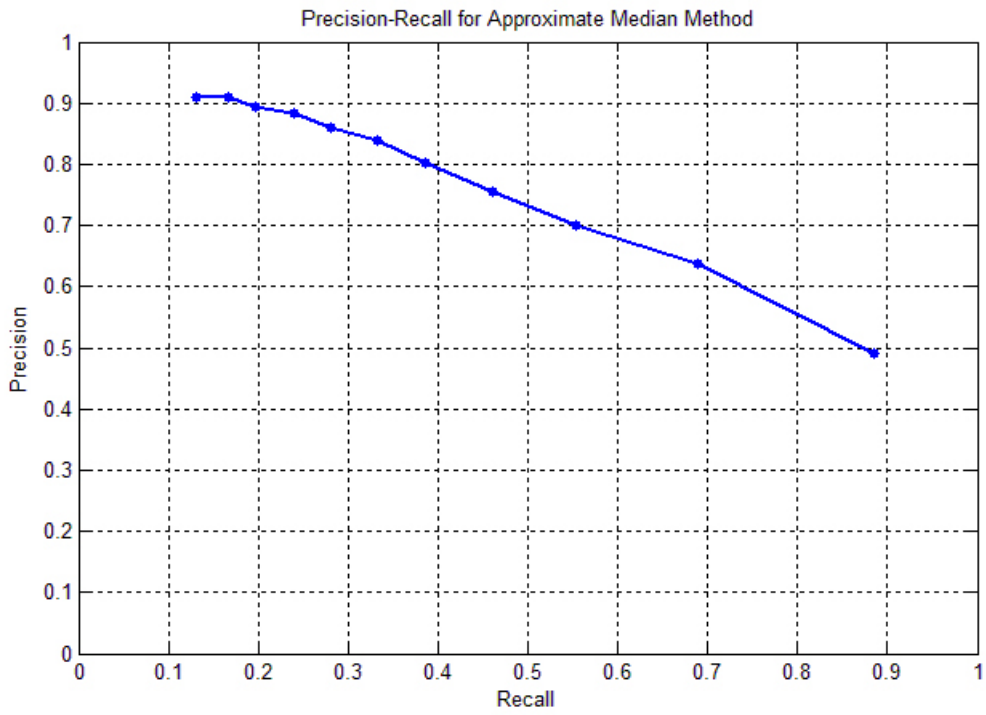


Figure 3.8: Approximate Median Precision-Recall

3.4 Evaluation of the Results

Figures 3.9, 3.10 and 3.11 show the comparison of the results between the approximate median case and the frame differencing case. From the results it can be seen that the approximate median method overall works better than the frame differencing method. This is due to the fact that the approximate median method reaches the highest possible value of precision and lowest possible value of recall with varying threshold quicker than that shown by the frame differencing method. However, the computational power required by the approximate median method is much higher as compared to frame differencing. Therefore, frame differencing is a possible algorithm that can be used in real-time with the drone.

From the results in figures 3.1 and 3.5 it can also be seen that the approximate median method does a better job is separating the whole object from the background and there are considerably fewer oscillations detected for the specific threshold values as compared to the frame differencing method.

For the approximate median method, it can be seen that for a threshold value ranging around 90

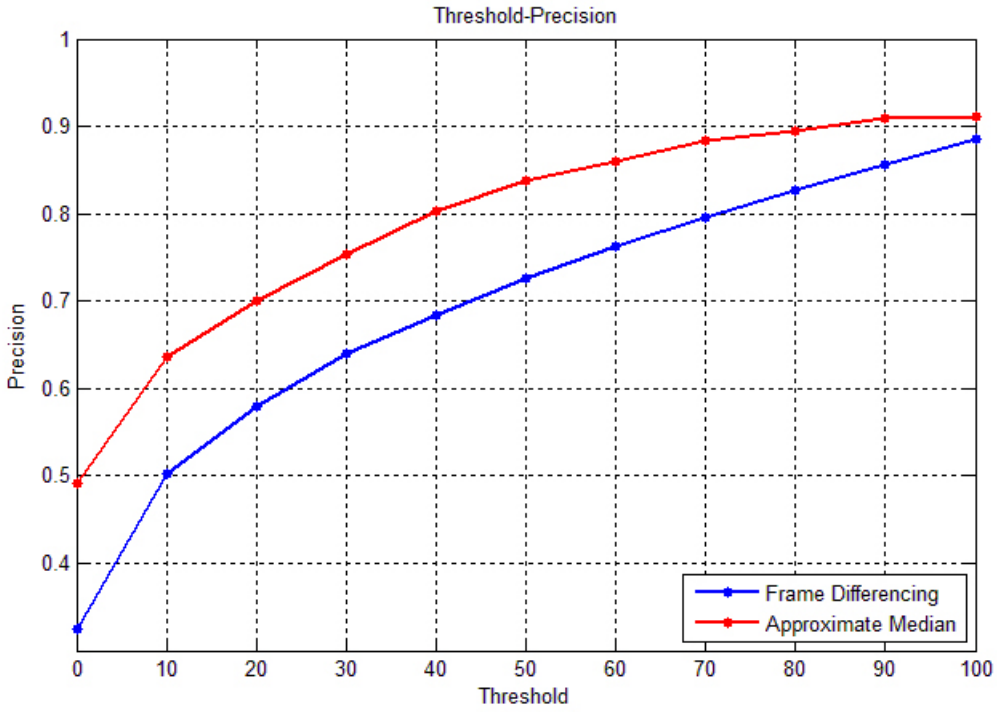


Figure 3.9: Precision: Frame Differencing vs Approximate Median

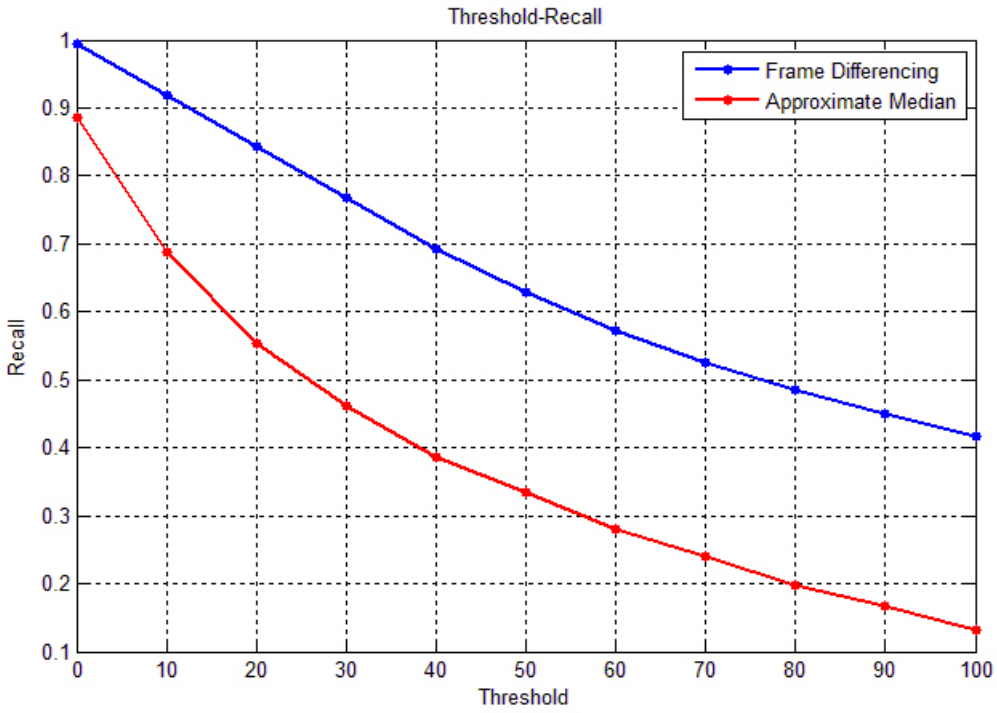


Figure 3.10: Recall: Frame Differencing vs Approximate Median

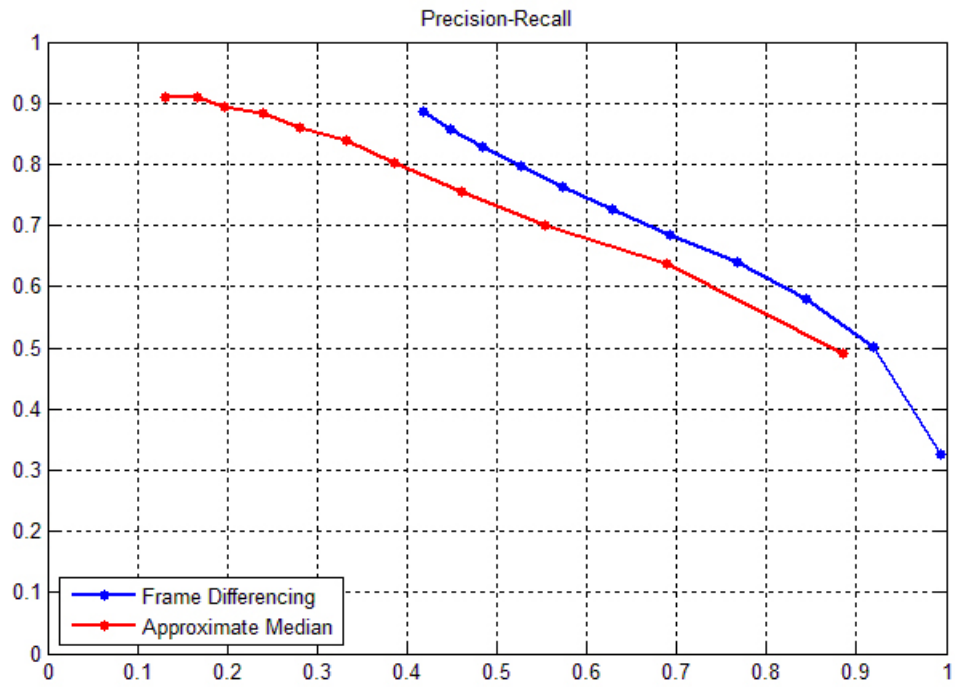


Figure 3.11: Precision-Recall: Frame Differencing vs Approximate Median

gives the best result in terms of the fewest number of false negatives and positives whilst detecting the person in the background. If the threshold is increased further than this, the result is that the number of oscillations detected reduce to near zero, however the amount of the person detected is also reduced since a higher number of detected pixels do not match up to the higher threshold value and hence the results become ambiguous. For the frame differencing method it can be seen that around a threshold level of 160 produces a reasonable result with a minimum number of oscillations and the person of interest also detected.

However, in both methods there is a lot of noise and false negatives and positives. Therefore a form of either pre-processing or post-processing is required in order to improve the overall results and help the reduction of false negatives and positives.

3.5 Pre-processing: Median filter implementation

Median filters can be incorporated into video processing algorithms in order to improve the overall quality of the results obtained. A median filter is used in order to reduce specific kinds of noise

that are normally seen in videos from cameras such as that for the AR Drone.

A median filter works for example by first creating a neighbourhood windows i.e. a 3×3 neighbourhood with the individual pixels sorted from smallest to largest. Then each entry is replaced by the median of the surrounding pixel entries. This is basically ordering the pixels in order from smallest to largest and then choosing the median (middle) value and applying it to the neighbourhood.

Median filters are a preferred way of pre-processing a video in order to remove noise since it allows removal of noise but preserving the edges to a very successful extent.

3.6 Evaluation of the Frame Differencing Algorithm with Median Filter Implementation

In theory the implementation of a median filter in pre-processing the background and foreground image would enable the removal of possible excessive noise. In the case of the oscillations problem it is the movement of the tree in the background. The movement of the leaves causes a number of false positives being part of the background subtracted image. By using the median filter it is possible in theory to reduce the number of detected false positives. An example of how the image is altered when it is median filtered is shown in figure 3.12, where the second image shows the filtered

Figure 3.12: Left: Non-median filtered and Right: Median-filtered

image with the noise removed. It can bee seen much of the noise has been removed while keeping the overall structure of the tree intact. This should allow better results being obtained from the background subtraction. Table 3.3 shows the results obtained using the frame differencing method with implementation of a median filter. The results of the background subtracted images is shown in figure 3.13, where each image shows the result with an increase of the threshold by 10 from 0 till 200.

By comparing the results of the precision, it can be seen that the median filtered frame differencing

Threshold	Recall	Precision	F-measure
0	0.9951	0.33	1.32
10	0.9262	0.5285	2.1139
20	0.8449	0.6333	2.5333
30	0.7662	0.7277	2.9109
40	0.688	0.7936	3.1746
50	0.6239	0.8448	3.3792
60	0.5667	0.8831	3.5326
70	0.5181	0.9109	3.6436
80	0.4788	0.9297	3.7187
90	0.4497	0.9468	3.7872
100	0.4066	0.9545	3.818
110	0.3692	0.9598	3.8391
120	0.3467	0.971	3.8838
130	0.3187	0.9766	3.9062
140	0.2848	0.9836	3.9342
150	0.2398	0.9874	3.9496

Table 3.3: Precision, Recall and F-Measure results for frame differencing with median filter implementation

results in achieving a precision of 90% by using a threshold of around 65 which is significantly lower than that of the non filtered simple frame differencing, which results in achieving a precision above 90% at around a threshold of 110. Hence the results of the median filtered background subtraction are overall better. Furthermore a significant amount of noise is reduced in each increment of threshold as compared to the non-filtered results.

Figure 3.13: Frame differencing with median filtering. Results are for threshold being varied from 0-200

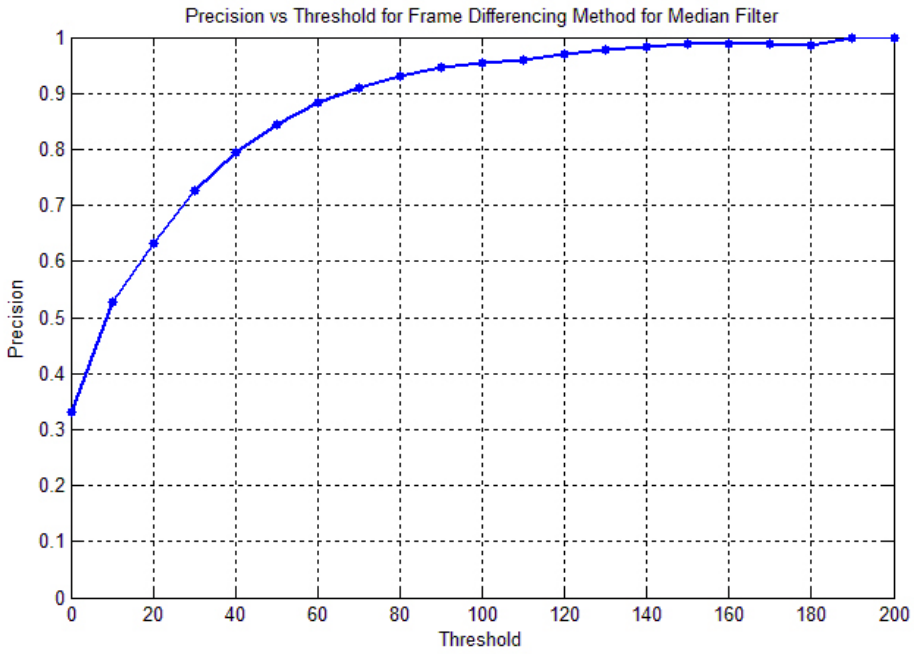


Figure 3.14: Frame Differencing with median filtering: Precision v Threshold

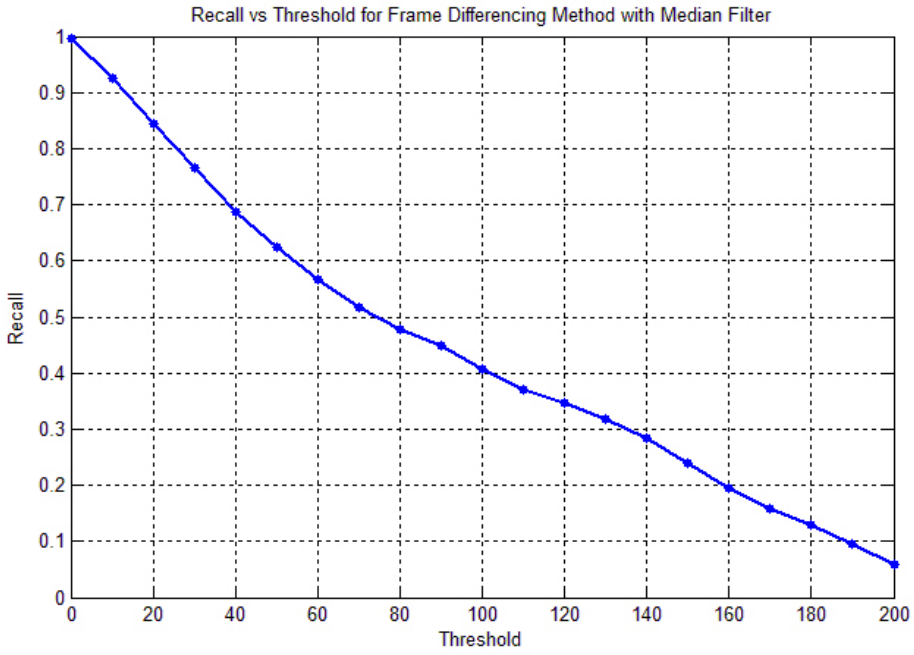


Figure 3.15: Frame Differencing with median filtering: Recall v Threshold

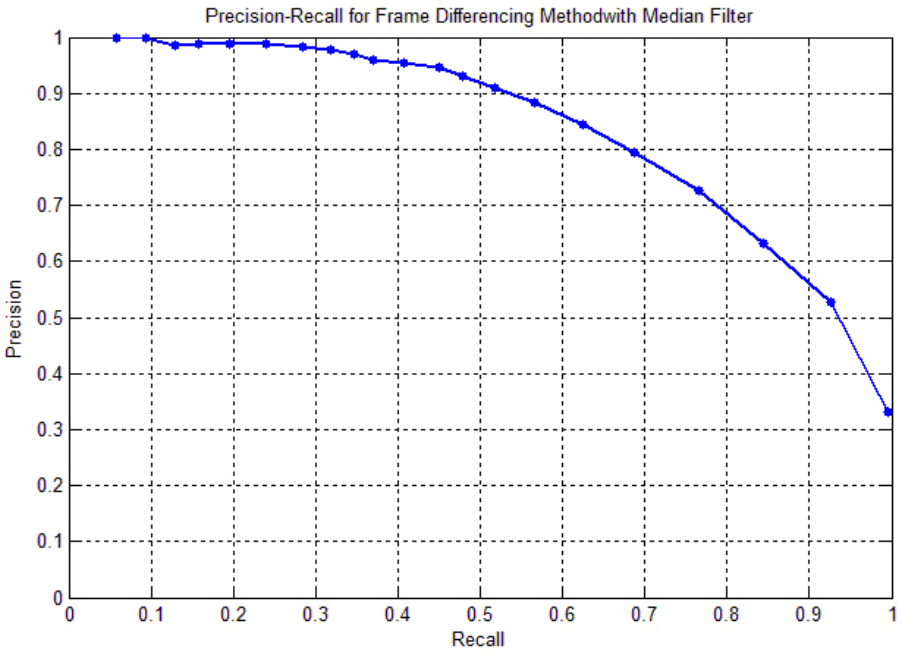


Figure 3.16: Frame Differencing with median filtering: Precision-Recall

3.7 Problems with current implementation

Although the frame differencing method and approximate median method of background subtraction does a decent job on video processing problems involving moving objects or oscillations, it does not perform correctly for more extreme problems in background subtraction. For example changes in illumination whether sudden or gradual and for moving camera case. This is due to the fact that these algorithms depend on a background model which is static. This is a bigger problem in the frame differencing case since the model is either a single image from which all the subtraction of the following frames are done or it is either the previous frame. The problem with having a single background model is that eventually illumination will change causing the change in illumination to be falsely detected as the foreground. In the case when the background model is updated frame by frame, the issue of unable to deal with changing illuminations is fixed to a good extent, however, the actual foreground objects are not fully detected due to the high refresh rate of the background model. It is necessary to design a background subtraction algorithm that will be able to adapt to the different challenging conditions as well as be able to detect the actual foreground objects.

Improved Background Subtraction Algorithms

The frame differencing method and approximate median method are traditional methods to perform background subtraction. The benefit of these methods are the fact the they have a modest computational load when being used and also are able to detect the moving objects. However, they have many different problems. For example, the frame differencing and background subtraction algorithms cannot cope with changes in illumination, whether rapid or gradual. This is due to the usage of a fixed static background in the frame differencing method as mentioned previously. In the case of the approximate median method, the algorithm is able to perform better as compared to the frame differencing method in changing illuminations, however, the algorithm still does not respond well to very quick changes in illumination.

Furthermore, both these algorithms cannot cope with shadows or reflections. The shadows of objects must be classified as background so that the moving objects are detected correctly. In this section further background subtraction methods will be discussed that would enable to fix some of these issues being faced.

4.1 Adaptive Background Subtraction

An adaptive background subtraction algorithm was implemented in order to address the problems associated with illumination changes and other background subtraction problems. The algorithm works by updating the background model for every frame using a background model formula. This is closely similar to the running average method of background modelling and is given as shown in equation 4.1.

$$B(x, y, t) = (1 - \alpha)I(x, y, t - 1) + \alpha B(x, y, t) \tag{4.1}$$

Where is the learning rate. The value of α is varied between 0 and 1. The learning rate would determine the rate at which the background is updated. The Wallflower data set for the time of day was used in order to test the algorithm. The results of the algorithm are as follows. It

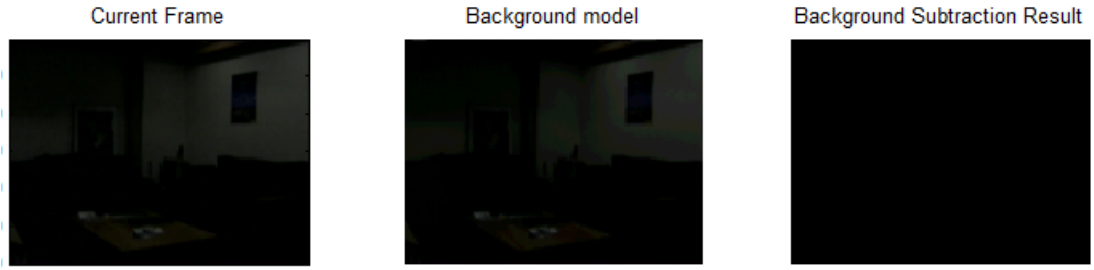


Figure 4.1: Adaptive Background subtraction at certain time t

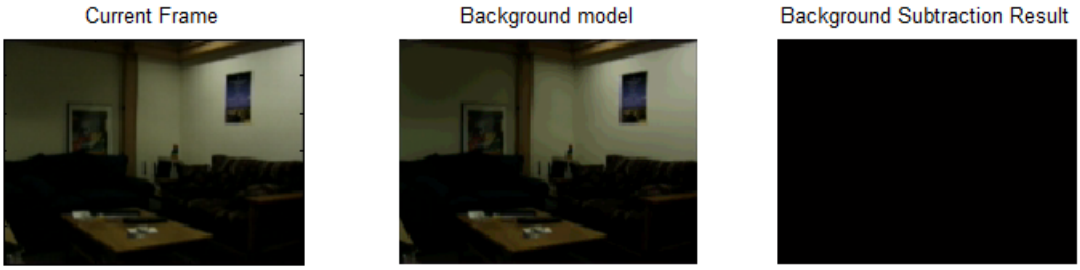


Figure 4.2: Adaptive Background subtraction at a later time $t + N$

can be seen from figures 4.1 and 4.2 that the background model is slightly more darker than the current frame due to the intensity of light increasing. Our purpose for this algorithm is to adapt the background so that the change in illumination is not detected as the foreground when the background subtraction actually takes place.

It can be seen that from the results of the background subtraction, no foreground has been detected. This means that the algorithm is successfully implementing the changes in the background model and hence the background subtraction does not produce any false positives as desired. However, this algorithm is not perfect even though the illumination changes are being accounted for. Due to the nature of the background modelling, the background is being changed every frame and a new background model is being formed. This causes incorrect foreground detection since the background refresh rate is relatively high and so for an actual object that is moving, for example a person across the room, the background subtraction may not detect the whole person as shown in figure 4.3. It can be seen that the person as a whole is detected, however, much of the persons

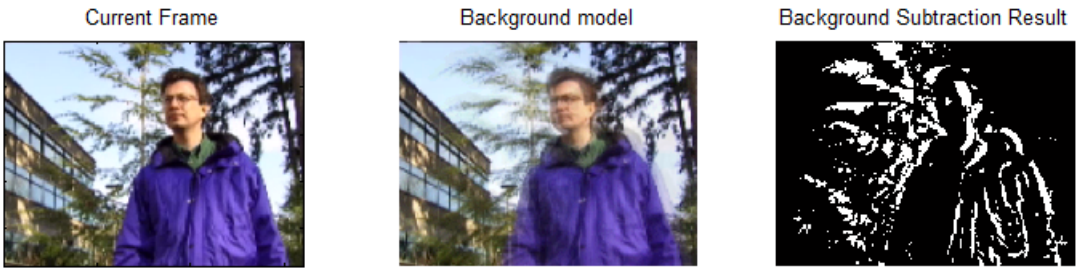


Figure 4.3: Adaptive Background subtraction: Wallflower data set waving trees challenge

clothing is modelled as part of the background incorrectly due to the high refresh rate of the background model. This refresh rate can be reduced by increasing the learning rate, however, the problem of updating the background model at a slower rate would cause and increase of false positives in illumination changes.

Hence one way to improve this result is to develop a different background modelling method so that the background model would be updated at a slower rate, but not too slow as so there would be incorrect results from changing illumination. The model would need to be designed different for different applications. For example, this current background updating model would work well in the moving camera case of the drone, since the background model would be updated at a good rate and hence the drone would be capable of handling changes in the background.

This background subtraction method also works well with very quick changes in illumination since the background model is updated at a fast rate.

In terms of the possibility of using this algorithm in real-time for applications in the AR Drone, it is possible to use this algorithm since it is very efficient and at its core uses the basic frame differencing method to produce the background subtracted result. The only difference between the frame differencing method and this adaptive background method is the fact that the background model is being updated in the adaptive background case, whereas in the frame differencing case the background is usually fixed.

4.2 ViBE Background Subtraction Algorithm

The Vibe background subtraction algorithm is based around the background model to be modelled over a series of frames in which a background is constructed from the ground up in real time. Initially there is no background model. The algorithm forms the model by comparing the current frame with previous frames, and then slowly over time the algorithm distinguishes the background from foreground and then background subtraction process is done. The Stuttgart Artificial background subtraction data set was used for testing this algorithm. The background and the current frame are shown in figures 4.4 and 4.5 respectively. The results for the background subtraction using

Figure 4.4: Background model used for initializing ViBE algorithm

the ViBE algorithm is shown in figures 4.6, 4.7 and 4.8. It can be seen that the moving object (i.e. the van) is separated from the foreground very clearly and the result is visually superior to the frame differencing and approximate median methods. However, as mentioned previously, the Vibe algorithm makes a background model from ground up. Figures 4.7 and 4.8 show the initial background subtraction results and the results for background subtraction after a few seconds respectively. It can be seen that initially when background subtraction was performed on the current frame, almost all the of frame was classified as the foreground. This is due to the fact that the background still has to be modelled. The second image shows that now some regions of the background have been modelled and so this process will continue until the entire background

Figure 4.5: Foreground frame for which foreground frame is being analyzed

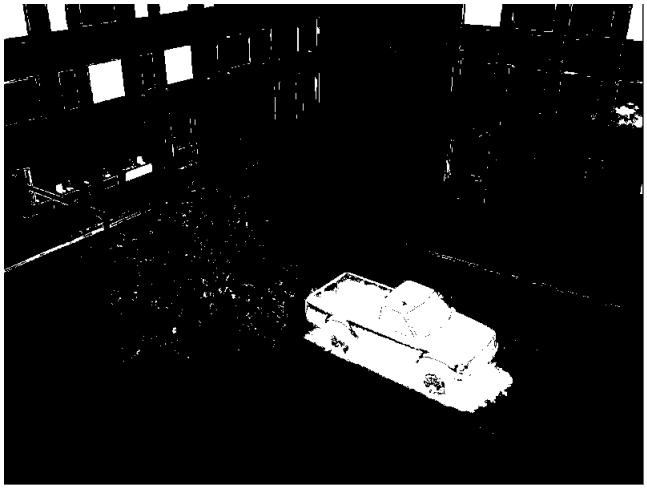


Figure 4.6: Result obtained from background subtraction using ViBE

is modelled. After the background modelling the Vibe algorithm can easily and accurately detect moving objects and foreground entities.

The benefit of the Vibe algorithm is that it is accurate and is able to correctly respond to changes in illumination. The background model is dynamic so therefore if there are any sudden changes in the background the background model can adapt. The only downside of this algorithm is that it requires a period of time before the background is modelled and hence cannot be used in moving

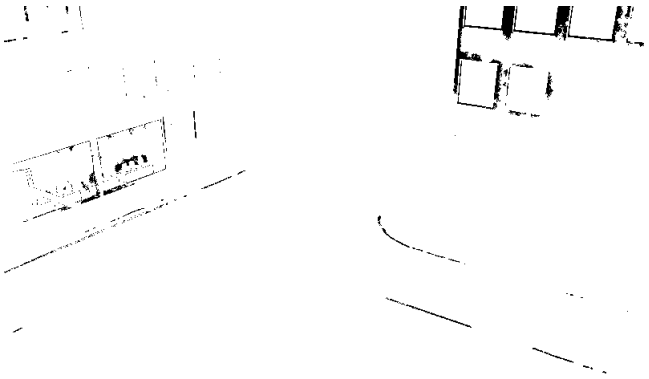


Figure 4.7: Result when algorithm initializes

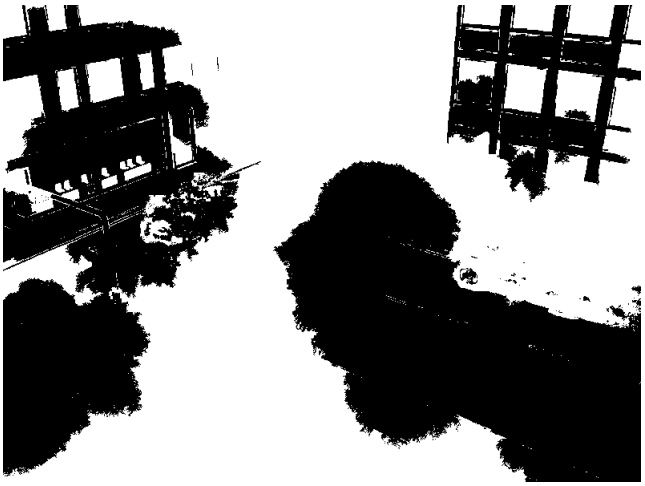


Figure 4.8: Result at a later time which shows the background being modelled so the results are improving

camera cases like the drone. However, this algorithm works in real-time and hence in theory can be applied on the drone.

This algorithm can be used for drone purposes in cases where the drone is performing some form of surveillance and is mostly hovering. Since the drone is hovering there will be time for the background to be modelled and hence it is possible to use the Vibe algorithm in drone surveillance.

4.3 Mixture of Gaussians (GMM)

A mixture of Gaussians background subtraction algorithm was created in C++ using OpenCV. The SABS dataset was tested as was the case for the Vibe algorithm. The result of the foreground mask is shown in figure 4.9. It can be seen by comparing figures 4.6 and 4.9 that compared to the

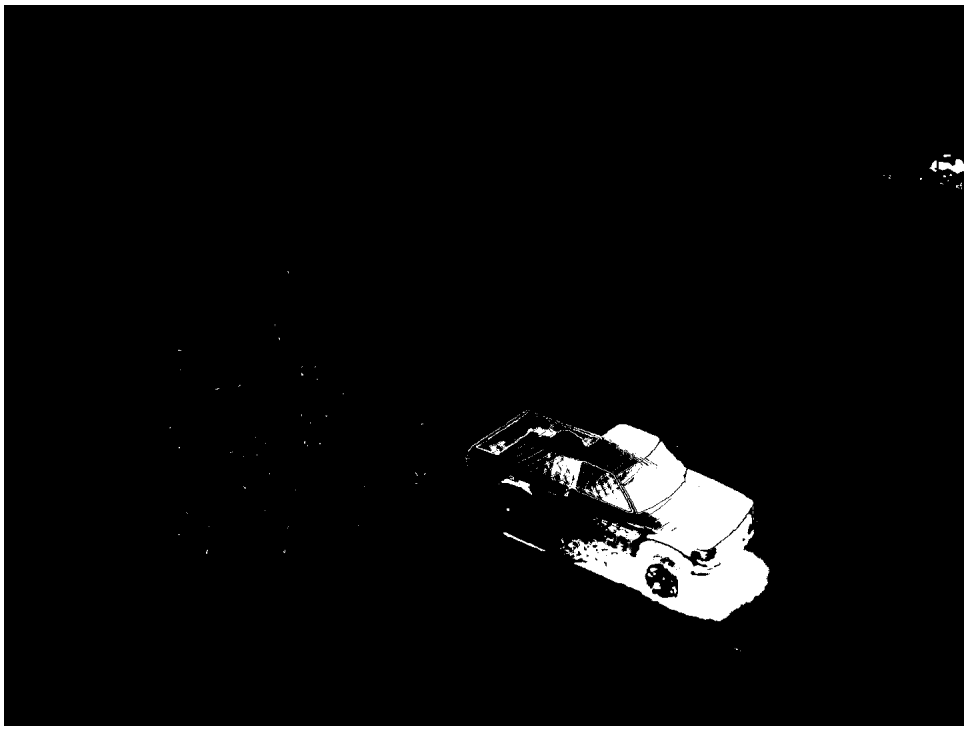


Figure 4.9: Mixture of Gaussians result for background subtraction

Vibe algorithm, the mixture of Gaussians results in a better removal of false positives. Much less of the background and noise is detected in the foreground mask. Also the mixture of Gaussians results in higher removal of oscillations (waving of the tree).

The Vibe algorithm however does a better job in detected the whole moving object whilst the mixture of Gaussians results in partial detection of the moving object and hence an increased number of false negatvies. The benefit of the mixture of Gaussians model is that the background model is instantly developed and also it is a much better model as compared to Vibe hence it can be used further is moving camera cases if developed for the purpose. However, the mixture of Gaussians is a memory and processor intensive background subtraction method. On average 55 ms were spent on processing an individual frame whilst in the Vibe algorithm around 19 ms

were spent on average. Hence it is theoretically possible to use the mixture of Gaussians method of background subtraction in real-time applications for the drone, but practically the processing time would cause problems in device response to tracking and detecting important events. The drone camera also records at 24 frames/sec and hence it would not be possible to use the mixture of Gaussians method for the AR Drone 2.0 unless modified to perform quicker.

Further problems that are in both algorithms include that both algorithms were unable to detect and remove shadows. The shadow beneath the car was not detected as part of the background as desired. Hence for shadow removal a different algorithm will have to be developed that can successfully detect and categorize shadows as background objects. The challenge in this particular dataset to remove the background from being detected as foreground is the similarity of the colour of the shadow and the road.

4.4 Performance Evaluation for ViBE and GMM

A precision-recall test was carried out for both the mixture of Gaussians as well as the Vibe algorithms. The results of the precision and recall for the Vibe and GMM algorithms are given in table 1.1. The ground truth used to obtain these results is shown in figure 4.10. It can be seen

Method	*Precision*	*Recall*
Vibe	0.5268	0.9216
Mixture of Gaussians	0.6928	0.5999

Table 4.1: GMM vs ViBE

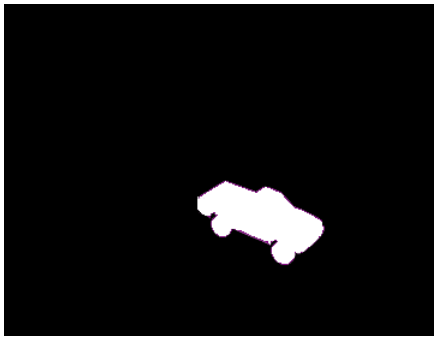


Figure 4.10: Ground truth used for Precision-Recall

overall the precision and recall for the mixture of Gaussians method is better than the precision

and recall from the Vibe algorithm. This is largely due to the lower amount of shadow detection and false positives in the mixture of Gaussians result as compared with the Vibe algorithm. The vibe algorithm on visual inspection although does do a better job in separating the whole moving object from the background, however the high amount of false positives impacts the performance of the algorithm and reduces the precision and recall. The advantage of the Vibe algorithm is the lower processing power required as well as the possibility of using it in real-time applications.

4.5 Gaussian Mixture Model and Traditional Method Comparisons

Figure 4.11 shows the result of the background subtraction using mixture of Gaussians for the Wallflower dataset. Table 4.2 shows the results for the OpenCV implementation of the background

Figure 4.11: Mixture of Gaussians result for the Wallflower data set waving trees challenge

subtraction algorithm using mixture of Gaussians for the waving trees case as well as the results for the frame differencing method and approximate median method for thresholds of 50. Figures 4.12

Method	**Precision**	**Recall**
Frame Differencing	0.7268	0.6285
Approximate Median	0.8383	0.3333
Mixture of Gaussians	0.7743	0.9031

Table 4.2: Precision vs Recall for the Frame Differencing, Approximate Median and Mixture of Gaussians method

and 4.13 show the results of the frame differencing method and the approximate median method for thresholds of 50. It can be seen from comparing figures 4.11, 4.12 and 4.13 that overall the mixture of Gaussians yields the best result in terms of fully detecting the object of interest. Frame Differencing then does the next best job, however, frame differencing results in a large quantity of

Figure 4.12: Frame differencing result for threshold of 50

Figure 4.13: Approximate median result for threshold of 50

false positives. The approximate median method correctly identifies the object of interest and its edges as well, but, does not necessarily segregate the whole object from the background.

From the precision and recall results in table 4.2 it can be seen the the mixture of Gaussians has a higher precision than that of the frame differencing method but lower that the approximate median. This may be due to the object although being detected much more visually, but there are a lower number of correctly identified foreground pixels as compared with the approximate median method. Despite that the mixture of Gaussians method has a superior result since it is based on a fix set of parameters whilst the approximate median method depends on the amount of threshold which has to be ideal for the best results.

From table 4.2 it can be seen that the overall recall of the mixture of Gaussian method is the highest. This is due to the way the mixture of Gaussians method models the background. The background modelling process is adaptive. When the current frame data is compared with the ground truth, the foreground still contains information from previous frames. This causes an increase in the number of incorrectly detected foreground pixels. In order to calculate the precision and recall, the foreground mask and the ground truth was subtracted in order to determine the number of incorrectly detected foreground pixels. The result of this image is shown in figure 4.14. It can be seen that in the incorrectly detected pixel data, previous motion of the detected object

Figure 4.14: Incorrectly detected foreground pixels in the mixture of Gaussians result

is still part of the image. This is due to the background still being re-modelled whilst the object has moved. Due to this extra information there is a higher amount of recall.

Figures 4.15 and 4.16 show the difference between the foreground mask and the ground truth for the frame differencing method and the approximate median method respectively. Hence it can

Figure 4.15: Incorrectly detected foreground pixels in the frame differencing result

Figure 4.16: Incorrectly detected foreground pixels in the approximate median result

be visually determined that the approximate median method results in the least amount of false positives and so its recall is low as compared with the other two algorithms.

AR Drone Implementation Methods

Several background subtraction methods have been discussed and their relative performance has also been analysed. It is now necessary to implement these algorithms on the AR Drone. Many methods can be used in order to implement the algorithms. One method that was researched was based around utilizing the camera of the AR Drone and making it act as a webcam. This would have enabled use of the camera directly in MATLAB using the Image Acquisition Toolbox. This would have allowed to implement the background subtraction methods in Simulink directly by utilizing the image acquisition toolbox extensions in Simulink. However, the problem with this method is that it is highly difficult to convert the AR Drone as a webcam since no official Windows driver support is available and the video feed is encrypted over a TCP connection. It is possible to make the video feed appear in a browser, however, it is not possible to use the browser to perform background subtraction tasks. Figure 5.1 shows the Simulink implementation of the frame differencing method where the from Video Device block can be reconfigured to take the video source from a webcam (image acquisition toolbox) or existing video. Another method available to

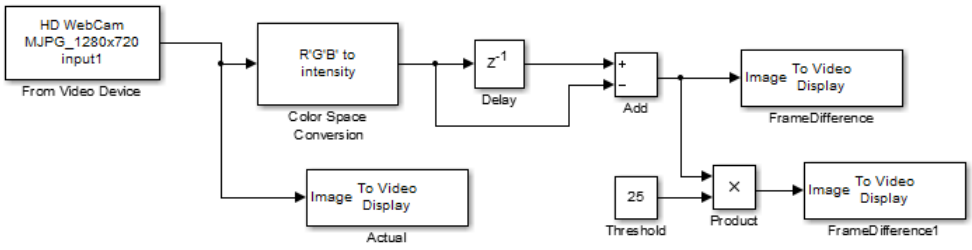


Figure 5.1: Simulink model for the frame differencing method

utilize was LabVIEW. LabVIEW has a third-party development kit for AR Drone which can be used to obtain the video feed and further program the AR Drone allowing remote control of the AR Drone using a custom controller. However, the actual implementation of even the simplest frame differencing is highly difficult in LabVIEW since the LabVIEW software is intended to be used for embedded computer programming rather than an image processing tool. One other possibility of

linking MATLAB with the LabVIEW environment was used in order to somehow obtain the AR Drone video feed and bring it into MATLAB for processing. However, due to compatibility of the AR Drone version (Version 2) and Simulink compatibility issues with LabVIEW, it was not possible to implement video processing this way. Figure 5.2 shows a simple visual interface utilizing the AR Drone toolbox for LabVIEW which can be used in order to obtain the drone video. The control cluster in figure 5.2 can reconfigured in order to utilize a custom controller in order to control the drone or use LabVIEW directly in order to do so. Another public codec known as ffmpeg was also

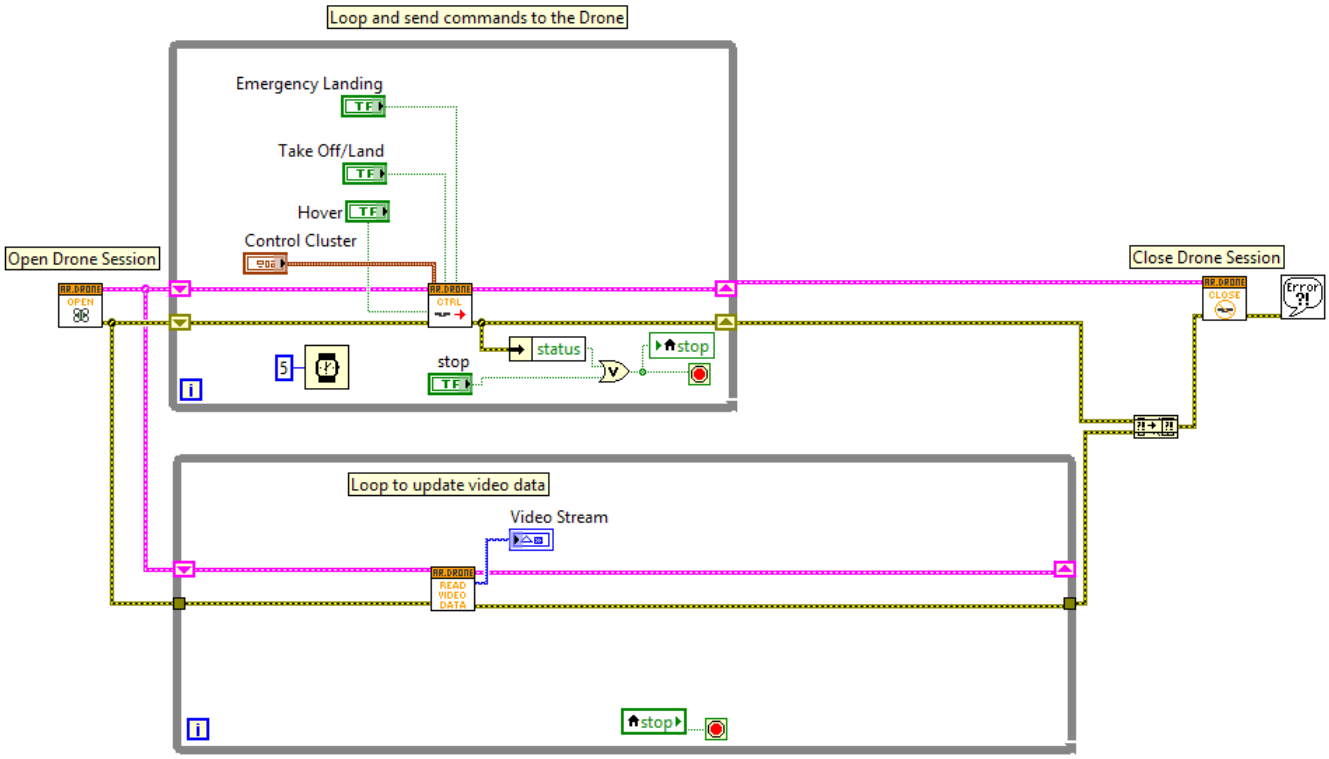


Figure 5.2: LabVIEW model to obtain and display video stream from the AR Drone

used to successfully obtain the AR Drone video feed, however similar issues prevented its successful use in performing background subtraction.

The only actual working solution that was seen possible to implement these algorithms onto the AR Drone (albeit processing would take place on a PC at ground) was using Ubuntu (Linux) in order to use OpenCV and C++ to connect with the AR Drone. Once a video link was established using OpenCV to obtain the video frames from the drone, it is then possible to extend the code in order to implement whatever background subtraction algorithm desired.

However, in order to implement any of the algorithm developed, it is necessary to be able to make those methods be able to alter the background so that the movement of the drone would still result in acceptable results from background subtraction. This would require a robust method of background modelling so that the background is updated as the drone moves.

In the following section, some possible implementations for the moving camera problem are discussed and the viability of using those methods in order to make the background subtraction algorithms work on the drone are also discussed.

Moving Camera Background Subtraction

Most of the background subtraction algorithms centre around having a fixed static camera so that the background model would also be fixed. However, in reality many applications exist where background subtraction is required for moving camera. An example is the AR Drone. However, even static cameras can ever be always static. For example strong winds could cause shaking in an otherwise static camera. Hand held cameras are another form of moving camera. If background subtraction algorithms are to be implemented for applications related to background subtraction, it is necessary to add a registration step before background subtraction that would remodel the background so any misalignments would be accounted for due to the moving camera. In order to tackle the moving camera problem, a generalized background subtraction algorithm was proposed by Kwak *et al* [35]. This method was based around creating a prior appearance model along with the current observation model sequentially in order to compensate for camera movement. Another method is used in [36, 37] which utilizes motion segmentation. The method of motion segmentation is based around cancelling the camera movement by estimating the background motion in order to separate the foreground objects. However the flaw with using this method for motion background subtraction is that this method depends on the background motion to be assumed to move in a single plane which is usually not the case such as in the AR Drone. Hence motion segmentation is not a valid form of background subtraction for the AR Drone. More recent motion background subtraction methods have based their research around video mosaicking.

A generalized background subtraction algorithm is meant to perform background subtraction for moving cameras and usually utilize the process of video mosaicking in order to perform background subtraction. The background model is estimated by modelling the entirety of the scene utilizing video mosaicking techniques such as RANSAC and FAST. The mosaic is created in real-time and the background subtraction process is unchanged in the sense that if a current background subtraction method is used with video mosaicking techniques, the background would still be pre-processed as the

algorithm requires it, however, the background will be estimated by video mosaicking techniques. Hence, all currently discussed background subtraction techniques can be used in conjunction with video mosaicking to perform generalized background subtraction. However, not all methods would be ideal due to the increased computational load due to the mosaicking process. Only the algorithms which are fast enough could be implemented with mosacking since it is required to perform real-time background subtraction on the AR Drone.

A. Romanoni *et al.* in their paper proposed a generalized background subtraction method that uses the RANSAC algorithm to perform background subtraction. In theory any background subtraction method can be used in conjuction with RANSAC, however, RANSAC is only an algorithm used to mosaic images uses a series of corner detection functions to adjust misalignment's. RANSAC is not a perfect process, and so at the registration step RANSAC will always produce misalignment's and hence false positives in the background.

RANSAC is known as RANdom SAmple Concensus. The RANSAC algorithm produces a background model via making a mosaic by detecting corner features in the scene. It does this by comparing the current and previous frames. The current frame would have moved in positive in relation to the previous frame and so RANSAC works by detecting and labelling the corners in both the current and previous frames. The corners that RANSAC deems to have moved considerable are taken in order to make the mosaic and the ones which are deemed to be mostly unchanged in position are rejected and so the image mosaic is produced. The current or later frame is known as the frame to be registered and the previous frame is known as the current background model or the reference frame.

Once RANSAC algorithm has completed its process the background subtraction methods discussed can be used then to perform background subtraction. The results will be satisfactory, however for much better results with fewer false positives in the background, A. Romanoni *et al.* in their paper used Temporal + Spatio-Temporal techniques to remove misalignments and perform the final background subtraction.

Temporal approaches are mostly defined by classical background subtraction methods. Background subtraction methods such as approximate median, mixture of Gaussians or adaptive background subtraction algorithm all model each pixels' history and intensity separately to that of the other

pixels. To summarize, classical background subtraction methods model each pixel individually by only considering the *temporal* changes. It is due to this the aforementioned false positives are caused when classical background subtraction methods are used in generalized background subtraction algorithms.

Spatio-Temporal approaches were combined in order to model is pixels' history and intensity in relation to its neighbourhood. Hence this spatio-temporal approach improves the overall results obtained.

In order to determine which algorithms are possible candidates to be used with video mosaicking algorithms, four different algorithms which include the frame differencing method, adaptive background method, mixture of Gaussians method and the approximate median method were tested for a 95 second video. The execution time for each method was tested where the execution time is the number of frames processed each milisecond. The results are given in Table 6.1.

The faster the algorithm is the higher the value of execution time. It can be seen that the

Method	**Execution time frame/ms**	**Total time taken**
Static Frame Differencing	0.161	17m 61s
Frame Differencing	0.193	14s 71ms
Static Frame Differencing with Median Filter	0.057	49s 57ms
Frame Differencing with Median Filter	0.144	19m 17ms
Adaptive Background Subtraction	0.075	37s 90ms
Mixture of Gaussians	0.119	23s 9ms
Approximate Median	0.149	18s 98ms
Approximate Median with Median Filter	0.138	20s 55ms

Table 6.1: Execution time for different background subtraction methods (All tested on OpenCV using C++)

frame differencing method has the highest number of frames processed in a unit time and adaptive background subtraction had the lowest. These results can be used to determine if a particular algorithm is fast enough to be used in real-time for applications in the drone. However, it is not possible to firmly state it being possible. This is due to the fast these results are for the individual algorithms, and not when they are combined with moving camera algorithms. The moving camera mosaicking algorithms would increase the processing power and hence the frames processed a unit time would drop. Further problems exist, for example, the video used to test the algorithms was a 240p resolution video. The AR Drone 2.0 utilizes a 720p HD video stream which is considerably

higher resolution as compared to the test video. Hence, this factor combined with the exclusion of video mosaicking algorithms in the tests cause difficulty in determining real-time usage possibility.

Furthermore, the viability of using the aforementioned background subtraction techniques also depends on how the processing would be carried out. Firstly, the processing carried out for the results in table 6.1 were done on a high end PC. It may be required to perform the processing on-line on the drone. In that case it might not be possible to carry out some of the background subtraction algorithms as some algorithms may have high complexity and hence if embedded systems in the AR Drone are used, they may not be able to process the algorithms.

Finally, coding imperfections also cause problems in determining real-time viability. For example, the mixture of Gaussians has been determined to be a faster algorithm as compared to the adaptive background subtraction algorithm. In reality the adaptive background subtraction algorithm is many times simpler as compared to the mixture of Gaussians and so theoretically it should result in higher execution times time the mixture of Gaussians. However, this does not seem to be the case. Hence, this may be due to inefficient code used to test the algorithm. Therefore, these combination of factors prevent fully determining which algorithms could be used in real-time on the drone.

Based on the results, the frame differencing and approximate median method, regardless of performing them with or without the median filter seem to be the best methods to utilize in real-time due to their low complexity and low computational requirements. This is if the processing is to be done using on-board embedded systems. However, if a powerful ground computer is to be used, it opens up the possibility of using the other algorithms as well.

Conclusion

Several different background subtraction algorithms have been researched upon and results of their implementation are explained and evaluated for a static background case. Furthermore it has been determined the possibility of using these background subtraction algorithms in camera motion cases using motion segmentation or video mosaicking by implementing a registration step before performing the background subtraction so that camera motion can be compensated for. Several algorithms have been identified as possible candidates for use in the AR Drone including the Vibe algorithm, approximate median algorithm and the adaptive background subtraction algorithm due to their fast response and capability to adapt to changing backgrounds.

In terms of acheiving the best possible results, it has been seen that statistical methods such as the mixture of Gaussian method of background subtraction overall produces the best results. This is due to the fact that the mixture of Gaussians, as it was seen in figures 4.10 and 4.11, had the best result in terms of detecting the object as well as segmenting the whole of the object. Whereas other algorithms may have had overall better results, but for most of the algorithms there was mostly a larger number of false negatives and positives as compared to the mixture of Gaussians. Hence the mixture of Gaussians is superior to the other algorithms discussed on a pure background subtraction perspective. However, for real-time implementation, depending on how the implementation may be done, mixture of Gaussians method may not be the best method due to high computational requirements. A problem with the mixture of Gaussians is the slow rate of background model adapting. Despite the flaws with the mixture of Gaussians, it still provided the best results. Another good algorithm that is easier to implement with lower computational power is the ViBE algorithm and is a real contender for use in real-time applications.

Future Work

For future work in this project, further work may be carried out either in MATLAB or OpenCV in order to make video mosaicking background subtraction work. MATLAB already has video mosaicking capability using the FAST algorithm. Basic forms of background subtraction can also be programmed to work real-time using the drone camera. In this dissertation several methods to acquire video feed had been researched, however, besides using OpenCV none of these methods were seen to be a viable way for real-time video processing. However, if the firmware of the AR Drone be modified so that the video feed may be extracted into MATLAB, it would enable to perform various background subtraction algorithms directly. Using OpenCV the methods already developed can be programmed in OpenCV using MATLAB functions to convert directly to OpenCV or reprogram the algorithms in native C++ so that they may be implemented in OpenCV.

Appendices

9.1 Appendix 1 - Frame Differencing Code

Below is the source code used to perform frame differencing in MATLAB.

```
1  %Frame Differencing
2  clc;
3  close all;
4  clear;
5  try
6  source = VideoReader('WavingTrees.mp4'); %Read the video file
7  numberOfFrames = source.NumberOfFrames; %calculate the number of
8  %frames in the video
9  vidHeight = source.Height; %determine height in pixels for the video
10 vidWidth = source.Width; %determine width in pixels for the video
11 figure('Color',[1 1 1]); %figure produced with results set to have white
12 %background
13 for frame = 1 : numberOfFrames %frame 1 till end will be read
14     thisFrame = read(source, frame); %thisFrame is the current frame
15     bgmodel = read(source, 1); %background model has been set as
16     % the first frame. Alternatively can be set as frame-1 to get the
17     %previous frame
18     hImage = subplot(2, 2, 1);
19     image(thisFrame);
20     title('Current Frame')
21     drawnow; %drawnow forces MATLAB to refresh the current frame
22     Background=bgmodel;%Setting the background
23 % Display the changing/adapting background.
24  subplot(2, 2, 2);
25  imshow(Background);
26     title('Background model') %if frame-1 used for background
27     %need to add drawnow to change the background model
```

```
% Calculate a difference between this frame and the background.
 differenceImage = thisFrame - uint8(Background);%  frame
    %differencinf step
 grayImage = rgb2gray(differenceImage); % Convert to greyscale
    threshold = 70; %Threshold is set here
    binaryImage = grayImage > threshold; %using frame differencing algoritthm
    %results above the threshold are seperated from that below to get final
    % result
% Plot the binary image.
 subplot(2, 2, 3);
 imshow(binaryImage);
    title('Background Subtraction Result')
end
end
```

9.2 Appendix 2 - Approximate Median Code

Below is the source code used to perform approximate median background subtraction in MATLAB.

```
clear all
clc
source = VideoReader('WavingTrees.mp4'); %Video file to be analyzed is input
%here
th = 25; %The threshold is defined here
Background = read(source, 1); %The background is defined by reading the first frame
BGGreyscale = rgb2gray(Background); %The background is converted to greyscale
BGIntensity = double(BGGreyscale); %The background is then converted to pure b/w
thisFramesize = size(Background);
vidHeight = thisFramesize(1); %height of the video in pixels
vidWidth = thisFramesize(2); %width of the video in pixels
thisFrame = zeros(vidHeight, vidWidth); %foreground is initialized as zeros (pure black)
numFrames = get(source, 'NumberOfFrames'); %calculating the number of frames
for i = 2:numFrames
Foreground = read(source,i); %this code will read every frame in sequence
    FGGreyscale = rgb2gray(Foreground); %same pre-processing to get greyscale foreground
    frameDifference = abs(double(FGGreyscale) - double(BGIntensity)); %frame differencing
    for x=1:vidWidth
        for y=1:vidHeight
            if ((frameDifference(y,x) > th)) %Similar to frame differencin the
                %frame differenced image is compared to a threshold
                thisFrame(y,x) = FGGreyscale(y,x);
            else
                thisFrame(y,x) = 0; %if above threshold then 1 else 0
            end
            if (FGGreyscale(y,x) > BGIntensity(y,x))
                BGIntensity(y,x) = BGIntensity(y,x) + 1;  %This is where the approximate
                %median algorithm described in the dissertation is applied
            elseif (FGGreyscale(y,x) < BGIntensity(y,x))
                BGIntensity(y,x) = BGIntensity(y,x) - 1;  %half the background pixels
                %are categorized as foreground and half as background to
                %get effect of approximate median
            end
        end
    end
    figure(1),subplot(3,1,1),imshow(Foreground)
```

```
37        subplot(3,1,2),imshow(uint8(BGIntensity))
38        subplot(3,1,3),imshow(uint8(thisFrame)) %results are plotted live
39
40  end
```

9.3 Appendix 3 - Adaptive Background Subtraction Code

Below is the source code used to perform adaptive background subtraction in MATLAB.

```
%Adaptive Background subtraction
clc;    close all;
clear;
try
source = VideoReader('WavingTrees.mp4'); %video file being analyzed
numberOfFrames = source.NumberOfFrames; %number of frames calculated
vidHeight = source.Height; %video height in pixels
vidWidth = source.Width; %video width in pixels
figure('Color',[1 1 1]); %white background setting for results display
for i = 1 : numberOfFrames %read from frame 1 till end
    thisFrame = read(source, i); %current frame
    hImage = subplot(2, 2, 1);
    image(thisFrame); %current frame being displayed
    title('Current Frame')
    drawnow; %drawnow forces updating
alpha = 0.90;
    if i == 1
    Background = thisFrame;
    else
    Background = (1-alpha)* thisFrame + alpha * Background;
        %background is updating using the algorithm
    end
 subplot(2, 2, 2);
 imshow(Background); %background model is displayed
    title('Background model')
 framediffer = thisFrame - uint8(Background); %after applying algorithm
    %normal frame differencing is performed
 grayImage = rgb2gray(framediffer);
    threshold = 25; % a threshold value desired is set
    foregroundmask = grayImage > threshold; %the values of the foreground
    %mask are compared with a threshold
 subplot(2, 2, 3);
 imshow(foregroundmask); %threshold mask is displayed
    title('Background Subtraction Result')
end
end
```

9.4 Appendix 4 - Precision, Recall and F-Measure Code

This section shows the code snippet used to perform the basic precision-recall calculations. Many various variants of the code were actually used and this is just the basic structure for the code. This code can be used if the foreground mask, ground truth and backgrounds are available.

```
fg(fg>0)=255; %foreground mask values changed to either 0 or 255
pr = imread('perfect.bmp'); %Ground truth input
pr_gr = rgb2gray(pr); %converting ground truth to a similar format to
%that used for fg and bg masks
pr_bw = double(pr_gr); %intensity conversion of the ground truth
fg_pr = imsubtract(fg,pr_bw); %the fg mask and ground truth are subtracted
%in order to get the image matrix that represents the total number of in-
%correctly identified foreground puxels
middleColumn = height/2;
leftHalfOnes = nnz(fg_pr(:,1:middleColumn)); %nnz function is used in order
%to determine the total number of non-zero elements (i.e. the total number
%of foreground classified pixels
rightHalfOnes = nnz(fg_pr(:,middleColumn+1:end));
pr_bw(pr_bw>0)=255; %ground truth results are also converted to either a 0
%value or a value of 255
%Now comparing fg and pr_bw
PC_FG = (pr_bw == 255 & fg == 255); %a matrix is formed where the fg
%and ground truth are compared. In the cases where the value matches
%i.e. correctly clssified foreground pixel, the matrix will take a value
%of 1 for the corresponding pixel, otherwise zero
noOfCorrectFGPixels=nnz(PC_FG);%hence using nnz, number of correctly
%identified fg pixels
noOfIncorrectFGPixels = leftHalfOnes + rightHalfOnes;%incorrectly identified
noOfGTPixels = nnz(pr_bw); %number of pixels in the ground truth
noOfPixelsinFG = nnz(fg); %number of pixels in the fg mask identified as
%foreground pixels
Recall= noOfCorrectFGPixels/noOfGTPixels; %Recall equation
Precision = noOfCorrectFGPixels/noOfPixelsinFG; %Precision equation
F_measure = 2*(Recall*Precision/Recall+Precision); %f-measure equation
```

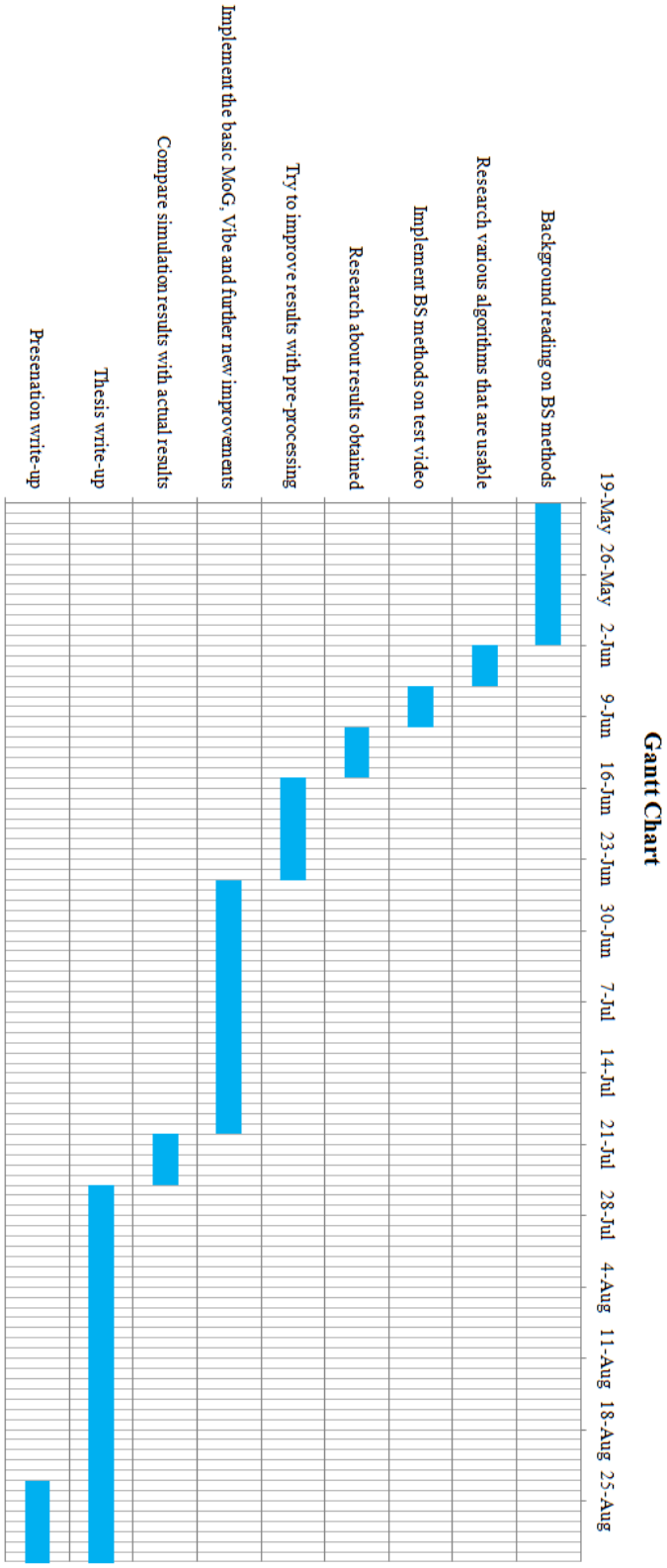
9.5 Appendix 5 - General Median Filter Implementation

This code is for median filter implementation for the frame differencing method. A similar code is used for approximate median.

```
clear all
clc
source = VideoReader('WavingTrees.mp4'); %Video file to be analyzed is input
%here
th = 25; %The threshold is defined here
Background = read(source, 1); %The background is defined by reading the first frame
BGGreyscale = rgb2gray(Background); %The background is converted to greyscale
bg_no = imnoise(bg_gr,'salt & pepper',0.02);
bg_fi = filter2(fspecial('average',3),bg_no)/255;
bg_mf = medfilt2(bg_no,[3 3]);
bg_bw = bg_mf;
BGIntensity = double(bg_bw); %The background is then converted to pure b/w
thisFramesize = size(Background);
vidHeight = thisFramesize(1); %height of the video in pixels
vidWidth = thisFramesize(2); %width of the video in pixels
thisFrame = zeros(vidHeight, vidWidth); %foreground is initialized as zeros (pure black)
numFrames = get(source, 'NumberOfFrames'); %calculating the number of frames
for i = 2:numFrames
Foreground = read(source,i); %this code will read every frame in sequence
    fg_gr = rgb2gray(Foreground);
    fg_no = imnoise(fg_gr,'salt & pepper',0.02);
    fr_bw = medfilt2(fg_no,[3 3]);
    FGGreyscale = fr_bw;
    frameDifference = abs(double(FGGreyscale) - double(BGIntensity)); %frame differencing
    for x=1:vidWidth
        for y=1:vidHeight
            if ((frameDifference(y,x) > th)) %Similar to frame differencin the
                %frame differenced image is compared to a threshold
                thisFrame(y,x) = FGGreyscale(y,x);
            else
                thisFrame(y,x) = 0; %if above threshold then 1 else 0
            end
            if (FGGreyscale(y,x) > BGIntensity(y,x))
                BGIntensity(y,x) = BGIntensity(y,x) + 1;  %This is where the approximate
```

```
35                  %median algorithm described in the dissertation is applied
36              elseif (FGGreyscale(y,x) < BGIntensity(y,x))
37                  BGIntensity(y,x) = BGIntensity(y,x) - 1;  %half the background pixels
38                  %are categorized as foreground and half as background to
39                  %get effect of approximate median
40              end
41          end
42      end
43      figure(1),subplot(3,1,1),imshow(Foreground)
44      subplot(3,1,2),imshow(uint8(BGIntensity))
45      subplot(3,1,3),imshow(uint8(thisFrame)) %results are plotted live
46
47  end
```

9.6 Appendix 6 - Gantt Chart



References

[1] A. Gibiansky. Quadcopter Dynamics, Simulation, and Control. http://xtal.tk/wp-content/uploads/2014/04/Quadcopter-Dynamics-Simulation-and-Control.pdf, 2014. [Online; accessed 23-Aug-2014].

[2] R. Feltman. The Future of Sports Photography: Drones. http://www.theatlantic.com/technology/archive/2014/02/the-future-of-sports-photography-drones/283896/. [Online; accessed 1-Sep-2014].

[3] N. Ungerleider. Unmanned Drones Go From Afghanistan To Hollywood". UAV Drones. http://www.fastcompany.com/1816578/unmanned-drones-go-afghanistan-hollywood. [Online; accessed 1-Sep-2014].

[4] P. Time. Manufacturers Market Drones Before the Law Specifies How They Can Be Used. http://activistdefense.wordpress.com/2013/02/16/manufacturers-market-drones-before-the-law-specifies-how-they-can-be-used/. [Online; accessed 1-Sep-2014].

[5] K. Jason. North Dakota Man Sentenced to Jail In Controversial Drone-Arrest Case. http://www.usnews.com/news/articles/2014/01/15/north-dakota-man-sentenced-to-jail-in-controversial-drone-arrest-case. [Online; accessed 1-Sep-2014].

[6] Our UAV. {http://www.energy-business-review.com/companies/universal_wing_technologies_inc-_gd_1262}. [Online; accessed 1-Sep-2014].

[7] V. Cheng and N. Kehtarnavaz. A smart camera application: Dsp-based people detection and tracking. *Journal of Electronic Imaging*, 9(3)(336-346), 2000.

[8] H. Saji T. Inaguma and H. Nakatani. Hand motion tracking based on a constraint of three-dimensional continuity. *Journal of Electronic Imaging*, 14(1), 2005.

[9] D. Makris and T. Ellis. Path detection in video surveillance. *Image and Vision Computing*, 20(895-903), 2002.

[10] X. Wang E. Grimson and K. Tieu. Learning semantic scene models from observing activity in visual surveillance. *Transactions on Systems, Man and Cybernetics, Part B*, 35(397-408), 2005.

[11] Q. Zhou and J. Aggarwal. Tracking and classifying moving objects from video. *Performance Evaluation of Tracking Systems Workshop*, 2001.

[12] S.C.S. Cheung and C. Kamath. Robust techniques for background subtraction in urban traffic video. *Visual Communications and Image Processing*, 5308(880-893), 2004.

[13] T. Bouwmans. Traditional and recent approaches in background modeling for foreground detection: An overview. *Computer Science Review*, I1-I2(31-66), 2014.

[14] V. Murino M. Cristani, M. Farenzena D. Bloisi. Background subtraction for automated multisensor surveillance: a comprehensive review. *Advances in Signal Processing*, 2010.

[15] C.R. Wren. Pfinder: real-time tracking of the human body. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(780-785), 1997.

[16] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. *International conference on Computer Vision and Pattern Recognition*, 1999.

[17] Y. Benezeth. Comparative study of background subtraction algorithms. *Journal of Electronic Imaging*, 19(7), 2010.

[18] D. Harwood A Elgammal and L. Davis. Non-parametric model for background subtraction. *European Conference on Computer Vision*, (751-767), 2000.

[19] A.Mittal and N. Paragios. Motion-based background subtraction using adaptive kernel density estimation. *Proceedings of the international conference on Computer Vision and Pattern Recognition*, 2004.

[20] D. Harwood K. Kim, T.H. Chalidabhongse and L. Davis. Real-time foreground-background segmentation using codebook model. *Real-Time Imaging*, 11(172-185), 2005.

[21] T. Bouwmans. Traditional and recent approaches in background modeling for foreground detection: An overview. *Computer Science Review*, 11-12(41-51), 2014.

[22] Chen Y. Yamazaki M, Xu G. Detection of moving objects by independent component analysis. *ACCV*, (467-478), 2006.

[23] Lai C. Tsai D. Independent component analysis-based background subtraction for indoor surveillance. *IEEE Trans Image Proc IP*, 18(158-167), 2009.

[24] Tao H. Tang F. Fast linear discriminant analysis using binary bases. *18th Inter Conf on Pattern Recognition*, 2, 2006.

[25] Zhou S.K. and Chellapa R. Multiple-exemplar discriminant analysis for face recognition. *IEEE Xplore*, (1-3).

[26] T. Bouwmans. Type-2 fuzzy mixture of gaussians model: Application to background modelings. *Las Vegas : Etats-Unis d'Amerique*, (1-10), 2008.

[27] L. Wang W. Hu, T. Tan and S. Maybank. A survey on visual surveillance of object motion and behaviors. *Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 34(3)(334-352), 2006.

[28] S. Hadadan S. Panahi, S. Sheikhi and N. Gheissari. Evaluation of background subtraction methods. *International conference on Digital Image Computing: Techniques and Applications*, (357-364), 2008.

[29] B. Brumitt K. Toyama, J. Krumm and B. Meyers. Wallflower: principles and practice of background maintenance. *International Conference on Computer Visions*, 1(255-261), 1999.

[30] B. Emile Y. Benezeth and C. Rosenberger. Comparative study on foreground detection algorithms for human detection. *International Conference on Image and Graphics*, (661-666), 2007.

[31] P. Ribeiro E. Andrade P. Moreno S. Pesnel T. List R. Emonet R.B. Fisher J.S. Victor D. Hall, J. Nascimento and J.L. Crowley. Comparison of target detection algorithms using

adaptive background models. *International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, (113-120), 2005.

[32] M. Van Droogenbroeck O. Barnich. Vibe: A universal background subtraction algorithm for video sequences. *IEEE Image Processing*, (1709 - 1724), 2011.

[33] J. Krumm. Wallflower: Test Images. `http://research.microsoft.com/en-us/um/people/jckrumm/wallflower/testimages.htm`, 2014. [Online; accessed 30-Aug-2014].

[34] Sebastian Brutzer, Benjamin Höferlin, and Gunther Heidemann. Evaluation of background subtraction techniques for video surveillance. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1937–1944. IEEE, 2011.

[35] O. Javed Y. Sheikh and T. Kanade. Background subtraction based on coocurence of image variations. *In CVPR*, 2003.

[36] E. Hayman and J. O. Eklundh. Statistical background subtraction for a mobile observer. *In ICCV*, 2003.

[37] A. Mittal and D. Huttenlocher. Scene modeling for wide area surveillance and image synthesis. *In CVPR*, 2000.