University of Johannesburg

A Comparison of Background Subtraction Techniques Under Sudden Illumination Changes for Video Surveillance

Author:
Cornelius JF Reyneke
200723049

Supervisors:
Prof. André L. Nel
Philip E. Robinson

A thesis submitted in partial fulfilment of the requirements for the degree of
Magister Ingeneriae

at the

School of Electrical and Electronic Engineering
Faculty of Engineering and the Built Environment

August 2015

Declaration of Authorship

I, CJF Reyneke, declare that this thesis titled, 'A Comparison of Background Subtraction Techniques Under Sudden Illumination Changes for Video Surveillance' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: ______________________

Date: ______________________

"In science men have discovered an activity of the very highest value in which they are no longer, as in art, dependent for progress upon the appearance of continually greater genius, for in science the successors stand upon the shoulders of their predecessors; where one man of supreme genius has invented a method, a thousand lesser men can apply it."

Bertrand Russell

UNIVERSITY OF JOHANNESBURG

Abstract

Faculty of Engineering and the Built Environment
School of Electrical and Electronic Engineering

Magister Ingeneriae

A Comparison of Background Subtraction Techniques Under Sudden Illumination Changes for Video Surveillance

by Cornelius JF Reyneke
200723049

An investigation of background modeling techniques that have potential to be robust to sudden and gradual illumination changes is presented. The first makes use of a modified local binary pattern that considers both spatial texture and colour information. The second uses a combination of a frame-based Gaussianity Test and a pixel-based Shading Model to handle sudden illumination changes. The third solution is an extension of a popular kernel density estimation (KDE) technique from the temporal to spatio-temporal domain using 9-dimensional data points instead of pixel intensity values and a discrete hyperspherical kernel instead of a Gaussian kernel. The solutions are compared against one another with regard to classification accuracy and computation time. The most robust of these techniques is improved upon, and implemented on a GPU with to achieve real-time processing speeds.

Acknowledgements

I would like to thank my supervisors and mentors, Prof. André Nel, Philip Robinson and Yuko Roodt, as well as my parents, Cor and Aléta Reyneke, without whom this dissertation would not have been possible.

This masters dissertation was funded by:
The Council of Scientific and Industrial Research

Contents

Abbreviations

COD	**C**ontiguous **O**utliers **D**etection
CS-LBP	**C**entre **S**ymmetrical **L**ocal **B**inary **P**attern
DRMF	**D**irect **R**obust **M**atrix **F**actorization
GMM	**G**aussian **M**ixture **M**odels
GRASTA	**G**rassmanian **R**obust **A**daptive **S**ubspace **T**racking **A**lgorithm
IMMC	**I**ncremental **M**aximum **M**argin **C**riterion
KDE	**K**ernel **D**ensity **E**stimation
LBP	**L**ocal **B**inary **P**attern
PCA	**P**rinciple **C**omponent **A**nalysis
LDA	**L**inear **D**iscriminative **A**nalysis
LRM	**L**ow **R**ank **M**inimization
pROST	**p**-norm **R**obust **S**ubspace **T**racking
SCBP	**S**patial **C**olour **B**inary **P**attern
SVDD	**S**upport **V**ector **D**ata **D**escription
SVM	**S**upport **V**ector **M**achine
SVR	**S**upport **V**ector **R**egression
t-GRASTA	**t**ransformed **G**rassmanian **R**obust **A**daptive **S**ubspace **T**racking **A**lgorithm

Dedicated to my parents

Chapter 1

Introduction and Problem Statement

1.1 Background

The field of computer vision is the study of techniques to extract meaningful information from digital images and videos repeatably, efficiently and accurately. It includes methods to acquire, process, analyze and understand images [1]. The primary goal of computer vision is to develop a system that can describe the world that is seen in one or more images and to reconstruct its properties [2].

There are a number of fields of study in computer vision as is shown in Figure 1.1. One of these is object detection. Its main aim is the detection and identification of salient information or objects from an image or video sequence. Object detection has a wide variety of applications including intelligent visual surveillance, intelligent visual observation of animals and insects, optical motion capture, human-machine interaction, content-based video coding automated systems and manufacturing processes [3].

Background subtraction techniques are a popular form of object detection in computer vision systems for video sequences and have become a fundamental component for many of its applications. They are useful because one can identify object regions within cluttered and undefined scenes. The goal is to classify the background in a scene so that only foreground objects remain for further analysis; tracking a herd of animals in an outdoor environment is an example of this [4]. This is achieved by subtracting a background image from an observed image. However, when simple background subtraction techniques are implemented in scenes with dynamic backgrounds or fluctuating lighting conditions, problems begin to arise [5]. As an example, scenes that contain swaying leaves or moving

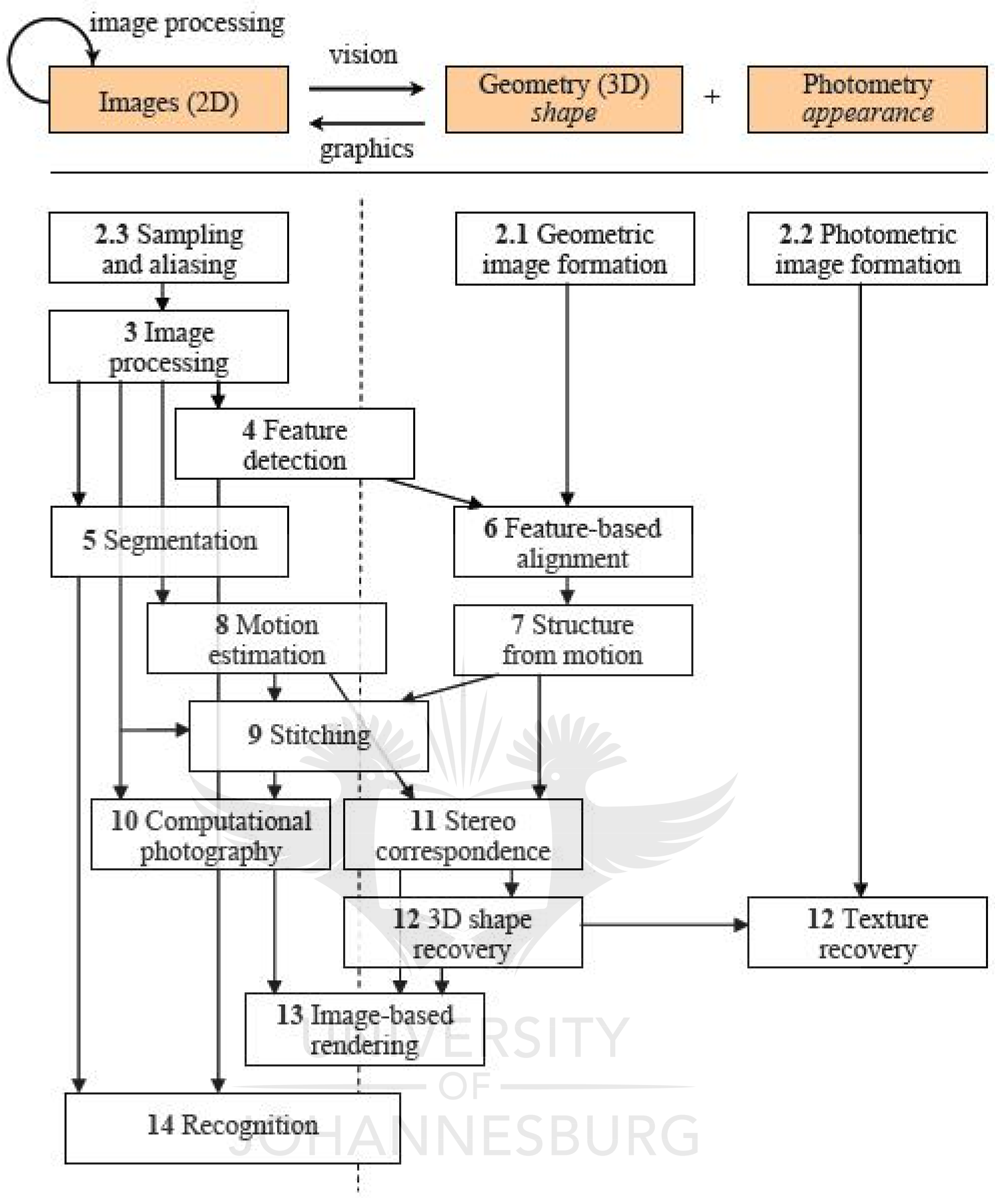

FIGURE 1.1: An illustration of the relationship between images, geometry and photometry [2].

clouds cast shadows and can easily be misinterpreted as additional objects. In order to obtain an accurate background model it has to be updated regularly in order to adapt to the temporal changes.

A number of background modeling challenges exist including noisy input, camera jitter, automatic camera adjustments, bootstrapping, camouflage, foreground aperture, moved background objects, inserted background objects, dynamic backgrounds, beginning moving objects, sleeping foreground objects, shadows and illumination changes [3].

1.2 Problem statement

Numerous background modeling techniques have been proposed to address gradual illumination changes in a scene. Some of the state-of-the-art methods include Gaussian Mixture Models (GMM), Codebooks, Eigenbackgrounds, Shading Models (SM), Statistical Circular Shift Moments (SCSM), and Local Binary Patterns (LBP) [6, 7, 8, 9, 10, 11]. However sudden illumination changes, such as light sources being turned on and off or curtains being opened or closed, are still a very challenging problem and an active area of research. In recent years a number of new segmentation techniques have been proposed with the potential to be robust to sudden illumination changes and small background dynamics at the cost of being computationally expensive. It is now necessary to compare the performance and characterize the capabilities of this new set of proposed techniques.

1.3 Purpose of the study

This dissertation aims to measure and compare a number of candidate solutions for background modeling under sudden illumination changes. Once identified and implemented a comparison and analysis of their characteristics will be discussed. Furthermore, the superior mechanisms of these solutions will then be incorporated into a new algorithm, with the hopes of outperforming previously developed solutions.

1.4 Scope

For the scope of this dissertation, the algorithms that will be investigated will be constrained to video sequences of a stationary scene. A scene can be indoor or outdoor and can contain moving vegetation or sudden illumination changes. No prior information regarding the background will be available; an on-line adaptive background model will be built and maintained.

1.5 Research methodology

We implement the Design Science Research Paradigm which provides a set of techniques with which to analyse research in information systems. It involves the design of novel or innovative solutions and the analysis of the use and/or performance of such solutions to improve existing information systems [12].

A literature review will first be completed to identify three techniques that have potential to handle sudden illumination changes. These solutions will be implemented and then tested using a publicly available dataset. These contain scenes under gradual and sudden illumination changes as well as scenes with dynamic backgrounds like waving trees. The best solution will be distinguished with regard to accuracy and computation time. Using this solution as a basis, and incorporating useful modules from all three solutions, a new, superior solution will be developed. Furthermore, this solution will be implemented on a Graphics Processing Unit (GPU) for real-time application.

1.6 Dissertation overview

In summary, this dissertation will investigate background modeling techniques that have potential to be robust against sudden and gradual illumination changes for a stationary scene in Chapter 2. Furthermore, the most robust parts of these techniques will be identified in Chapters 3 and 4, improved upon in Chapter 5, and implemented on a GPU in order to achieve real-time processing speeds. Finally, some conclusions and insights are discussed in Chapter 6.

Chapter 2

Literature Review

This chapter will provide an overview of the theory and concepts related to background subtraction and illumination change. This dissertation will primarily focus on background modeling and maintenance techniques, the sub-categories of which will be explored in greater detail. Some sub-categories are not immediately relevant in terms of their capacity to perform in real-time or handle sudden illumination changes (section 2.1.9.6 and beyond) and these have only been discussed very briefly.

2.1 Background subtraction concepts

2.1.1 Image

An image (as shown in Figure 2.1) is defined as a two-dimensional array that corresponds to the grid of pixels that comprise a digital image. Each pixel has four normalized values corresponding to the red channel, green channel, blue channel and pixel intensity.

$$\bar{P}(x,y) = \langle r(x,y), g(x,y), b(x,y), I(x,y)\rangle \tag{2.1}$$

where $r(x,y), g(x,y), b(x,y)$ and $I(x,y)$ are the red, green, blue and intensity values at coordinates (x,y) and $(x,y) \in \Omega$ where Ω is the image domain.

$I(x,y)$ is a gamma-corrected greyscale value that can be computed with the following equation (Figure 2.2) [14]:

$$I(x,y) = 0.2126r(x,y) + 0.7152g(x,y) + 0.0722b(x,y) \tag{2.2}$$

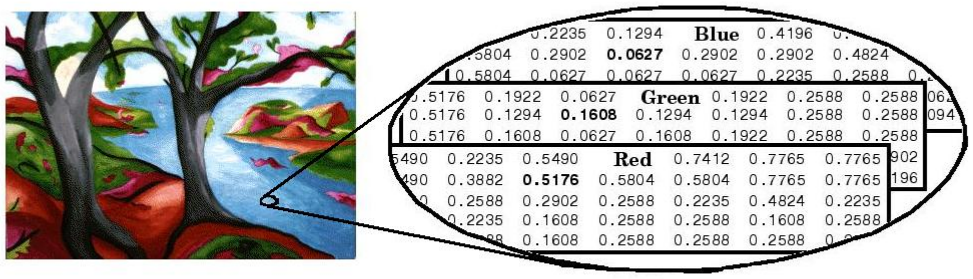

FIGURE 2.1: Each pixel in an image has four values corresponding to the red, blue, green and intensity channel [13].

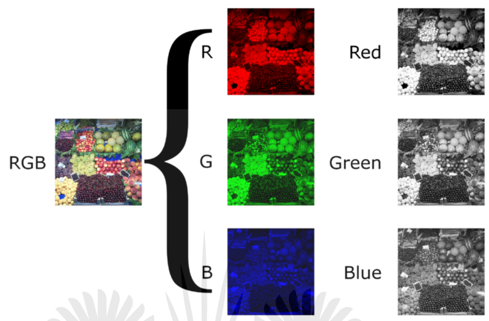

FIGURE 2.2: The grayscale intensity value is calculated using a weighted sum of the red, blue and green colour channels [15].

2.1.2 Video

A video (as shown in Figure 2.3) is defined as a temporally ordered sequence of images:

$$\bar{V}(x, y, t) = \bar{P}_t(x, y) = \langle r_t(x, y), g_t(x, y), b_t(x, y), I_t(x, y)\rangle \tag{2.3}$$

where $r_t(x, y), g_t(x, y), b_t(x, y)$ and $I_t(x, y)$ are the red, green, blue and intensity values at coordinates (x, y) and time t.

2.1.3 Feature

A feature is a specific structure in an image and acts as a descriptor of image information. There are typically five types of features which are commonly used: colour, edge, depth, motion and texture [3] (Figure 2.4. Features are selected based on their relevance to the computational task to be solved. As an example, colour features are very discriminatory but have limitations when encountering shadow, camouflage and illumination changes and are thus context-dependent. A set of multiple features is known as a feature vector and all the possible features comprise the feature space:

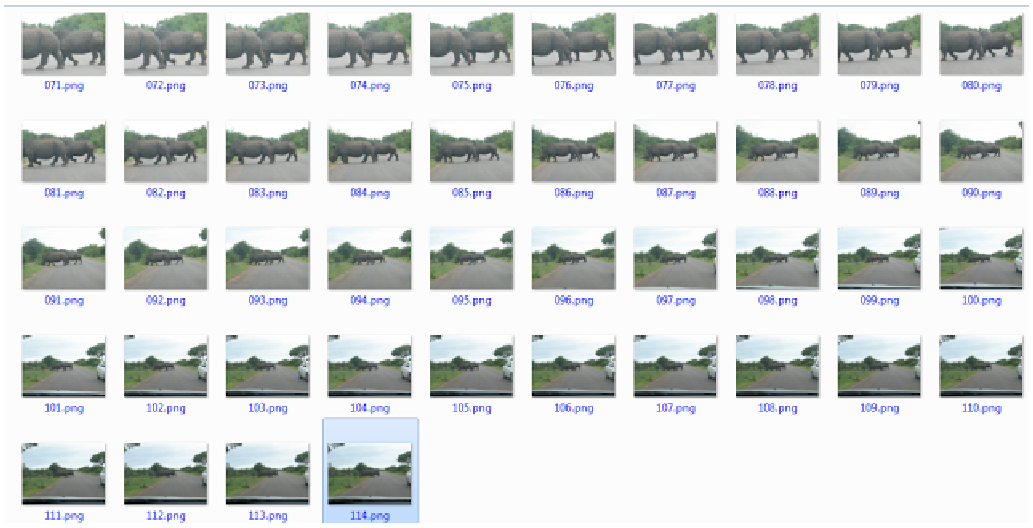

FIGURE 2.3: A video is a temporally ordered sequence of images [16].

$$\bar{E}(x, y) = \langle e_1(x, y), e_2(x, y), ..., e_i(x, y)\rangle \tag{2.4}$$

where $\bar{E}(x, y)$ is a feature vector that comprises i features where Δ is the feature space:

$$\bar{E}(x, y) \subset \{e_1(x, y), e_2(x, y), ..., e_i(x, y)\} \in \Delta \tag{2.5}$$

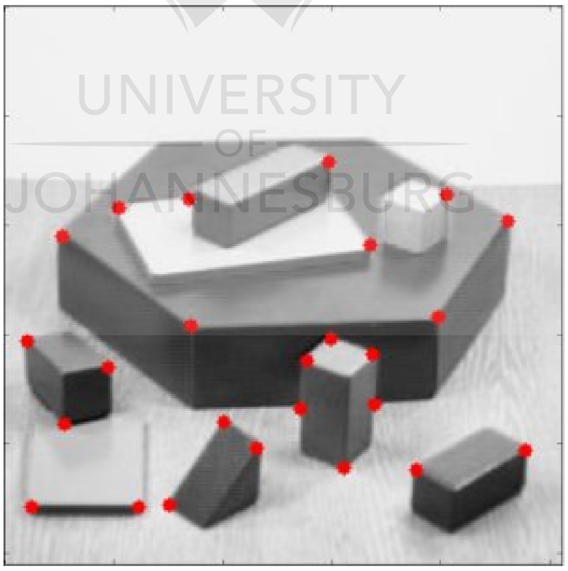

FIGURE 2.4: Example of corner features (shown in red) [17].

It is common to use a combination of features, such as a colour feature along with an edge feature to be more robust to local illumination changes.

Features have different sizes depending on what scale they operate at such as a pixel, block or cluster. The size of a feature influences it's precision, tolerance to noise and size of image structure it can describe.

Furthermore, different strategies can be employed, such as a multi-level approach, in order to achieve a desired property of a background model such as scale-invariance. Figure 2.5 shows how a Difference-of-Gaussian (DOG) Approach can be implemented to extract different scale levels as shown in Figure 2.6.

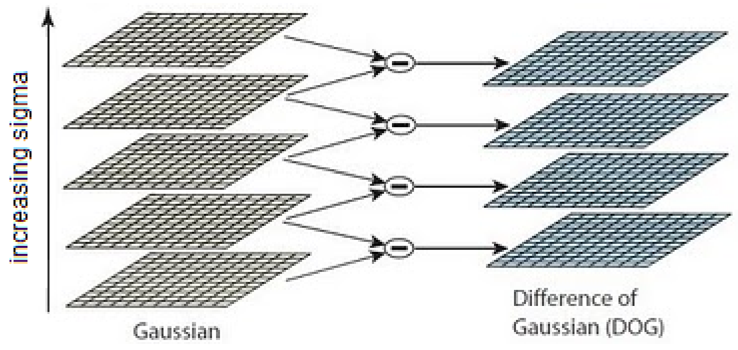

FIGURE 2.5: The DOG scale space technique [18].

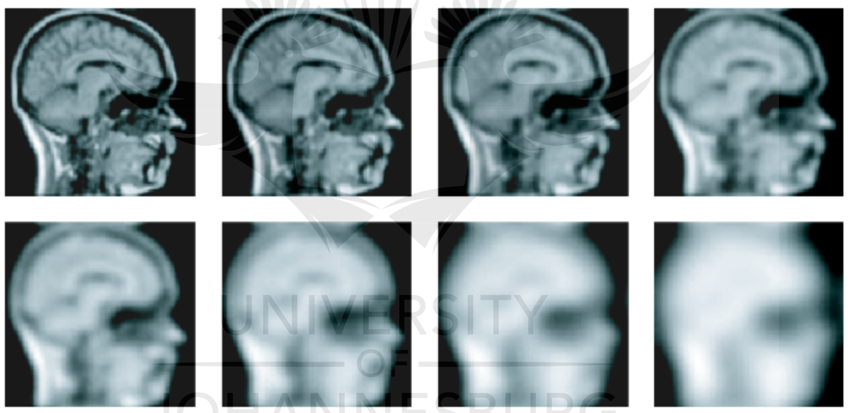

FIGURE 2.6: Different levels of a DOG scale space with increasing sigma from left-to-right and top-to-bottom [19].

Features, like ones that describe edges or colour, often only make use of low-order statistics (constant or linear terms) [20]. The colour or greyscale histogram, local binary pattern (LBP) and moments are examples of higher-order statistics that can be implemented as features.

2.1.3.1 Greyscale histogram

A greyscale histogram is an array of size N which corresponds to the number of greyscale intensity bins in a digital image. N is commonly 256 due to an 8-bit image representation. The histogram can be defined by the following discrete function:

$$h(r_k) = n_k \tag{2.6}$$

where r_k is the k^{th} grey level and n_k is the number of pixels in the image having the gray level r_k. Figure 2.7 shows a typical histogram.

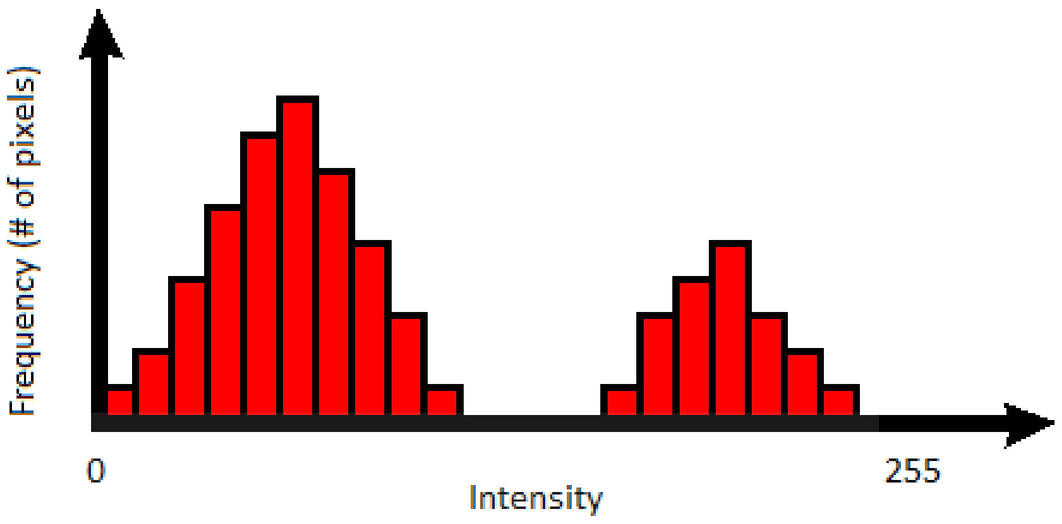

FIGURE 2.7: An illustrated greyscale histogram showing frequency (number of pixels) versus intensity bin value (0-255) [21].

Similarly, colour histograms can be employed as colour features. The histogram is ubiquitous in image processing and a very useful statistic in brightness and contrast enhancement techniques as shown in Figure 2.8.

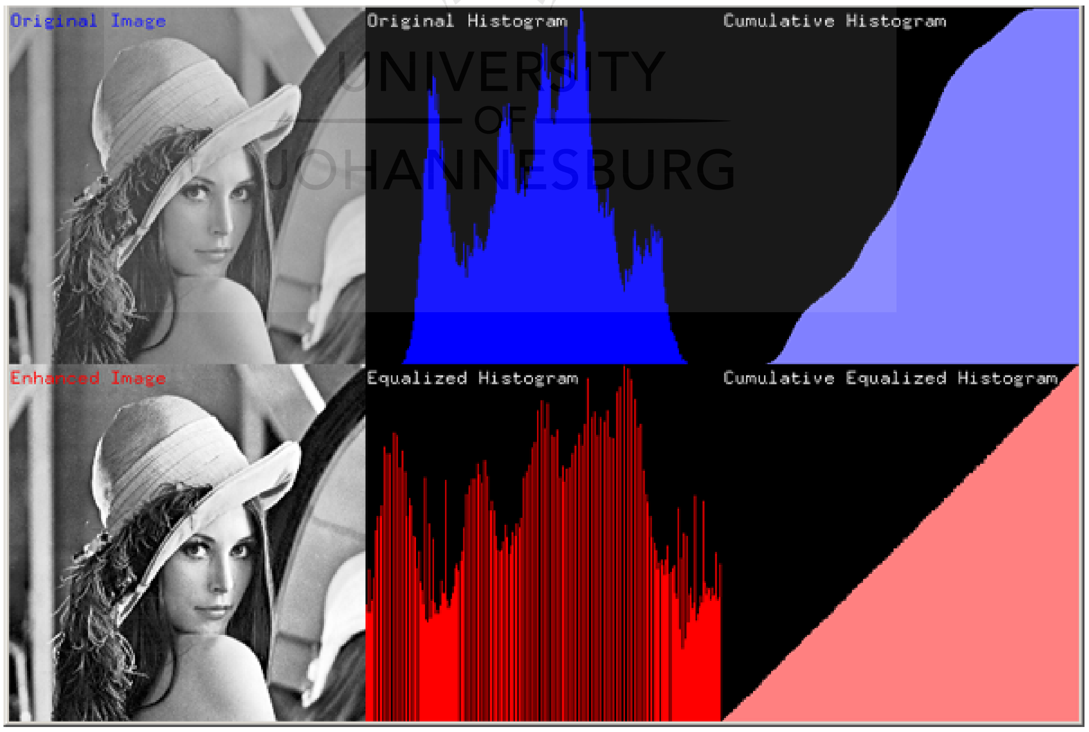

FIGURE 2.8: Image Enhancement using histogram equalization [21].

2.1.3.2 Local binary pattern

The LBP is well-known as a texture descriptor and is defined as follows [11]:

$$LBP_{N,R}(x_c, y_c) = \sum_{i=0}^{N-1} 2^i s(g_i - g_c) \tag{2.7}$$

$$s(x) = \begin{cases} 1, & x >= 0 \\ 0, & x < 0 \end{cases}$$

where g_c is the grey value of the centre pixel (x_c, y_c) and N is the number of neighbours to be compared. The neighbours are evenly distributed on a circle around the centre pixel with radius R (Figure 2.9). If a neighbour value does not fall exactly on a pixel it is estimated using bilinear interpolation.

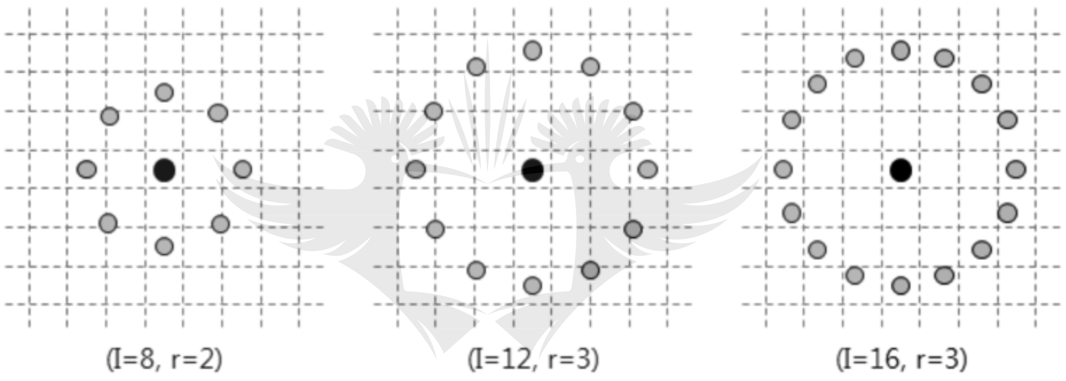

FIGURE 2.9: An illustration of how an LBP computed for different numbers of neighbours, N, and at different radii, R. $N = 8$ and $R = 2$ for the example on the left, $N = 12$ and $R = 3$ for the middle example and $N = 16$ and $R = 3$ for the example on the right [22].

The LBP is advantageous because it is tolerant of illumination changes. It employs the difference of pixel intensities relative to one another instead of to an absolute value; when the lighting changes in a scene this difference value is affected far less than if it were an intensity value alone. An illustration of how the LBP is calculated is provided in Figure 2.10.

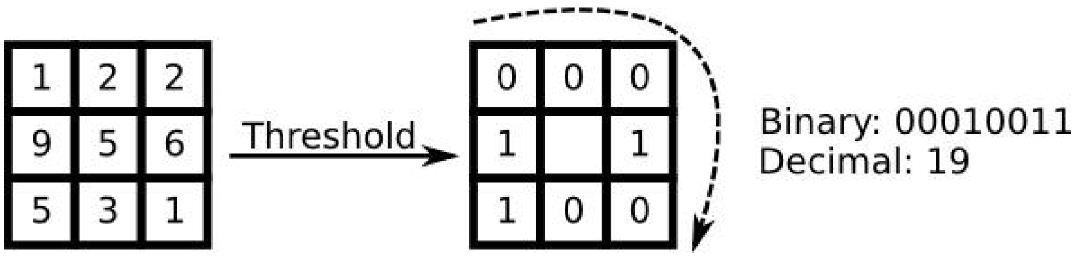

FIGURE 2.10: An illustration of how the LBP is calculated [23].

A variation on the classic LBP has been suggested by Zhou *et al.* [5] and is described by Equation 2.8; the pattern is modified to include colour information.

$$\begin{aligned} \mathrm{SCBP}_{2N,R}(x_c, y_c) = LBP_{N,R}(x_c, y_c) + 2^{N+1} f(R_c, G_c|\gamma) + \\ 2^{N+2} f(G_c, B_c|\gamma) + 2^{N+3} f(B_c, R_c|\gamma) \end{aligned} \tag{2.8}$$

$$f(a, b|\gamma) = \begin{cases} 1, & a > \gamma b \\ 0, & \text{otherwise} \end{cases} \tag{2.9}$$

where R_c, G_c and B_c are the three colour channels of the centre pixel (x_c, y_c) and $\gamma > 1$ is a noise suppression factor.

Furthermore, to simplify computation, Zhou *et al.* modify the pattern so that it does not compare the centre-pixel to its immediate neighbours, but rather that neighbours of the centre-pixel (across from one another) are compared (Figure 2.11). Also, one of the colour channels is excluded from the computation as the information present in these channels is highly correlative.

$$\mathrm{SCBP}_{2N,R}(x_c, y_c) = CS\_LBP_{2N,R}(x_c, y_c) + 2^{N+1} f(R_c, G_c|\gamma) + 2^{N+2} f(G_c, B_c|\gamma) \tag{2.10}$$

where the CS-LBP is defined as:

$$\mathrm{CS\_LBP}_{2N,R}(x_c, y_c) = \sum_{i=0}^{N-1} 2^i s(g_i - g_{i+N}) \tag{2.11}$$

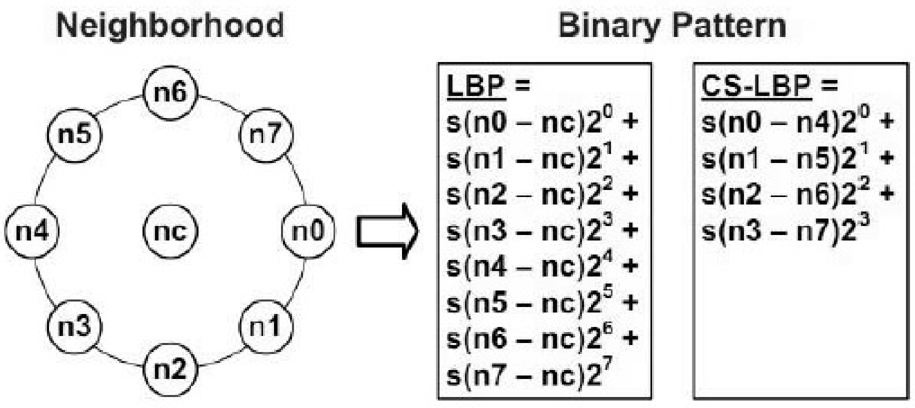

FIGURE 2.11: An illustration of how the LBP and SC-LBP is calculated [5].

2.1.3.3 Moments

A moment is a quantitative measure used to describe the shape of the distribution of a set of points [24]. For a probability distribution such as pixel intensity values, the zeroth moment corresponds to the total probability. The normalized first, second, third and fourth moments correspond to the mean, variance, skewness and kurtosis (flatness) of the probability distribution.

For a discrete two-dimensional probability distribution the k^{th} moment is defined by Equation 2.12.

$$\frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}X(i,j)^k \tag{2.12}$$

Where $X(i,j)$ is a sample of a probability distribution, X, at index (i,j).

2.1.4 Fuzzy logic

Critical situations at various stages of the background subtraction process give rise to uncertainties and imprecision. To account for this a number of fuzzy[1] methods have been proposed at each of these steps. Those pertinent to background modeling include fuzzy background modeling and fuzzy features. A background to fuzzy sets and logic can be found in Zadeh [25].

Fuzzy background modeling encompasses the use of a fuzzy running average or Type-2 fuzzy Mixture-of-Gaussians to model the background [26, 27]. Various fuzzy features are used to model the background such as the fuzzy correlogram or the fuzzy transformed co-occurrence vector.

2.1.5 Background subtraction

Background subtraction is implemented in order to distinguish foreground pixels from background pixels. It typically comprises four steps: Background initialization, background modeling, foreground detection and background maintenance. A flow chart illustrating the background subtraction process is provided in Figure 2.12 [28].

[1]Fuzzy logic is a form of logic where variables have truth values that are not only *true* or *false*, but can range between 0 and 1. It is well-suited to decision making based on approximate rather than exact thresholds.

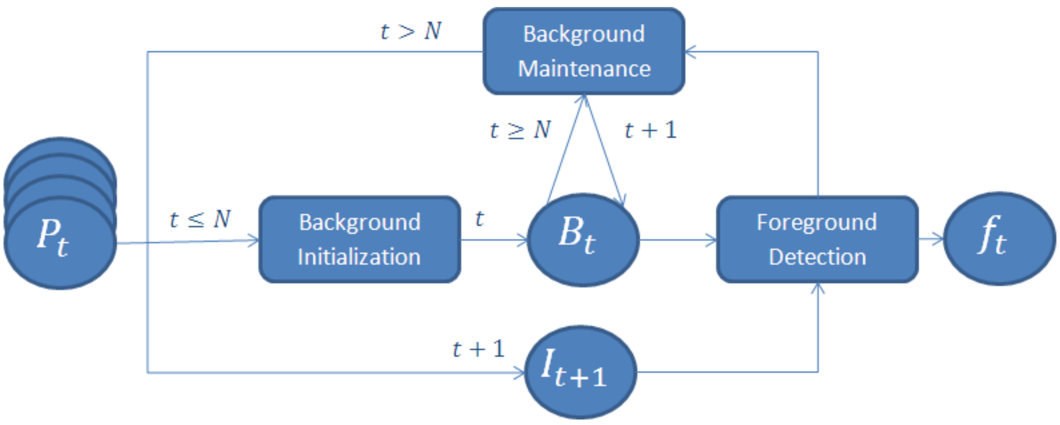

FIGURE 2.12: The background subtraction process [3]. N is the number of frames used for the background initialization, B_t and f_t is the background model and foreground mask at time t, respectively.

2.1.6 Foreground detection

Foreground detection involves the task of classifying a pixel as foreground or background. This is achieved through per-pixel multiplication of the input image and a foreground mask, $f_t(x, y)$. The result of this multiplication is known as a foreground image, $F_t(x, y)$:

$$F_t(x, y) = I_t(x, y) \circ f_t(x, y) \tag{2.13}$$

$$f_t(x, y) \subset \begin{cases} 1, & \text{if } \mathit{foreground} \\ 0, & \text{if } \mathit{background} \end{cases}$$

where $(x, y) \in \Omega$ at time t.

The foreground mask is usually determined using a difference frame, $D_t(x, y)$, of the current image, $I_t(x, y)$ and the background model image, $B_t(x, y)$ (Figure 2.13), which is defined as [3]:

$$D_t(x, y) = d(I_t(x, y), B_{t-1}(x, y)) = |I_t(x, y) - B_{t-1}(x, y)| \tag{2.14}$$

where $\{D_t(x, y), B_t(x, y)\} \in \Omega$. $d(a, b)$ need not necessarily be the absolute difference as in Equation 2.14; in fact any distance measure can be used.

Once a background model is initialized a difference image is taken and each pixel is classified as foreground or background depending on a threshold, T, as in Equation 2.15.

$$f_t(x, y) \subset \begin{cases} 1, & \text{if } D_t(x, y) > T \\ 0, & \mathit{otherwise} \end{cases} \tag{2.15}$$

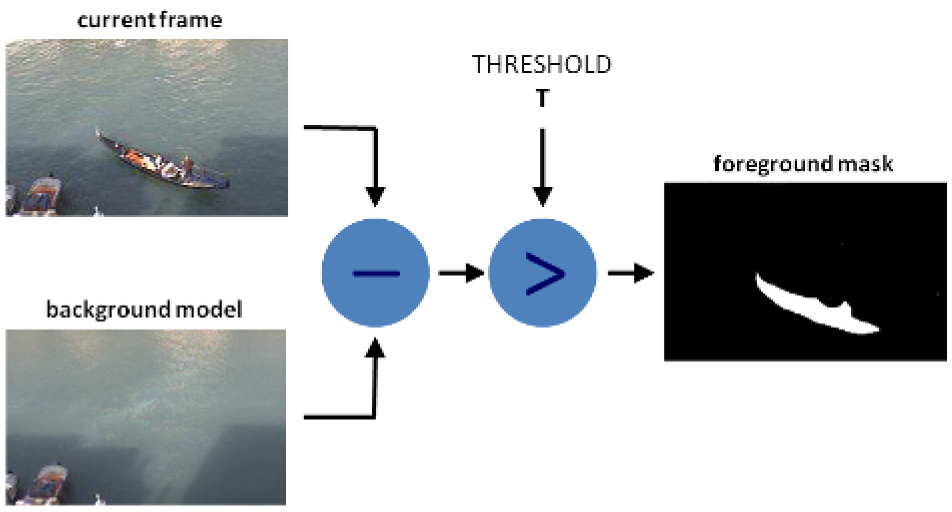

FIGURE 2.13: The frame-differencing process used to obtain a foreground mask [29].

T can be pre-defined or adaptively calculated. Such thresholding methods can typically be categorized into 5 groups: spatial, histogram shape-based, clustering-based, entropy-based and object-attribute-based [30]. Furthermore, a single global threshold or number of local thresholds can be computed such as for a region or for each pixel.

Spatial methods implement higher-order probability statistics to find correlations between pixels or regions.

Histogram shape-based methods analyze the peaks and valleys of a pixel intensity histogram to deduce a threshold. Otsu as well as Ridler and Calvard have proposed such methods [31, 32]. We discuss Ridler and Calvard's method in more detail.

2.1.6.1 Ridler and Calvard's method

A threshold, T_k, is computed using an automatic, iterative method similiar to the popular method proposed by Otsu [31]. It is computationally inexpensive but has the disadvantage of assuming that the scene is bimodal. This assumption predicts that there will be two distinct brightness regions in the image represented by two peaks in the grey-level histogram of the input image. These regions correspond to the object and its surroundings and so it is then reasonable to select the threshold as the grey-level halfway between these two peaks (Figure 2.14).

The histogram of the current frame, $I_t(x, y)$ is segmented into two parts using a threshold, $T_{iterate}$, which is first set to the middle value (127) of the range of intensities. For each iteration, the sample means of the foreground pixel intensities and the sample means of the background pixel intensities are calculated and a new threshold is determined as

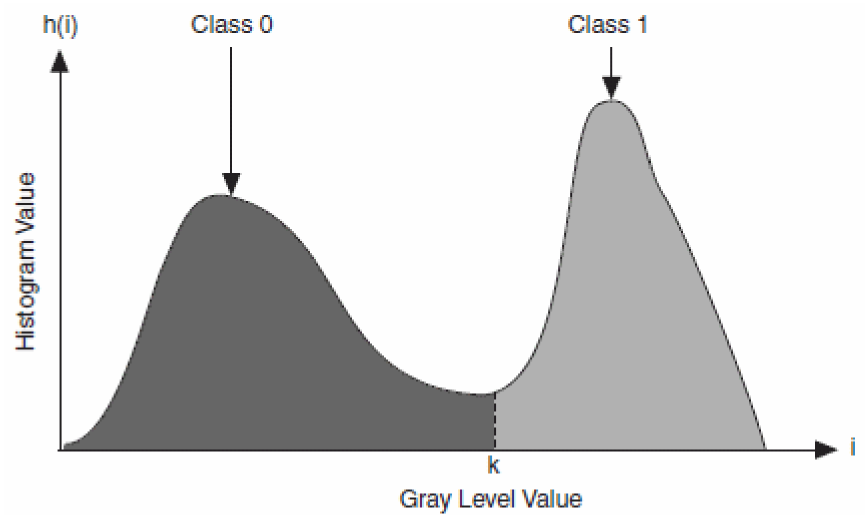

FIGURE 2.14: Selection of the threshold as the grey-level halfway between the two peaks of the histogram [33].

the average of these two means. The iterations stop once the threshold converges on a value, normally within about 4 iterations. The following formula describes this process:

$$T_{k+1} = \frac{\sum_{b=0}^{T_k} bn(b)}{2\sum_{b=0}^{T_k} n(b)} + \frac{\sum_{b=T_k+1}^{N} bn(b)}{2\sum_{b=T_k+1}^{N} n(b)} \tag{2.16}$$

where T_k is the threshold at the k^{th} iteration, b is the intensity value and $n(b)$ is the number of occurrences of the value b in the image such that $0 \leq b \leq N$.

2.1.7 Background maintenance

Background maintenance is necessary to ensure that a background model adapts to any changes that occur over time in a scene. This is an on-line learning process as is shown in Figure 2.12 and brings up a number of issues including learning rate, maintenance schemes and update frequency [28].

2.1.7.1 Learning rate

The learning rate specifies the speed at which the background model adapts to changes in a scene. A number of different methods for determining the learning rate have been proposed including: fixed, dynamic, statistical and fuzzy [34, 35]. For the scope of this dissertation the learning rates will be kept constant.

2.1.7.2 Maintenance scheme

There are three types of maintenance schemes: blind, selective, and fuzzy [3]. The blind update scheme is defined by the following formula:

$$B_{t+1}(x,y) = (1-\alpha)B_t(x,y) + \alpha I_t(x,y) \tag{2.17}$$

where α is a learning rate such that $\alpha \in [0,1]$.

The primary disadvantage of implementing this is that every pixel is updated in the same way; even foreground pixels are incorporated into the update which distorts the background model. The selective update scheme accounts for this using the following formulae [3]:

$$B_{t+1}(x,y) = \begin{cases} (1-\alpha)B_t(x,y) + \alpha I_t(x,y), & \text{if } f(x,y) \text{ is background} \\ (1-\beta)B_t(x,y) + \beta I_t(x,y), & \text{if } f(x,y) \text{ is foreground} \end{cases}$$

where β is is a learning rate such that $\beta \in [0,1]$.

β is chosen such that $\beta \gg \alpha$ so that the background model adapts quickly to a background pixel and slowly to a foreground one. β is often chosen to be 0. The disadvantage of doing this is that the selective update scheme is susceptible to false classifications which can lead to a permanently incorrect background model. This has given rise to the fuzzy adaptive update scheme which allows the update formula to be "graduated" based on the result of the foreground detection module [36].

2.1.7.3 Update frequency

Some authors favour not updating the background after every frame but rather only after significant changes have occurred [37, 38]. Tsai *et al.* explore a multi-model background maintenance framework which is influenced by dynamic and static pixels [39]. For the scope of this dissertation only a per-frame update frequency will be implemented.

2.1.8 Background initialization

The background initialization phase will only be briefly investigated; more complex initialization environments are beyond the scope of this dissertation.

Basic background initialization assumes that a sequence starts with a number, N, of training images that contain few to no moving objects. A mathematical process can then be performed on the training sequence. Some examples include the mean and median [40].

More advanced background initialization strategies allow for training sequences with scenes that have moving objects present. Temporal frame-differencing is one example of these types of initialization strategies.

2.1.8.1 Mean

The mean is very computationally inexpensive at the cost of producing a blurred background model. Furthermore, it is not at all robust to noise; its error is largely affected by the percentage and distribution of the noise [41]. It is defined as [40]:

$$B_t(x,y) = \frac{\sum_{t=0}^{N-1} I_t(x,y)}{N} \tag{2.18}$$

2.1.8.2 Median

The first N training frames are used to determine the temporal median from the input video. For any real probability distribution a median is defined as any real number m that satisfies the following inequalities [24]:

$$P(X \leq m) \geq \frac{1}{2} \quad \text{and} \quad P(X \geq m) \geq \frac{1}{2} \tag{2.19}$$

An advantage of using a median as opposed to an averaging method is that the resulting initialized background is more robust to outliers provided that the noise occupies less than 50% of the data [41]. This is typically achieved by first sorting the values using a selection sort or insertion sort and then selecting the middle value from the distribution. The insertion sort is less computationally expensive; it is $O(k)$ at its best, compared to the selection sort which is $O(n^2)$ at its best.

2.1.8.3 Temporal frame-differencing (persisting pixel)

Temporal frame-differencing avoids the assumption that a frame sequence starts with the absence of foreground objects [42]. Two different sets of frame-differencing formulae are used depending on a pixel's stability. Using Equations 2.20 and 2.21, a pixel is labeled as *stable* if it is persistently classified as part of the background for N_{stable} frames.

$$FD_t(x,y) = |I_t(x,y) - I_{t-1}(x,y)| \tag{2.20}$$

$$I(x,y) \subset \begin{cases} \text{foreground,} & \text{if } FD_t(x,y) > T \\ \text{background,} & \text{otherwise} \end{cases} \tag{2.21}$$

It is assumed that every pixel of the background will eventually be uncovered. Once classified as stable the initialized background pixel value is taken as the average of the N_{stable} pixels, which were persistently classified as part of the background, and the usual frame-differencing formulae, Equations 2.14 and 2.15, are used perform foreground detection.

2.1.9 Background modeling

Bouwmans [3] provides an excellent literature survey of more than 300 traditional and recent background modeling techniques. Using this as a reference, at least two example techniques of each sub-category will be discussed briefly.

2.1.9.1 Basic

These methods typically make use of a single mathematical process to model the background. Methods such as these prove to be too simple for dynamic backgrounds such as those with sudden illumination changes. Examples include the median, average and histogram analysis [43, 44, 45]. The median and average are computed similarly to equations 2.19 and 2.18 respectively.

2.1.9.2 Statistical

Statistical processes are used to distinguish background pixels from foreground pixels. These provide a good framework to construct a background model that is robust to dynamic backgrounds and numerous techniques have been developed. These can be further categorized into Gaussian, support vector and subspace learning models [46].

Gaussian models assume that a Gaussian function can be used to represent the history of the intensity values of a pixel. Background intensity values can then be predicted. More information regarding how this function is used is provided in Section 2.1.9.5. The Gaussian function is the probability distribution function (PDF) of a normally distributed random variable. It is defined as follows:

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{2.22}$$

where μ is the expected value and σ is the variance of the distribution.

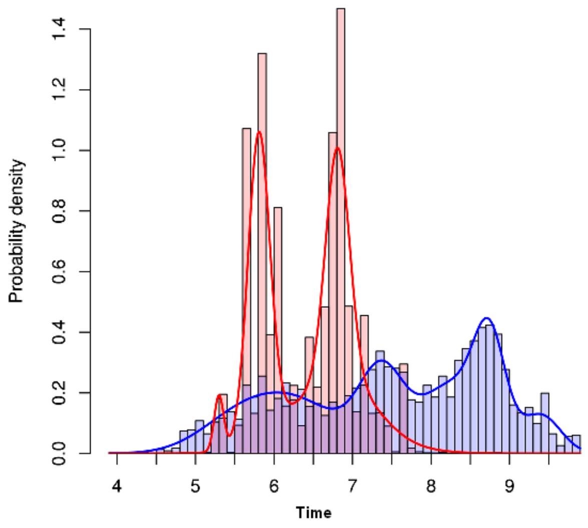

FIGURE 2.15: A mixture of Gaussian distributions can be employed to represent the history of pixel intensity values [47].

A number of variations have been proposed including a single Gaussian, Gaussian mixture model (GMM) (see Figure 2.15) and mixture of general Gaussians [6, 48, 49]. The choice of of the kernel function is crucial to its predictive performance. In order to exploit this authors have either employed parameter learning techniques or multiple models with different parameter values in order to improve classification accuracy [50]. Some authors have combined a Gaussian model with other approaches such as neural networks, kernel density estimation (KDE) and Markov random fields [51, 52, 53]. Techniques such as KDE model the history of pixel values to estimate the background pixel's values. Then, if any pixel in the current image deviates significantly from its predicted value it is classified as part of the foreground. A KDE estimator typically takes the following form:

$$p_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - x_i}{h}) \tag{2.23}$$

where $p_h(x)$ is the probability that the pixel belongs to the background. K is a scaled kernel (like the Gaussian mentioned in section 2.1.9.2) and h is the kernel bandwidth. n is the number of samples that are considered.

The bandwidth value, h, allows for a trade-off between false positives and false negatives. When h is too small, noisy estimates are produced and it is possible to misclassify a pixel as part of the foreground. Alternatively, if the bandwidth is too large the kernel is over-smoothed and it is possible to misclassify a pixel as part of the background. Figure 2.16 shows the effect of different bandwidth values for a PDF.

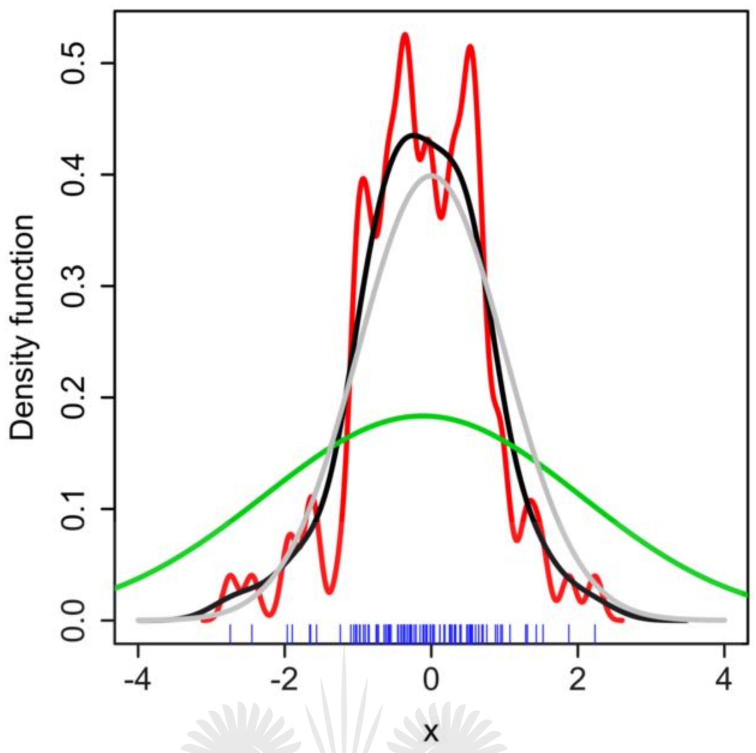

FIGURE 2.16: KDE with different bandwidths. Grey corresponds to the true density of a standard normal distribution. Red, black and green correspond to a KDE with h=0.05, 0.337 and 2 respectively [54].

The estimation is also influenced by n. The larger n is, the more closely the kernel is approximated, and the smaller the bandwidth will have to be to avoid over-smoothing. If there are few samples and h is too small, noisy estimates may result.

Support vector models implement sophisticated statistical models such as support vector machines (SVM), support vector regression (SVR) and support vector data description (SVDD) to model a background [55, 56, 57].

SVM's are similar to two-layer neural networks (see section 2.1.9.4). They makes use of an optical flow value[2] and inter-frame difference as background features to optimally separates data into two categories through the construction of an n-dimensional hyperplane [55]. Feature vectors located near the hyperplane are known as support vectors, which comprise the background model. The model parameters influence the accuracy of the model as shown in Figure 2.17.

SVR models each background pixel as a function of intensity [56]. SVDD does not model the background using a probability function but rather as an analytical description of

[2]Optical flow is the distribution of the apparent velocities of objects between two frames in a video sequence.

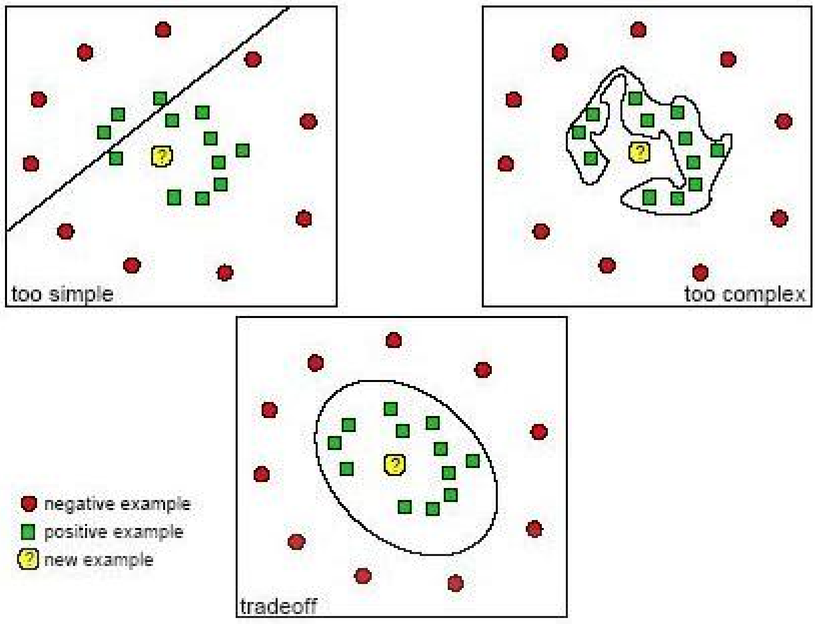

FIGURE 2.17: Under-fitting and over-fitting is dependant upon the selection of the model parameters [58].

the decision boundary used to classify background and foreground pixels. This has the advantage of the model not being bounded by the accuracy of the estimated PDF [57].

2.1.9.3 Cluster

The cluster model approach is a type of vector quantization and is based on the premise that each pixel in a frame can be temporally represented by clusters of features or tokens derived from the input image. A pixel is added to a cluster with the highest similarity. This similarity can be measured using a distance measure between points in the feature space (see Figure 2.18). Examples of cluster models include K-means, Codebook and basic sequential [7, 59, 60].

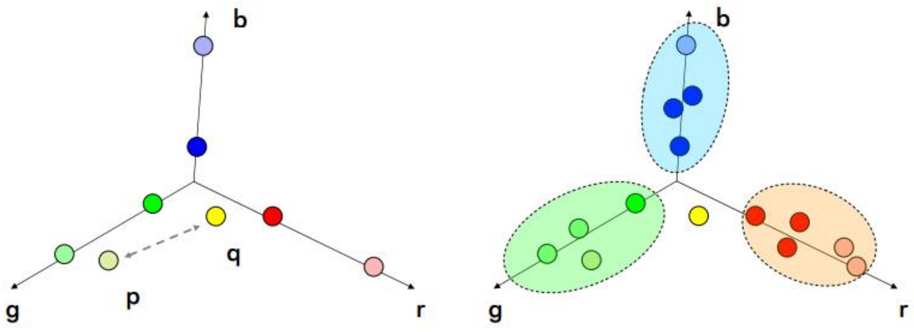

FIGURE 2.18: Background modeling using clusters of features.

```
Data: $k$ datapoints, $i$ clusters
Result: Allocated cluster centres
while Cluster centres have been modified do
    read kth datapoint;
    assign kth datapoint to cluster whose centre is nearest ;
    if ith cluster is empty then
        assign a random datapoint to empty cluster;
    else
        Replace the ith cluster with the mean of the datapoints in the cluster;
    end
end
```

Algorithm 1: The k-means clustering algorithm [61].

The K-means models assign a group of clusters to each pixel in a frame. The model adapts to illumination and background variations by ordering clusters according to how well they represent the background. A fixed number of clusters are chosen. Cluster-centres and point-cluster allocations are chosen in such a way as to minimize error [59]. Since there are too many possibilities, these are determined by means of K-means clustering (Algorithm 1). New pixels are classified according to whether or not their corresponding cluster is part of the background.

An advantage of k-means models is that they are computationally simple while a disadvantage is that they are sensitive to initialization and statistical outliers.

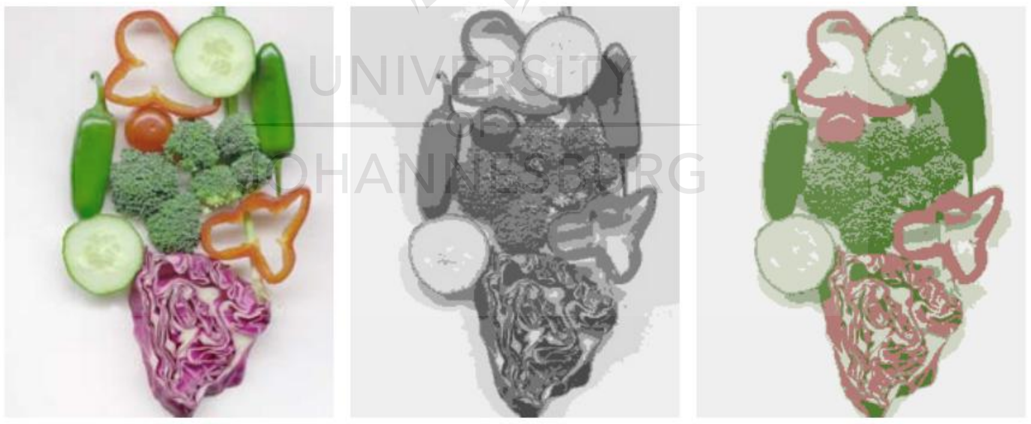

FIGURE 2.19: K-means clustering using: only intensity (middle) and only colour (right).

Codebook models implement the per-pixel construction of codewords that comprise a codebook [7]. These codewords are background features, such as brightness boundaries and colour distortion metrics, and represent a compressed form of the background model. The number of codewords for a pixel is a function of the behaviour of the pixel. A pixel is classified as foreground if its colour distortion is less than a specified threshold and if its brightness lies within a range specific to the pixel. The compressive nature of the codebook model ensure inexpensive computations.

Advantages of Codebook-based background modeling include better handing of local and global illumination changes as well as a reduction in artifacts that have resulted from image acquisition, digitization and compression (Figure 2.20).

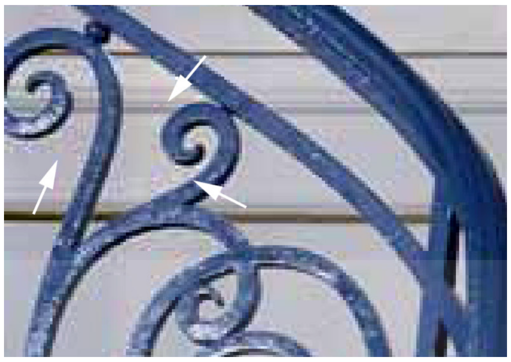

FIGURE 2.20: An example of block artifacts that have resulted from image compression.

A number of improvements on this original approach have been proposed that incorporate different colour models as well as multi-layer, block-based, multi-scale and hierarchical approaches [62, 63, 64, 65].

2.1.9.4 Neural networks

The neural network approach uses what are known as "training patterns" to estimate statistical parameters [1]. A set of patterns, called a "training set", is used to obtain a decision function through a process called learning. The decision function is employed to classify a pixel or region as background or foreground. A model is constructed with the means of the weights of a neural network that have been trained on a number of clean frames. Weights are assigned and background features are used to compute pattern vectors. These weights and vectors are then fed into the decision function. The decision function is defined by equation 2.24 and illustrated by Figure 2.21.

$$d(\bar{x}) = \sum_{i=1}^{n} w_i x_i + w_{n+1} \tag{2.24}$$

where w_i is the weight and x_i is the pattern vector at index i.

If $d(\bar{x})$ exceeds a specified threshold the pixel or region is classified as part of the background. Figure 2.22 shows a number of training patterns used to recognize a specific shape. Noisy versions of these shapes are included to make the model more robust.

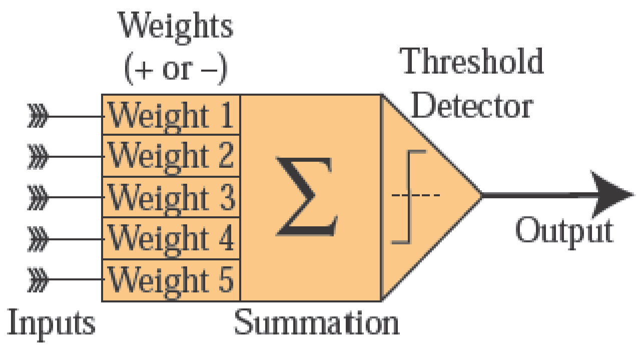

FIGURE 2.21: An illustration of the decision function or "threshold logic unit"[66]. For the sake of simplicity the w_{n+1} term (see Equation 2.24) is not included.

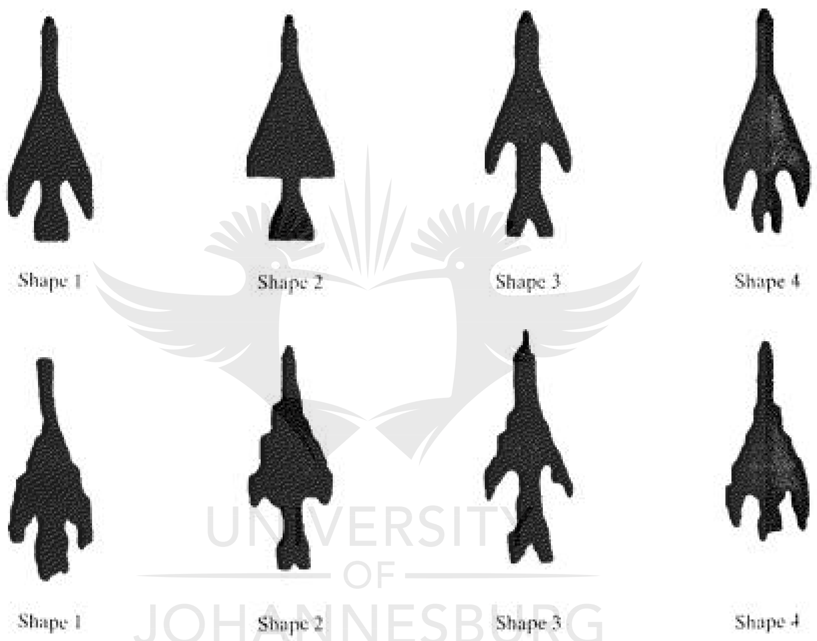

FIGURE 2.22: A variety of shapes used as training data [1].

Examples of neural networks that have been applied to background modeling include general regression, multivalued, competitive, dipolar competitive, self-organizing and growing hierarchical self-organizing neural networks [67, 68, 69, 70, 71, 72].

General regression neural networks implement an unsupervised Bayesian classifier in a neural network architecture to model the background [67]. The weights of the network aid in updating and maintaining the background model effectively.

Multivalued neural networks are able to detect and correct some of the errors that are produced by the mixture of Gaussians algorithm [68]. This type of network is able to represent non-numerical states which is particularly useful when designating qualitative labels such as "foreground", "background" or "shadow".

2.1.9.5 Linear systems theory: Estimation

Linear systems theory, a subset of control theory, describes linear functions that can be used to estimate a discrete-time signal [2]. When applied to background modeling the history of pixel values is used to estimate the background pixel's values. A number of estimators have also been explored such as the Wiener, Kalman and Chebychev filter [73, 74, 75]. However, they are well-suited to handle only gradual illumination changes and will not be explored in much detail here [3].

The Wiener filter is a linear minimum mean square error (MMSE) estimator that is typically used to remove unwanted noise from a signal (Figure 2.23). It assumes that an image is a sample from a correlated Gaussian random noise field combined with a statistical model of the measurement process [73]. Its main advantage is that it reduces the uncertainty of a pixel value by accounting for the pixel time variation. A disadvantage is that the history of values is susceptible to corruption by moving objects.

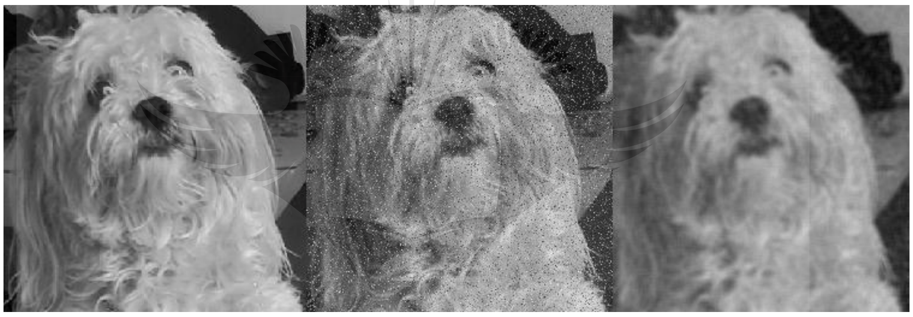

FIGURE 2.23: The left image is the original and the middle image is the original to which white noise has been added. The right image is the result of a Wiener filter applied to the middle image [76].

The Kalman filter is an optimal estimator of the state of processes. It is based on the premise that a process can be modeled by a linear system and that the process and measurement noise is white and has a zero-mean Gaussian distribution [74]. The Kalman filter is keeps track of the estimated state of the system and the variance of the estimate. The estimate is updated using measurements and a state transition model. Equation 2.25 describes the Kalman filter.

$$\bar{E}_k = K_k Z_k + (1 - K_k)\bar{E}_{k-1} \tag{2.25}$$

where $\bar{E}_k$ is the current estimation, K_k is the Kalman gain, Z_k is the measured value and $\bar{E}_{k-1}$ is the previous estimate.

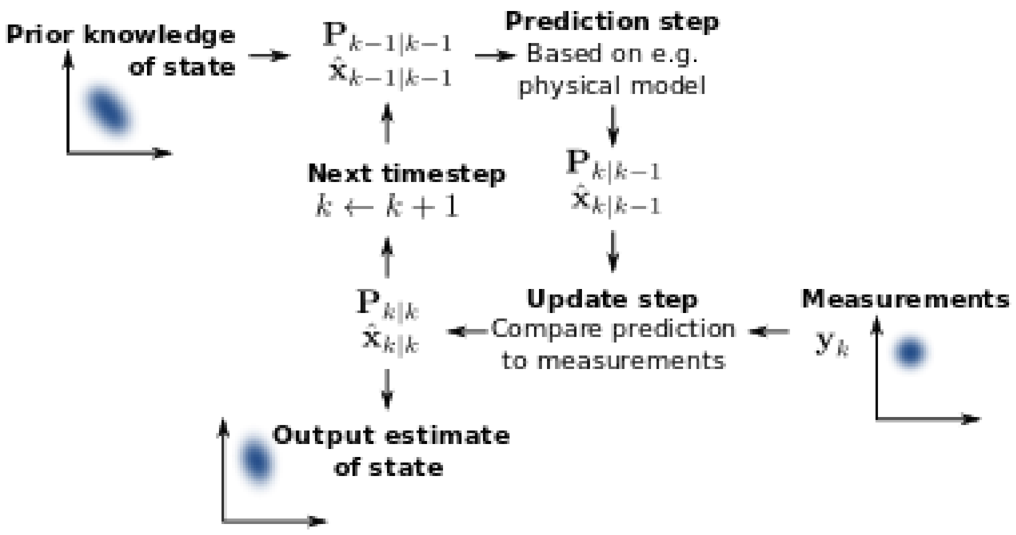
Prior knowledge of state
$\mathbf{P}_{k-1|k-1}$ $\hat{\mathbf{x}}_{k-1|k-1}$
Prediction step
Based on e.g. physical model
Next timestep
$k \leftarrow k+1$
$\mathbf{P}_{k|k-1}$ $\hat{\mathbf{x}}_{k|k-1}$
$\mathbf{P}_{k|k}$ $\hat{\mathbf{x}}_{k|k}$
Update step
Compare prediction to measurements
Measurements
$\mathbf{y}_k$
Output estimate of state

FIGURE 2.24: An overview of the Kalman filtering process [77].

2.1.9.6 Advanced statistical models

Complex statistical models are used to distinguish background from foreground. These include mixture, hybrid, non-parametric and multi-kernel models [74, 78, 79, 80]. Mixture models are very similar to Gaussian mixture models but make use of a distribution other than a Gaussian. Examples include Student-t and Dirichlet mixture models [56, 78].

The Dirichlet PDF is defined by the following equation:

$$n(\bar{x}) = Z \prod_i x_i^{\alpha_i - 1} \tag{2.26}$$

where Z is a constant chosen so that the integral of $n(\bar{x})$ is 1 and $\bar{\alpha}$ is a Dirichlet distribution.

Multi-kernel models incorporate different colour representations and multiple kernel representations to construct an enhanced feature space [80]. A grouping-based algorithm is then implemented on the space to classify pixels as background or foreground.

2.1.9.7 Discriminative and mixed subspace learning

There are two types of subspace learning methods: reconstructive [81] and discriminative [82]. A reconstructive approach considers the variability of the training data it has gathered while trying to be as informative as possible regarding the approximation of the original data. This means that reconstructive representations are not task-dependent.

On the contrary, a discriminative approach is task-dependent. Furthermore, it is computationally and spatially far more efficient and often provides better classifications than a reconstructive approach. Ultimately, the only advantage of reconstructive subspace learning is that it allows for unsupervised learning. Therefore, most authors implementing these types of subspace learning make use of discriminative or mixed subspace models.

The Incremental Maximum Margin Criterion (IMMC) [83] is a type of discriminative subspace learning algorithm that derives a subspace from sequential data samples. The model is updated incrementing the eigenvectors of the criterion matrix.

An example of a mixed subspace model is the one proposed by Marghes *et al.* [82]. They combine principle component analysis (PCA) (reconstructive) with Linear Discriminant Analysis (LDA) (discriminative). PCA is used to model the primary distribution of the pixel values and LDA is used to distinguish background pixels from foreground pixels.

2.1.9.8 Robust subspace models

A robust subspace model distinguished background pixels from foreground pixels using low-rank and sparse decomposition. This is achieved using Robust Principal Components Analysis (RPCA), Robust Non-negative Matrix Factorization (RNMF) or Robust Orthonormal Subspace Learning (ROSL) [83, 84, 85, 86, 87].

RPCA is the most popular approach; it implements Principal Component Pursuit (PCP) to decompose a data matrix, A, and a sparse noise matrix, S, such that a data matrix can be described as:

$$A = L + S \tag{2.27}$$

Using A as the training sequence, foreground objects are represented by the correlated sparse outliers (S) and background pixels are modeled by the low-rank subspace (L).

RPCA via Sparsity Control implements a tunable parameter that varies the sparsity of the estimated matrix and hence, the number of outliers [88].

Bayesian RPCA derives an approximate representation of the noise present and also infers the sparse and low-rank components of the subspace model [74].

2.1.9.9 Subspace tracking

He *et al.* [89] propose an algorithm called Grassmanian Robust Adaptive Subspace Tracking Algorithm (GRASTA) which implements a Grassmanian manifold and a robust l_1-norm cost function. This function tracks and estimates non-stationary subspaces allowing for the on-line classification of background and foreground pixels. A number of improvements of this algorithm have been proposed including t-GRASTA (transformed-GRASTA) [90], pROST (l_p-norm Robust Subspace Tracking) [91] and GOSUS (Grassmanian Online Subspace Updates with Structured-sparsity) [92].

It is assumed that a video frame can be represented by the Equation 2.28.

$$\bar{V}_t(x,y) = \bar{U}_t\bar{w}_t(x,y) + \bar{s}_t(x,y) + Z_t(\bar{x},y) \tag{2.28}$$

where $\bar{U}_t$ is the subspace model vector, $\bar{w}_t$ is the weight vector, $\bar{s}_t$ is the sparse outlier vector and $\bar{Z}_t$ is the zero-mean Gaussian white noise vector.

pROST implements a smoothed l_p-norm which outperforms GRASTA in regard to multi-modal backgrounds [91].

GOSUS improves the accuracy of on-line subspace maintenance by incorporating a meaningful structured sparsity term [92].

2.1.9.10 Low Rank Minimization

Low Rank Minimization (LRM) is a very effective approach to data-mining, however the presence of outliers negatively affects its performance. Recently, an algorithm for robust matrix factorization has been introduced that makes LRM robust to outliers. LRM can now be formulated as a matrix approximation problem and can incorporate structural information of outliers (such as foreground objects) in order to find them more effectively. The approaches that have been developed include Contiguous Outliers Detection (COD), Direct Robust Matrix Factorization (DRMF), Direct Robust Matrix Factorization-Row, Probabilistic Robust Matrix Factorization and Bayesian Robust Matrix Factorization [92, 93, 94, 95].

COD estimates the low-rank and outlier support matrix using a unified framework known as DEtecting Contiguous Outliers in the LOw-rank Representation (DECOLOR) [92]. Assuming that the underlying background images in a video sequence are linearly correlated, the matrix of frames can be approximated using a low-rank matrix. Objects in motion can then be detected as outliers in the low-rank representation.

An advantage of formulating the problem in this way is that many of the assumptions regarding the behaviour of the foreground are unnecessary. Furthermore, this representation is more tolerant of global background variations. Another advantage is that DECOLOR does not require a training sequence.

DRMF is based on the assumption that a small portion of the matrix has been corrupted by arbitrary outliers [93]. The estimated model excludes outliers in order to get a reliable estimation of the true low-rank structure of the matrix. This is what distinguishes DRMF from conventional LRM.

2.1.9.11 Sparse models

There are a number of sparse model categories including structure sparsity (SS), dynamic group sparsity and dictionary models [96, 97, 98].

The SS approach is a natural extension of the standard sparsity concept in compressive sensing and statistical learning. It considers the model as an optimization problem which can be solved using techniques such as the one proposed by Cui *et al.* [96, 99]. An overview of this technique is provided in Figure 2.25.

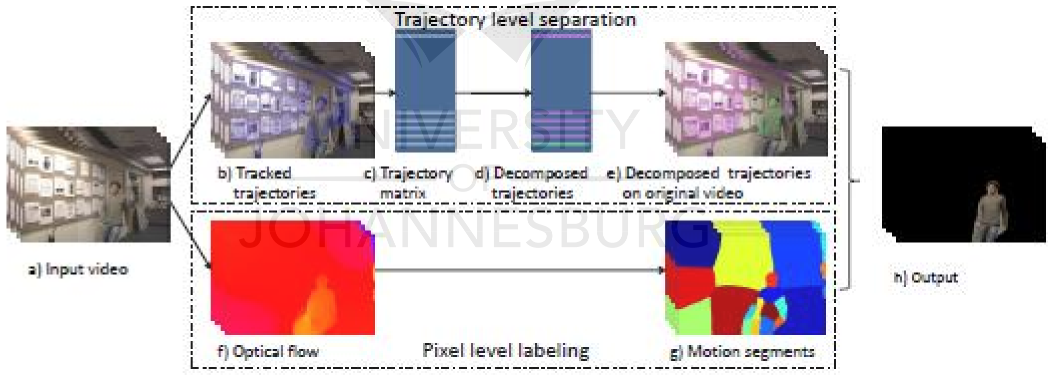

FIGURE 2.25: Overview of the framework implemented by Cui *et al.* [99].

Dictionary learning assumes that the current image can be sparsely represented as a linear combination of vectors, which comprises a dictionary [3]. A static background can then be decomposed into a foreground image and the sum of static background images. The foreground image is estimated as a sparse error; the dynamic foreground and static background can be modeled as signal samples that vary slowly in time with sparse corruption. Each "atom" in the dictionary represents a variation in the background model, which is learned from training frames [100].

2.1.9.12 Domain transform models

The premise of this approach is that background pixels can be better distinguished from foreground pixels by approaching the problem in a different domain. Various different transforms have been implemented including the Fast Fourier (FFT), Discrete Cosine (DCT), Walsh, Wavelet and Hadamard Transform [37, 101, 102, 103, 104], however, these techniques are typically too computationally expensive for real-time applications.

The FFT approach models multi-modal backgrounds as a number of spectral signatures. Signature inconsistencies are then used to detect changes in a scene.

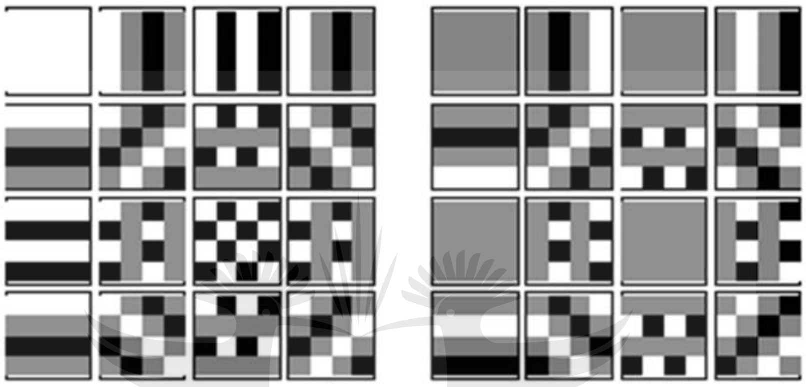

FIGURE 2.26: Illustration of the real (left set) and imaginary (right set) steps in the FFT transform [105] (top to bottom, left to right).

The Walsh Transform employs the mixture of Gaussian approach applied to multiple block sizes to model the background. This is achieved by using directional coefficients to derive the feature parameters of the Walsh Transform (Figure 2.27). These coefficients have a strong spatial correlation. An advantage of the Walsh transform is that it is computationally cheaper to perform than the discrete FFT and DCT.

2.2 Illumination change concepts

For the scope of this dissertation we only consider global illumination changes.

2.2.1 Sudden illumination changes

We define a sudden illumination change as a drastic change in illumination occurring within a single frame of a video sequence. These are caused by lights switching on or off, varying cloud cover and curtains opening or closing.

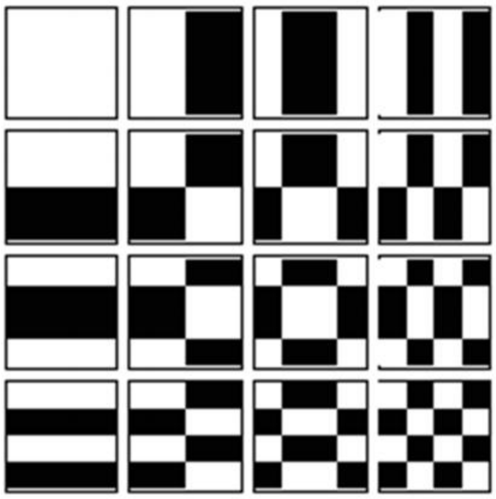

FIGURE 2.27: Illustration of the steps in the Walsh transform [105] (top to bottom, left to right).

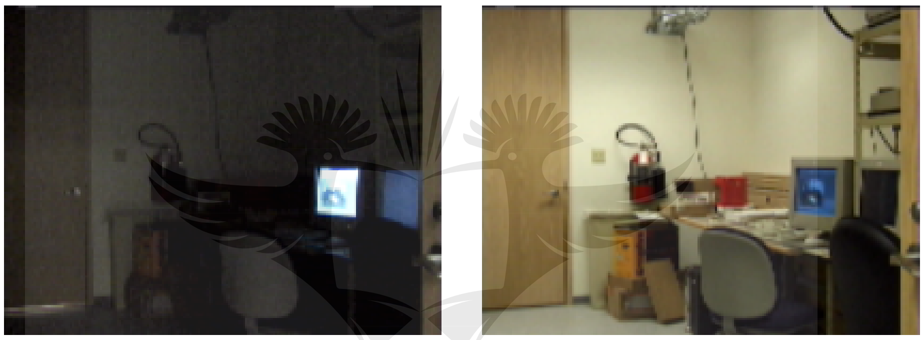

FIGURE 2.28: Sudden Illumination change example [73]. The left image shows the initial scene while the right image shows the same scene a single frame later.

2.2.2 Gradual illumination changes

We define a gradual illumination change as a steady change in illumination occurring over no more than 60 seconds of a video sequence. These are typical in outdoor environments and caused by the trajectory of the sun or cloud movement.

2.2.3 Phong shading model

The Phong shading model is a common model used to describe illumination [2]. It is based on the shading model used by computer graphics which assumes that a pixel intensity can be decomposed into an illumination value, L, and a shading coefficient, S [9]:

$$I(x,y) = L.S(x,y) \tag{2.29}$$

The shading coefficient is uniquely defined depending upon the reflectance of the surface material and the actual physical structure of the object.

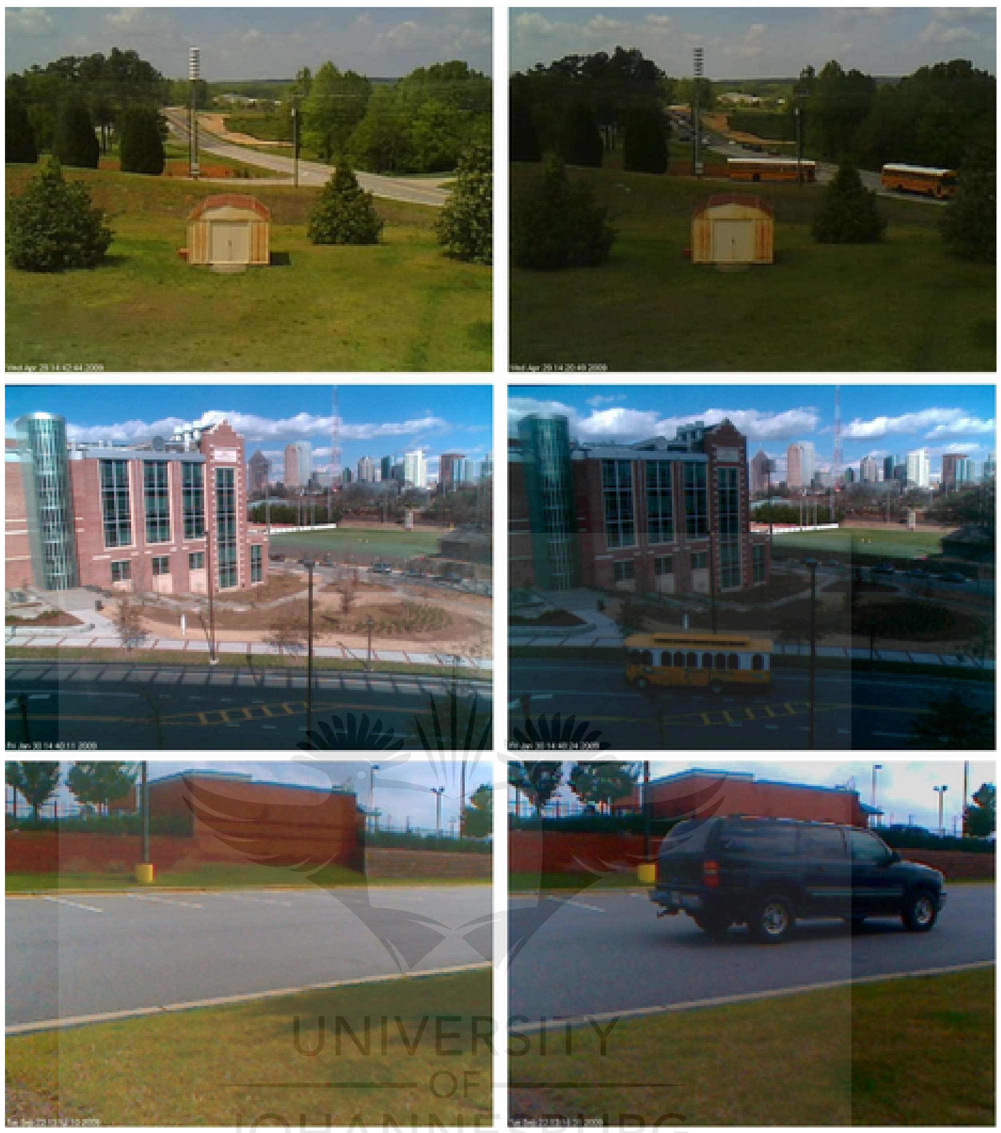

FIGURE 2.29: Gradual illumination change examples [106]. The left column shows the initial scene while the right column shows the same scene moments later.

The Phong shading model is described by the following equation [2]:

$$S(x, y) = C(x, y)[cos(i)(1 - d) + d] + W(i)[cos(s)]^n \tag{2.30}$$

where C_p is the reflection coefficient of the object located at point P for a specific wavelength, i is the incident angle, d is the environmental diffuse reflection coefficient and $W(i)$ is a function that provides the ratio of the specular reflected light and the incident light as a function of the incident angle. Parameter s is the angle between the direction of the reflected light and the line of sight. n is a power value which, for each material, models the specular reflected light. Figure 2.30 illustrates some of the terms used.

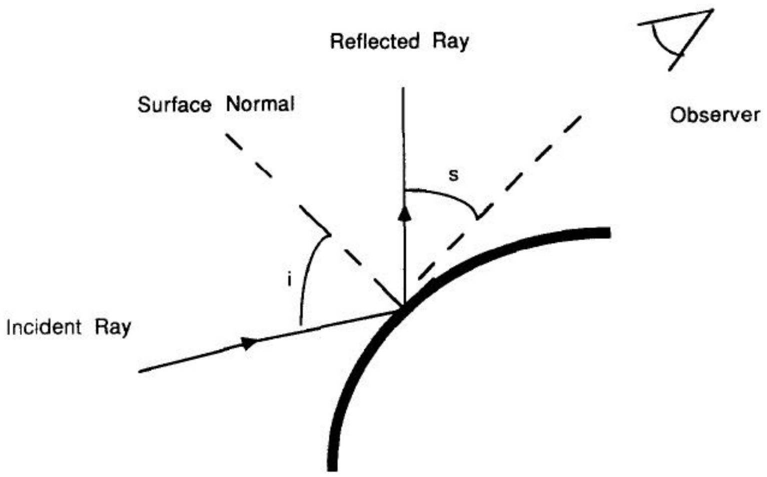

FIGURE 2.30: Light striking the surface of an object [9].

2.3 Binary morphology

Morphological operators provide a non-linear approach to modify the shape of geometrically related sets, as well as to reduce the amount of noise present, in a binary image. Traditional approaches to solving these tasks such as the use of linear systems are not as well-suited as morphological operators since they do not exploit the geometric aspects of an image [107]. The morphological concepts that will be discussed include erosion, dilation and structuring elements as well as two processes that make use of them, namely, "opening" and "closing".

2.3.1 Erosion

Each "1" pixel that is adjacent to a "0" pixel is changed into a "0" pixel. This is defined as follows [40]:

$$f(x, y) = \min\{I(x, y) \text{ and its neighbouring pixels}\} \tag{2.31}$$

2.3.2 Dilation

Each "0" pixel that is adjacent to an "1" pixel is changed into an "1" pixel. This is defined as follows [40]:

$$g(x, y) = \max\{I(x, y) \text{ and its neighbouring pixels}\} \tag{2.32}$$

2.3.3 Structuring element

The structuring element is a shape mask which determines which neighbouring pixels are considered in an erosion or dilation operation. It can be any shape or size provided it can be represented digitally and has an origin (like a box, disk or hexagon). The choice of structuring element affects the way in which a region grows or shrinks and the preservation of object contours. Examples of some structuring elements are provided in Figure 2.31.

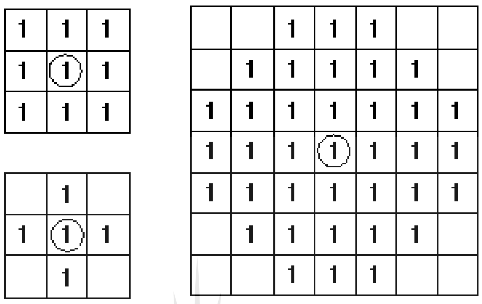

FIGURE 2.31: A few examples of structuring elements [108].

2.3.4 Opening

Opening is an erosion operation followed by a dilation operation. It is typically used to remove small isolated object pixels and to smooth an object's boundary without changing its shape or area (Figure). Opening is defined by Equation 2.33 [40]:

$$\text{Opening}(I(x,y)) = \max\{\min\{I(x,y) \text{ and neighbouring pixels}\} \text{ and neighbouring pixels}\} \tag{2.33}$$

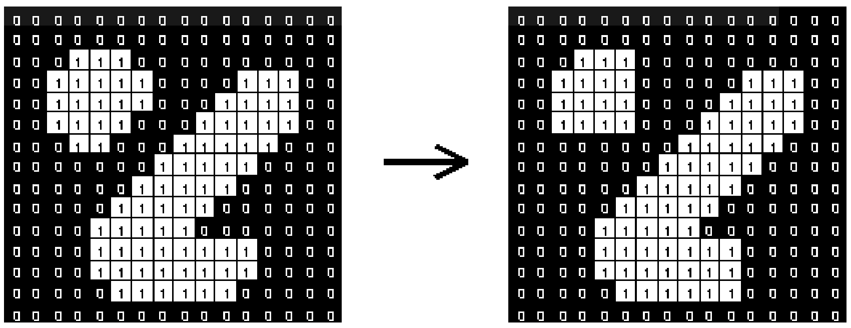

FIGURE 2.32: The effect of using a 3 × 3 square structuring element to perform an opening operation [109].

2.3.5 Closing

Closing is a dilation operation followed by an erosion operation. It is typically used to close small holes within an object without changing its shape or area (Figure). Closing is defined by Equation 2.34 [40].

$$\text{Closing}(I(x,y)) = \min\{\max\{I(x,y) \text{ and neighbouring pixels}\} \text{ and neighbouring pixels}\} \tag{2.34}$$

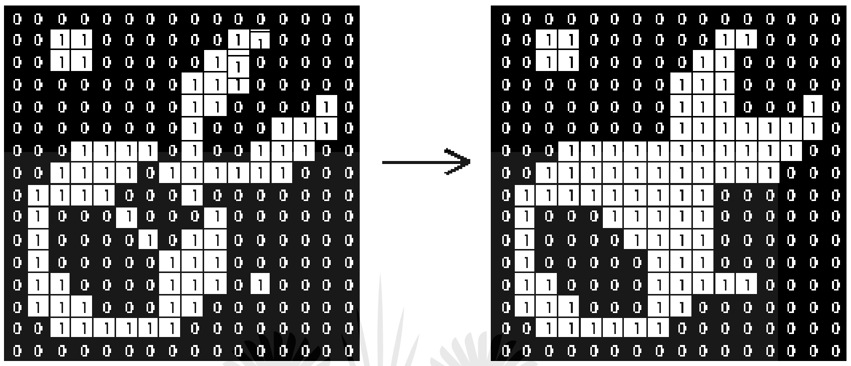

FIGURE 2.33: The effect of using a 3 × 3 square structuring element to perform a closing operation [110].

2.4 Review of sudden illumination change literature

Real-world scenes often contain dynamic backgrounds such as swaying trees, rippling water, illumination changes and noise. While a number of techniques are effective at handling these, sudden illumination changes such as a light source switching on/off or curtains opening/closing continue to be a challenging problem for background subtraction [42]. In recent years a number of new segmentation techniques have been developed to handle sudden illumination changes.

A number of texture-based methods have developed to solve the problem of sudden illumination changes. Heikkila *et al.*, Xie *et al.* and Pilet *et al.* make use of robust texture features [11, 36, 111, 112]. Heikkila *et al.* make use of local binary pattern (LBP) histograms as background statistics. Xie *et al.* assumes that pixel order values in local neighbourhoods are preserved in the presence of sudden illumination changes. They provide an output image by classifying each pixel by its probability of order consistency. Pilet *et al.* make use of texture and colour ratios to model the background and segment the foreground using an expectation-maximization framework. Texture-based features work well, but only in scenes with sufficient texture; texture-less objects prove to be a difficulty.

Another way of dealing with sudden illumination changes is to maintain a representative set of background models [36]. These record the appearance of the background under different lighting conditions and alternate between these models depending on observation. The techniques that make use of this approach mostly differ in their method of deciding which model should be used for the current observation. Toyama *et al.* [73], implement the Wallflower system which chooses the model as the one that produces the lowest number of foreground pixels. This proves to be an unreliable criterion for real-world scenes. Stenger *et al.* [113] make use of hidden Markov models but in most cases sharp changes occur without any discernible pattern. Also, Stenger *et al.* and Toyama *et al.* require off-line training procedures and consequently cannot incorporate new real-world scenes into their models during run-time [114]. Sun and Yuan [115] implements a hierarchical Gaussian Mixture Model (GMM) in a top-down pyramid structure. At each scale-level a mean pixel intensity is extracted and is matched to the best model of its upper-level GMM. While mean pixel intensity is useful for the detection of illumination changes, it is also sensitive to changes caused by the foreground. Additionally, the Hierarchical GMM does not exploit any spatial relationships among pixels which can thus output incoherent segmentation [36]. Dong *et al.* [116] employ PCA to build a number subspaces where each represent a single background appearance. The foreground is segmented by selecting the subspaces which produces minimum reconstruction error. However, their work does not discuss how the system reacts to repetitive background movements.

More recently, Zhou *et al.* [5], Ng *et al.* [42] and Hwang *et al.* [117] have developed techniques that have potential to be robust to sudden illumination changes. These will be discussed in more detail in the next chapter.

Chapter 3

Proposed Solutions

Here we will discuss the three algorithms that were identified in Chapter 2 in greater detail. An illustration of each proposed algorithm is provided as well as a description of its respective background subtraction steps.

3.1 Dynamic background subtraction using spatial-colour binary patterns

3.1.1 Introduction and algorithm overview

This background modeling technique was first developed by Heikkila *et al.* in 1999 using local binary pattern (LBP) histograms as background features [11]. It was improved upon by Heikkila *et al.* in 2006 and Zhang *et al.* in 2008 [118, 119]. Zhang *et al.* made use of a Spatio-temporal LBP [119] while Heikkila *et al.* implemented a Centre-Symmetrical LBP (CS-LBP) [118]. In 2010 Xue *et al.* combined these two approaches [120]. Zhou *et al.* extended the binary pattern used by Xue *et al.* to also consider colour information [5]. This novel texture feature is known as the Spatial-Colour Binary Pattern (SCBP); The LBP and SCBP are described in more detail in Chapter 2.1.3.2.

Each pixel is modeled as a group of adaptive SCBP histograms calculated over a circular region around the pixel. Pixels in a new frame are then labeled as foreground or background depending on a proximity measure between its SCBP histogram and those of the model. An adaptive threshold is maintained for each pixel to improve both the tolerance of dynamic regions and the sensitivity of static regions. Furthermore, a contour refinement model is introduced. It employs a statistical operator to reduce false positives and improve the legibility of foreground object contours.

The diagram shown below provides a visual representation of the algorithm proposed by Zhou *et al.* For the sake of simplicity all processes relating to this solution apply to a single pixel and are performed on all the pixels in an image.

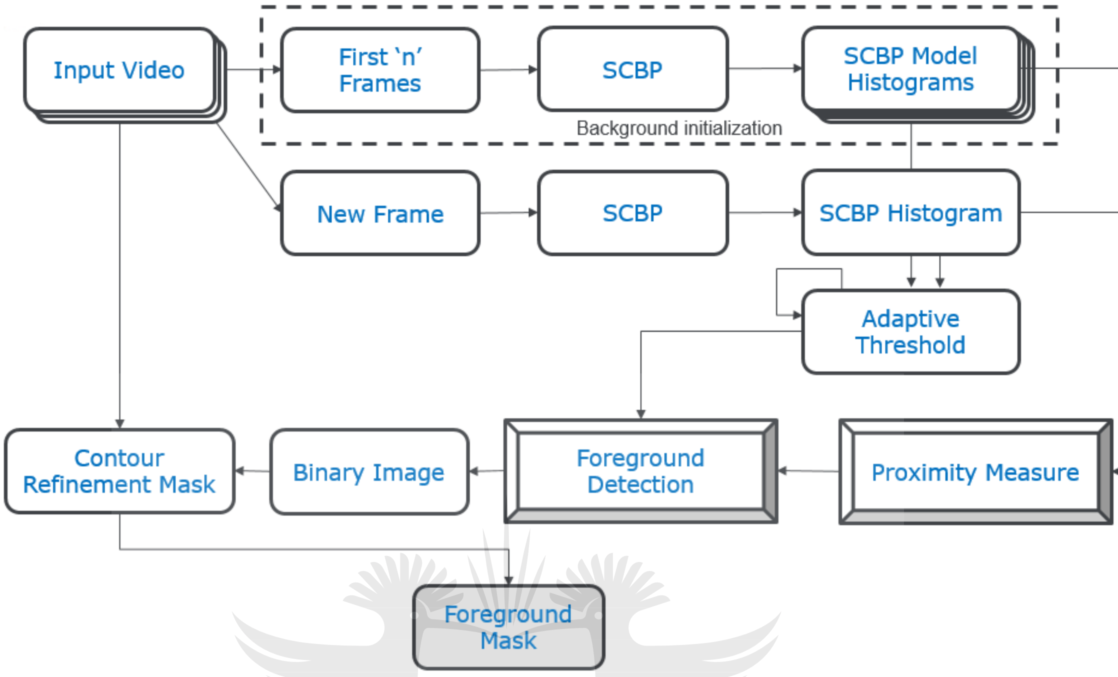

FIGURE 3.1: Overview of the Algorithm proposed by Zhou *et al.*

3.1.2 Background initialization

The temporal median technique (Section 2.1.8.2) is implemented to compute a median background pixel value. An insertion sort algorithm is employed which has $O(n)$ time complexity at its best. Once this is completed, a SCBP histogram is computed over a circular region of radius R_{region} around the pixel and a model consisting of K SCBP histograms is built.

However, all of the histograms in the model are not necessarily produced by background processes so the persistence of each histogram must be considered when deciding if it will be included as part of the background model.

A weighting technique is employed to achieve this. Each of the model histograms is assigned its own weight, such that $w_0 + w_1 + ... + w_K = 1.0$ where the magnitude of the weights are in descending order. When initialized these histograms will have identical bin values but will begin to differ as their respective bins and weights are updated. The weight of a histogram is increased if it is especially similar to that of a new pixel. The persistence of a model histogram is directly related to its weight; the larger it is the higher the probability it has of being a background histogram and being included in the background model.

In order to determine which of these model histograms will be included as part of the background model, the value for B is first determined using Equation 3.1. The value computed for B determines the number of corresponding model histograms that are selected to be part of the background model.

$$w_0 + w_1 + ... + w_B \leq T_B \tag{3.1}$$

where the weights have been sorted into descending order. T_B is a fixed threshold and is dependent upon the number of histograms that make up the model.

3.1.3 Background modeling

A SCBP histogram is calculated for each new pixel and then compared to the model histograms using a proximity measure as described by Equation 3.2. This measure adds the mutual minimum histogram bins of the current frame and each SCBP histogram in the model. An advantage of the proximity measure is that it explicitly neglects features that only occur in one of the histograms. Furthermore, it is not very computationally expensive having a time complexity of $O(n)$ for the number of histogram bins.

$$\cap\,(\bar{a}, \bar{b}) = \sum_{n=0}^{N-1} min(a_n, b_n) \tag{3.2}$$

where $\bar{a}$ and $\bar{b}$ are histograms and N is the number of bins in each histogram.

3.1.4 Foreground detection

If the proximity measure value for at least one of the B background model histograms is greater than a threshold, T_p, the pixel is classified as background, otherwise the pixel is classified as foreground.

T_p is an adaptive threshold that is maintained (for each pixel). The advantage of this is that static regions become more sensitive while dynamic regions have a higher tolerance to noise (Figure 3.2). The threshold is updated as follows:

$$T_p(x, y) = \alpha_p(s(x, y) - 0.05) + (1 - \alpha_p)T_p(x, y) \tag{3.3}$$

where α_p is a learning rate such that $\alpha_p \in [0, 1]$. $s(x, y)$ is a similarity measure of the highest value between the current frame's SCBP histogram bins and those of the histograms which comprise the model. Zhou *et al.* do not specify which similarity measure they used; the L_1 or L_2 norm[1] can be used, 0.05 is an empirical value introduced by Zhou *et al.*

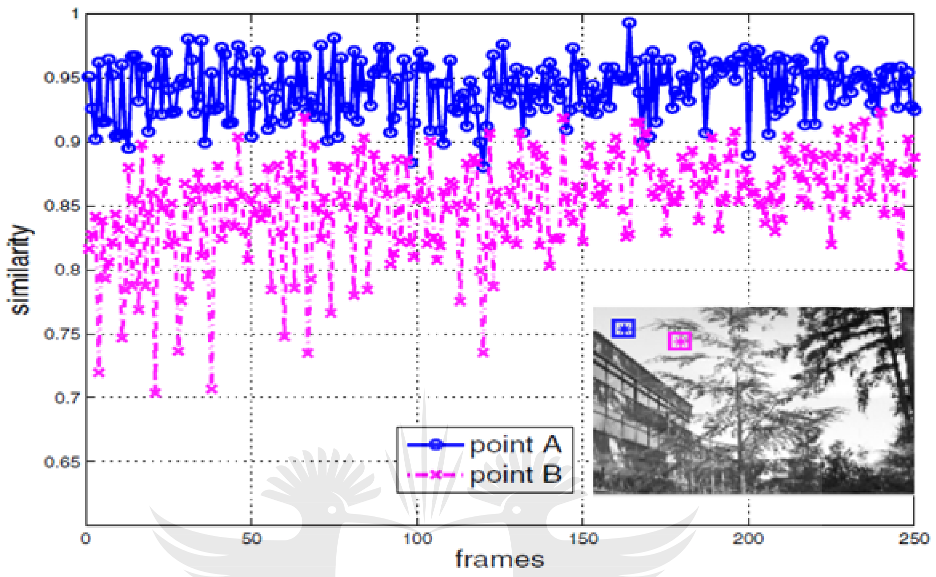

FIGURE 3.2: The adaptive theshold $T_p(x, y)$ is more sensitive to noise in static regions and more tolerant of noise in dynamic regions [5].

3.1.5 Contour refinement

A disadvantage of block-based processing is that pixels near the edges of objects are often misclassified resulting in illegible object contours. Zhou *et al.* refine these contours using a statistical operator to reduce false positives and negatives. These are based upon two assumptions. A pixel should only be successfully classified as part of the foreground if its intensity value deviates much from the average pixel intensity of its pixel neighbourhood and its colour composition changes much from that of its pixel neighbourhood.

Therefore, a binary mask is constructed as follows and is convolved with the output of the foreground detection module:

$$\Omega_i = \begin{cases} 1, & \text{if } [d_i >= \xi \text{std}_i] \text{ and } [d_i/\bar{g}_i \geq \varepsilon_1], \\ 1, & \text{if} |(r_i, g_i, b_i) - (\bar{r}_i, \bar{g}_i, \bar{b}_i)| \geq \varepsilon_2, \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

[1] A norm is a function that assigns a (strictly positive) length or size to each vector in a vector space.

where $d_i = abs(g_i - \bar{g}_i)$ is the absolute deviation of intensity from the average and r, g and b are chromaticity coordinates calculated by $r = R/(R+G+B)$, $g = G/(R+G+B)$ and $b = B/(R+G+B)$. $\bar{r}_i$, $\bar{g}_i$, $\bar{b}_i$ and $\bar{g}_i$ are average values computed over the same region as that of the SCBP histogram. The parameters ξ, ε_1 and ε_2 are empirically set tuning parameters.

Figure 3.3 shows how the module can improve the output by making object contours more legible.

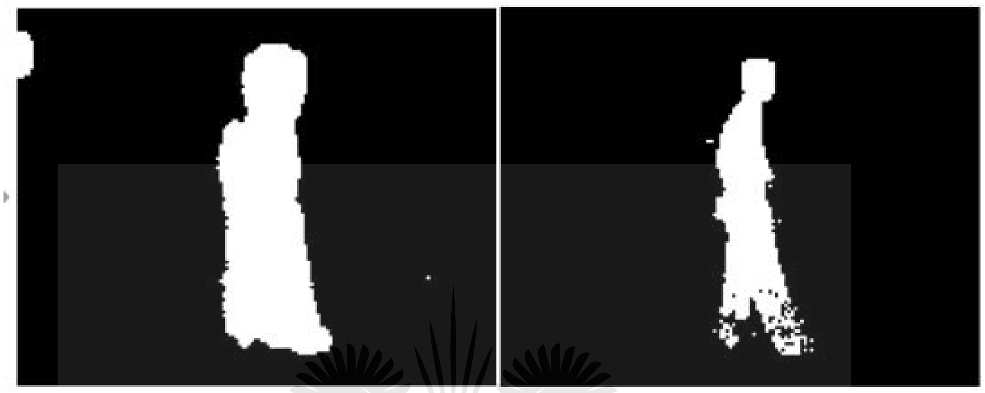

FIGURE 3.3: The foreground mask before (left) and after (right) contour refinement [5].

3.1.6 Background maintenance

3.1.6.1 Background model

The model is updated selectively depending on the value of the calculated proximity measures. If the proximity measure values of all the K model histograms are below the threshold, T_p, the histogram with the smallest weight has its bins replaced by those of the current pixel and is given a small initial weight of 0.01. No further processing is required in this case.

However, If the proximity measure value for at least one of the K model histograms is greater than T_p then only the histogram that produced the highest proximity measure is updated using Equation 3.5.

$$\bar{m_K} = \alpha_b \bar{h} + (1 - \alpha_b)\bar{m_K} \tag{3.5}$$

where $\bar{m_K}$ is the model SCBP histogram, $\bar{h}$ is the current frame SCBP histogram and α_b is a learning rate such that $\alpha_b \in [0, 1]$.

Furthermore, the weights of all K model histograms are updated as follows:

$$w_K = \alpha_w M_K + (1 - \alpha_w) w_K \tag{3.6}$$

where α_w is a learning rate such that $\alpha_w \in [0, 1]$. M_K has a value of 1 for the model histogram with the highest proximity measure and 0 for the rest.

3.1.6.2 Contour refinement model

The average and standard deviation of the resulting background pixels are updated as follows:

$$\bar{g_i} = \beta g_i + (1 - \beta)\bar{g_i} \tag{3.7}$$

$$std_i = \sqrt{\beta(g_i - \bar{g_i})^2 + (1 - \beta) std_i^2} \tag{3.8}$$

The chromaticity coordinates, $\bar{r_i}, \bar{g_i}, \bar{b_i}$, are updated in the same way as was done for $\bar{g_i}$.

3.2 Background subtraction using a shading model and a Gaussianity Test

3.2.1 Introduction and algorithm overview

Ng *et al.* make use of a technique which was first developed by Ojeda *et al.* [121] and can be applied to any time series data that is invertible and causal[2]. This technique employs a Gaussianity Test which determines if a set of samples are part of a Gaussian distribution. Gurcan *et al.* [122, 123] applied this test to mammogram images for the detection of micro-calcification.

The approach proposed by Ng *et al.* implements a hierarchical framework that uses a combination of a pixel-based shading model and a block-based Gaussianity Test. The Gaussianity Test allows for object detection while the shading model handles illumination changes. Figure 3.4 provides a visual representation of the algorithm proposed by Ng *et al.*

3.2.2 Background initialization

Temporal frame-differencing is implemented to initialize the background model (see Section 2.1.8.3).

[2]Data is "causal" if the future value of a variable is a mathematical function of past values.

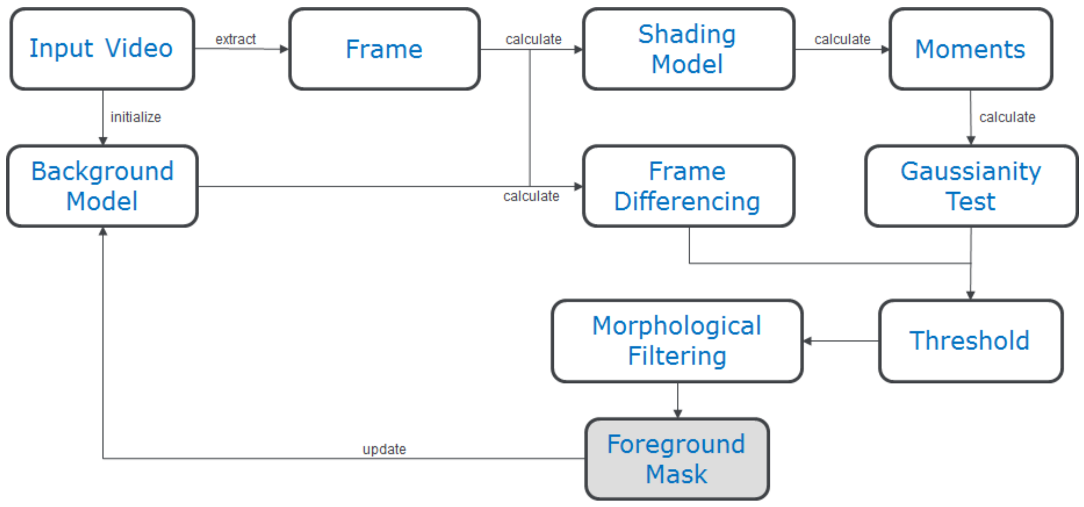

FIGURE 3.4: Overview of the algorithm proposed by Ng *et al.*

3.2.3 Background modeling and foreground detection

3.2.3.1 Gaussianity Test

The method proposed by Ng *et al.* is based on the assumption that sensor noise is both spatially Gaussian, and temporally uncorrelated. Then, if a difference frame is taken (Equation 2.14), only Gaussian noise and foreground objects should remain since the sum of independent Gaussian random variables is Gaussian [42]. Under these assumptions, they deduce that background pixels will be Gaussian distributed and foreground pixels will be non-Gaussian distributed. Therefore background pixels can be distinguished from foreground pixels using a Gaussianity Test performed on a difference frame, D_t.

If a set of samples, such as a region of pixels, is Gaussian distributed, the non-central moments of the distribution converge on their theoretical values as the sample size tends to infinity. These theoretical values are as follows:

$$\begin{aligned} J_1 &\rightarrow \mu \\ J_2 &\rightarrow \sigma^2 + \mu^2 \\ J_3 &\rightarrow \sigma^3 + 3\sigma^2\mu \\ J_4 &\rightarrow \mu^4 + 6\mu^2\sigma^2 + 3\sigma^4 \end{aligned} \tag{3.9}$$

Therefore, a Gaussianity Test is deduced such that its resultant test statistic is expected to be close to zero when a set of samples is Gaussian distributed. Ng *et al.* define this statistic similarly to Gurcan *et al.*, using Equation 3.10 [122, 123]; if the theoretical values of the non-central moments are used Equation 3.10 will be equal to 0.

$$H(J_1, J_2, J_4) = J_4 + 2J_1^4 - 3J_2^2 \tag{3.10}$$

Where J_k is the k^{th} moment of a probability distribution. Moments are discussed in greater detail in Section 2.1.3.3.

Ng *et al.* compute these moments and corresponding Gaussianity Test for each $M \times M$ non-overlapping block in an image (also known as a tile). Figure 3.5 illustrates this. Consequently, J_k is a moment defined by Equation 3.11:

$$\hat{J}_k(x,y) = \frac{1}{MN} \sum_{m=-\frac{M-1}{2}}^{\frac{N-1}{2}} \sum_{n=-\frac{N-1}{2}}^{\frac{N-1}{2}} [I_t(x+m, y+n)]^k \tag{3.11}$$

where $I_t(x,y)$ is a sample of an input image I at index (x,y), k is the moment number and $M \times N$ is the block size.

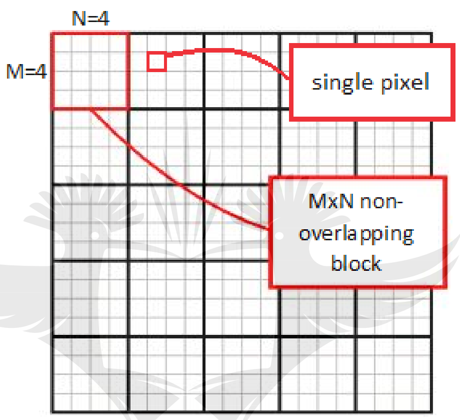

FIGURE 3.5: An Illustration of a non-overlapping block or tile [124].

3.2.4 Foreground detection

If a set of samples in a block has a Gaussianity Test statistic that is greater than a predefined threshold, τ, then the block is considered to contain foreground pixels. This is described by Equation 3.12. Foreground detection is only completed for pixels within these blocks.

$$\text{block} = \begin{cases} \text{contains foreground pixels,} & \text{if } H > \tau \\ \text{contains background pixels,} & \text{otherwise} \end{cases} \tag{3.12}$$

Foreground detection is done using the frame-differencing method discussed in Chapter 2.1.6. Ng *et al.* employ an adaptive threshold, $T_{fd,ad}$, when using this method as discussed in Chapter 2.1.6.1.

Once the foreground mask has been generated, morphological filtering is performed on the foreground mask in order to remove noise. Morphological operations are discussed in greater detail in Chapter 2.3. Ng *et al.* perform one closing operation followed by one opening operation [40].

3.2.4.1 Shading model

The change in pixel intensity values caused by photometric distortion is not constant - even for global illumination changes. Hence, the previous assumption that background regions are Gaussian distributed does not hold true in the presence of gradual and sudden illumination changes. Ng *et al.* introduce a shading model proposed by Skifstad and Jain [9] (derived from the Phong shading model described in Section 2.2.3) in order to make it robust to sudden and gradual illumination changes.

The model states that a pixel intensity can be decomposed into a product of a shading coefficient $S(x, y)$ and an illumination value L_i [9]:

$$I(x, y) = L_i S(x, y) \tag{3.13}$$

It is then assumed that if there is no physical change between two frames, such as a moving object, the ratio shown in Equation 3.14 will be constant and independent of the shading coefficients [9]:

$$R(x, y) = \frac{I_1(x, y)}{I_2(x, y)} = \frac{L_{i,1}}{L_{i,2}} \tag{3.14}$$

Ng *et al.* modify their original assumption in order to account for illumination changes; if no foreground objects exist in a scene, the ratio of pixel intensities between two frames should remain constant and therefore be Gaussian distributed [42]. So by extending the Gaussianity Test to include the shading model the background model can be made robust to sudden illumination changes. Equation 3.10 is modified follows:

$$\hat{J}_k(x, y) = \frac{1}{M^2} \sum_{m=-\frac{M-1}{2}}^{\frac{M-1}{2}} \sum_{m=-\frac{M-1}{2}}^{\frac{M-1}{2}} [R_{gt}(x + m, y + n)]^k \tag{3.15}$$

where

$$R_{gt}(x, y) = \frac{I_t(x, y)}{B_{t-1}(x, y)} \tag{3.16}$$

3.2.4.2 Shading model investigation

In the event of a light-to-dark sudden illumination change, such as the sun moving behind clouds, $R_{gt}(x, y) \in [0, 1]$; if a dark-to-light sudden illumination change occurs,

$R_{gt}(x,y) \in [1,\infty)$. Furthermore, the former tends toward 0 when an illumination change has occurred while the latter tends toward ∞. This means that the resultant Gaussianity test statistics for these two scenarios will vary greatly, even if the change in intensity values is identical. It is therefore unlikely that correct foreground region classifications will be made for these scenarios if they employ the same Gaussianity test threshold, τ.

After performing empirical tests we found that the ideal threshold of each scenario can vary by several orders of magnitude. Furthermore, our tests show that it is possible to produce negative Gaussianity test statistics when $R_{gt}(x,y) \in [0,1]$. Ng *et al.* do not discuss these possibilities; the test statistic threshold that they implement is positive and very large ($\tau = 1 \times 10^5$) which suggests that they only anticipate lighting changes from dark to light. Since both types of lighting changes are possible it is worth investigating the effect that this oversight has on classification accuracy. It is also worth investigating the classification of the Gaussianity test when it employs a shading model that forces $R_{gt}(x,y) \in [0,1]$ as in Equation 3.17 and $R_{gt}(x,y) \in [1,\infty)$ as in Equation 3.18.

$$R_{gt}(x,y) = \begin{cases} \frac{I_t(x,y)}{B_{t-1}(x,y)}, & \text{if } I_t(x,y) < B_{t-1}(x,y) \\ \frac{B_{t-1}(x,y)}{I_t(x,y)}, & \text{otherwise} \end{cases} \tag{3.17}$$

$$R_{gt}(x,y) = \begin{cases} \frac{B_{t-1}(x,y)}{I_t(x,y)}, & \text{if } I_t(x,y) < B_{t-1}(x,y) \\ \frac{I_t(x,y)}{B_{t-1}(x,y)}, & \text{otherwise} \end{cases} \tag{3.18}$$

We investigate both types of sudden illumination change as well as a scenario where illumination change is not present as a control group. Furthermore we investigate two types of sample distribution changes: A change from a Gaussian to non-Gaussian distributed block (which should yield a foreground region classification) as well as from a Gaussian to Gaussian distributed block (which should yield a background region classification). It is not necessary to investigate changes from a non-Gaussian to Gaussian distributed block and from a non-Gaussian to non-Gaussian distributed block; we assume that our background model does not contain foreground objects and therefore the shading model ratios within any block should always be Gaussian distributed. Hence, a total of eighteen Gaussianity Test statistics are computed.

Gaussian samples are generated using a Gaussian random number generator while non-Gaussian samples are generated using a random number generator that uses atmospheric noise[3] as its source [125]. The mean of a Gaussian distribution changes from 0.9 to 0.1 when simulating an illumination change from light-to-dark and vice versa. Each Gaussian distribution has a standard deviation of 0.03. Each non-Gaussian distribution changes from values that range between 0.8 and 1.0 to values that range between 0.0 and 0.2 when simulating an illumination change from light-to-dark and vice versa. When a sudden illumination change does not occur, as is the case for our control group, the distribution mean, or range, remains constant.

Our investigation considers Two 17×17 blocks that represent the same region in a scene before and after a sudden illumination change has (or has not) occurred. As mentioned, a block should only be classified as background if the samples within it remain Gaussian distributed and, ideally, these classifications should not differ in the presence of a sudden illumination change.

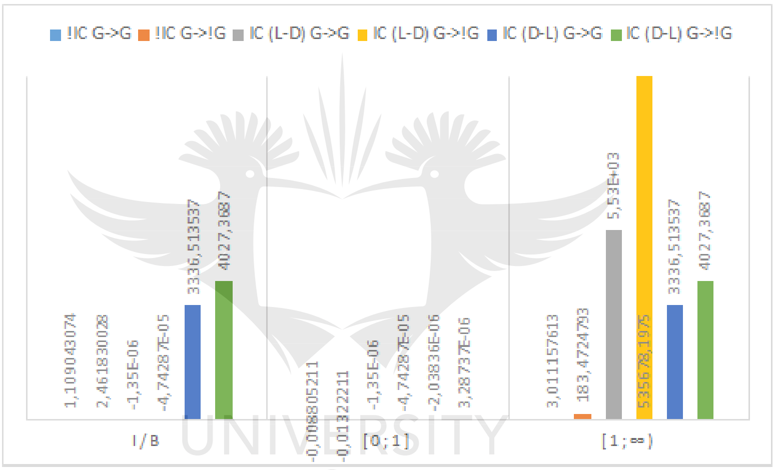

FIGURE 3.6: The Gaussianity test statistics produced using Equations 3.16, 3.17 and 3.18, respectively.

The results of our investigation are shown in Figure 3.6. In all three scenarios, and for all three equations, blocks that remain Gaussian distributed produce test statistics that are lower than when a block changes to a non-Gaussian distribution. This verifies Ng *et al.*'s assumption regarding the premise of their background model.

We first consider the scenarios were illumination changes are present. As we suspected, Equation 3.16 produces two different Gaussianity Test statistics that differ by many orders of magnitude, for the two types of sudden illumination change. This is also true for Equation 3.18, however the statistics are within the same order of magnitude. Of all three equations, Equation 3.17 is the only one that produces Gaussianity Test statistics that can be distinguish foreground from background blocks using the same threshold.

[3]Atmospheric noise is radio noise caused by natural atmospheric processes resulting primarily from lightning discharges in thunderstorms.

This is due to the scaling that takes place by the various equations; Equation 3.17 performs better because the shading model is normalized.

However, even if Equation 3.17 is used, another threshold is necessary for the control group scenario. For this reason, as well as the need for more legible foreground object contours, Ng *et al.* perform an additional frame-differencing step before generating a foreground mask.

It should be noted that the Gaussianity Test statistics that were produced are not large enough to be effectively distinguished using the threshold value employed by Ng *et al.*. Ng *et al.* mention that this value should be empirically set for the sequence at hand and depends on the amount of variation in the background. It is also possible that the Gaussianity Test is too lenient for the specific changes in distribution we used to simulate illumination changes (or the absence thereof); it is reasonable to assume that more obvious changes in sample distribution will produce test statistics capable of being distinguished by the threshold that Ng *et al.* use.

It is interesting to note that Equation 3.17 produces test statistics for the control group that are larger than those produced for the other two scenarios while the opposite is true for the other two equations. This is possibly due to the fact that Equation 3.17 tends toward 0 when an illumination change has occurred while the other two equations tend towards ∞ (if we choose to ignore Equation 3.16 since it tends to both). We were unable to draw a conclusion by examining Equation 3.10 analytically and therefore cannot be certain of this or whether the control group will always produce larger statistics. However, if this is the case Equation 3.17 is once again superior to the other two equations because the control group will still be classified as a possible foreground block and can be distinguished using the frame-differencing module.

In conclusion, we have validated our suspicion that Ng *et al.*'s oversight regard the shading model does negatively influence the classification accuracy of their solution. Furthermore, we found Equation 3.17 to be superior to the other two equations. This will be taken into consideration when we investigate possible improvements to Ng *et al.*'s solution in Chapter 5.

3.2.5 Background maintenance

Ng *et al.* do not explicitly describe the background maintenance technique that they chose to implement. Their solution is a continuation of their older work which makes use of a modified form of the selective maintenance scheme described in Section 2.1.7.2. Their older work makes use of a small fixed threshold in order to distinguish pixels that have a particularly high likelihood of belonging to the background [40]. This is described by Equation 3.19.

$$B_t(x,y) = \begin{cases} B_{t-1}(x,y), & \text{if } D_t(x,y) \geq T_a \\ I_t(x,y), & \text{if } D_t(x,y) < T_f \\ \alpha I_t(x,y) + (1-\alpha)B_{t-1}(x,y), & \text{if } T_f \leq D_t(x,y) < T_a \end{cases} \tag{3.19}$$

where T_f is fixed and smaller than T_a and α is a learning rate such that $\alpha \in [0,1]$

Ng *et al.*'s latest work does not explain how this maintenance scheme should behave after having introduced the Gaussianity test module to their solution. We take the liberty of implementing the same equation, but only for regions that have been classified as possibly containing foreground pixels by the Gaussianity test. If this is not the case, the pixel takes on the value of the current frame.

3.3 Non-parametric KDE

3.3.1 Algorithm overview

This solution is an extension of the popular kernel density estimation (KDE) technique first proposed by Elgammal *et al.* [126]. This technique maintains two non-parametric background models, long-term and short-term, in order to exploit their respective advantages at eliminating false positive detections.

Vemulapalli and Aravind extends the model from the temporal to spatio-temporal domain by using 9-dimensional data points. In order to overcome the obvious increase in computational complexity that this would cause, a hyper-spherical kernel is used instead of the typical Gaussian kernel [127].

Furthermore, a modification is made to the short-term model in order to handle sudden illumination changes; if a sudden illumination change is detected, the model is updated differently so that it adapts to the change quickly [127].

The diagram shown in Figure 3.7 provides a visual representation of the algorithm proposed by Vemulapalli and Aravind.

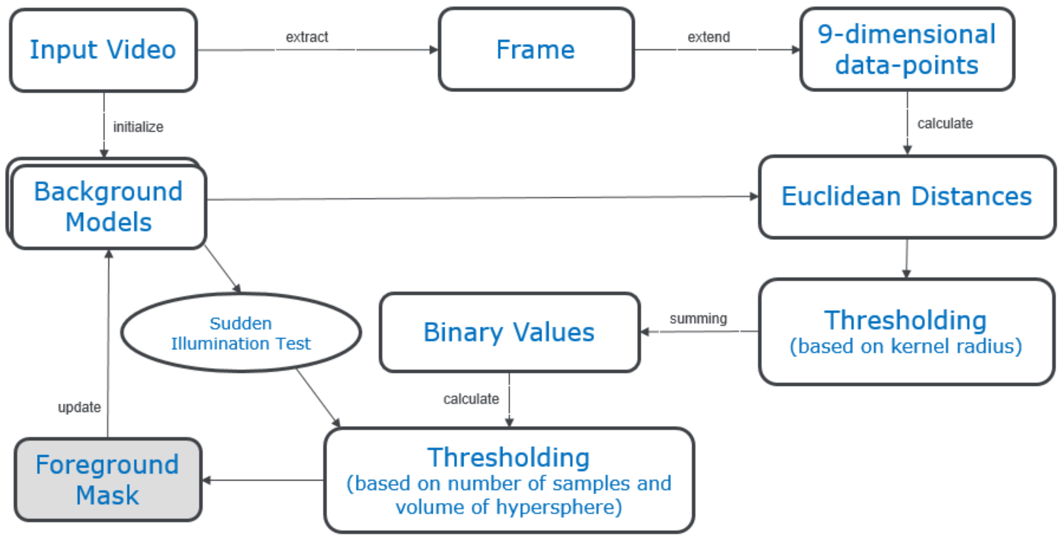

FIGURE 3.7: Overview of the algorithm proposed by Vemulapalli and Aravind.

3.3.2 Background initialization

The background model is initialized once sufficient frames have passed to fill a preset window of size W. Once this is complete the first background modeling and foreground detection pass is performed, providing a binary output. The long-term and short-term models are initialized, each eventually containing N samples (where $W > N$), using the update mechanisms described in the background maintenance section.

3.3.3 Background modeling

In order to employ a spatio-temporal approach each frame is organized into 9-dimensional data points; 3×3 blocks centred at each pixel in the frame with coordinates (x, y). Each pass of the background modeling module entails comparing the data points of the current frame, $F_0(x, y)$ with those of the previous frames, $F_{1...N}(x, y)$.

So, for each new frame a series of $N - 1$ Euclidean distances are calculated by comparing each current pixel's data point to its past data-point values. The higher the value of a Euclidean distance, the higher the probability that the current pixel is part of the foreground. These distances are then thresholded to determine if they lie within the radius of the discrete hyperspherical kernel, r. This radius is a function of the amount of variation present in the background; If r is too small the estimated function will not be smooth. This means that the estimated probability density is strongly dependent on the observed pixel values and false positives are likely. On the contrary, if r is too large the estimated function will be too smooth, increasing the likelihood of false negatives.

The $N-1$ binary outputs of this module are summed to produce a type of confidence measure, M, of whether the current pixel belongs to the background. This is described by Equation 3.20.

$$M = \sum_{i=1}^{N} \phi\left(\frac{||F_0(x,y) - F_i(x,y)||}{r}\right) \tag{3.20}$$

where r is the radius of the hyper-sphere and

$$\phi(u) = \begin{cases} 1, & \text{if } u \leq 1, \\ 0, & \text{otherwise} \end{cases} \tag{3.21}$$

and $||F_0(x,y) - F_i(x,y)||$ is the Euclidean distance between the data points $F_0(x,y)$ and $F_i(x,y)$. M is then thresholded using a value, T, as described by Equation 3.22.

$$\frac{M}{N} \leq T \tag{3.22}$$

Furthermore, a modification is made to the short-term model in order to handle sudden illumination changes. If a sudden illumination change is detected, the model is updated differently so that it adapts to the change quickly [127]. This is discussed in more detail in Section 3.3.5.

3.3.4 Foreground detection

The outputs of both the long-term and short-term models are used as inputs to the foreground detection module, the output of which is described by Table 3.1. If the two models agree on an output, the resultant foreground mask will obviously have the same output. If only the long-term model predicts foreground, the foreground mask will prefer the prediction of the short-term model. In the event of the short-term model predicting foreground and the long-term model predicting a background, a check is performed to see if the two models agree on the output of any of the neighbouring pixels being foreground. If this is the case, the pixel is classified as a foreground pixel.

Long-term model	Short-term model	Output
$O_l(x,y) = 0$	$O_s(x,y) = 0$	$O_{fd}(x,y) = 0$
$O_l(x,y) = 0$	$O_s(x,y) = 1$	$O_{fd}(x,y) = O'_{fd}(x,y)$
$O_l(x,y) = 1$	$O_s(x,y) = 0$	$O_{fd}(x,y) = 0$
$O_l(x,y) = 1$	$O_s(x,y) = 1$	$O_{fd}(x,y) = 1$

Table 3.1: The output of the foreground detection module, $O_{fd}(x,y)$, which combines the output of the short-term model, $O_s(x,y)$, and long-term model, $O_l(x,y)$. $O'_{fd}(x,y)$ is described by Equation 3.23.

$$O'_{fd}(x,y) = \begin{cases} 1, & \text{if } \sum_{i=-1}^{1}\sum_{j=-1}^{1} O_s(x-i,y-j)O_l(x-i,y-j) \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{3.23}$$

3.3.5 Background maintenance

The long-term and short-term models are updated using a blind update and selective update mechanism, respectively. The blind update adds a new 9-dimensional data point, $F_i(x, y)$, to the sample set regardless of whether it belongs to the background or foreground while the selective update adds the data-point only if it belongs to the background. This means that the long-term model will always contain N samples while the short-term model can contain any number of samples up to a maximum of N (selected from a window of size W). When a new data point is added the oldest data point is removed from the sample set.

In the event of a sudden illumination change most of the frame will be falsely classified as foreground and will continue to be classified as such unless the background model adapts to the new lighting conditions. Vemulapalli and Aravind checks whether the percentage of pixels that have been classified as foreground in the previous frame, α, exceeds a threshold, T_f. If this is the case the short-term model is updated using the blind update mechanism and will continue to do so for new frames until until the percentage falls below a certain threshold. Eventually, once enough frames have passed which have the new lighting conditions present, the resultant classification of the short-term model will disagree with the long-term model. As shown in Table 3.1, if the short-term model classifies the pixel as background and disagrees with the long-term model, then the short-term model takes preference. This ensures that background model adapts quickly to the new lighting conditions. However, the success of this mechanism is dependent on the relationship between N (which is derived from W) and α. The test requires that by the time α falls below T_f enough frames have been blindly updated to the short-term model to correctly classify future pixels under the new lighting conditions.

Chapter 4

Experiments and Discussion

4.1 Introduction

The three proposed solutions that were investigated in chapter 3 are implemented on a GPU. Statistics are gathered in order to determine which of the three solutions is superior with regard to classification accuracy and computation time. Details are provided concerning the experimental procedure, equipment, dataset and metrics that were employed. Finally, the results of the experiments are presented and discussed.

A substantial amount of computational resources is required to process images, especially since the size of images are ever-increasing [128]. GPU implementation is advantageous because it provides higher computational power at a lower cost [129]. Furthermore, as is shown in Figure 4.1 and 4.2, it is also reasonable to assume that this advantage will not diminish soon [130].

4.2 Experimental procedure

4.2.1 Selection of tuning parameters

Each of the solutions have a number of tuning parameters. These are set to have the same values as was employed by the original authors.

Zhou *et al.* set $R_{region} = 9$, $R = 2$, $N = 4$, $K = 4$, $T_p = 0.65$, $T_B = 0.7$, $\alpha_b = \alpha_w = \beta = 0.01$, $\alpha_p = 0.9$, $\xi = 2.5$ and $\varepsilon_1 = \varepsilon_2 = 0.2$. They do not specify which similarity measure they used; we empirically set this to be the L_2 norm. Zhou *et al.* also do not specify how they initialized the weights of the model histograms; after having investigated both

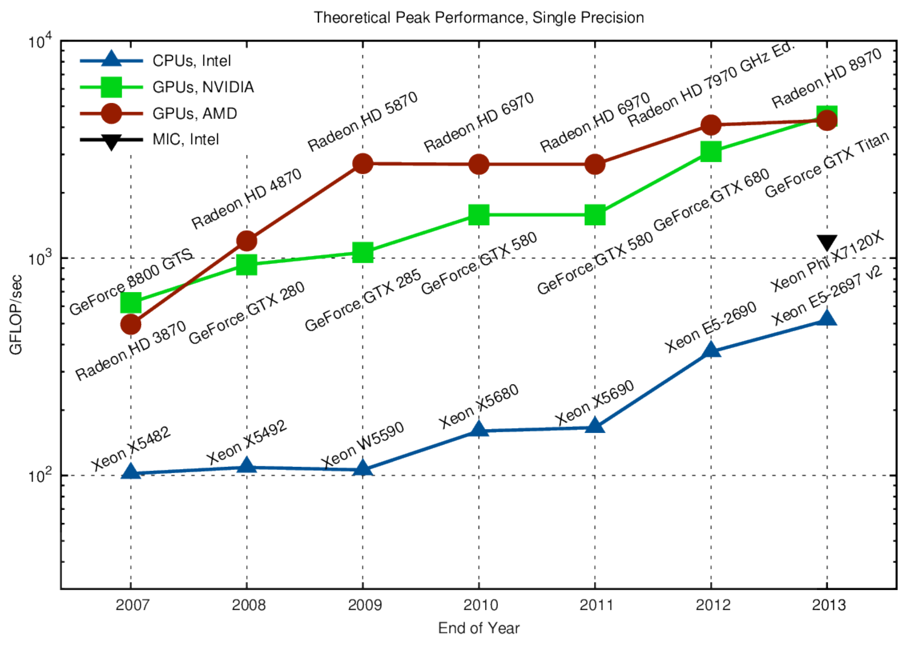

FIGURE 4.1: Theoretical peak performance for single precision [130]

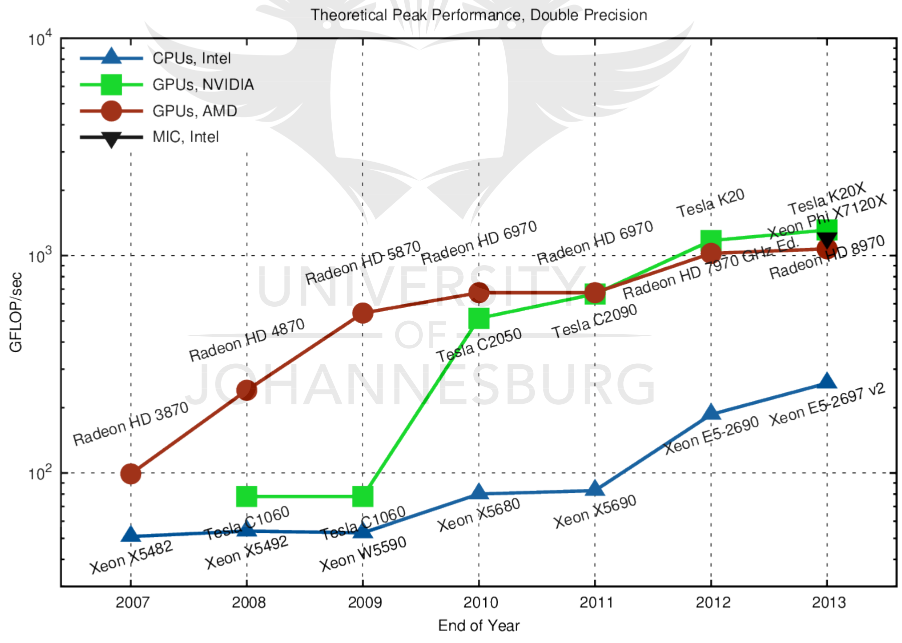

FIGURE 4.2: Theoretical peak performance for double precision [130]

linearly and exponentially decreasing values, we empirically set $w_0 = 0.567$, $w_1 = 0.321$, $w_2 = 0.103$, $w_3 = 0.011$ so that the values decrease exponentially.

Ng *et al.* set $M = 17$ and $\alpha = 0.1$. The value for τ is empirically set for the dataset at hand. We select a value of $\tau = 1 \text{ x } 10^4$.

Vemulapalli and Aravind set $W = 250$, $N = 50$ and $\alpha = 75\%$. However, for the the

"Waving Trees" sequence we set $W = 200$ and $N = 20$ since the 247th frame is used for the ground truth. They do not specify which parameters they used for the hypersphere radius, r, and the threshold, T. We set $r = 1$ and $T = \mu + k\sigma$ where μ is the mean and σ is the standard deviation of the greyscale input image. k is a positive integer which is empirically set to 6.

4.2.2 Metrics

4.2.2.1 Classification accuracy

In order to evaluate the classification accuracy we make use of the detection rate (DR), false alarm rate (FAR) and precision (P) statistics. The formulae for these are provided below:

$$DR = \frac{\#tp}{\#tp + \#fn} \tag{4.1}$$

$$FAR = \frac{\#fp}{\#fp + \#tn} \tag{4.2}$$

$$P = \frac{\#tp}{\#tp + \#fp} \tag{4.3}$$

where $\#tp$ is the number of correctly classified foreground pixels (true positives), $\#tn$ is the number of correctly classified background pixels (true negatives), $\#fp$ is the number of incorrectly classified foreground pixels (false positives) and $\#tn$ is the number of incorrectly classified background pixels (true negatives).

The DR statistic provides the percentage of true foreground pixels that have been correctly classified. The FAR statistic provides the percentage of true background pixels that have been incorrectly classified. The P statistic provides the percentage of all the foreground pixels that were detected that have been correctly classified.

4.2.2.2 Computation time

In order to measure the computation time we employ the frame-rate statistic (FR) (equation 4.4).

$$FR = \frac{\#frames}{time} \tag{4.4}$$

where $\#frames$ is the total number of frames in the video and $time$ is the time taken to complete the computation in seconds. This is measured using a standard timer provided by the programming environment.

4.2.3 Hardware and software

The three proposed techniques that were investigated were implemented using OpenCL $v1.1$ and GLSL $v4.4$ running on an NVIDIA GeForce GTX 760 GPU and an Intel Core i7-4770K CPU @ 3.5 Ghz with 8192 GB of RAM.

4.2.4 Dataset

The use of a publicly available data ensures that our experiments are repeatable and allows others to confirm our results. Three sequences from the publicly available Wallflower dataset [73] are used. They contain real-life scenes of typical surveillance environments. Furthermore, a hand-segmented ground-truth [1] is provided for a single frame at a critical point in the sequence. Each frame in a sequence is made up of 160×120 pixels. Low resolutions such as these are often used in surveillance systems which require large amounts of memory.

While the Wallflower dataset provides a ground-truth for a only one frame per sequence, we hand-segment an additional 14 sequential ground-truths for each of the three sequences. This ensures that the results obtained in our experiments are statistically sound while also providing a way to measure the persistence of the accuracy of each solution.

The first sequence is named “Waving Trees” and contains a scene with a typical dynamic background. It comprises 286 frames where ground-truths are provided for frames 243 to 257. These correspond to a person entering the foreground with a tree waving in the background.

The second sequence is named “Time of Day” and contains a scene with gradual illumination changes. It comprises 5889 frames where ground-truths are provided for frames 1841 to 1855. These correspond to a person entering a room and sitting down on a couch after enough time has passed to allow for a gradual illumination change.

The third sequence is named “Light Switch” and contains a scene with sudden illumination changes. It has 2714 frames where ground-truths are provided for frames 1856 to 1870. These correspond to a person entering a room and sitting at a desk after having switched on a light (sudden illumination change).

Example frames from all three sequences are provided in Figure 4.3.

[1]The “ground truth” refers to an ideal output image used to measure the accuracy of the background/foreground classification of a solution.

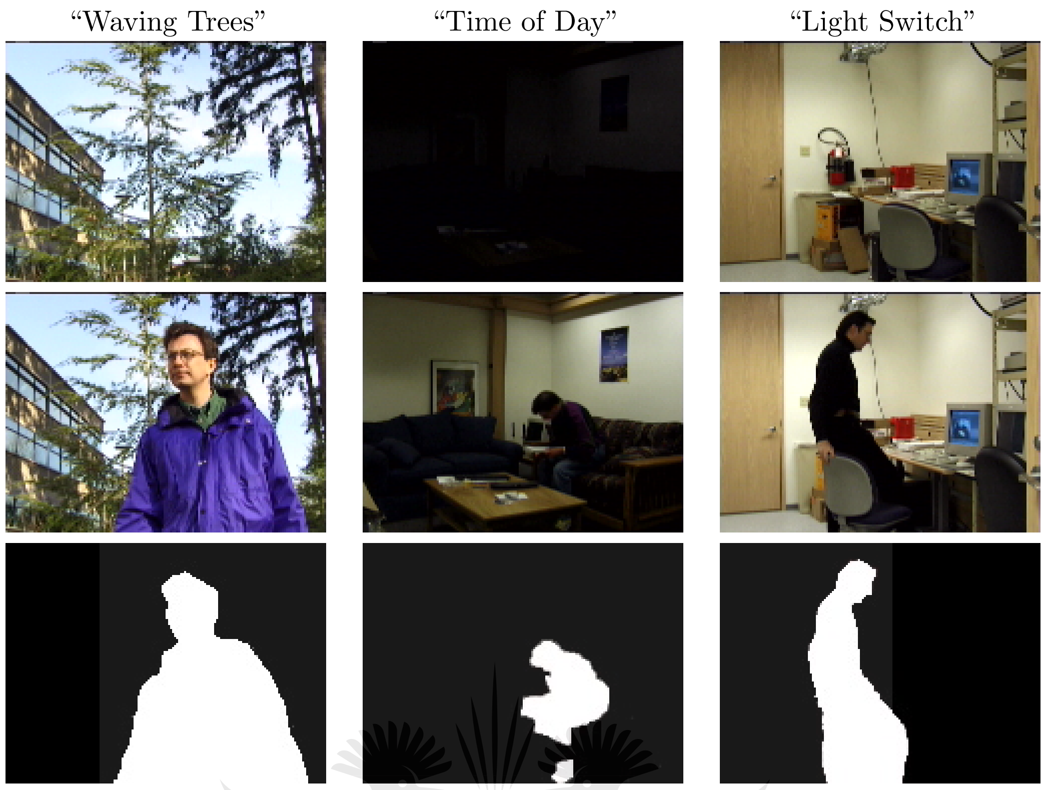

FIGURE 4.3: Three sequences from the Wallflower dataset. The top row represents the first training image while the middle row represents one of the images that will be evaluated. The bottom row shows one of the hand-segmented images that will be used as a ground truth in our experiments [73].

4.3 Experimental results

4.3.1 Experiment 1: Classification accuracy

The experimental results of the "Waving Trees", "Time of Day" and "Light Switch" sequences regarding classification accuracy are provided in Figures 4.4, 4.5 and 4.6, respectively. These are average values, each with a standard deviation, computed using the DR, FAR and P values corresponding to the 15 hand-segmented ground-truths.

4.3.2 Experiment 2: Computation time

The results of the computation time experiment are provided in Figure 4.8. These are average FPS values computed from 15 instances.

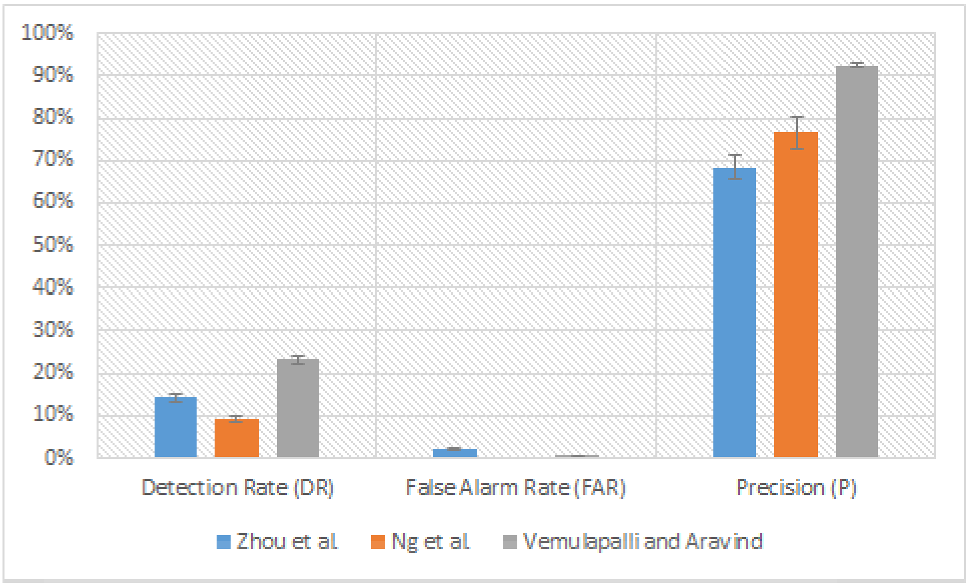

FIGURE 4.4: Classification accuracy results for the "Waving Trees" sequence.

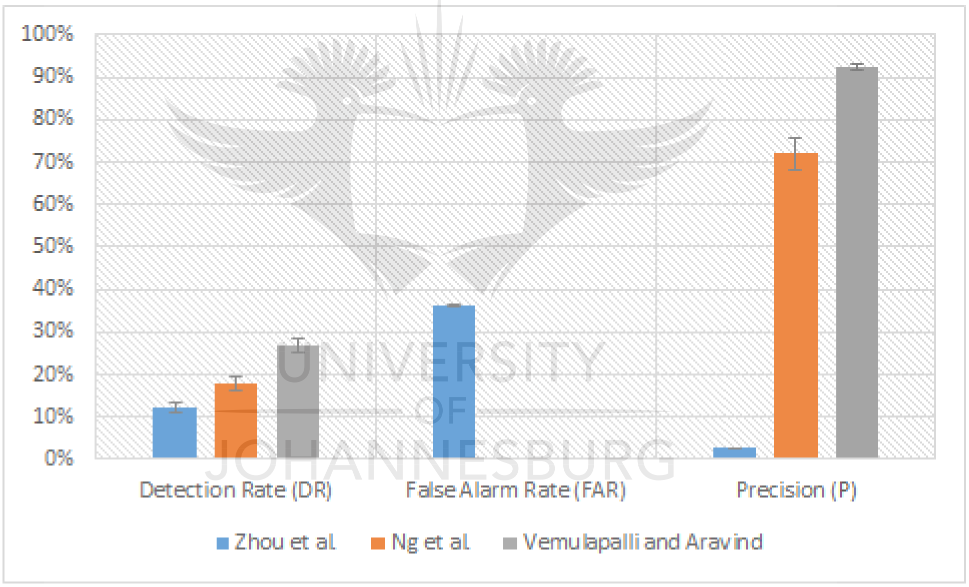

FIGURE 4.5: Classification accuracy results for the "Time of Day" sequence.

4.4 Statistical analysis of results

4.4.1 Experiment 1: Classification accuracy

4.4.1.1 "Waving Trees"

From the results shown in Figure 4.4 we can see that Vemulapalli and Aravind's solution provides the best detection rate which is 1.62 times higher than Zhou *et al.*'s solution and 2.48 times higher than Ng *et al.*'s solution. Ng *et al.*'s detection rate has the smallest

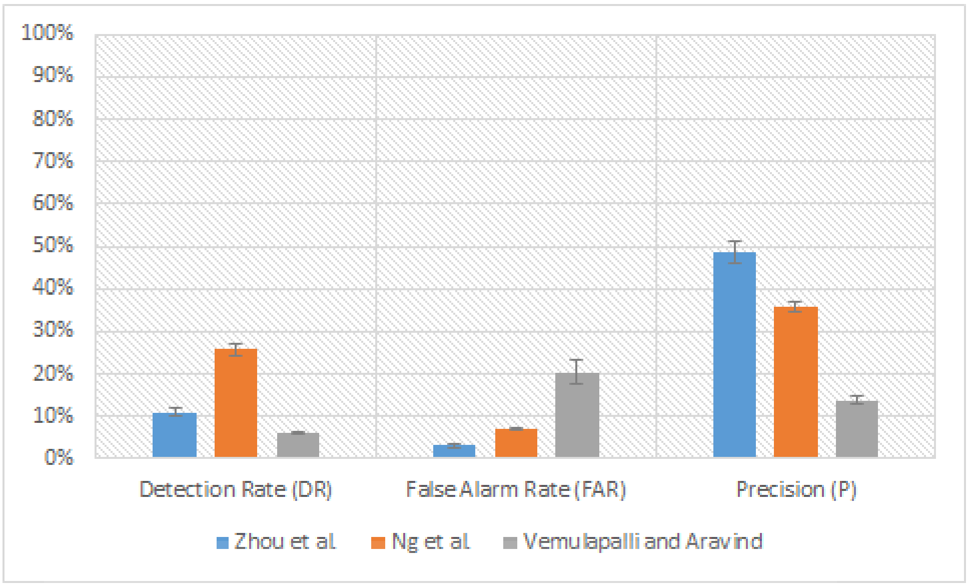

FIGURE 4.6: Classification accuracy results for the "Lightswitch" sequence.

standard deviation which is 1.65 times less than Vemulapalli and Aravind's solution and 2.04 times less than Zhou *et al.*'s solution.

Ng *et al.*'s solution provides the best false alarm rate which is 4.82 times less than Vemulapalli and Aravind's solution and 13.87 times less than Zhou *et al.*'s solution. Ng *et al.*'s false alarm rate has the smallest standard deviation which is 3.92 times less than Vemulapalli and Aravind's solution and 99.34 times less than Zhou *et al.*'s solution

Vemulapalli and Aravind's solution also provides the best precision which is 1.21 times higher than Ng *et al.*'s solution and 1.35 times higher than Zhou *et al.*'s solution. Their solution also provides the smallest precision standard deviation which is 23.88 times less than Zhou *et al.*'s solution and 43.4 times less than Ng *et al.*'s solution.

4.4.1.2 "Time of Day"

From the results shown in Figure 4.5 we can see that Vemulapalli and Aravind's solution provides the best detection rate which is 1.51 times higher than Ng *et al.*'s solution and 2.22 times higher than Zhou *et al.*'s solution. Zhou *et al.*'s detection rate has the smallest standard deviation which is 1.74 times less than Ng *et al.*'s solution and 1.9 times less than Vemulapalli and Aravind's solution.

Vemulapalli and Aravind's solution provides the best false alarm rate which is 1.64 times less than Ng *et al.*'s solution and 307.27 times less than Zhou *et al.*'s solution. Vemulapalli and Aravind's false alarm rate has the smallest standard deviation which

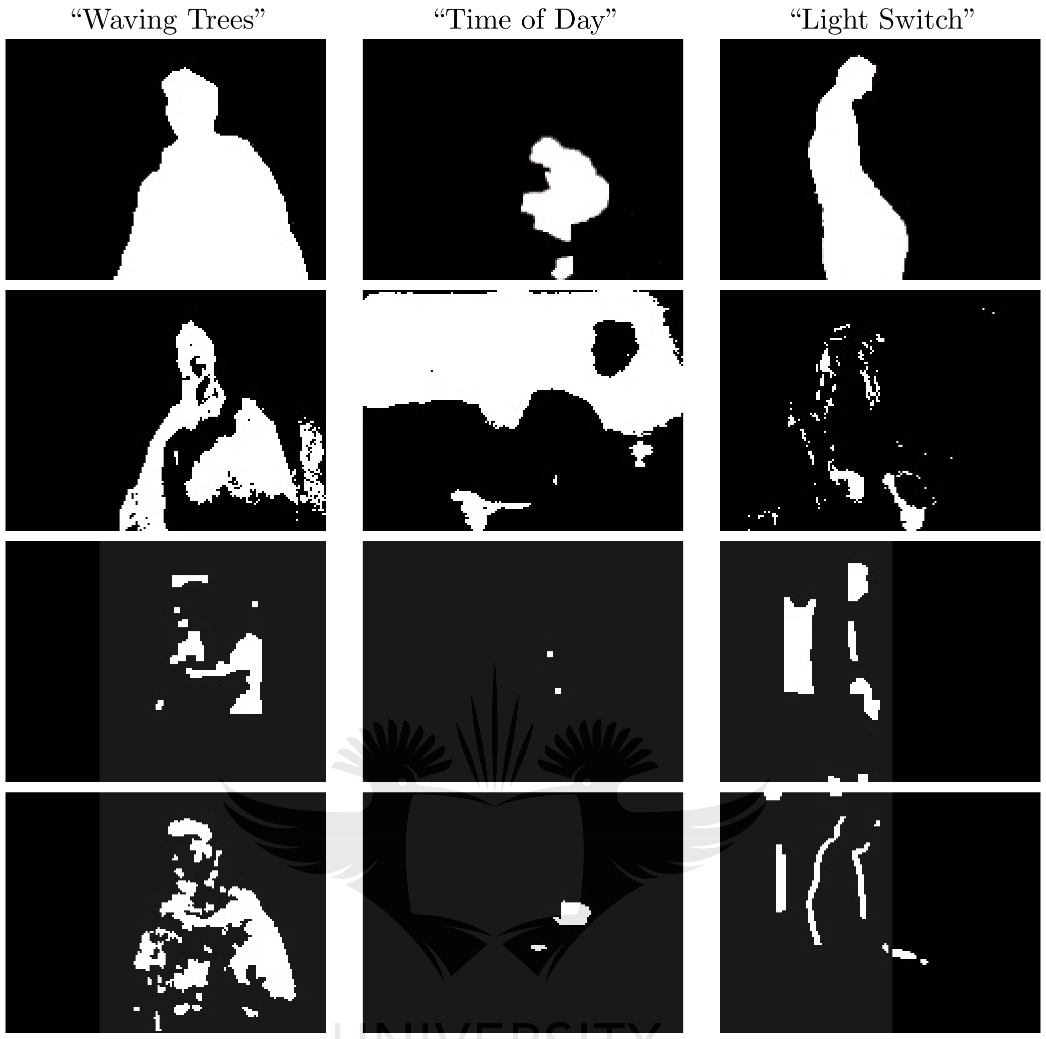

Figure 4.7: Foreground segmentation masks of proposed solutions. The columns correspond to the "Waving Trees" (frame 247), "Time of Day" (frame 1850) and "Light Switch" (frame 1865) sequences respectively. The first row represents the ground truths while the remaining rows correspond to the outputs of the solutions proposed by Zhou *et al.*, Ng *et al.* and Vemulapalli and Aravind respectively.

is 39.37 times less than Ng *et al.*'s solution and 1212.69 times less than Zhou *et al.*'s solution.

Vemulapalli and Aravind's solution also provides the best precision which is 1.28 times higher than Ng *et al.*'s solution and 36.54 times higher than Zhou *et al.*'s solution. Zhou *et al.*'s solution provides the smallest precision standard deviation which is 9.12 times less than Vemulapalli and Aravind's solution and 303.42 times less than Ng *et al.*'s solution.

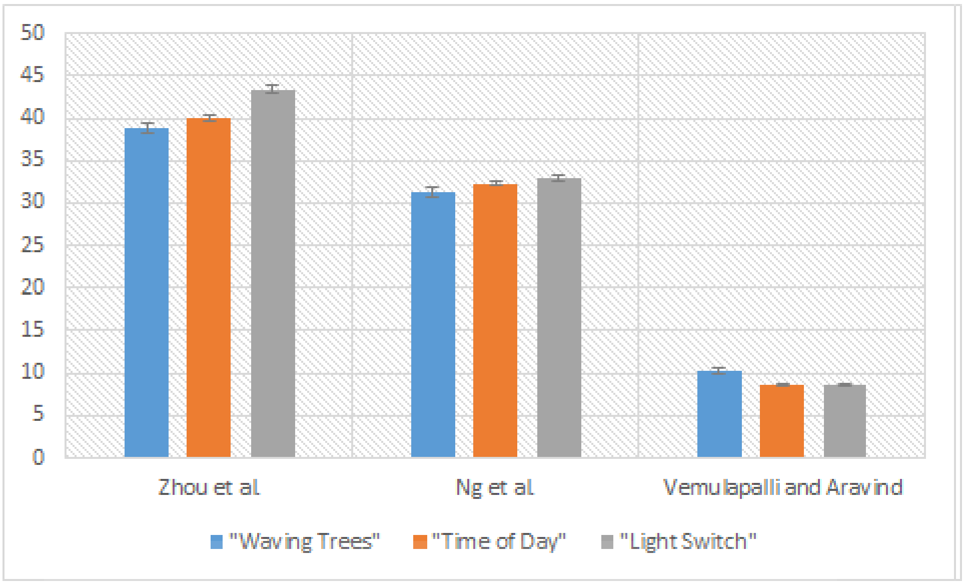

FIGURE 4.8: Computation time results, in frames-per-second (FPS), for all three solutions and all three sequences.

4.4.1.3 "Light Switch"

From the results shown in Figure 4.6 we can see that Ng *et al.*'s solution provides the best detection rate which is 2.33 times higher than Zhou *et al.*'s solution and 4.23 times higher than Vemulapalli and Aravind's solution. Vemulapalli and Aravind's detection rate has the smallest standard deviation which is 6.26 times less than Zhou *et al.*'s solution and 22.38 times less than Ng *et al.*'s solution.

Zhou *et al.*'s solution provides the best false alarm rate which is 2.27 times less than Ng *et al.*'s solution and 6.44 times less than Vemulapalli and Aravind's solution. Ng *et al.*'s false alarm rate has the smallest standard deviation which is 4.28 times less than Zhou *et al.*'s solution and 162.06 times less than Vemulapalli and Aravind's solution.

Zhou *et al.*'s solution also provides the best precision which is 1.36 times higher than Ng *et al.*'s solution and 3.50 times higher than Vemulapalli and Aravind's solution. Vemulapalli and Aravind's solution provides the smallest precision standard deviation which is 1.22 times less than Ng *et al.*'s solution and 6.40 times less than Zhou *et al.*'s solution.

4.4.2 Experiment 2: Computation time

4.4.2.1 "Waving Trees"

From the results shown in Figure 4.8 we can see that Zhou *et al.*'s solution provides the best FPS value which is 1.24 times higher than Ng *et al.*'s solution and 3.81 times higher than Vemulapalli and Aravind's solution. Vemulapalli and Aravind's solution also has the smallest standard deviation which is 1.56 times less than Zhou *et al.*'s solution and 1.63 times less than Ng *et al.*'s solution.

4.4.2.2 "Time of Day"

From the results shown in Figure 4.8 we can see that Zhou *et al.*'s solution provides the best FPS value which is 1.24 times higher than Ng *et al.*'s solution and 4.66 times higher than Vemulapalli and Aravind's solution. Vemulapalli and Aravind's solution has the smallest standard deviation which is 2.59 times less than Ng *et al.*'s solution and 3.1 times less than Zhou *et al.*'s solution.

4.4.2.3 "Light Switch"

From the results shown in Figure 4.8 we can see that Zhou *et al.*'s solution provides the best FPS value which is 1.32 times higher than Ng *et al.*'s solution and 5.05 times more than Vemulapalli and Aravind's solution. Vemulapalli and Aravind's solution has the smallest standard deviation which is 2.73 times less than Ng *et al.*'s solution and 5.72 times less than Zhou *et al.*'s solution.

4.5 Discussion of results

4.5.1 Experiment 1: Classification accuracy

4.5.1.1 Zhou *et al.*'s solution

Zhou *et al.*'s solution performs particularly poorly in the "Time of Day" sequence. This is attributed to the presence of large uniform regions, slow learning rates, α_b and α_w, as well as a value for the threshold, T_B, that is too strict.

Zhou *et al.*'s solution is primarily based on texture and colour features; regions that have very little texture and a fairly uniform colour, like the wall in the background, provide insufficient information and cause misclassifications.

The learning rates, α_b and α_w are not quick enough for the background model to adapt to the gradual illumination changes present in the sequence. If α_b is too slow the model histogram with the highest proximity measure value does not incorporate new background information quickly enough and causes false positives. On the contrary, if α_b were too quick false negatives would be likely since not all model histograms are necessarily caused by background processes . However, model histograms that do not describe the background accurately enough are soon replaced by the histogram values from the current frame. This leads us to assume that α_b does not have as much of an effect on the accuracy of the background model as does α_w.

α_w and T_B affect the number of model histograms that are included in the background model as is shown by Equation 3.1. If α_w is too slow, or T_B is too lenient, too many model histograms are included in the background model and, since a pixel is classified as background if even one of these suggests so, false negatives are likely. On the contrary, if α_w is too fast, or T_B is too strict, too few model histograms are included in the background model which increases the likelihood of false positives.

This, however, does not explain why Zhou *et al.*'s solution provides better classification accuracy for the "Light Switch" sequence since the scene has similar uniform regions as in the "Time of Day" sequence and is also negatively influenced by the chosen parameter values. The reason for this is due to the nature of the "Light Switch" sequence. Throughout the sequence two sudden illumination changes take place before the series of frames that are tested for classification accuracy; one from light to dark and vice versa. When the first change occurs the background model adapts too slowly and accepts a large portion of the scene as foreground. This persists until the next illumination change occurs which effectively reverts the scene to one similar to where no lighting changes had occurred yet. This allows Zhou *et al.*'s solution to, misleading, provide better classification accuracy than what is expected.

4.5.1.2 Ng *et al.*'s solution

Ng *et al.*'s solution performs adequately for all three sequences. However, the solution would fair much worse for the "Time of Day" sequence if it underwent a gradual illumination change from light to dark. Similarly, the solution would fair much worse for the "Light Switch" sequence if the classification accuracy statistics were computed after a sudden illumination change from light to dark. This is due to an oversight in their shading model and is discussed in greater detail in Chapter 3.2.4.2.

4.5.1.3 Vemulapalli and Aravind's solution

Vemulapalli and Aravind's solution performs particularly poorly in the "Light Switch" sequence. A trade-off exists between the classification accuracy of the this sequence in relation to the other two sequences; the classification accuracy of the "Light Switch" sequence is greatly improved when the window size of the background model, W, is reduced. However, this negatively influences the classification accuracy of the "Waving Trees" and "Time of Day" sequences. This is because of the sudden illumination change test that is performed.

As mentioned in Chapter 3.3.5, in order for this mechanism to work properly it requires that, when a sudden illumination change has occurred, enough frames have been blindly updated to the short-term model before the percentage of foreground pixels in the frame, α, falls below a certain threshold, T_f. This depends on the relationship between the number of samples in the short-term model, N (derived from the window size, W), and the value of α; if W is too big the background model will continue to misclassify background pixels as foreground. However, if W is too small the background model will misclassify foreground pixels as background because insufficient information is stored by it to account for gradual illumination changes and background dynamics. Vemulapalli and Aravind's solution would benefit if two background models were maintained, for light and dark illumination conditions, and switched when a sudden illumination change is detected.

The trade-off between the classification accuracy of the this sequence in relation to the other two sequences is further influenced by the value of the hyperspherical kernel radius, r. In the "Light Switch" sequence r is too large and, as mentioned in Chapter 3.3.3, causes false negatives. If r were to be made smaller it would negatively influence the classification accuracy of the other two sequences, causing false positives. Vemulapalli and Aravind's solution would benefit if the value of r were adaptive depending on the amount of variation present in the background.

4.5.2 Experiment 2: Computation time

The results from our experiments show us that of all the solutions, the one proposed by Zhou *et al.* is the most computationally efficient followed by Ng *et al.* and Vemulapalli and Aravind respectively. This order is consistent for all of the sequences. The standard deviation of these values is due to the fluctuating amount of available computer resources which is caused by background processes running on the computer.

4.5.2.1 Zhou *et al.*'s solution

From Figure 4.8 it is evident that Zhou *et al.*'s solution provides the largest FPS value for the "Light Switch" sequence followed by the "Time of Day" sequence and "Waving Trees" sequence, respectively.

This is influenced by the amount of background variation present in a sequence; the more dynamic it is the more often one of the model histograms will have its bins replaced by those of the current frame. This requires more memory operations which will negatively affect the FPS value of the sequence.

4.5.2.2 Ng *et al.*'s solution

From Figure 4.8 it is evident that Ng *et al.*'s solution provides the largest FPS value for the "Light Switch" sequence followed by the "Time of Day" sequence and "Waving Trees" sequence, respectively. This is influenced by the amount of background variation present in a scene as well as the percentage of frames in the sequence that contain foreground objects. Ng *et al.*'s solution only performs frame-differencing (which is a computationally expensive module) on foreground blocks detected by the Gaussianity Test. Since sequences with less background variation or foreground objects require less processing, they will compute a larger number of FPS. This is reflected by the FPS values; the "Waving Trees" sequence has the highest number of foreground blocks followed by the "Time of Day" sequence and "Light Switch" sequence, respectively.

4.5.2.3 Vemulapalli and Aravind's solution

From Figure 4.8 it is evident that Vemulapalli and Aravind's solution computes the least number of FPS of all the solutions. Their solution has much larger memory requirements because it maintains a history of 250 past frames. These FPS values improve considerably when the window size, W, is decreased.

The "Waving Trees" sequence produces a larger FPS value than the other two sequences; this is because the W was reduced in order to accommodate the fact that the ground truth sequence already commences from the 243^{th} frame.

4.6 Conclusion

Having discussed the weaknesses of all three solutions we consider which solution to select in light of possible improvements that could be introduced to each solution.

Zhou *et al.*'s solution would benefit from parameter automation because different learning rates are required for different types of lighting changes. However, it is still problematic when untextured and uniform areas are present in a scene. Their solution would require another type of feature in order to overcome this.

Ng *et al.*'s solution is the only one of the three that is capable of producing competitive classification accuracy and computation time statistics without modifications. As discussed in Chapter 3.2.4.2, its classification accuracy can be improved by modifying the shading model module. Its classification accuracy can be further improved if the moments used by the Gaussianity Test are computed for a neighbourhood around each pixel instead of for tiles. However, this will increase the computation time of the solution.

Vemulapalli and Aravind's solution would be the most accurate of all the solution if we were able to automate the window size, W, so that it is reduced when a sudden illumination change is detected, or if two interchangeable background models were maintained for light and dark illumination conditions. However, it is very computationally expensive compared to the other two solutions.

We decide to use Ng *et al.*'s solution as a basis for a new solution that will hopefully outperform all three solutions.

Chapter 5

Best Solution and Improvements

5.1 Introduction

From the experimental results gathered in chapter 4 we decided that the algorithm proposed by Ng *et al.* is best suited to our problem space. In this chapter we investigate possible modifications to this solution in order to make it more robust to sudden illumination changes and less computationally expensive. We then verify these improvements by means of a number of experiments.

5.2 Proposed improvements

This section describes possible improvements to Ng *et al.*'s solution that were considered; Figure 5.1 illustrates the modules of the solution that are investigated.

5.2.1 Shading Model

In Chapter 3.2.4.2 we present evidence which suggests that more accurate classifications are possible if Equation 3.17 is implemented instead of Equation 3.16. We, therefore, employ this equation and make two modifications to the Gaussianity Test module to accommodate it.

Equation 3.17 requires the absolute value of each Gaussianity test statistic to be considered when comparing it to the Gaussianity test threshold. Figure 5.1 shows the part of the Ng *et al.*'s solution that is modified (marked in yellow).

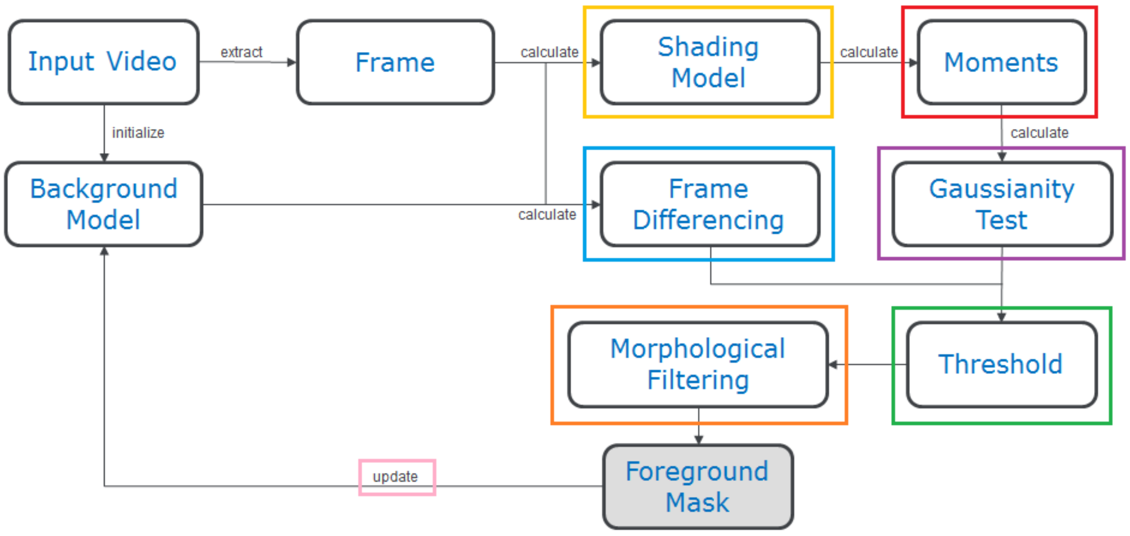

FIGURE 5.1: Ng *et al.*'s solution with coloured boxes indicating proposed modifications.

5.2.2 Calculation of moments

We compute the moments, and hence the Gaussianity Test, using $M \times M$ local pixel neighbourhoods (region surrounding a pixel) instead of non-overlapping blocks (tiles). These produce better object contours than the original block-based method but are more expensive to compute; GPU implementation affords us this.

In order to produce a moment value for a local pixel neighbourhood Equation 3.11 is used as before, but for a sliding window of size $M \times M$ (see Figure 5.2). Figure 5.1 shows the part of the Ng *et al.*'s solution that is modified (marked in red).

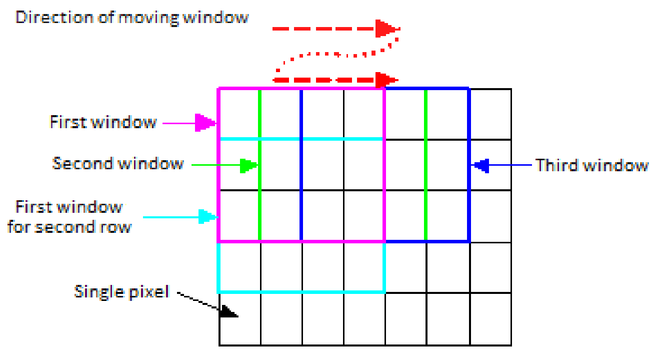

FIGURE 5.2: An illustration of a sliding window in an image [131].

5.2.3 Computation time

Ng *et al.* originally compute the Gaussianity Test for a block region and then generate a foreground mask using frame differencing, but only in the blocks that qualify as containing foreground pixels according to the Gaussianity Test. The shading model investigation in Chapter 3.2.4.2 shows that a Gaussianity test that employs Equation 3.17 as its shading model is capable of accurately distinguishing foreground from background, using a single Gaussianity test threshold, in all scenarios except when no sudden illumination changes are present. While our solution would still benefit from the frame-differencing module we decide to exclude it from the background modeling phase of our solution for the sake of computational complexity. We believe this to be a reasonable trade-off since the legibility of foreground objects, and hence the classification accuracy of our solution, is greatly improved by the sliding window to the computation of moments (as described in Section 5.2.2). Furthermore, Figure 5.3 shows that the classification accuracy results do not change drastically while Figure 5.4 shows the improvement to computation time that this change makes. Figure 5.1 shows the part of the Ng *et al.*'s solution that is modified (marked in blue).

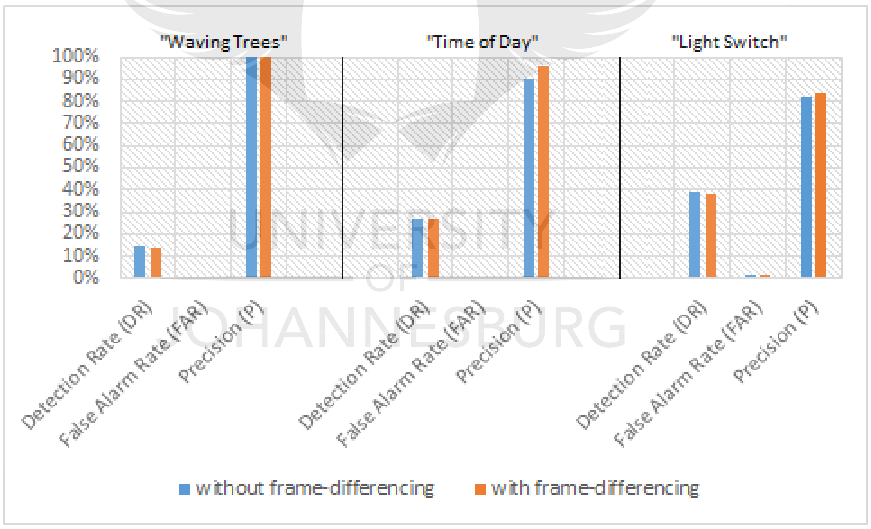

FIGURE 5.3: Comparison of classification accuracy statistics when M=17.

The absence of the frame-differencing module from background modeling phase requires an adjustment to be made to the background maintenance and morphological filtering module of our solution.

5.2.3.1 Background maintenance

Since we no longer include the frame-differencing module for background modeling phase, the background maintenance scheme described in Chapter 3.3.5 is no longer valid. We

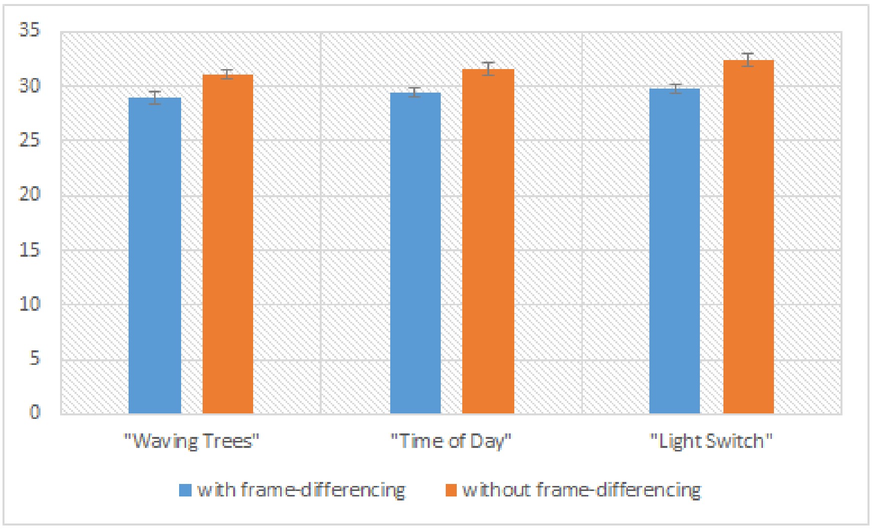

FIGURE 5.4: Comparison of computation time statistics, in frames-per-second (FPS), when M=17.

implement the selective maintenance scheme discussed in Chapter 2.1.7.2. Figure 5.1 shows the part of the Ng *et al.*'s solution that is modified (marked in pink).

5.2.3.2 Morphological operations

Our new pixel neighbourhood strategy for the computation of moments produces foreground regions that are slightly larger than actual foreground objects. Ng *et al.*'s original solution produces a more sparse foreground mask which benefits from morphological operations that first grow (opening) the foreground regions before shrinking them (closing). We employ these same morphological operations but in reverse order. Figure 5.1 shows the part of the Ng *et al.*'s solution that is modified (marked in orange).

5.2.4 Parameter selection

Ng *et al.*'s solution has three major parameters: the size of the pixel neighbourhood that is sampled for the computation of moments, M, the learning rate of the background maintenance module, α, and the Gaussianity Test threshold, τ.

The value of the sum of Gaussianity test moments, and hence the test statistic value, is scaled according to the value of M, according to Equation 3.15. The ideal Gaussianity Test threshold only depends on the amount of variation present in the background and is therefore empirically set. The Gaussianity test statistics that result from the use of Equation 3.10 suggest that the threshold, τ, should be far smaller than the one suggested by Ng *et al.*. We set $\tau = 0.01$ for all three sequences despite the fact that the "Waving

Trees" sequence would benefit from a slightly larger value for τ. Figure 5.1 shows the part of the Ng *et al.*'s solution that is modified (marked in purple).

We assume that M and α are independent of one another and can therefore be optimized separately. When optimizing M we set $\alpha = 0.1$ and when optimizing α we set $M = 17$.

5.2.4.1 Sliding window pixel neighbourhood for moment computation (M)

We compute the same classification accuracy statistics as in Chapter 4.2.2.1 but for a range of M values. As before, classification accuracy statistics are gathered for all three sequences. The results of this experiment are shown in Figures 5.5, 5.6 and 5.7.

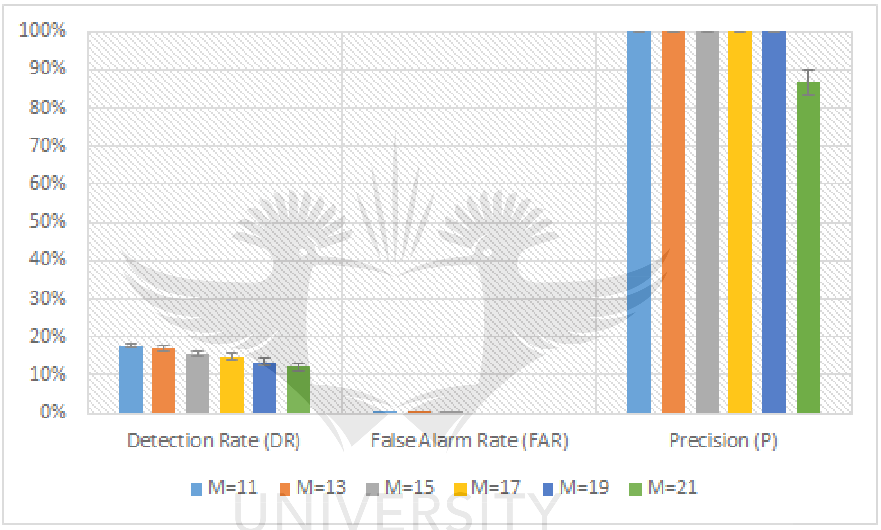

FIGURE 5.5: The results of the parameter optimization experiments for M, including the detection rate, false alarm rate and precision for the "Waving Trees" dataset.

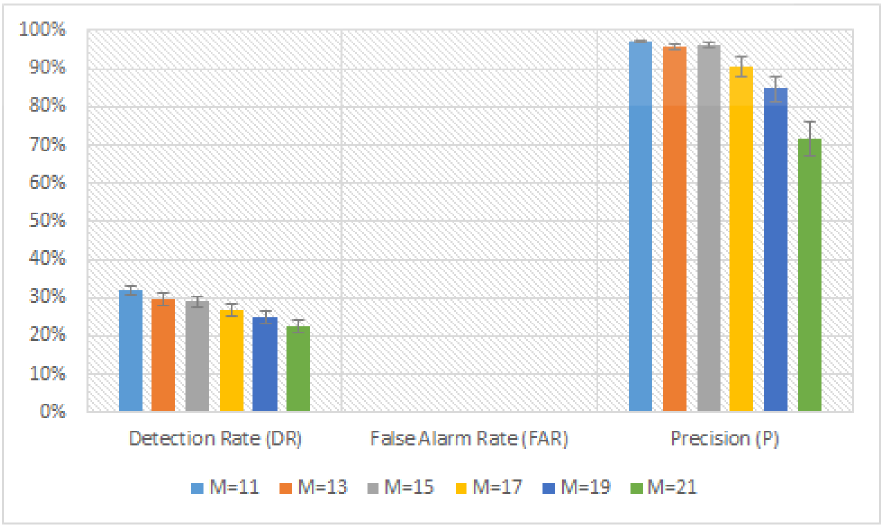

FIGURE 5.6: The results of the parameter optimization experiments for M, including the detection rate, false alarm rate and precision for the "Time of Day" dataset.

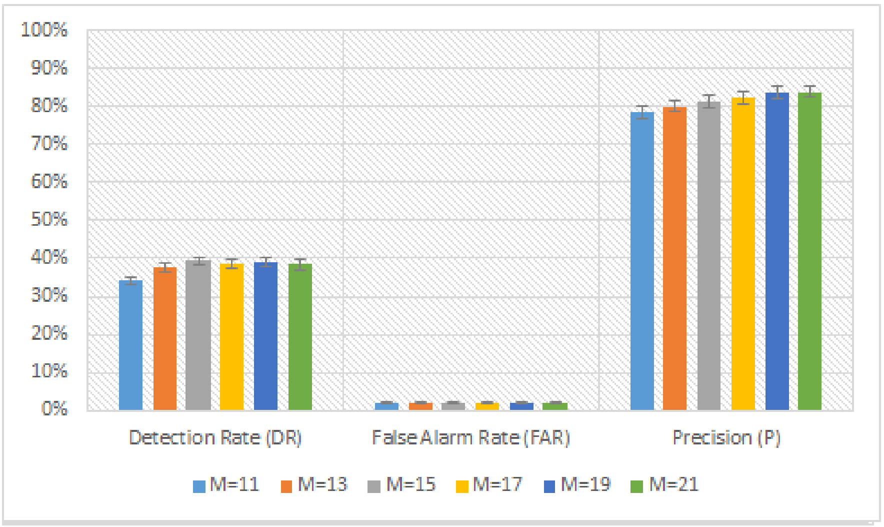

FIGURE 5.7: The results of the parameter optimization experiments for M, including the detection rate, false alarm rate and precision for the "Light Switch" dataset.

In the "Waving Trees" dataset it is evident that the detection rate, detection rate standard deviation, false alarm rate and precision of the solution increases as M decreases. We decide that $M = 11$ is the optimal value for this sequence.

In the "Time of Day" dataset it is evident that, similarly to the "Waving Trees" dataset, the detection rate, detection rate standard deviation, false alarm rate and precision of the solution increases as M decreases. It is also evident from the precision statistic, and its standard deviation, that the solution quickly deteriorates as M increases above 15. We decide that $M = 11$ is the optimal value for this sequence.

In the "Light Switch" dataset we see that the precision statistic varies mostly noticeably and increases as M increases while the detection rate peaks at $M = 15$. The false alarm rate show no obvious correlation. We decide that $M = 19$ is the optimal value for this sequence.

From these results of all three sequences we can see that a trade-off exists between the classification accuracy statistics of the "Light Switch" sequence and those of the "Waving Trees" and "Time of Day" sequences. Taking this into consideration we select $M = 15$ as the optimal parameter value since it is the largest M value possible, so as to accommodate the "Light Switch" sequence, before the accuracy of the "Time of Day" sequence deteriorates too drastically.

Preliminary testing indicates that the increase in computation time as M increases is negligible. It is therefore not necessary to gather and compare computation time statistics, for the range of M values, for the purpose of parameter optimization.

Figure 5.1 shows the part of the Ng *et al.*'s solution that is modified (marked in red).

5.2.4.2 Learning rate (α)

We implement the same parameter optimization strategy as for M; classification accuracy statistics for a range of α values are computed, for all three sequences. The results of this experiment are shown in Figures 5.8, 5.9 and 5.10.

Figure 5.8 indicates that the classification accuracy statistics of the "Waving Trees" sequence is negatively affected as α increases while the statistics corresponding to the "Light Switch" sequence are positively affected up to a learning rate of 0.7 for its detection rate and 0.4 for its precision. The detection rate of the "Time of Day" sequence increases slowly as α increase while its precision peaks at $\alpha = 0.1$ The false alarm rate shows no obvious correlation. We decide that when $\alpha = 0.3$ the solution is best able to accommodate all three sequences.

We assume that the magnitude of the learning rate does not affect computation time since the same number of computations are performed regardless of α's value. It is therefore not necessary to gather and compare computation time statistics, for the range of α values, for the purpose of parameter optimization.

Figure 5.1 shows the part of the Ng *et al.*'s solution that is modified (marked in pink).

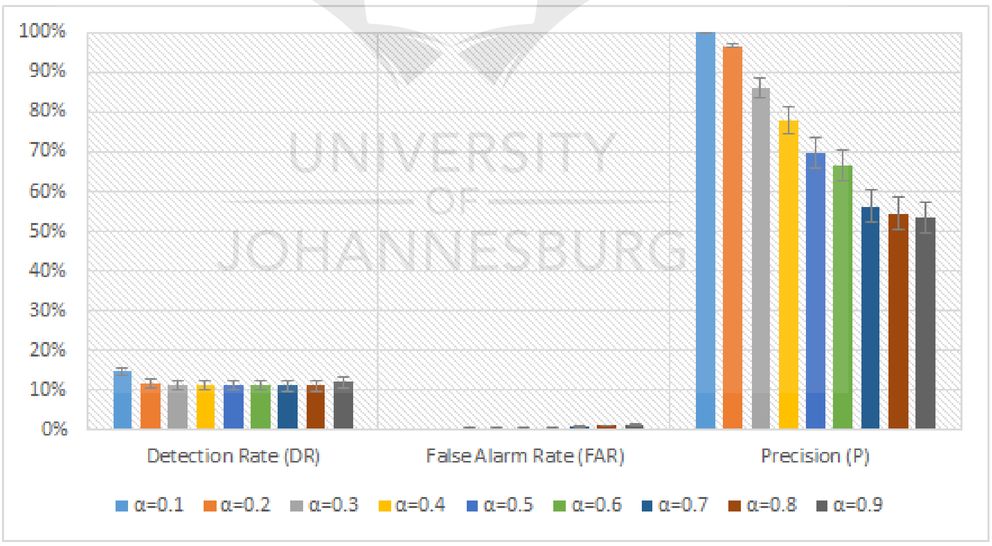

FIGURE 5.8: The results of the parameter optimization experiments for α, including the detection rate, false alarm rate and precision for the "Waving Trees" dataset.

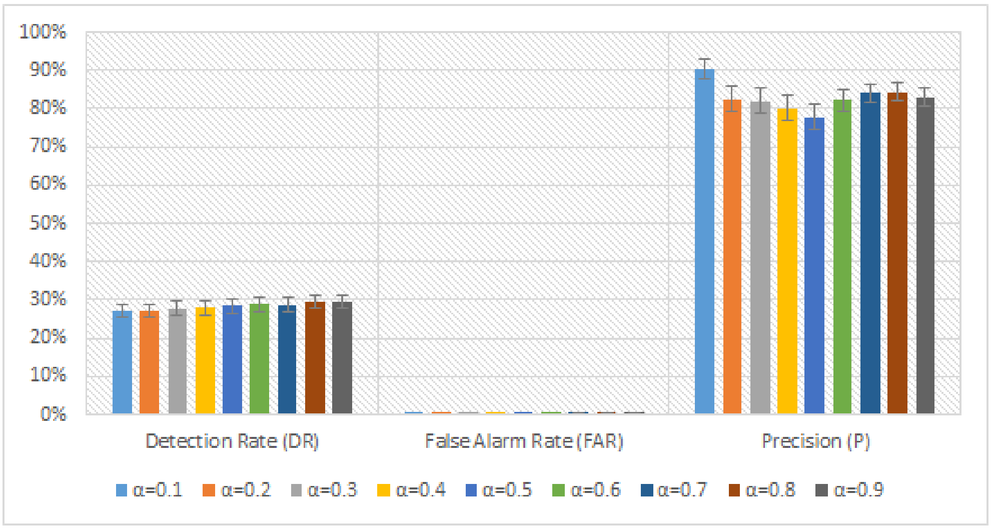

FIGURE 5.9: The results of the parameter optimization experiments for α, including the detection rate, false alarm rate and precision for the "Time of Day" dataset.

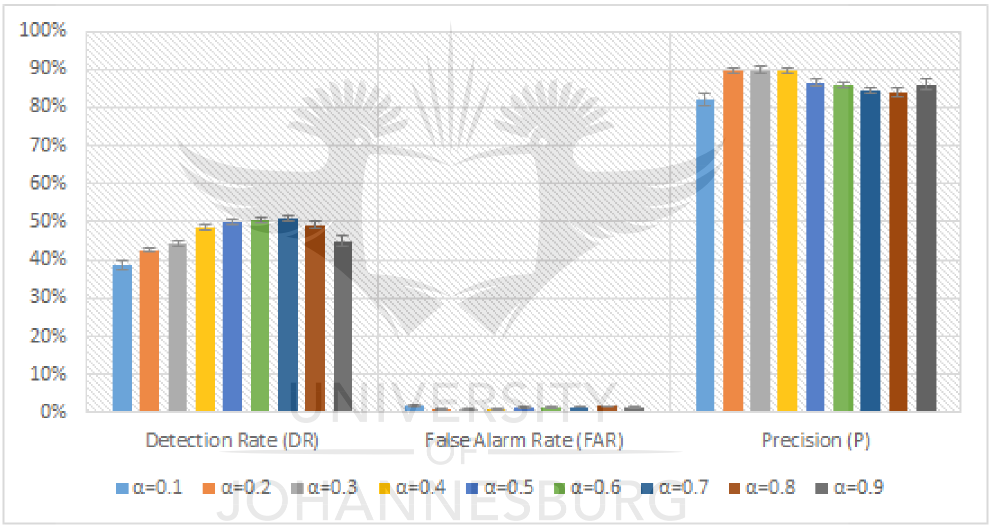

FIGURE 5.10: The results of the parameter optimization experiments for α, including the detection rate, false alarm rate and precision for the "Light Switch" dataset.

5.3 Experimental results

5.3.1 Classification accuracy

The new classification accuracy statistics of the "Waving Trees", "Time of Day" and "Light Switch" sequences are provided inFigures 5.11, 5.12 and 5.13, respectively. These are average values computed using the DR, FAR and P values corresponding to the 15 hand-segmented ground-truths.

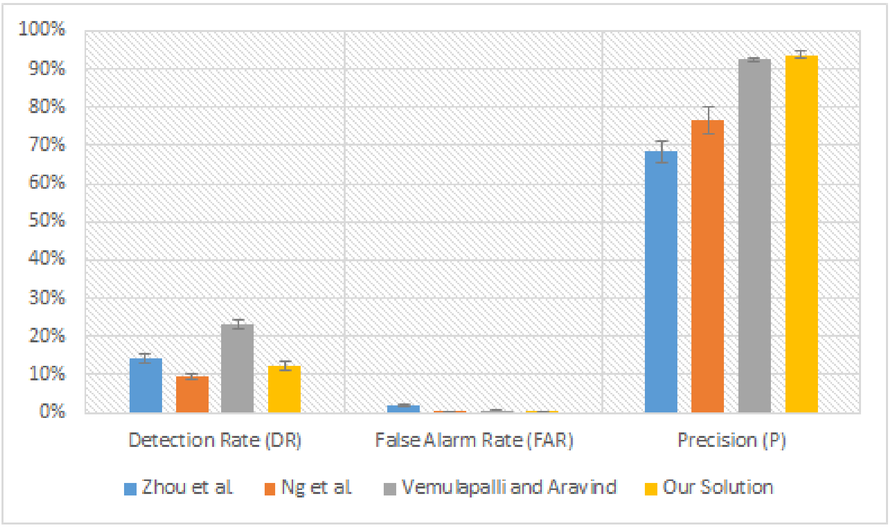

FIGURE 5.11: The results of the classification accuracy experiments for the "Waving Trees" sequence.

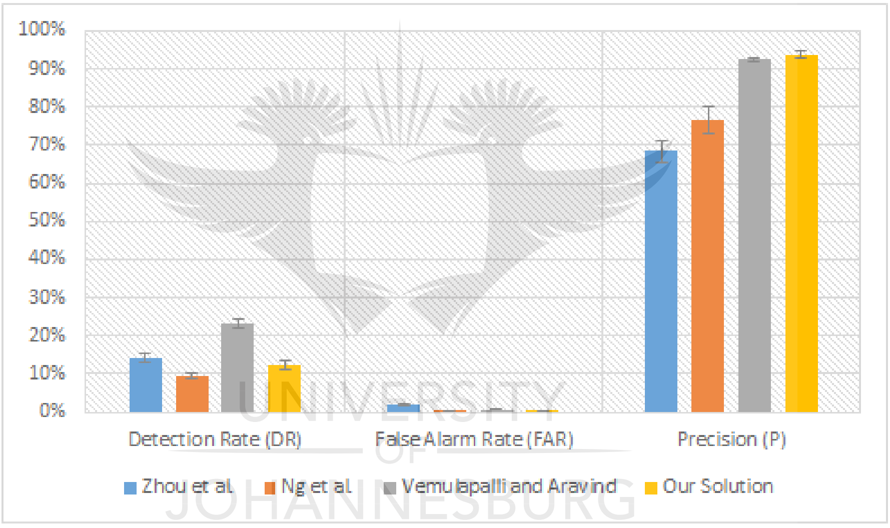

FIGURE 5.12: The results of the classification accuracy experiments for the "Time of Day" sequence.

5.3.2 Computation time

The new computation time statistics are provided in Figure 5.14. These are average FPS values computed from 15 instances.

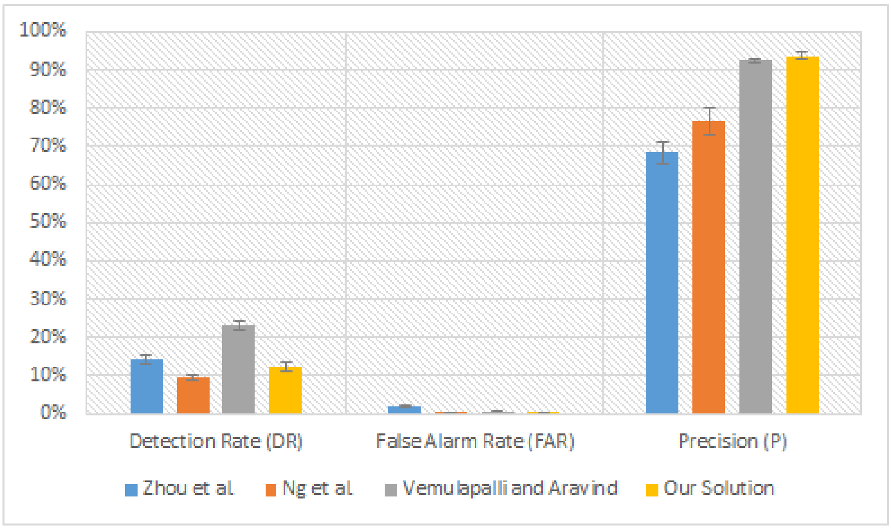

FIGURE 5.13: The results of the classification accuracy experiments for the "Light Switch" sequence.

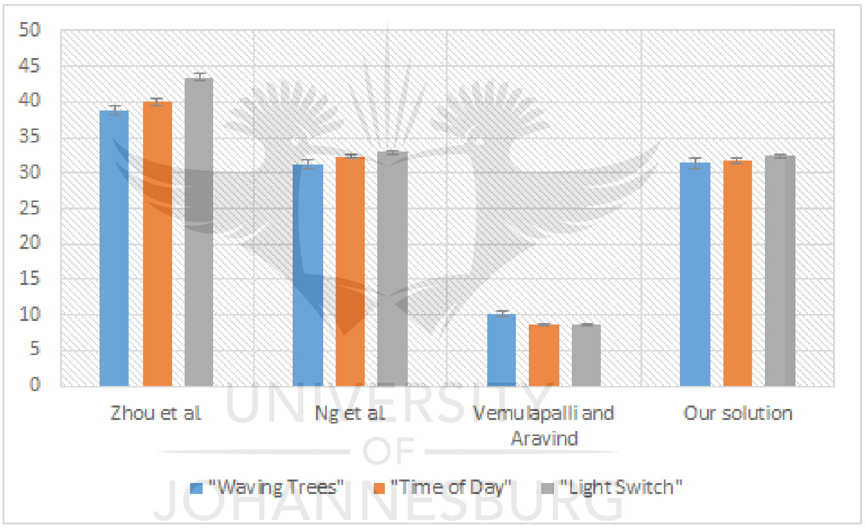

FIGURE 5.14: The results of the computation time experiment for all three sequences.

5.4 Statistical analysis of results

5.4.1 Experiment 1: Classification accuracy

5.4.1.1 "Waving Trees"

From the results shown in Figure 5.11 we can see that Vemulapalli and Aravind's solution still provides the best detection rate which is 1.87 times higher than our solution followed by Zhou *et al.*'s solution which is 1.15 times higher than our solution. Our detection rate is 1.32 times higher than Ng *et al.*'s. Ng *et al.*'s detection rate has the smallest standard deviation which is 1.65 times less than ours, followed by Vemulapalli and Aravind's detection rate standard deviation which is 1.45 times less than ours. Our detection rate standard deviation is 1.05 times less than Zhou *et al.*'s.

Ng *et al.*'s solution still provides the best false alarm rate which is 2.39 times less than our solution. Our false alarm rate is 2.01 times less than Vemulapalli and Aravind's solution and 5.8 times less than Zhou *et al.*'s solution. Ng *et al.*'s detection rate still has the smallest standard deviation which is 8.77 times less than ours, followed by Vemulapalli and Aravind's detection rate standard deviation which is 3 times less than ours. Our detection rate standard deviation is 11.33 times less than Zhou *et al.*'s.

Our solution provides the best precision which is 1.01 times higher than Vemulapalli and Aravind's solution, 1.22 times higher Ng *et al.*'s solution and 1.37 times higher than Zhou *et al.*'s solution. Vemulapalli and Aravind's solution still provides the smallest precision standard deviation which is 2.83 times less than ours. Our precision standard deviation is 8.45 times less than Zhou *et al.*'s solution and 15.36 times less Ng *et al.*'s solution.

5.4.1.2 "Time of Day"

From the results shown in Figure 5.12 we can see that our solution provides the best detection rate which is 1.08 times higher than Vemulapalli and Aravind's solution, 1.64 times higher than Ng *et al.*'s solution and 2.4 times higher than Zhou *et al.*'s solution. Our detection rate has the largest standard deviation which is 1.1 times higher than Vemulapalli and Aravind's solution, 1.2 times higher than Ng *et al.*'s solution and 2.09 times higher Zhou *et al.*'s solution.

Our solution provides the best false alarm rate which is 1.11 times less than Vemulapalli and Aravind's solution, 1.83 times less than Ng *et al.*'s solution and 341.4 times less than Zhou *et al.*'s solution. Vemulapalli and Aravind's false alarm rate has the smallest standard deviation which is 10.97 times less than our solution. Our false alarm rate standard deviation is 4.5 times less than Ng *et al.*'s solution and 110.55 times less than Zhou *et al.*'s solution.

Our solution provides the best precision which is 1.02 times higher than Vemulapalli and Aravind's solution, 1.31 times higher Ng *et al.*'s solution and 37.35 times higher than Zhou *et al.*'s solution. Zhou *et al.*'s solution still provides the smallest precision standard deviation which is 26.41 times less than our solution, followed by Vemulapalli and Aravind's solution which is 2.87 times less than ours. Our precision standard deviation is 11.49 times less Ng *et al.*'s solution.

5.4.1.3 "Light Switch"

From the results shown in Figure 5.13 we can see that our solution provides the best detection rate which is 1.81 times higher than Ng *et al.*'s solution, 4.22 times higher than Zhou *et al.*'s solution and 7.67 times higher than Vemulapalli and Aravind's solution. Vemulapalli and Aravind's detection rate has the smallest standard deviation which is 3.77 times less than our solution. Our solution has the smallest detection rate standard deviation which is 1.66 times less than Zhou *et al.*'s solution and 5.94 times less than Ng *et al.*'s solution.

Our solution provides the best false alarm rate which is 3.95 times less than Zhou *et al.*'s, 8.95 times less than Ng *et al.*'s solution and 25.42 times less than Vemulapalli and Aravind's solution solution. Our false alarm rate has the smallest standard deviation which is 13.53 times less than Ng *et al.*'s solution, 57.96 times less than Zhou *et al.*'s solution and 2193.04 times less than Vemulapalli and Aravind's solution.

Our solution provides the best precision which is 1.88 times higher than Zhou *et al.*'s solution, 2.54 times higher Ng *et al.*'s solution and 6.58 times higher than Vemulapalli and Aravind's solution. Our solution has the smallest precision standard deviation which is 2.13 times less than Vemulapalli and Aravind's solution, 2.61 times less than Ng *et al.*'s solution and 13.64 times less than Zhou *et al.*'s solution.

5.4.2 Computation time

From the results shown in Figure 5.14 we can see that our solution provides computation time statistics very similar to those produced by Ng *et al.*'s solution. It is capable of processing a 160×120 pixel sequence at more than 31.36 frames per second using a desktop computer.

5.5 Discussion of results

5.5.1 Classification accuracy

Our solution has improved upon Ng *et al.*'s solution for all classification accuracy statistics except for the detection rate standard deviation, false alarm rate and false alarm rate standard deviation of the "Waving Trees" sequence as well as the detection rate standard deviation of the "Time of Day" sequence.

Furthermore, our solution is superior to all solutions with regard to the precision of the "Waving Trees" sequence, the detection rate, false alarm rate, precision and precision standard deviation of the "Time of Day" sequence as well as all the classification accuracy statistics of the "Light Switch" sequence.

Our solution is superior to all of the other solutions with regard to its minimum performance. This is a valuable characteristic since it reflects the versatility of our solution. It produced a minimum detection rate of 12.39%, a maximum false alarm rate of 0.8%, a minimum precision of $91,01\%$ and a maximum standard deviation of 1.73%.

The main weakness of our solution is stationary foreground objects. This occurs in the "Waving Trees" sequence when the person pauses in front of the tree and in both the "Time of Day" and "Light Switch' sequence when the person sits down. The solution adapts very quickly to new background pixel values in order to tolerate illumination changes. When a foreground object stops moving the background model quickly assimilates it into the background model. The object is misclassified as background until it moves again.

5.5.2 Computation time

The increase in computational complexity caused by the sliding window approach to the computation of moments is counteracted by the decrease in computational complexity caused by the exclusion of the frame-differencing module from the background modeling phase of our solution. The resulting computation speed of our solution is competitive when compared to the other solutions.

5.6 Conclusion

In this chapter we set out to improve the solution provided by Ng *et al.* in order to make it better than all three of the solutions that were investigated. We believe we were successful in this endeavor, having compiled a solution with superior minimum classification accuracy as well as a competitive processing speed.

Chapter 6

Conclusions and Future Work

6.1 Introduction

This chapter will summarize and discuss the dissertation as a whole. Finally, potential future avenues of research are discussed.

6.2 Conclusions

In the field of computer vision, numerous background modeling techniques have successfully addressed the presence of gradual illumination changes in a scene. However, the presence of sudden illumination changes continues to be a challenging problem.

The aim of this dissertation was to identify potential solutions to this problem in recent literature, characterize their capabilities and compare their respective performances when encountering a challenging dataset. Furthermore, this dissertation aimed to improve one of these solutions with the hopes of outperforming all of them. We constrained any potential solution to only make use of a single video sequence of a stationary scene as an input, and to employ an on-line adaptive background model exclusively.

We identified and investigated three solutions and implemented them on a GPU. We then compared each to one another with respect to classification accuracy and computation time. This was accomplished using statistical analysis performed on a number of commonly-used metrics. We determined that the solution proposed by Ng *et al.* is the best-suited for our problem space; it is the only solution capable of producing competitive classification accuracy and computation time statistics for all the sequences in a challenging dataset.

Potential improvements to this solution were investigated, implemented and verified. We were able to assemble a solution capable of producing a minimum detection rate of 12.39%, a maximum false alarm rate of 0.8% and a minimum precision of $91,01\%$, all with a standard deviation of no more than 1.73%. It is superior to all of the other solutions with regard to its minimum accuracy of classification. Furthermore, the solution is capable of computing 160×120 frames at a minimum of 31.36 FPS.

6.3 Future work

Future work will involve further investigation into robust classification in the presence of sudden illumination changes with hopes to further improve Ng *et al.*'s solution with regard to persistent, accurate foreground detection as well as computational efficiency. It is an active area of research in computer vision and it will be beneficial to review recently published literature.

6.3.1 Multiple Gaussianity Tests

The Gaussianity Test that is employed by the improved solution is tuned to a specific bandwidth, standard deviation and scale. The success of these selected parameters can be influenced by camera properties or the size of foreground objects. We believe it will be beneficial to explore the possible of using multiple or successive Gaussianity Tests in order to make the solution more robust.

6.3.2 Parameter selection automation

In Chapters 4 and 5 we found that solutions often benefit from different parameter values for different sequences. These are often influenced by the amount of variation in the background of a sequence or the type of lighting changes that occur. We believe it possible to automate the Gaussianity Test threshold, τ and pixel neighbourhood for the computation of moments, M, according to these phenomena, which will make the solution more adaptable to different environments.

6.3.3 Real-time computation

Algorithm optimization will be investigated to reduce the computation time of the improved solution.

6.3.4 Datasets

We will investigate how the improved solution reacts to other datasets. While we were were able to optimize the parameter tuning/selection for the "Wallflower" dataset, other datasets may vary with respect to the amount of noise present, frame size or the types of background dynamics that are encountered and the performance of our solution may also vary.

Appendix A

PRASA 2014 Paper

Comparison of Background Subtraction Techniques Under Sudden Illumination Changes

C.J.F. Reyneke, P.E. Robinson and A.L. Nel
Department of Electrical and Electronic Engineering Science
Faculty of Engineering and the Built Environment
University of Johannesburg
email: corius.reyneke@gmail.com; {philipr, andren}@uj.ac.za

***Abstract*—This paper investigates three background modelling techniques that have potential to be robust against sudden and gradual illumination changes for a single, stationary camera. The first makes use of a modified local binary pattern that considers both spatial texture and colour information. The second uses a combination of a frame-based Gaussianity Test and a pixel-based Shading Model to handle sudden illumination changes. The third solution is an extension of a popular kernel density estimation (KDE) technique from the temporal to spatio-temporal domain using 9-dimensional data points instead of pixel intensity values and a discrete hyperspherical kernel instead of a Gaussian kernel.**

A number of experiments were performed to provide a comparison of these techniques in regard to classfication accuracy.

***Index Terms*—background subtraction, sudden illumination changes.**

I. Introduction

Background subtraction techniques have traditionally been applied to object detection in computer vision systems and have since become a fundamental component for many applications ranging from human pose estimation to video surveillance. The goal is to remove the background in a scene so that only the interesting objects remain for further analysis or tracking. Techniques such as these are especially useful when they can identify object regions without prior information and when they can perform in real-time.

Real-life scenes often contain dynamic backgrounds such as swaying trees, rippling water, illumination changes and noise. While a number of techniques are effective at handling these, sudden illumination changes such as a light source switching on/off or curtains opening/closing continue to be a challenging problem for background subtraction [1]. In recent years a number of new segmentation techniques have been developed that are robust to sudden illumination changes but only for certain scenes. Our aim is to eventually identify the best-performing solution, improve upon it, and implement it on a GPU for real-time application.

II. Related Work

A number of texture-based methods have developed to solve the problem of sudden illumination changes. Heikkila [2], Xie *et al.* [3] and Pilet *et al.* [4] make use of robust texture features [5]. Heikkila makes use of local binary pattern histograms as background statistics. Xie *et al.* assumes that pixel order values in local neighbourhoods are preserved in the presence of sudden illumination changes. They provide an output image by classifying each pixel by its probability of order consistency [3]. Pilet *et al.* make use of texture and colour ratios to model the background and segment the foreground using an expectation-maximization framework [4]. Texture-based features work well, but only in scenes with sufficient texture; untextured objects prove to be a difficulty.

Another way of dealing with sudden illumination changes is to maintain a representative set of background models [5]. These record the appearance of the background under different lighting conditions and alternate between these models depending on observation. The techniques that make use of this approach mostly differ in their method of deciding which model should be used for the current observation. Toyama *et al.* [6], implement the Wallflower system which chooses the model as the one that produces the lowest number of foreground pixels. This proves to be an unreliable criterion for real-world scenes. Stenger *et al.* [7] make use of hidden Markov models but in most cases, sharp changes occur without any discernible pattern. Also, Stenger *et al.* and Toyama *et al.* require off-line training procedures and consequently cannot incorporate new real-world scenes into their models during run-time [8]. Sun [9] implements a hierarchical Gaussian Mixture Model (GMM) in a top-down pyramid structure. At each scale-level a mean pixel intensity is extracted and is matched to the best model of its upper-level GMM. While mean pixel intensity is useful for the detection of illumination changes, it is also sensitive to changes caused by the foreground. Additionally, the Hierarchical GMM does not exploit any spatial relationships among pixels which can output incoherent segmentation [5]. Dong *et al.* [10] employ principle component analysis (PCA) to build a number of subspaces where each represent a single background appearance. The foreground is segmented by selecting the subspaces which produces minimum reconstruction error. However, their work does not discuss how the system reacts to repetitive background movements.

More recently, Zhou *et al.* [11], Ng *et al.* [1] and Vemulapalli [12] have developed techniques that have potential to handle, and even be robust to, sudden illumination changes. These will be discussed in more detail in section III.

III. Proposed Solutions

A. Background Modeling using Spatial-Colour Binary Patterns (SCBP)

This approach makes used of a novel feature extraction operator, the Spatial-Colour Binary Pattern (SCBP), which takes spatial texture and colour information into consideration [11]. It is an extension of a local binary pattern which is adapted to be centre-symmetrical and to consider only two colour channels for the sake of computational efficiency. For the sake of simplicity all processes relating to this solution apply to a single pixel and are performed on all the pixels in an image.

$$\begin{aligned}\text{SCBP}_{2N,R}(x_c, y_c) = CSLBP_{2N,R}(x_c, y_c)\\ +2^{N+1}f(R_c, G_c|\gamma) + 2^{N+2}f(G_c, B_c|\gamma)\end{aligned} \tag{1}$$

$$f(a, b|\gamma) = \begin{cases} 1, & a > \gamma b \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Where R_c, G_c and B_c are the three colour channels of the centre pixel (x_c, y_c) and $\gamma > 1$ is a noise suppression factor. The Centre-Symmetrical Local Binary Pattern (CSLBP) is defined as:

$$\text{CSLBP}_{2N,R}(x_c, y_c) = \sum_{i=0}^{N-1} 2^i s(g_i - g_{i+N}) \tag{3}$$

$$s(x) = \begin{cases} 1, & x >= 0 \\ 0, & x < 0 \end{cases} \tag{4}$$

Where g_i is the grey value of the neighbouring pixel at index i and N is the number of neighbours to be compared. The neighbours are evenly distributed on a circle around the centre pixel with radius R. If a neighbour value does not fall exactly on a pixel it is estimated using bilinear interpolation.

An SCBP histogram is computed over a circular region of radius R_{region} around the pixel. Using this as a feature vector a model consisting of K SCBP histograms is built, each with their own weight, such that $w_0 + w_1 + w_K = 1.0$ in decreasing order. At the start these model histograms will be identical but will begin to differ as their respective weights are updated.

An SCBP histogram is calculated for each new frame and then compared to the model histograms using a proximity measure. This measure adds the mutual minimum histogram bins of the current frame and each model histogram that comprise the background model. The proximity measure is defined as follows:

$$\cap(\bar{a}, \bar{b}) = \sum_{n=0}^{N-1} min(a_n, b_n) \tag{5}$$

Where $\bar{a}$ and $\bar{b}$ are histograms and N is the number of bins in each histogram.

The model is updated selectively depending on the value of the calculated proximity measures. If all the proximity measures are below the threshold, T_p, the model histogram with the lowest weight has its bins replaced with those of the current frame. If at least one proximity measure is above the threshold then only the background histogram that produced the highest proximity measure is updated using the following formula:

$$\bar{m_k} = \alpha_b \bar{h} + (1 - \alpha_b)\bar{m_k} \tag{6}$$

Where $\bar{m_k}$ is the model SCBP histogram, $\bar{h_k}$ is the current frame SCBP histogram and α_b is a learning rate such that $\alpha_b \in [0, 1]$.

Furthermore, the weights of the model are updated as follows:

$$w_k = \alpha_w M_k + (1 - \alpha_w)w_k \tag{7}$$

Where α_w is a learning rate such that $\alpha_w \in [0, 1]$ and M_k is 1 for the best-matching histogram and 0 for the rest.

T_p is an adaptive threshold that is maintained (for each pixel). The advantage of this is that static regions become more sensitive while dynamic regions have a higher tolerance. The threshold is updated as follows:

$$T_p(x, y) = \alpha_p(s(x, y) - 0.05) + (1 - \alpha_p)T_p(x, y) \tag{8}$$

Where α_p is a learning rate such that $\alpha_p \in [0, 1]$ and $s(x, y)$ is a similarity measure of the highest value between the current frame's SCBP histogram bins and those of the model histograms.

In order to determine the foreground mask the value for n in the following equation is first determined.

$$w_0 + w_1 + ... + w_n \le T_w \tag{9}$$

Where the weights have been sorted into descending order. T_w is a fixed threshold and is dependent upon the number of histograms that make up the model. The calculated value for n determines the number of corresponding model histograms which are selected to be part of the background model. The advantage of using this weighted technique is that the persistence of a model histogram is directly related its weight. Persistence needs to be considered because all of the model histograms are not necessarily produced by background processes; the bigger the weight of a model histogram, the higher the probability it has of being a background histogram.

If the proximity measure values for all the background model histograms are smaller than the threshold T_w the pixel is classified as background. If the proximity value for at least one of the background models is greater than T_w then the pixel is classified as foreground.

Furthermore, object contours are refined using a statistical operator to reduce false positives. These are based upon two assumptions. A pixel should only be successfully classified as part of the foreground if its intensity value deviates much from the average pixel intensity of its pixel neighbourhood and its colour composition changes much from that of its pixel neighbourhood. Ergo, a binary mask is constructed as follows and is convolved with the output of the foreground detection module:

$$\Omega_i = \begin{cases} 1, & \text{if } [d_i >= \xi \text{std}_i]\&[d_i/\bar{g}_i \ge \varepsilon_1], \\ 1, & \text{if}||(r_i, g_i, b_i) - (\bar{r}_i, \bar{g}_i, \bar{b}_i)||_2 \ge \varepsilon_2, \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

Where $d_i = abs(g_i - \bar{g_i})$ is the absolute deviation of intensity from the average and r, g and b are chromaticity coordinates calculated by $r = R/(R+G+B)$, $g = G/(R+G+B)$ and $b = B/(R+G+B)$. The parameters ξ, ε_1 and ε_2 are tuning parameters.

Finally, the average and standard deviation of the resulting background pixels are updated as follows:

$$\bar{g_i} = \beta g_i + (1-\beta)\bar{g_i} \tag{11}$$

$$std_i = \sqrt{\beta(g_i - \bar{g_i})^2 + (1-\beta)std_i^2} \tag{12}$$

Where β is a learning rate such that $\beta \in [0,1]$ The chromaticity coordinates, $\bar{r_i}, \bar{g_i}, \bar{b_i}$, are updated in the same way as was done for $\bar{g_i}$.

B. Background Modeling using a Shading Model and a Gaussianity Test

The method proposed by Ng *et al* implements a hierarchical framework that uses a combination of a pixel-based Shading Model and a block-based Gaussianity Test [1]. This approach is based on the assumption that camera noise is both spatially Gaussian, and temporally uncorrelated. If the difference of two consecutive frames are taken, only Gaussian noise and foreground objects should remain. Under these assumptions, they deduce that background pixels will be Gaussian distributed and foreground pixels will be non-Gaussian distributed. Therefore background pixels can be distinguished from foreground pixels using a Gaussianity test.

The Gaussianity Test statistic is defined as follows:

$$H(J_1, J_2, J_4) = J_4 + 2J_1^4 - 3J_2^2 \tag{13}$$

Where J_k is a moment defined by the following equation:

$$\hat{J}_k(x,y) = \frac{1}{M^2} \sum_{m=-\frac{M-1}{2}}^{\frac{M-1}{2}} \sum_{n=1-\frac{M-1}{2}}^{\frac{M-1}{2}} [D_t(x+m, y+n)]^k \tag{14}$$

The Gaussianity Test statistic is expected to be close to zero when a set of samples is Gaussian distributed. If a set of samples in a block of size MxM has a Gaussianity Test statistic that is greater than a predefined threshold, τ, then the block is considered to contain foreground pixels.

$$\text{block} = \begin{cases} \text{foreground}, & \text{if } H > \tau \\ \text{background}, & \text{otherwise} \end{cases} \tag{15}$$

However, this assumption does not perform well in the presence of sudden illumination changes. A shading model is implemented to handle these.

Ng *et al* extend the Gaussianity test with a shading model proposed by Skifstad [13] in order to make it robust to sudden illumination changes. The shading model is necessary because the previous assumption that background regions are Gaussian distributed does not hold true in the presence of sudden illumination changes.

The shading model assumes that a pixel intensity can be decomposed into an illumination value and a shading coefficient. It is also assumed that if there is no physical change between two frames, such as a moving object, then the ratio of pixel intensities will be constant and independent of the shading coefficients of the frames:

$$R(x,y) = \frac{I_1(x,y)}{I_2(x,y)} = \frac{L_{i,1}}{L_{i,2}} \tag{16}$$

Under this assumption, if no foreground objects exist in a difference frame, the ratio of pixel intensities should remain constant and therefore be Gaussian distributed. Now, by employing the shading model as an input to the Gaussianity test module, the background model can be made robust to sudden illumination changes. The equation used to generate the moments used in the Gaussianity Test statistic is modified to make use of the pixel intensity ratio:

$$\hat{J}_k(x,y) = \frac{1}{M^2} \sum_{m=-\frac{M-1}{2}}^{\frac{M-1}{2}} \sum_{n=1-\frac{M-1}{2}}^{\frac{M-1}{2}} [R_{gt}(x+m, y+n)]^k \tag{17}$$

Where

$$R_{gt}(x,y) = \frac{BM_{t-1}(x,y)}{I_t(x,y)} \tag{18}$$

The foreground mask is obtained using the following equations:

$$D_t(x,y) = |I_t(x,y) - BM_{t-1}(x,y)| \tag{19}$$

$$(x,y) \subset \begin{cases} \text{foreground}, & \text{if } D_t(x,y) > T_a \\ \text{background}, & \text{otherwise} \end{cases} \tag{20}$$

Where $BM_t(x,y)$ is the intensity value of the background model at the coordinates (x,y) and time t, $I_t(x,y)$ is the intensity value of the current pixel at the coordinates (x,y) and time t and T_a is an adaptive threshold. This equation is only employed in the foreground blocks as classified by the Gaussianity test.

T_a is an adaptive threshold which is calculated using an automatic, iterative method first proposed by Ridler [14]. This method is computationally inexpensive but has the disadvantage of assuming that the scene is bimodal. This assumption predicts that there will be two distinct brightness regions in the image represented by two peaks in the grey-level histogram of the input image. These regions correspond to the object and its surroundings and so it is then reasonable to select the threshold as the grey-level half-way between these two peaks.

The histogram of the current frame, $I_t(x,y)$ is segmented into two parts using a threshold, $T_{iterate}$, which is first set to the middle value (127) of the range of intensities. For each iteration, the sample means of the foreground pixel intensities and the sample means of the background pixel intensities are calculated and a new threshold is determined as the average of these two means. The iterations stop once the threshold converges on a value, normally within about 4 iterations. The following formula describes this process:

$$T_{k+1} = \frac{\sum_{b=0}^{T_k} bn(b)}{2\sum_{b=0}^{T_k} n(b)} + \frac{\sum_{b=T_{k+1}}^{N} bn(b)}{2\sum_{b=T_{k+1}}^{N} n(b)} \tag{21}$$

Where T_k is the threshold at the k^{th} iteration, b is the intensity value and $n(b)$ is the number of occurrences of the value b in the image such that $0 \leq b \leq N$.

Once the foreground mask has been segmented, morphological filtering is performed on the foreground mask in order to remove noise. Ng *et al.* perform one closing operation followed by one opening operation.

The values of the background pixels are updated using the following formula:

$$BM_t(x,y) =$$

$$\begin{cases} BM_{t-1}(x,y), & \text{if } D_t(x,y) \geq T_a \\ I_t(x,y), & \text{if } D_t(x,y) < T_f \\ \alpha I_t(x,y)+ & \\ (1-\alpha)BM_{t-1}(x,y), & \text{if } T_f \leq D_t(x,y) < T_a \end{cases} \tag{22}$$

Where T_f is fixed and smaller than T_a and α is a learning rate such that $\in [0,1]$

C. Background Modeling using Non-parametric Kernel Density Estimation

The solution proposed by Vemulapalli is an extension of the popular kernel density estimation (KDE) technique first proposed by Elgammal *et al.* [15]. They extend the background model from the temporal to spatio-temporal domain by using 3x3 blocks centred at each pixel as 9-dimensional data points instead of individual pixel intensity values [12]. In order to overcome the obvious increase in computational complexity that this would cause, a hyper-spherical kernel is used instead of the typical Gaussian kernel. Each pass of the background modeling module entails comparing the data points of the current frame, $F_0(x,y)$ with those of the previous frames, $F_{i...N}(x,y)$ selected from a window of size $N = 50$. The Euclidean distance is then employed to compare the data points instead of the typical pixel subtraction as used by Elgammal *et al.* Furthermore, two non-parametric background models, long-term and short-term, in order to exploit their respective advantages at eliminating false positive detections.

So, for each new frame a series of $N-1$ Euclidean distances are calculated by comparing each current pixel's data point to its past data-point values. The higher the value of a Euclidean distance, the higher the probability that the current pixel is part of the foreground. These distances are then thresholded to determine if they lie within the radius of the discrete hyperspherical kernel. This radius is a function of the amount of variation present in the background.

$$M = \sum_{i=1}^{N} \phi\left(\frac{||F_0(x,y) - F_i(x,y)||}{r}\right) \tag{23}$$

Where r is the radius of the hyper-sphere and

$$\phi(u) = \begin{cases} 1, & \text{if } u \leq 1, \\ 0, & \text{otherwise} \end{cases} \tag{24}$$

$||F_0(x,y) - F_i(x,y)||$ is the Euclidean distance between the data points $F_0(x,y)$ and $F_i(x,y)$.

The $N-1$ binary outputs of this module are then summed to produce a type of confidence measure, M of whether the current pixel belongs to the background. This sum is then thresholded using a value, T:

$$\frac{M}{N} \leq T \tag{25}$$

The long-term and short-term models are updated using a blind update and selective update mechanism respectively. The blind update adds a new 9-dimensional data point, $F_i(x,y)$, to the sample set regardless of whether it belongs to the background or foreground while the selective update adds the data-point only if it belongs to the background. When a new data point is added the oldest data point is removed from the sample set. The output of both the long-term and short-term models are used as inputs to the foreground detection module. The output of the module is described by the following table:

Long-term model	Short-term model	Output
$O_l(x,y) = 0$	$O_s(x,y) = 0$	$O_{fd}(x,y) = 0$
$O_l(x,y) = 0$	$O_s(x,y) = 1$	$O_{fd}(x,y) = O'_{fd}(x,y)$
$O_l(x,y) = 1$	$O_s(x,y) = 0$	$O_{fd}(x,y) = 0$
$O_l(x,y) = 1$	$O_s(x,y) = 1$	$O_{fd}(x,y) = 1$

TABLE I: The output of the foreground detection module which combines the output of the short-term and long-term background models.

Where $O_l(x,y) = 1$ is the output of the long-term model, $O_s(x,y) = 1$ is the output of the short-term model and $O_{fd}(x,y) = 1$ is the output of the foreground detection module where:

$$O'_{fd}(x,y) = \begin{cases} 1, & \text{if } \sum_{i=-1}^{1}\sum_{j=-1}^{1} \\ & O_s(x-i,y-j)O_l(x-i,y-j) \\ & \neq 0, \\ 0, & \text{otherwise} \end{cases} \tag{26}$$

If the two models agree on an output, the resultant foreground mask will obviously have the same output. If only the long-term model predicts foreground, the foreground mask will prefer the prediction of the short-term model. In the event of the short-term model predicting foreground and the long-term model predicting a background, a check is performed to see if the two models agree on the output of any of the neighbouring pixels being foreground. If this is the case, the pixel is classified as a foreground.

In the event of a sudden illumination change most of the frame will be classified as foreground and will remain so unitl the long term model adapts to the new lighting conditions. Vemulapalli checks whether more than a certain percentage α of the frame is declared as foreground. If this is the case the short-term model is updated using the blind update mechanism so that it avoids false detections and adapts to the new lighting conditions quickly.

IV. Experimental Methodology

A. Dataset

These techniques will be tested with respect to the accuracy of their outputs. In order to accomplish this three sequences from the publicly available Wallflower dataset [6] are used.

The first sequence is named ”Waving Trees” and contains a scene with a typical dynamic background. It has 286 frames where a ground truth is provided for the 247th frame. The second sequence is named ”Time of Day” and contains a scene with gradual illumination changes. It has 5889 frames where a ground truth is provided for the 1850th frame. The third sequence is named ”Light Switch” and contains a scene with sudden illumination changes. It has 2714 frames where a ground truth is provided for the 1865th frame.

B. *Metrics*

For the evaluation of the output accuracy we make use of the detection rate (DR), false alarm rate (FAR) and precision (P) statistics. The formulae for these are provided below:

$$DR = \frac{\#true\_positives}{\#true\_positives + \#false\_negatives} \tag{27}$$

$$FAR = \frac{\#false\_positives}{\#false\_positives + \#true\_negatives} \tag{28}$$

$$P = \frac{\#true\_positives}{\#true\_positives + \#false\_positives} \tag{29}$$

Where $\#true\_positives$ is the number of correctly classified foreground pixels, $\#true\_negatives$ is the number of correctly classified background pixels, $\#false\_positives$ is the number of incorrectly classified foreground pixels and $\#true\_negatives$ is the number of incorrectly classified background pixels.

C. *Selection of Tuning Parameters*

Zhou *et al.* set $R_{region} = 9$, $R = 2$, $N = 4$, $K = 4$, $T_P = 0.65$, $T_B = 0.7$, $\alpha_b = \alpha_w = \beta = 0.01$, $\alpha_p = 0.9$, $\xi = 2.5$ and $\varepsilon_1 = \varepsilon_2 = 0.2$. Zhou *et al* do not specify which similarity measure they used; we investigated two, the L_1 Norm and the Square L_2 Norm. The latter was determined to be best by qualitatively comparing their output. Zhou *et al.* also did not specify how they initialized the weights of the model histograms; we investigated two methods: using a values that decrease linearly and values that decrease exponentially. The latter was determined to be the best by qualitative analysis. Using the exponential curve $w_0 = 0.567$, $w_1 = 0.321$, $w_2 = 0.103$, $w_3 = 0.011$.

Ng *et al.* set $M = 17$ and $\alpha = 0.1$. The value for τ is set empirically for the dataset at hand. For the experiments they perform on the PETS 2006 dataset they set $\tau = 1 \times 10^5$. We set $\tau = 1 \times 10^3$.

Vemulapalli sets $W = 250$, $N = 50$ and $\alpha = 75\%$. However, for the the Waving Trees sequence we set $W = 200$ and $N = 20$ since the 247th frame is used for the ground truth. Vemulapalli does not specify which parameters they used for the hypersphere radius, r, and the threshold, T. We set $r = 1$ and $T = \mu + k\sigma$ where μ is the mean and σ is the standard deviation of the values obtained for M in a frame. k is a positive integer which is set to 6.

V. Experimental Results

A. *Waving Trees*

From these results shown in *fig. 1* we can see that the Zhou *et al.* provides the best detection rate, moderate precision and worst false alarm rate. Ng *et al.* provides the lowest false alarm rate, but the worst precision and detection rate. Vemulapalli provides the best precision and moderate detection and false alarm rates.

B. *Time of Day*

From these results shown in *fig. 2* we can see that Zhou has the worst performance; having the worst detection rate, precision and false alarm rate. Ng *et al.* has a superior precision and false alarm rate as well as a moderate detection rate. Vemulapalli provides the best detection rate and values only slightly worse than Ng *et al.* in regard to precision and false alarm rate.

C. *Light Switch*

From these results shown in *fig. 3* we can see that Zhou *et al.* provides the best detection rate, moderate precision and a moderate false alarm rate. Ng *et al.* has the best precision and false alarm rate, but the worst detection rate. Vemulapalli has a moderate detection rate, but the worst precision and false alarm rate.

The poor performance of the solution proposed by Vemulapalli is largely due to the fact that the sudden illumination check is not triggered by the video sequence. Hence, the blind update mechanism for the short-term model is not employed and the model does not adapt to the new lighting conditions quickly enough.

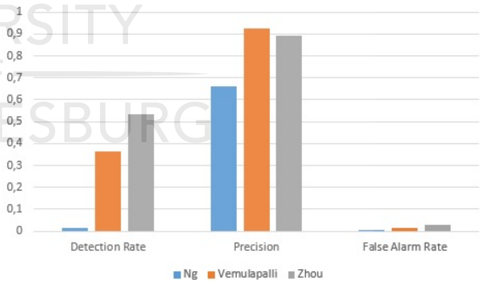

Fig. 1: Results of ”Waving Trees” sequence.

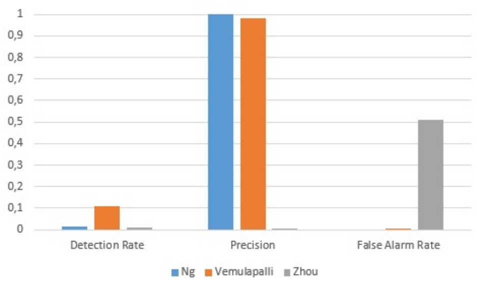

Fig. 2: Results of ”Time of Day” sequence.

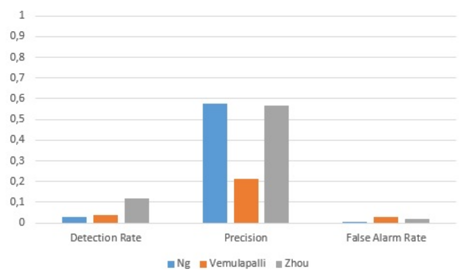

Fig. 3: Results of ”Light Switch” sequence.

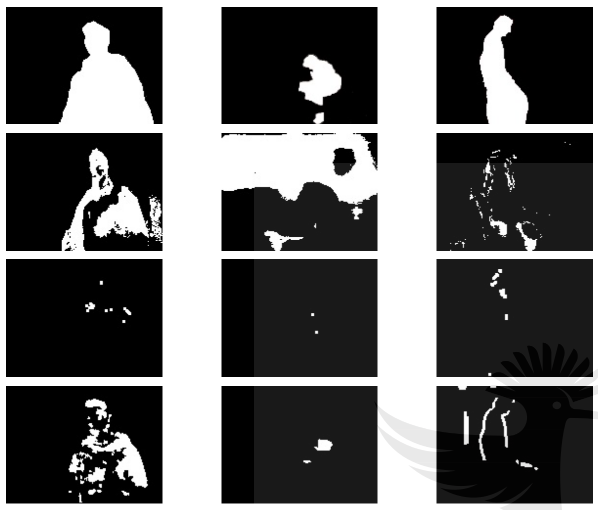

Fig. 4: Foreground segmentation masks of proposed solutions. The columns correspond to the ”Waving Trees”, ”Time of Day” and ”Light Switch” sequences respectively. The first row represents the ground truths while the remaining rows correspond to the outputs of the solutions proposed by Zhou *et al.*, Ng *et al.* and Vemulapalli respectively.

VI. Conclusion

This paper investigates three background modelling techniques that are robust against sudden and gradual illumination changes for a single, stationary camera. The first makes use of a modified local binary pattern that considers both spatial texture and colour information. The second uses a combination of a frame-based Gaussianity Test and a pixel-based Shading Model to handle sudden illumination changes. The third solution is an extension of a popular kernel density estimation (KDE) technique from the temporal to spatio-temporal domain using 9-dimensional data points instead of pixel intensity values and a discrete hyperspherical kernel instead of a Gaussian kernel.

A number of experiments were then performed which provide a comparison of these techniques in regard to classification accuracy.

The SCBP histogram feature approach performs well for simple dynamic backgrounds, but not for scenes that contain any type of illumination changes.

The Shading Model and Gaussianity Test approach provides a sparse foreground mask that is very accurate for all three sequences, but has a poor detection rate.

The KDE approach performs well for simple dynamic backgrounds and scenes that contain gradual illumination changes. However, the mechanism employed to handle sudden illumination changes does not work well due to the use of an unreliable criterion for sudden illumination detection.

VII. Future Work

We plan to further investigate the solution proposed by Ng *et al.* and Vemulapalli. Both have potential to be improved through automatic parameter selection and possibly by integrating the strengths of all three the solutions that were investigated.

References

[1] K. K. Ng, S. Srivastava, and E. Delp, “Foreground segmentation with sudden illumination changes using a shading model and a gaussianity test,” in *Image and Signal Processing and Analysis (ISPA), 7th International Symposium on*, September 2011, pp. 236–240.

[2] M. Heikkila and M. Petikainen, “A texture-based method for modelling the background and detecting moving objects,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 657–662, 2006.

[3] B. Xie, V. Ramesh, and T. Boult, “Sudden illumination change detection using order consistency,” *Image and Vision Computing*, vol. 22, no. 2, pp. 117–125, 2004.

[4] J. Pilet, C. Strecha, and P. Fua, “Making background subtraction robust to sudden illumination changes,” in *Computer Vision–ECCV 2008*. Springer, pp. 567–580, 2008.

[5] J. Li and Z. Miao, “Foreground segmentation for dynamic scenes with sudden illumination changes,” *Image Processing, IET*, vol. 6, no. 5, pp. 606–615, July 2012.

[6] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, “Wallflower: principles and practice of background maintenance,” in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 1, pp. 255-261, 1999.

[7] B. Stenger, V. Ramesh, N. Paragios, F. Coetzee, and J. M. Buhmann, “Topology free hidden markov models: Application to background modeling,” in *Computer Vision, ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1, pp. 294–301. IEEE, 2001.

[8] X. Zhao, W. He, S. Luo, and L. Zhang, “Mrf-based adaptive approach for foreground segmentation under sudden illumination change,” in *Information, Communications Signal Processing, 2007 6th International Conference on*, pp. 1-4, Dec 2007.

[9] Y. Sun and B. Yuan, “Hierarchical gmm to handle sharp changes in moving object detection,” *Electronics Letters*, vol. 40, no. 13, pp. 801–802, 2004.

[10] Y. Dong, T. Han, and G. N. DeSouza, “Illumination invariant foreground detection using multi-subspace learning,” *International journal of knowledge-based and intelligent engineering systems*, vol. 14, no. 1, pp. 31–41, 2010.

[11] W. Zhou, Y. Liu, W. Zhang, L. Zhuang, and N. Yu, “Dynamic background subtraction using spatial-color binary patterns,” in *Image and Graphics (ICIG), 2011 Sixth International Conference on*, pp. 314-319, Aug 2011.

[12] R. Vemulapalli and R. Aravind, “Spatio-temporal nonparametric background modeling and subtraction,” in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, 1145-1152, Sept 2009.

[13] K. Skifstad and R. Jain, “Illumination independent change detection for real world image sequences,” *Computer Vision, Graphics, and Image Processing*, vol. 46, no. 3, pp. 387–399, June 1989.

[14] T. W. Ridler and S. Calvard, “Picture thresholding using an iterative selection method,” *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 8, no. 8, pp. 630–632, Aug 1978.

[15] A. M. Elgammal, D. Harwood, and L. S. Davis, “Non-parametric model for background subtraction,” in *Proceedings of the 6th European Conference on Computer Vision-Part II*, ser. ECCV ’00. Springer-Verlag, 2000, pp. 751–767.

Bibliography

[1] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2001.

[2] R. Szeliski, *Computer vision: algorithms and applications.* Springer, 2010.

[3] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Computer Science Review*, vol. 11–12, no. 0, pp. 31 – 66, 2014.

[4] V. Rabaud and S. Belongie, "Counting crowded moving objects," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 1. IEEE, 2006, pp. 705–711.

[5] W. Zhou, Y. Liu, W. Zhang, L. Zhuang, and N. Yu, "Dynamic background subtraction using spatial-color binary patterns," in *Image and Graphics (ICIG), 2011 Sixth International Conference on*, Aug 2011, pp. 314–319.

[6] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, August 2000.

[7] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. S. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172–185, 2005.

[8] N. M. Oliver, B. Rosario, and A. P. Pentland, "A Bayesian computer vision system for modelling human interactions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 831–843, 2000.

[9] K. Skifstad and R. Jain, "Illumination independent change detection for real world image sequences," *Comput. Vision Graph. Image Process.*, vol. 46, no. 3, pp. 387–399, jun 1989.

[10] S. Liu and C. Fu, “Statistical change detection with moments under time-varying illumination,” *IEEE Transactions on Image Processing*, vol. 7, no. 9, pp. 1258–1268, September 1998.

[11] M. Heikkila and M. Petikainen, “A texture-based method for modelling the background and detecting moving objects,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 657–662, 2006.

[12] B. Kuechler and V. Vaishnavi, “On theory development in design science research: anatomy of a research project,” *European Journal of Information Systems*, vol. 17, no. 5, pp. 489–504, 2008.

[13] (2014) Matlab: Creating plots. (last accessed: 15 November 2014). [Online]. Available: http://www.mathworks.com/help/matlab/creating_plots/chimage8.gif

[14] C. Ware, *Information visualization: perception for design.* Elsevier, 2013.

[15] (2014) Greyscale. (last accessed: 15 November 2014). [Online]. Available: http://en.wikipedia.org/wiki/File:Beyoglu_4671_tricolor.png

[16] (2013) Digital iVision Labs! (last accessed: 15 November 2014). [Online]. Available: http://www.divilabs.com/2013_12_07_archive.html

[17] (2014) Shi-tomasi corner detector & good features to track - opencv 3.0.0-dev documentation. (last accessed: 15 November 2014). [Online]. Available: http://docs.opencv.org/trunk/doc/py_tutorials/py_feature2d/py_shi_tomasi/py_shi_tomasi.html

[18] (2013) [research] feature detectors and descriptors. (last accessed: 15 November 2014). [Online]. Available: http://littlecheesecake.me/blog/13804625/feature-detectors-and-descriptors

[19] (2014) Scale space - a 1000 ways of inventing the gaussian. (last accessed: 15 November 2014). [Online]. Available: http://cseweb.ucsd.edu/classes/fa02/cse252c/scale.pdf

[20] A. Agarwal and B. Triggs, “Hyperfeatures–multilevel local coding for visual recognition,” in *Computer Vision–ECCV 2006.* Springer, 2006, pp. 30–43.

[21] (2003) Image analysis - intensity histogram. (last accessed: 15 November 2014). [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/histgram.htm

[22] K. Kim, J. Kim, J. Choi, J. Kim, and S. Lee, “Depth camera-based 3d hand gesture controls with immersive tactile feedback for natural mid-air gesture interactions,” *Sensors*, vol. 15, no. 1, pp. 1022–1046, 2015.

[23] (2014) Face recognition with opencv - opencv 2.4.9.0 documentation. (last accessed: 15 November 2014). [Online]. Available: http://docs.opencv.org/modules/contrib/doc/facerec/facerec_tutorial.html

[24] P. Z. Peebles, J. Read, and P. Read, *Probability, random variables, and random signal principles*. McGraw-Hill New York, 1987.

[25] L. A. Zadeh, "Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic," *Fuzzy sets and systems*, vol. 90, no. 2, pp. 111–127, 1997.

[26] F. el Baf, T. Bouwmans, and B. Vachon, "Type-2 fuzzy mixture of Gaussians model: application to background modeling," in *Advances in Visual Computing*. Springer, 2008, pp. 772–781.

[27] ——, "Fuzzy statistical modeling of dynamic backgrounds for moving object detection in infrared videos," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2009, pp. 60–65.

[28] T. Bouwmans, "Recent advanced statistical background modeling for foreground detection: A systematic survey," *Recent Patents on Computer Science*, vol. 4, no. 3, pp. 147–176, September 2011.

[29] (2014) How to use background subtraction methods - opencv 3.0.0-dev documentation - go. (last accessed: 15 November 2014). [Online]. Available: http://docs.opencv.org/trunk/doc/tutorials/video/background_subtraction/background_subtraction.html

[30] M. Sezgin *et al.*, "Survey over image thresholding techniques and quantitative performance evaluation," *Journal of Electronic imaging*, vol. 13, no. 1, pp. 146–168, 2004.

[31] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, Jan 1979.

[32] T. W. Ridler and S. Calvard, "Picture thresholding using an iterative selection method," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 8, no. 8, pp. 630–632, Aug 1978.

[33] (2011) Thresholding - NI Vision Concepts Help - National Instruments. (last accessed: 15 November 2014). [Online]. Available: http://zone.ni.com/reference/en-XX/help/372916L-01/nivisionconcepts/thresholding/

[34] M. H. Sigari, N. Mozayani, and H. R. Pourreza, "Fuzzy running average and fuzzy background subtraction: concepts and application," *International Journal of Computer Science and Network Security*, vol. 8, no. 2, pp. 138–143, 2008.

[35] L. Maddalena and A. Petrosino, "A fuzzy spatial coherence-based approach to background/foreground separation for moving object detection," *Neural Computing and Applications*, vol. 19, no. 2, pp. 179–186, 2010.

[36] J. Li and Z. Miao, "Foreground segmentation for dynamic scenes with sudden illumination changes," *Image Processing, IET*, vol. 6, no. 5, pp. 606–615, July 2012.

[37] F. Porikli and O. Tuzel, "Human body tracking by adaptive background models and mean-shift analysis," in *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance.* IEEE, 2003, pp. 1–9.

[38] D. Magee, "Tracking multiple vehicles using foreground, background and motion models," *Image and Vision Computing*, vol. 22, pp. 143–155, 2001.

[39] T.-H. Tsai, W.-T. Sheu, and C.-Y. Lin, "Foreground object detection based on multi-model background maintenance," in *Multimedia Workshops, Proceedings of the 9th IEEE International Symposium on*, Dec 2007, pp. 151–159.

[40] K. K. Ng and E. J. Delp, "Object tracking initialization using automatic moving object detection," in *IS&T/SPIE Electronic Imaging.* International Society for Optics and Photonics, 2010, pp. 75 430M–75 430M.

[41] H. Wang and D. Suter, "Background initialization with a new robust statistical approach," in *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2nd Joint IEEE International Workshop on*, 2005, pp. 153–159.

[42] K. K. Ng, S. Srivastava, and E. Delp, "Foreground segmentation with sudden illumination changes using a shading model and a Gaussianity test," in *Image and Signal Processing and Analysis (ISPA), 7th International Symposium on*, September 2011, pp. 236–240.

[43] N. J. McFarlane and C. P. Schofield, "Segmentation and tracking of piglets in images," *Machine Vision and Applications*, vol. 8, no. 3, pp. 187–193, 1995.

[44] B. Lee and M. Hedley, "Background estimation for video surveillance," in *Image and vision computing New Zealand*, 2002, pp. 315–320.

[45] J. Zheng, Y. Wang, N. L. Nihan, and M. E. Hallenbeck, "Extracting roadway background image: Model-based approach," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1944, no. 1, pp. 82–88, 2006.

[46] T. Bouwmans, F. el Baf, B. Vachon *et al.*, "Statistical background modeling for foreground detection: A survey," *Handbook of Pattern Recognition and Computer Vision*, pp. 181–199, 2010.

[47] (2014) University of Tuebingen: 4D Flexible Atom-Pair Kernel. (last accessed: 15 November 2014). [Online]. Available: http://www.ra.cs.uni-tuebingen.de/software/4DFAP/hist.png

[48] C. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, "Pfinder: real-time tracking of the human body," in *Automatic Face and Gesture Recognition, Proceedings of the Second International Conference on*, Oct 1996, pp. 51–56.

[49] M. S. Allili, N. Bouguila, and D. Ziou, "A robust video foreground segmentation by using generalized Gaussian mixture modeling," in *Proceedings of the Fourth Canadian Conference on Computer and Robot Vision*. IEEE Computer Society, 2007, pp. 503–509.

[50] C. Archambeau and F. Bach, "Multiple gaussian process models," *arXiv preprint arXiv:1110.5238*, 2011.

[51] M. Alvar, A. Rodriguez-Calvo, A. Sanchez-Miralles, and A. Arranz, "Mixture of merged Gaussian algorithm using RTDENN," *Machine Vision and Applications*, pp. 1–12, 2013.

[52] Y. Sheikh and M. Shah, "Bayesian object detection in dynamic scenes," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2005, pp. 74–79.

[53] ——, "Bayesian modeling of dynamic scenes for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1778–1792, 2005.

[54] (2014) Kernel density estimation. (last accessed: 15 November 2014). [Online]. Available: http://en.wikipedia.org/wiki/Kernel_density_estimation

[55] H.-H. Lin, T.-L. Liu, and J.-H. Chuang, "A probabilistic SVM approach for background scene initialization," in *Image Processing, Proceedings of the International Conference on*, vol. 3, June 2002, pp. 893–896 vol.3.

[56] J. Wang, G. Bebis, and R. Miller, "Robust video-based surveillance by integrating target detection with tracking," in *Computer Vision and Pattern Recognition Workshop*, June 2006, pp. 137–137.

[57] A. Tavakkoli, M. Nicolescu, and G. Bebis, “A novelty detection approach for foreground region detection in videos with quasi-stationary backgrounds,” in *Advances in Visual Computing, Proceedings of the Second International Conference on*, ser. ISVC’06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 40–49.

[58] (2011) Opencv - working with opencv: July 2011. (last accessed: 15 November 2014). [Online]. Available: http://areshopencv.blogspot.com/2011_07_01_archive.html

[59] D. Butler, S. Sridharan, and V. M. Bove Jr, “Real-time adaptive background segmentation,” in *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP’03). 2003 IEEE International Conference on*, vol. 3. IEEE, 2003, pp. 345–349.

[60] M. Xiao, C. Han, and X. Kang, “A background reconstruction for dynamic scenes,” in *Information Fusion, 9th International Conference on.* IEEE, 2006, pp. 1–7.

[61] M. J. Zaki and W. Meira Jr, *Data mining and analysis: fundamental concepts and algorithms.* Cambridge University Press, 2014.

[62] M. H. Sigari and M. Fathy, “Real-time background modeling/subtraction using two-layer codebook model,” in *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 1. IEEE, 2008.

[63] X. Deng, J. Bu, Z. Yang, C. Chen, and Y. Liu, “A block-based background model for video surveillance,” in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008, pp. 1013–1016.

[64] J. Xu, N. Jiang, and S. Goto, “Block-based codebook model with oriented-gradient feature for real-time foreground detection,” in *Multimedia Signal Processing (MMSP), 13th International Workshop on.* IEEE, 2011, pp. 1–6.

[65] J.-M. Guo, Y.-F. Liu, C.-H. Hsia, M.-H. Shih, and C.-S. Hsu, “Hierarchical method for foreground detection using codebook model,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 6, pp. 804–815, 2011.

[66] J. C. Russ and R. P. Woods, “The image processing handbook,” *Journal of Computer Assisted Tomography*, vol. 19, no. 6, pp. 979–981, 1995.

[67] D. Culibrk, O. Marques, D. Socek, H. Kalva, and B. Furht, “Neural network approach to background modeling for video object segmentation,” *IEEE Transactions on Neural Networks*, vol. 18, no. 6, pp. 1614–1627, Nov 2007.

[68] R. M. Luque, D. Lopez-Rodriguez, E. Mérida-Casermeiro, and E. J. Palomo, "Video object segmentation with multivalued neural networks," in *Hybrid Intelligent Systems, 2008. HIS'08. Eighth International Conference on.* IEEE, 2008, pp. 613–618.

[69] R. M. Luque, E. Domínguez, E. J. Palomo, and J. Muñoz, "A neural network approach for video object segmentation in traffic surveillance," in *Image Analysis and Recognition.* Springer, 2008, pp. 151–158.

[70] R. M. Luque, D. López-Rodríguez, E. Dominguez, and E. J. Palomo, "A dipolar competitive neural network for video segmentation," in *Advances in Artificial Intelligence–IBERAMIA 2008.* Springer, 2008, pp. 103–112.

[71] L. Maddalena and A. Petrosino, "A self-organizing approach to background subtraction for visual surveillance applications," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1168–1177, 2008.

[72] E. J. Palomo, E. Domínguez, R. M. Luque, and J. Muñoz, "Image hierarchical segmentation based on a GHSOM," in *Neural Information Processing.* Springer, 2009, pp. 743–750.

[73] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: principles and practice of background maintenance," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 1, 1999, pp. 255–261 vol.1.

[74] J. Ding, M. Li, K. Huang, and T. Tan, "Modeling complex scenes for accurate moving objects segmentation," in *Computer Vision–ACCV 2010.* Springer, 2011, pp. 82–94.

[75] R. Chang, T. Gandhi, and M. M. Trivedi, "Vision modules for a multi-sensory bridge monitoring approach," in *Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on.* IEEE, 2004, pp. 971–976.

[76] (2014) Wiener filter (image). (last accessed: 15 November 2014). [Online]. Available: http://en.wikipedia.org/wiki/Wiener_filter

[77] (2014) Kalman filter (image). (last accessed: 15 November 2014). [Online]. Available: http://en.wikipedia.org/wiki/Kalman_filter

[78] D. Mukherjee and J. Wu, "Real-time video segmentation using student's t mixture model," *Procedia Computer Science*, vol. 10, pp. 153–160, 2012.

[79] Y. He, D. Wang, and M. Zhu, "Background subtraction based on nonparametric bayesian estimation," in *Digital Image Processing, 3rd International Conference on.* International Society for Optics and Photonics, 2011, pp. 80 090G–80 090G.

[80] S. Molina-Giraldo, J. C. González, A. M. Álvarez-Meza, and G. Castellanos-Domínguez, "Video segmentation based on multi-kernel learning and feature relevance analysis for object classification." in *ICPRAM*, 2013, pp. 396–401.

[81] D. Farcas, C. Marghes, and T. Bouwmans, "Background subtraction via incremental maximum margin criterion: a discriminative subspace approach," *Machine Vision and Applications*, vol. 23, no. 6, pp. 1083–1101, 2012.

[82] C. Marghes, T. Bouwmans, and R. Vasiu, "Background modeling and foreground detection via a reconstructive and discriminative subspace learning approach," in *Computer Vision, and Pattern Recognition (IPCV), International Conference on Image Processing*, 2012.

[83] J. Yan, B. Zhang, S. Yan, Q. Yang, H. Li, Z. Chen, W. Xi, W. Fan, W.-Y. Ma, and Q. Cheng, "IMMC: incremental maximum margin criterion," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 2004, pp. 725–730.

[84] F. Tang and H. Tao, "Fast linear discriminant analysis using binary bases," *Pattern Recognition Letters*, vol. 28, no. 16, pp. 2209–2218, 2007.

[85] D. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Computation*, vol. 16, no. 12, pp. 2639–2664, 2004.

[86] D. Chen and L. Zhang, "An incremental linear discriminant analysis using fixed point method," in *Advances in Neural Networks-ISNN 2006.* Springer, 2006, pp. 1334–1339.

[87] T.-K. Kim, K.-Y. K. Wong, B. Stenger, J. Kittler, and R. Cipolla, "Incremental linear discriminant analysis using sufficient spanning set approximations," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, June 2007, pp. 1–8.

[88] G. Mateos and G. B. Giannakis, "Sparsity control for robust principal component analysis," in *Signals, Systems and Computers (ASILOMAR), 2010 Conference Record of the Forty Fourth Asilomar Conference on.* IEEE, 2010, pp. 1925–1929.

[89] J. He, L. Balzano, and J. C. S. Lui, "Online robust subspace tracking from partial information," *CoRR*, vol. abs/1109.3827, 2011.

[90] J. He, D. Zhang, L. Balzano, and T. Tao, "Iterative Grassmannian optimization for robust image alignment," *Image and Vision Computing*, 2014.

[91] F. Seidel, C. Hage, and M. Kleinsteuber, "pROST: A smoothed lp-norm robust online subspace tracking method for realtime background subtraction in video," *arXiv preprint arXiv:1302.2073*, 2013.

[92] J. Xu, V. K. Ithapu, L. Mukherjee, J. M. Rehg, and V. Singh, "GOSUS: Grassmannian online subspace updates with structured-sparsity," in *Computer Vision (ICCV), 2013 IEEE International Conference on.* IEEE, 2013, pp. 3376–3383.

[93] X. Zhou, C. Yang, and W. Yu, "Moving object detection by detecting contiguous outliers in the low-rank representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 3, pp. 597–610, 2013.

[94] L. Xiong, X. Chen, and J. Schneider, "Direct robust matrix factorization for anomaly detection," in *Data Mining (ICDM), 11th International Conference on.* IEEE, 2010.

[95] N. Wang, T. Yao, J. Wang, and D.-Y. Yeung, "A probabilistic approach to robust matrix factorization," in *Computer Vision–ECCV 2012.* Springer, 2012, pp. 126–139.

[96] J. Huang, T. Zhang, and D. Metaxas, "Learning with structured sparsity," *The Journal of Machine Learning Research*, vol. 12, pp. 3371–3412, 2011.

[97] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

[98] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[99] X. Cui, J. Huang, S. Zhang, and D. N. Metaxas, "Background subtraction using low rank and group sparsity constraints," in *Computer Vision–ECCV 2012.* Springer, 2012, pp. 612–625.

[100] R. Sivalingam, A. D'Souza, M. Bazakos, R. Miezianko, V. Morellas, and N. Papanikolopoulos, "Dictionary learning for robust background modeling," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, May 2011, pp. 4234–4239.

[101] C. R. Wren and F. Porikli, "Waviz: Spectral similarity for object detection," in *Performance Evaluation of Tracking and Surveillance, IEEE International Workshop on*, 2005, pp. 55–61.

[102] H. Tezuka and T. Nishitani, "A precise and stable foreground segmentation using fine-to-coarse approach in transform domain," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on.* IEEE, 2008, pp. 2732–2735.

[103] T. Gao, Z. G. Liu, S. H. Yue, J. Zhang, J. Q. Mei, and W. C. Gao, "Robust background subtraction in traffic video sequence," *Journal of Central South University of Technology*, vol. 17, pp. 187–195, 2010.

[104] D. Baltieri, R. Vezzani, and R. Cucchiara, "Fast background initialization with recursive Hadamard Transform," in *Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on.* IEEE, 2010, pp. 165–171.

[105] (2014) Digital image processing: Image transforms. (last accessed: 15 November 2014). [Online]. Available: http://ee.sharif.edu/~dip/Files/DIPTransformForPrint.pdf

[106] (2014) Illumination change compensation. (last accessed: 15 November 2014). [Online]. Available: http://www.ece.gatech.edu/research/pica/GTILT.html

[107] P. Maragos and L. F. Pessoa, "Morphological filtering for image enhancement and detection," *analysis*, vol. 13, p. 12, 1999.

[108] (2003) Glossary - structuring elements. (last accessed: 15 November 2014). [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/strctel.htm

[109] (2003) Morphology - opening. (last accessed: 15 November 2014). [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/open.htm

[110] (2003) Morphology - closing. (last accessed: 15 November 2014). [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/close.htm

[111] B. Xie, V. Ramesh, and T. Boult, "Sudden illumination change detection using order consistency," *Image and Vision Computing*, vol. 22, no. 2, pp. 117–125, 2004.

[112] J. Pilet, C. Strecha, and P. Fua, "Making background subtraction robust to sudden illumination changes," in *Computer Vision–ECCV 2008.* Springer, 2008, pp. 567–580.

[113] B. Stenger, V. Ramesh, N. Paragios, F. Coetzee, and J. M. Buhmann, "Topology free hidden markov models: Application to background modeling," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1. IEEE, 2001, pp. 294–301.

[114] X. Zhao, W. He, S. Luo, and L. Zhang, "MRF-based adaptive approach for foreground segmentation under sudden illumination change," in *Information, Communications Signal Processing, 2007 6th International Conference on*, Dec 2007, pp. 1–4.

[115] Y. Sun and B. Yuan, "Hierarchical GMM to handle sharp changes in moving object detection," *Electronics Letters*, vol. 40, no. 13, pp. 801–802, 2004.

[116] Y. Dong, T. Han, and G. N. DeSouza, "Illumination invariant foreground detection using multi-subspace learning," *International Journal of Knowledge-based and Intelligent Engineering Systems*, vol. 14, no. 1, pp. 31–41, 2010.

[117] Y. Hwang, K. Sung, J. Chae, Y. Park, and I.-S. Kweon, "Robust background maintenance by estimating global intensity level changes for dynamic scenes," *Intelligent Service Robotics*, vol. 2, no. 3, pp. 187–194, 2009.

[118] M. Heikkilä, M. Pietikäinen, and C. Schmid, "Description of interest regions with center-symmetric local binary patterns," in *Computer Vision, Graphics and Image Processing*. Springer, 2006, pp. 58–69.

[119] S. Zhang, H. Yao, and S. Liu, "Dynamic background modeling and subtraction using spatio-temporal local binary patterns," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*. IEEE, 2008, pp. 1556–1559.

[120] G. Xue, J. Sun, and L. Song, "Dynamic background subtraction based on spatial extended center-symmetric local binary pattern," in *Multimedia and Expo (ICME), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1050–1054.

[121] R. Ojeda, J. Cardoso, and E. Moulines, "Asymptotically invariant gaussianity test for causal invertible time series," in *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, vol. 5, Apr 1997, pp. 3713–3716 vol.5.

[122] M. Nafi Gurcan, Y. Yardimci, and A. Enis Cetin, "Influence function based Gaussianity tests for detection of microcalcifications in mammogram images," in *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, vol. 3, 1999, pp. 407–411.

[123] M. N. Gurcan, Y. C. Yardimci, and E. A. Cetin, "2D adaptive prediction-based Gaussianity tests in microcalcification detection," in *Proc. SPIE*, vol. 3309, 1998, pp. 625–633.

[124] (2014) Introduction to neural networks using matlab 6.0 pdf download. (last accessed: 15 November 2014). [Online]. Available: http://carsoncitypizza.com/introduction-to-neural-networks-using-matlab-6-0-pdf-download/

[125] (2013) Random.org - true random number service. (last accessed: 15 November 2014). [Online]. Available: https://sites.google.com/site/hsi2013logan99/computer-vision/implementation

[126] A. M. Elgammal, D. Harwood, and L. S. Davis, "Non-parametric model for background subtraction," in *Computer Vision, 6th European Conference on*, ser. ECCV '00. Springer-Verlag, 2000, pp. 751–767.

[127] R. Vemulapalli and R. Aravind, "Spatio-temporal nonparametric background modeling and subtraction," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, Sept 2009, pp. 1145–1152.

[128] Y. Roodt, W. Visser, and W. Clarke, "Image processing on the GPU: Implementing the canny edge detection algorithm," in *International Symposium of the Pattern Recognition Association of South Africa*, 2007.

[129] M. Houston, "General-purpose computation on graphics hardware. siggraph 2007 GPGPU course," 2007.

[130] (2013) CPU, GPU and MIC hardware characteristics over time. (last accessed: 15 November 2014). [Online]. Available: http://www.karlrupp.net/2013/06/cpu-gpu-and-mic-hardware-characteristics-over-time/

[131] (2005) Data intensive swos speedup using fpga board. (last accessed: 15 November 2014). [Online]. Available: http://www.coe.neu.edu/Research/rcl/projects/SWO/index.htm