SINGLE-PIXEL ROBUST APPROACH FOR BACKGROUND SUBTRACTION FOR FAST PEOPLE-COUNTING AND DIRECTION ESTIMATION

by

Adedolapo Olaide Adegboye

Submitted in partial fulfilment of the requirements for the degree

Master of Engineering (Computer Engineering)

in the

Department of Electrical, Electronic and Computer Engineering
Faculty of Engineering, Built Environment and Information Technology
UNIVERSITY OF PRETORIA

March 2013

SUMMARY

SINGLE-PIXEL ROBUST APPROACH FOR BACKGROUND SUBTRACTION FOR FAST PEOPLE-COUNTING AND DIRECTION ESTIMATION

by

Adedolapo Olaide Adegboye

Supervisor(s):	Prof. Gerhard Hancke (University of Pretoria, South Africa)
Department:	Electrical, Electronic and Computer Engineering
University:	University of Pretoria
Degree:	Master of Engineering (Computer Engineering)
Keywords:	people counting, background subtraction, direction estimation, feature extraction, morphological processing, virtual line.

People counting system involves the process of counting and estimating the number of people in a scene. The counting system has a number of useful applications, ranging from pedestrian traffic surveillance and monitoring the number of people that enters and leaves shopping malls to commercial buildings, vehicles and a number of other security-related applications. Over the years, significant progress has been made. However, people counting systems still have not overcome a number of challenges such as occlusions, human pose and direction, multiple people detection, varying lighting and weather conditions.

The aim of this research is to present an optimal solution that is invariant to the challenges. That is, the outcome of the results will not be affected by the challenges. Also, the solution will handle the trade-off between the counting accuracy and the time it takes to implement the counting process. As a result, a new background subtraction method known as single pixel method is proposed. This is where useful features are collected from each scene using frame difference method. Then, these features are reduced into single pixels. The single pixels are then used to estimate the total number of people in the scene. Furthermore, a virtual-line direction-estimation method is presented where the directions in which the people are heading are estimated prior to counting.

OPSOMMING

ENKELLOPEND-PIXEL ROBUUSTE BENADERING VIR AGTERGROND AFTREKKING VIR VINNIGE MENSE TEL EN RIGTING RAMING

deur

Adedolapo Olaide Adegboye

Studieleier(s):	Prof. Gerhard Hancke (University of Pretoria, South Africa)
Departement:	Elektriese, Elektroniese en Rekenaar-Ingenieurswese
Universiteit:	Universiteit van Pretoria
Graad:	Magister in Ingenieurswese (Rekenaaringenieurswese)
Sleutelwoorde:	mense tel, agtergrond aftrekking, rigting skatting, Feature Extraction, morfologiese verwerking, virtuele lyn.

Mense-telstelsels behels die proses van die tel en die beraming van die aantal mense op ’n toneel. Die telstelsel het ’n aantal nuttige toepassings wat wissel van voetgangerverkeer toesig en die monitering van die aantal mense wat binnekom en verlaat tot winkelsentrums, kommersiële geboue, voertuie, en ’n aantal ander sekuriteit-verwante programme. Oor die jare is beduidende vordering gemaak. Daar is egter ’n aantal uitdagings wat mense-telstelsels nog nie oorkom het nie, soos afsluiting, menslike inhou en rigting, die opsporing van veelvoudige mense, wisselende beligting en weerstoestande.

Die doel van hierdie navorsing is om ’n optimale oplossing aan te bied, wat invariant is teen die uitdagings. Met ander woorde, die uitdagings sal nie die resultate affekteer nie. Die oplossing sal ook die uitruil tussen die tel akkuraatheid en die implementeringstyd van die telproses hanteer. As gevolg hiervan, is ’n nuwe agtergrondaftrekkingsmetode, wat bekend staan as ’n enkele beeldelement metode, voorgestel. Dit is waar die nuttige funksies van elke toneel, met behulp van die raamverskilmetode ingesamel word. Dan word hierdie eienskappe in enkele beeldelemente verminder. Die enkele beeldelemente word dan gebruik om die totale aantal mense in die toneel te skat. Verder is daar van ’n virtuele-lyn rigting-skatting metode gebruik gemaak wat die rigtings waarin die mense beweeg vooraf beraam.

ACKNOWLEDGEMENTS

I would like to extend my appreciation to the following people for their support in my journey to the completion of masters:

- Prof. Gerhard Hancke for his great supervision, guidance and support throughout the research.
- University of Pretoria for granting me a postgraduate bursary.
- My family and friends who motivated and supported me during the writing of the dissertation and implementation of results.
- Rogerio dos Santos and Bruno Silva, my friends, for their constant support and suggestions during the creation, implementation and testing of the system and helping to review and edit of the dissertation.
- Dr. Conrad Booysen, my pastor and my friend, for the support and language editing of the dissertation.
- Lammy, Jade, Mom and Dad for the countless sleepless nights filled with prayers, motivation and support throughout my undergraduate and postgraduate studies.
- Most importantly, the Lord Almighty for providing the opportunity to start and successfully finish this research in time.

LIST OF ABBREVIATIONS

DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DWT	Discrete Wavelet Transform
EKF	Extended Kalman Filter
FWT	Fast Wavelet Transform
GMM	Gaussian Mixture Model
HSV	Hue, saturation and value
HWT	Haar Wavelet Transform
KD	Kernel Density
KDE	Kernel Density Estimation
KLT	Kanade Lucas Tomasi
LOI	Line of interest
MoG	Mixture of Gaussians
PDF	Probability density function
POI	Point of Interest
RGB	Red, green, blue
ROI	Region of interest
SE	Structuring elements
SIR	Sequential importance re-sampling

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

"*Engineering is the art of modelling materials*
we do not wholly understand,
into shapes we cannot precisely analyse
so as to withstand forces we cannot properly assess,
in such a way that the public has no reason
to suspect the extent of our ignorance."

Dr AR Dykes - British Institution of Structural Engineers, 1976.

1.1 PROBLEM STATEMENT

1.1.1 Context of the problem

Visual surveillance is an important security application where people's activities can be monitored and recorded. People counting is an extract of the visual surveillance application and it serves as an estimation process where different point of interest (POI) areas are monitored and the number of people in those areas are counted and recorded. The monitored areas vary between outdoor and indoor environments such as buildings, shopping malls, hotels, residential areas, universities, streets, parks and other recreational areas.

People counting is quickly becoming a significant aspect in computer vision because it is widely implemented in various areas such as visual surveillance, intelligent transportation systems, people monitoring, behaviour recognition, direction estimation and security access control.

Due to advances in intelligent systems, people counting has attracted a lot of attention in recent years where extensive studies and algorithmic proposals have been successfully performed. However, it

is deemed as one of the challenging and key problems in robust real-time systems where restrictions such as occlusions, human pose and direction, multiple people detection, lighting and weather conditions, are present.

1.1.2 Research gap

Several research studies have been published in order to solve the problem of counting the number of people present in a scene using a video camera. In a typical scenario, an area is monitored and each person in the scene is counted and recorded after a period of time.

As easy as this sounds, the counting process faces a number of challenges or restrictions that limit its performance. Firstly, the people must be moving. If little or no movement is observed from the person, the person is considered as the background image. Secondly, the background of the scene must be simple, i.e. non-complex scenes and the image resolution should be high quality [1]. Furthermore, the counting process could be affected when non-human movements are observed and the complexity of the scene could greatly increase if multiple targets are observed.

Other challenges include partial and full occlusions, imperfect foreground extraction, false detections, illumination changes, human pose and direction, to name but a few [1]-[3]. One of the most difficult challenges is occlusions which mostly occurs when a person is partially hidden from the camera's field of view or when two or more people are moving at the same pace in the scene. In the process, crowds of people are entering or exiting an area at the same time so it is very difficult to distinguish between all the humans in the scene.

With regards to human pose and direction, the field of view is an important aspect in people detection. If a top view is presented, only the head (sometimes shoulder) of the person is observed and detected. If a side view is presented, then most of the parts of the body can be observed [4]. These are some of the challenges that need to be properly addressed before people counting can be successfully implemented.

There have been many advances that aim to solve these challenges affiliated with people counting but the solutions are highly application specific so they fall short in robust cases. The attempts either solve one of the challenges [1], [5]-[11] or bypass them using hardware-based methods [11], [12] and direction-flow methods [13]. People counting algorithm consists of other image processing al-

gorithms which increases the complexity of the algorithm. Therefore, there is a trade-off between the complexity and performance of the algorithm; that is, an increase in the performance accuracy of the algorithm will unfortunately result in an increase in the complexity of the algorithm.

1.2 RESEARCH OBJECTIVE AND QUESTIONS

People counting algorithms are mostly implemented in different scenes for security reasons. Some of the reasons are: 1) the need to monitor the number of people entering and exiting a building so that when an emergency occurs, it is easier to know the number of people that have exited the building and quickly locate the people left in the building; or 2) for traffic surveillance where pedestrian traffic flow can be estimated. As mentioned earlier, people counting faces significant restrictions which have to be properly addressed and resolved.

In the ideal scenario, a proposed people counting system should be capable of:

- successfully counting the number of people at a specific location in the scene,
- addressing and solving all the above-mentioned significant restrictions that people counting systems face, and
- being implementable under as many different scenarios as possible such as simple and complex cases of public, private, outdoor and indoor environments.

Due to time constraints, man-power and funding, this research will be unable to solve all the above-mentioned challenges that people counting systems are faced with. Therefore, the following constraints were imposed on the operational processes of the system.

1. The system can be implemented for outdoor and indoor applications; however, success will only be guaranteed for public-outdoor and private-indoor environments.
2. The system will be invariant to significant restrictions such as occlusions, illumination conditions, human pose and direction; however the performance of the system will be diminished when heavy occlusions are observed.

Given those restraints, the primary objectives are defined as: 1) the implementation of robust people counting algorithm that is not application specific; 2) that is invariant to occlusions, illumination

conditions, human pose and direction; 3) that provides direction estimation of the people; 4) with low complexity and high performance accuracy.

These primary objectives are correlated with the research questions provided as follows:

- How do we develop a robust people counting algorithm that is application-specific?
- How do we deal with restrictions that affect the performance accuracy of people-counting algorithm?
- How do we determine the direction in which a person is moving, prior to counting, without increasing the complexity of the algorithm?
- How do we reduce time, increase speed and performance accuracy of people-counting algorithm?
- How do we implement a robust background segmentation algorithm that significantly improves the performance accuracies of people-counting and related image processing algorithms?

1.3 HYPOTHESIS AND APPROACH

The hypothesis presented in this research states that the accuracies of people counting and other related image processing systems are significantly reduced where the above-mentioned challenges and restrictions are present. It is mandatory to solve these challenges in order to increase accuracy of the proposed system. Therefore, the approach taken in the implementation is explained as follows.

An important step in people-counting is background subtraction where targets are detected either by extracting their necessary features or isolating their shapes. The targets can be easily counted once they are accurately detected so it is very crucial to only extract useful and necessary features from the scenes. Therefore, a robust and effective background subtraction method is proposed using single-pixel method. In this method, background and other unnecessary objects are removed by observing and extracting frame differences from each frame. Then, each extracted frame difference is collected as useful features and reduced into single pixels; allowing easier counting and tracking of the people in the scenes.

Furthermore, a virtual-line direction-estimation method is proposed where the directions in which the people are heading are estimated prior to counting. Suppose multiple persons are observed in the scenes and it is required to determine if the people are entering or leaving a building; a vertical

or horizontal virtual line is drawn at the entrance point of the scene. The people are tracked by collecting the *xy*-coordinates of their single pixels. If vertical line is drawn, the system will compare the *x*-coordinates of the single pixels with the *x*-coordinates of the virtual line. If the *x*-coordinates are increasing, then we know that the person is trying to enter the building. If decreasing *x*-coordinates are observed, then the person is trying to exit the building. As a result, if the person′s direction is towards the building, the person is counted as entering the building. If not, the person is counted as exiting the building. If the person never gets to the building, the person is counted but "unassigned". Alternatively, if a horizontal line is drawn, the process is repeated for *y*-coordinates.

1.4 RESEARCH CONTRIBUTION

The main research contribution of the dissertation is the single-pixel method which answers the research objectives that are questioned in the study. The single-pixel method is a process that is able to accurately: 1) extract useful and necessary features in scenes; 2) count the number of people that are in different scenes with a minimized-low complexity; and 3) estimate the directions in which the people are heading with low complexity and high accuracy. As a result, it employs better accuracies for people counting and other related computer vision algorithms.

The method can be implemented on inputs with varying environmental conditions. These environments include indoor and outdoor areas; public places such as state libraries, stadiums, parks and fitness centres; and private places such as residential areas. Also, it is invariant to most significant challenges concerned with people counting algorithm such as partial and heavy occlusions, varying lighting conditions, different human pose and direction. As a result, an effective and robust people-counting and direction estimation approach is presented.

Other contribution includes the functionality of single-pixel method with other related image processing and computer vision algorithms.

1.5 OVERVIEW OF STUDY

The rest of the dissertation is organised as follows: *Chapter 2* presents an overview of state-of-art people counting algorithms while *Chapter 3* provides a literature study on background subtraction methods. The various approaches for the implementation of background subtraction methods are discussed. *Chapter 4* provides an overview on morphological image processing and *Chapter 5* provides

a literature study on state-of-art tracking techniques. *Chapter 6* provides the complete detailed approach and implementation of the developed people counting system while *Chapter 7* presents the experimental results and performance evaluation. The dissertation concludes with *Chapter 8* which presents the summary of the results achieved and future work in the research community.

CHAPTER 2 PEOPLE COUNTING ALGORITHMS

"When a distinguished but elderly scientist states that something is possible, he is almost certainly right. When he states that something is impossible, he is very probably wrong."

Arthur C. Clarke (1917 -), Clarke's first law

2.1 CHAPTER OBJECTIVES

People counting is an estimation process where the number of people in image scenes are observed and counted over a period of time in a frame-to-frame analysis. The counting process is divided into two classes, namely, region of interest (ROI) counting and line of interest (LOI) counting [14]. These two classes are explained in detail in this chapter.

2.2 LOI COUNTING

LOI counting involves the process of estimating the total number of people across a detection line over a certain period of time. It is implemented using a specific type of feature extraction-based method known as feature trajectories clustering [1], [14], [15]. This is the process where useful features are collected over a period of time across the frames and tracked into trajectories. Then, the feature trajectories are clustered into objects. The number of objects recorded in each scene is denoted as the number of people.

2.3 ROI COUNTING

ROI counting involves the process of estimating the total number of people in a specific region over a period of time and the counting process can be implemented through detection-based and feature extraction-based methods [16], [17].

2.3.1 Detection-based methods

Detection-based methods involve the process of individually detecting all the people present in the scene. The entire person or parts of their bodies can be detected, such as head, head-shoulder, face and upper body [4], [16], [18]-[21]. Also, the people can be detected by reconstructing their 3D human shapes [2], [3], [22]. Once detection has occurred, the total number of detected persons is counted and recorded.

Detection-based method is not popular with regards to implementation because its performance is strongly dependent on which detection algorithm is implemented [17]. Also, the performance of the algorithm is severely affected in the presence of occlusions, multi-target detection and varying lighting conditions [17], so the detection-based methods can be achieved using motion detection and analysis-based methods [23], [24]; template-based methods [25]; background differencing [26] and spatio-temporal methods [14], [27].

2.3.2 Feature-based methods

Feature extraction-based methods, also known as map or measurement-based methods [16] involve the process where various features are extracted from the input frames and the useful features are grouped according to their similarities into a feature vector. The feature vector is then employed to count the number of persons in the scene. Feature extraction-based method is the preferred approach in the implementation of people counting algorithms because it preserves privacy, are more robust and are not primarily dependent on detection.

The process of extracting useful features (feature vectors) in the input images is known as image representation or description. Significant features can be extracted using the following techniques: Discrete Fourier Transform (DFT) and Discrete Cosine Transform (DCT) [28] in the frequency domain filtering; Discrete Wavelet Transform (DWT), Fast Wavelet Transform (FWT) or Haar Wavelet

Transform [29] in time-frequency domain filtering; Gabor filtering (which can be implemented in both the spatial and spatial-frequency domain) is employed to obtain textual features [30], [31]; and in the time domain where local (spatial) features such as edges, perimeter and area [32] can be extracted.

Feature extraction-based methods can be further sub-divided into feature trajectories clustering [1], [15] and low-level feature regression methods [4]-[10], [13], [16], [32]-[34]. In low-level regression methods, sets of features (edges, textures) are extracted and grouped into a regression function which is used to estimate the crowd or people that are present in the scene [13]. The regression methods can range between linear regression [28], discrete regression, neural networks [8], [12], [13] and discrete classifier (SVM) [10].

CHAPTER 3 BACKGROUND SUBTRACTION

"Science is facts; just as houses are made of stones,
so is science made of facts;
but a pile of stones is not a house
and a collection of facts is not necessarily science."

Henri Poincare (1854 - 1912)

3.1 CHAPTER OVERVIEW

Background subtraction has received a lot of attention in recent years where various approaches have been proposed and successfully implemented. However, other factors contribute to the selection process of a suitable background subtraction algorithm. These factors include computational time and load complexity of the algorithm, illumination changes, bad weather conditions, detection of non-stationary background images, ghosts and shadows [35]. Therefore, this chapter provides detailed analysis and illustration of different methods that are followed in background subtraction process.

3.2 BACKGROUND SUBTRACTION

Background subtraction, also known as background segmentation, is a computational process where background pixels are subtracted from input images. As a result, useful features or foreground pixels are extracted from the images. In its simplest form, a background subtraction process is explained as follows. A background model is obtained. The input images are then observed and compared to the background model over a period of time in a frame-to-frame analysis. In each frame, the current image is subtracted from the background image and the result is thresholded accordingly to obtain

the desired result.

Essentially, the foreground pixels could be defined as useful object features contained in the images. In this case, the foreground pixels are denoted as moving people in the scene. Suppose a moving person can be observed in an outdoor scene, it will be required to identify the moving person whilst ignoring unnecessary movements that occur in varying weather conditions (wind causing moving leaves). So the approach that should be employed is the differentiation of the background model from the foreground pixels over a period of time in a frame-to-frame analysis.

Several background subtraction methods have been successfully implemented and most of these follow a simple flow diagram presented in Fig. 3.1. There are four major steps that constitute the implementation of background subtraction [36]: pre-processing; creation of a reference background model; foreground classification, which is the suitable subtraction process of the background image and the remaining images provided in the frame-to-frame analysis [37]; and data validation.

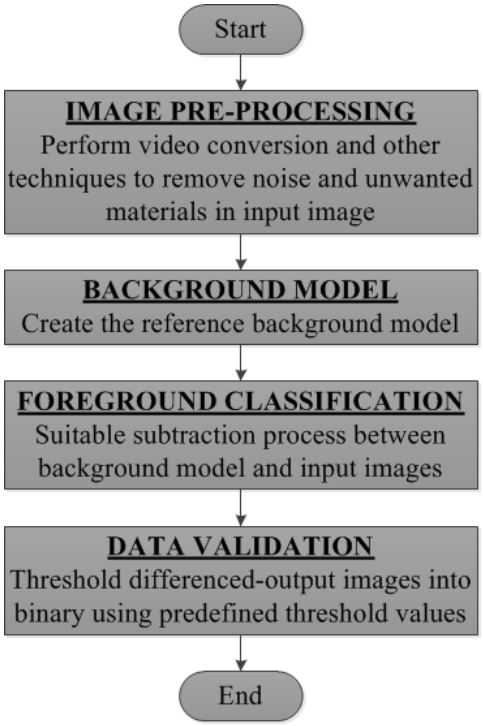


Figure 3.1: Illustration of the approach followed for the implementation of background subtraction.

In image pre-processing, the provided input (video) is converted into frames in order to perform frame-to-frame analysis over the period of time. The second step questions how well the background image can be modelled while the third step questions which distance metrics can be used to implement

a suitable subtraction process in order to classify the foreground pixels into target objects. These two questions generally serve as the key objectives in the background subtraction process, so detailed overviews of the background modeling and foreground classification are provided as follow.

3.3 BACKGROUND MODELING

Background modeling is regarded as a key step in background subtraction because it creates the background image. The background modeling can be further categorised into non-recursive and recursive methods. Usually, background pixels are assumed to be static and constant over time i.e. there are no significant lighting changes in the scenes, so non-recursive methods such as frame differencing [38], median filtering, non-parametric kernel density estimation (KDE) and sequential kernel density (KD) approximation, can employed in such cases.

However, in a real-world scenario, most of these non-recursive methods fail when lighting changes are observed in the scenes. Therefore, it is important for background models to allow variations in the background values. Several studies have been performed on these systems in order to allow robust changes in the scenes and these methods are known as recursive methods. Commonly-used recursive methods include approximate median filtering, Mixture of Gaussians (MoG) and Kalman filtering.

3.3.1 Non-recursive methods

Commonly-used non-recusive methods such as frame differencing [38], inter-frame differencing [39], median filtering [40], non-parametric kernel density estimation (KDE) are described below.

3.3.1.1 Frame Differencing

Frame differencing is one of the simplest methods of background subtraction. It involves the process where the initial frame of the frame-to-frame analysis is set as the background image at time t and the remaining frames are compared with the background frame over the period of time t. The current frame at time $t+1$ is subtracted from the background frame at time t and an absolute difference is obtained.

The absolute difference is then compared against a specified threshold and the result is returned as

either foreground or background pixels. The foreground pixels are regarded as the desired targets (the moving people in the scene in this case) and the background pixels are regarded as unnecessary materials in the images. As a result, this technique will be able to locate the position of moving people and track their movements [37]. The frame differencing procedure is formulated as follows.

Suppose the initial (background) frame is denoted as:

$$\mathbf{I}^b_{t-1}(x,y) \tag{3.1}$$

where time $t = 1$ and each current frame is denoted as:

$$\mathbf{I}^c_t(x,y) \tag{3.2}$$

where $t = 2,3,...,N$ and N is the total number of frames. The frame difference of the two images is represented in eq. 3.3 as:

$$\mathbf{I}^d_t(x,y) = \left\{ \begin{array}{ll} 1 & \text{if} |\mathbf{I}^c_t(x,y) - \mathbf{I}^b_{t-1}(x,y)| > \mathbf{T}_s \\ 0 & \text{otherwise,} \end{array} \right\} \tag{3.3}$$

where:

- $\mathbf{I}^d$ represents the frame-differenced output image,
- $\mathbf{I}^b$ represents the background image,
- $\mathbf{I}^c$ represents the current (or foreground) image,
- (x,y) represents the two-dimensional (rows and columns) form of the images,
- $\mathbf{T}_s$ is the specified threshold, and
- $||$ is the absolute difference of the two images.

The performance of the algorithm will significantly decrease and the accuracy of the results can be severely affected when varying illumination conditions are observed [37]. Changes in lighting conditions is highlighted as one of the key problems in frame differencing [38]. This problem can be bypassed by implementing frame differencing based on logarithmic intensities [41]. Other challenges include sensitivity to noise and the fact that properties of local consistency are not considered during

the implementation. Also, the method fails when limited or no movements are observed from the people in the scenes.

In summary, this background method works well under situations when there is no drastic change in the background i.e. lights are not suddenly switched off in an indoor scene. When significant changes are experienced, the method becomes ineffective and its performance decreases. So various methods were employed to cater for changes in scene lighting and weather conditions. These methods include: one-Gaussians [42], [43]; Gaussian mixture model (GMM) (also known as mixture of Gaussians) [44]; approximate median (temporal median filtering); non-parametric models; Kalman filter and particle filter. In these methods, there might be no need to define a prior threshold.

3.3.1.2 Inter-frame Differencing

W^4 is a background subtraction technique that caters for variation in the background values where the background is modelled by computing minimum and maximum intensity values, and the maximum difference between the frames (inter-frame difference) [39], [43].

3.3.1.3 Median Filtering

Median filtering is defined as one of the most commonly-used background modeling methods. It consists of a process where each background pixel is modeled as the median of all the pixel values collected over the period of time t of the frame-to-frame analysis [40]. The main disadvantage is that it is computationally expensive because median $O(LlogL)$ needs to be computed for each pixel [36], [45].

3.3.1.4 Non-parametric Kernel Density Estimation (KDE)

Non-parametric background modeling technique follows a slightly-different approach to other non-recursive methods that are described above. Instead of modeling each background pixel to the collection of its intensity values, a recent sample of all intensity values per pixel is collected and used to create a non-parametric estimate of the pixel probability density function [36], [46]. The probability density function is defined in eq. 3.4 as:

$$P(I_t) = \frac{1}{N}\sum_{i=1}^{N} K(I_t - I_i) \tag{3.4}$$

where:

- I_t is background image at time t,
- N is the number of frames, and
- K is kernel estimator function (Gaussian or Normal function)

3.3.2 Recursive Methods

Commonly-used recursive methods include approximate median filtering, Mixture of Gaussians (MoG) and Kalman filtering. These methods are described below.

3.3.2.1 Approximate Median Filtering

In approximate median filtering, the non-recursive median filter [40] is updated into a recursive process [47]. Instead of modeling each background pixel with the median of all the intensity values, an extra clause is provided to the median estimation process where the background pixel is compared to the median estimate. If the background pixel value is larger than the median estimate, a value of 1 is added to the current estimate [36], [47]. If the background pixel value is less than the median estimate, then the current estimate is decremented by 1.

3.3.2.2 One-Gaussian

One-Gaussian is a background subtraction process where the background image is modelled by obtaining each background pixel using Gaussian probability density function (pdf). Pfinder is one of the commonly-used one-Gaussian methods. Low-order statistics such as covariance matrix and spatial mean are used to represent each pixel location [42], [44], [45]. Alternatively, standard deviation can also be used. The probability density function is fitted for each pixel over a frame-to-frame analysis and in order to reduce the computational time, the pixel is updated by using an adaptive filter provided in eq. 3.5. The adaptive filter is otherwise known as the running (or cumulative average).

$$\mu_t = \alpha\gamma + (1-\alpha)\mu_{t-1} \tag{3.5}$$

where:

- μ_t is denoted as the spatial mean or average of the pixels at time t,
- μ_{t-1} is denoted as the previous spatial mean or average of the pixels at time $t-1$,
- α is the empirical weight and
- γ is the current pixel.

Once the background model is obtained, the subtraction process between the background and other images (also known as foreground classification) is implemented by comparing the current pixel to the updated average. This can be represented in eq. 3.6 as:

$$|I_t - \mu_t| > k\sigma_t \tag{3.6}$$

where k is denoted as the specific threshold and σ_t is the standard deviation of the pixel in time t.

3.3.2.3 Mixture of Gaussians

This is the background subtraction process where each background pixel is modelled with mixture of Gaussians [48] rather than one-Gaussian process described earlier. The mixture of Gaussians process is described as follows:

Suppose 50 frames of input images $I(x,y)$ are provided where (x,y) represents the spatial xy- coordinates of the input images. Each pixel's value is collected over the period of time t of the frame-to-frame analysis. As a result, 50 values will be collected per pixel. Therefore, the collection process can represented in eq. 3.7 as:

$$P_1, P_2, ..., P_t = I(x_0, y_0, i) \text{ where } 1 \leq i \leq t \tag{3.7}$$

where:

- $P_1, P_2, ..., P_t$ represents each pixel,
- i is the total period of time and
- I is the image sequence.

Thus, the probability of observing each pixel value P at time t in order to determine the occurrence of a colour is represented in eq. 3.8 as:

$$P(I_{p,t}) = \sum_{i=1}^{K} \omega_{i,p,t} * \eta(I_{p,t}, \mu_{i,p,t}, \Sigma_{i,p,t}) \tag{3.8}$$

where:

- K denotes the number of mixture of Gaussians distributions (typically ranges between 3 and 5),
- $I_{p,t}$ is the pixel value of the i^{th} Gaussian in the mixture K,
- $\mu_{i,p,t}$ is the spatial mean or average value of the i^{th} Gaussian in the mixture K over time t,
- $\Sigma_{i,p,t}$ is the covariance matrix of the i^{th} Gaussian in the mixture K over time t,
- $\omega_{i,p,t}$ is the weight of the i^{th} Gaussian in the mixture K, and
- η is a Gaussian probability density function.

The covariance matrix over K distributions can be represented in eq. 3.9 as:

$$\Sigma_{k,t} = \sigma_k^2 I \tag{3.9}$$

and the mixture of Gaussians is computed over K distributions. This background model approach is robust to scene lighting changes, when limited movements are experienced in the scene and when people are abruptly introduced or removed from the scene. However, it is computationally more expensive and time consuming than the earlier statistical background models described in previous sections of this chapter.

3.3.2.4 Kalman Filter

Kalman filter is widely implemented in several tracking systems [49], [50], [51], [52]. It is a mathematical process that is designed so that the position of a targeted object (or objects) in a current frame can be predicted from a series of previous noisy measurements [50]. Therefore, current state or location of the object can estimated by using its estimated-previous state and current measurements. The Kalman filter is regarded as an optimal linear estimator where mean and covariance values are calculated and are used to predict and estimate the locations of the target object. However, the assumption is only true on Gaussian-based noise [50]

Over the years, Kalman filtering has also been proposed for background modeling. The background model is estimated by using Kalman filtering process where estimated-previous state and current intensity values of the background pixels are computed over time t in a frame-to-frame analysis. Since it is a linear estimator, the noise should be Gaussian-based. Spatial derivatives (mean and covariance) and intensity values are then used to estimate the background pixels per frame [54]. Also, the estimation process can be implemented using intensity values and temporal derivatives [55].

The Kalman filter-based system is represented with a state-space model and the model is represented with two main variables, namely, posteriori state estimate and posteriori error covariance matrix. The posteriori (prior) state is estimated at time k. This is the collection of observations from the starting time till time t. Secondly, the posterior error covariance matrix is the measure of the estimated accuracy of state estimate from the starting time till time t. Fig. 3.2 shows the typical representation of a Kalman filter using graphical state-space model.

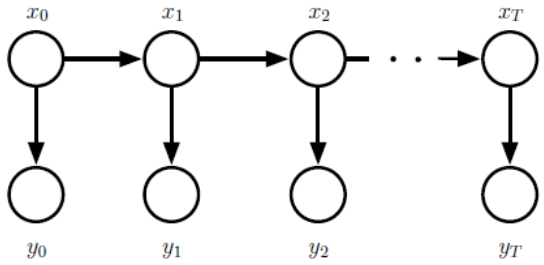


Figure 3.2: The typical representation of a Kalman filter using graphical state-space model [56].

In Fig. 3.2, all the states of the image frames are represented with **x** vector. Each x_t should contain

the mean and covariance of the states. Similarly, all the estimated states of the image frames are represented with **y**. That is, the locations of the target are estimated, predicted and returned as the estimated states. If a video sequence is provided, the video has to be converted into image frames $m = 1, 2, ..., N$. Suppose the second frame is the current frame, the posteriori state can be defined as the collection of observations from the initial frame till the second (current) frame while the posterior error-covariance matrix is the measure of the estimated accuracy of state estimate from the initial image frame until the second (current) frame. Fig. 3.3 shows the illustration of the state of state-space model.

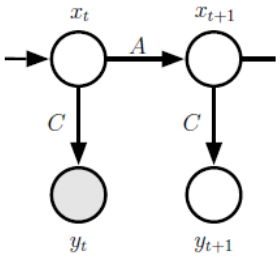


Figure 3.3: The illustration of the state of state-space model [56].

The Kalman filter algorithm is illustrated as follows. In order to create a background model, the state of all the objects in each scene has to be estimated. The estimation process is computed using a linear stochastic difference equation. The state transition matrix from frame $t-1$ to t is defined in eq. 3.10 as:

$$x_t = A\,x_{t-1} + B\,u_t + w_{t-1} \tag{3.10}$$

where x_t is the current state, A is the state transition matrix, w_{t-1} is the process Gaussian noise with zero mean and covariance matrix Q. Subsequently, the states are estimated using Kalman filtering by computing the measurements y_t. The measurements y_t can be expressed in terms of x_t. The measurements are defined in eq. 3.11 as:

$$y_t = C\,x_t + v_t \tag{3.11}$$

where v_t is the measurement Gaussian noise with zero mean and covariance matrix R and random variables w_k and v_k are independent of each other. The probability distribution of their covariance

matrices Q and R are defined in eq. 3.12 and eq. 3.13 as $p(w)$ and $p(v)$ respectively:

$$p(w) \sim N(0, Q) \tag{3.12}$$

$$p(v) \sim N(0, R) \tag{3.13}$$

The Kalman filter can be computed using two main stages: time update (predict) and measurement update [51]. The two stages are represented with two main equations which the Kalman filter is computed with.

Stage 1: Time Update (Predict)

In the time update stage, the current state (or location) of the target as well as the error covariance matrix are estimated at the current image frame m. The priori estimates $\hat{x}_t^-$ for each frame can be obtained by computing both the current state and error-covariance matrix estimates for each image frame. The priori (current) state estimate $\hat{x}_t^-$ can be computed using eq. 3.14:

$$\hat{x}_t^- = A\,\hat{x}_{t-1} + B\,u_t \tag{3.14}$$

Similarly, the priori covariance matrix estimate $\hat{P}_t^-$ for each image frame can be computed using eq. 3.15:

$$\hat{P}_t^- = A\,\hat{P}_{t-1}\,A^T + Q \tag{3.15}$$

Therefore, the priori estimates can be measured by computing the current state and error-covariance matrix estimates at the current frame of the frame-to-frame analysis. It is important to note that the priori estimates do not include the current observation information of the image frame.

Stage 2: Measurement Update

In the measurement update stage, the priori estimates are combined with the observation information at the current frame $m+1$. The combination is returned as the updated estimate which can be regarded as the posteriori state estimate. The first step is to compute the Kalman gain, K_t, using eq. 3.16:

$$K_t = P_t^- C^T \left(C P_t^- C^T + R\right)^{-1} \tag{3.16}$$

Once the Kalman gain is computed, it is used to compute the posterior state estimate $\hat{x}_t$. The posteriori state estimates can be measured by combining the predicted (priori) estimates and the observation information at the frame t. The posterior state estimate is defined in eq. 3.17:

$$\hat{x}_t = \hat{x}_t^- + K_t \left(y_t - C \hat{x}_t^-\right) \tag{3.17}$$

Similarly, the posterior error covariance matrix estimate is defined in eq. 3.18:

$$\hat{P}_t = (I - K_t H) \hat{P}_t^- \tag{3.18}$$

The observation information is needed for the successful implementation of tracking each moving object in all image frames, so that background model can be updated when the position of the moving object changes. At each image frame, the observation information includes:

- the *xy* center coordinates of the object's location,
- radius of the object's location,
- the change in the *xy* coordinates of the object's location during subsequent frame (time) intervals,
- the width and height of the object's bounding rectangle, and
- the change in the width and height of the object's bounding rectangle during subsequent frame intervals.

3.3.3 Foreground Classification

Once the background image has been modeled, it is required to correctly classify the useful information into foreground pixels. This can be achieved by implementing a suitable subtraction process

between the background model and the remaining input images that are provided in the frame-to-frame analysis [57]. Distance metrics such as Euclidean, Hamming, Mahalonobis, Manhattan and Bhattacharyya distance can be used to implement this subtraction process. In the simplest case, the difference between the background and input images per frame is computed over time t. If the resulting image is higher than a prior-defined threshold, the foreground pixels are retained. Otherwise, the pixels are discarded and set to 0. This process is represented using eq. 3.19:

$$|I_t(x,y) - B_t(x,y)| > T \tag{3.19}$$

where:

- I_t is the current image,
- B_t is the background image, and
- T is the specified threshold.

CHAPTER 4 MATHEMATICAL MORPHOLOGICAL PROCESSING

"A theory may be so rich in descriptive possibilities that it can be made to fit any data."

Phillip Johnson-Laird - The Computer and the Mind

4.1 INTRODUCTION

Mathematical morphological processing is a digital image processing approach that is based on shape. It serves as an image data-preserving tool because it retains the image shape while removing unnecessary and irrelevant features in the image [58]; otherwise, it extracts components that are useful for the representation and description of region shape of the image [59], [60]. These components include boundaries and skeletons of the images. Basic morphological operations such as erosion, dilation, opening and closing, are implemented on input images so that necessary features are highlighted. These features are critical for the direction estimation and counting of the people.

Originally, mathematical morphology was developed for binary images [61] so that set theory could be applied directly to two-level images i.e. foreground pixels at intensity value of 255 and background pixels at intensity value of 0. The mathematical morphological processes were then extended to greyscale images so that set operators such as set union, intersection and complementation, could also be used.

The general idea of mathematical morphology on an input image is to observe the geometric structure of the image then include and/or remove patterns from various locations in the image. The process operates on two sets: input image and its structuring element. The structuring element, also known as

the kernel or filter mask, is a neighbourhood of point (x,y) in the input image. The detailed overview of these morphological operations are explained in the following sections.

4.2 SET THEORY

The mathematical morphology is operated with set theory language. Some important set and logical operations are illustrated in the following sub-sections.

4.2.1 Basic set operations

Given **A**, a set composed of real numbers. If $a = (a_1, a_2)$ is an element of **A**, then:

$$a \in \mathbf{A} \tag{4.1}$$

If a is not an element of **A**:

$$a \notin \mathbf{A} \tag{4.2}$$

If a has no elements:

$$a \in \varnothing \tag{4.3}$$

If every element of set **A** is also an element of a set **B**, then **A** is a subset of **B**:

$$\mathbf{A} \subseteq \mathbf{B} \tag{4.4}$$

The union of two sets **A** and **B** is expressed as:

$$\mathbf{A} \cup \mathbf{B} \tag{4.5}$$

The intersection of the two sets **A** and **B** i.e. sets belonging to either or both is expressed in eq. 4.6 as:

$$\mathbf{A} \cap \mathbf{B} \tag{4.6}$$

Two sets are mutually exclusive if they have no common elements. This is expressed in eq. 4.7 as:

$$\mathbf{A} \cup \mathbf{B} = \varnothing \tag{4.7}$$

The compliment of set **A** i.e. sets of elements that are not in A is expressed in eq. 4.8 as:

$$\mathbf{A}^c = \{w | w \notin \mathbf{A}\} \tag{4.8}$$

The difference of two sets **A** and **B** is expressed in eq. 4.9 as:

$$\mathbf{A} - \mathbf{B} = \{w | w \notin \mathbf{A}, w \notin \mathbf{B}\} = \mathbf{A} \cap \mathbf{B}^c \tag{4.9}$$

Finally, U is a set universe; the set of all elements in a given application.

However, the preceding concepts are not applicable in greyscale images because the intensities of the pixels resulting from a set operation must be specified. The elements of a greyscale image can be represented by a set **A** whose elements are represented in the form (x, y, z), where x and y are the spatial coordinates and z is the intensity. This is expressed in eq. 4.10 as:

$$a = \{x, y, z\} \tag{4.10}$$

The compliment of set **A** i.e. sets of elements that are not in A is expressed in eq. 4.11 as:

$$\mathbf{A}^c = \{(x, y, K - z) | (x, y, z) \in \mathbf{A}\} \tag{4.11}$$

where intensities have been subtracted from a constant $\mathbf{K} = 2^k - 1$, where k is the number of intensity bits used to represent z. Therefore if A is an 8-bit greyscale image, the compliment of **A** is expressed in eq. 4.12 as:

$$\mathbf{A}^c = \{(x, y, 255 - z) | (x, y, z) \in \mathbf{A}\} \tag{4.12}$$

The union of two greyscale sets **A** and **B** are defined in eq. 4.13 as:

$$\mathbf{A} \cup \mathbf{B} = \{\max_z(a, b) | a \in \mathbf{A}, b \in \mathbf{B}\} \tag{4.13}$$

Fig. 4.1 illustrates the preceding concepts, where sets **A** and **B** are the sets of coordinates contained within the boundaries shown and the set universe, U is the set of coordinates contained within the rectangle provided.

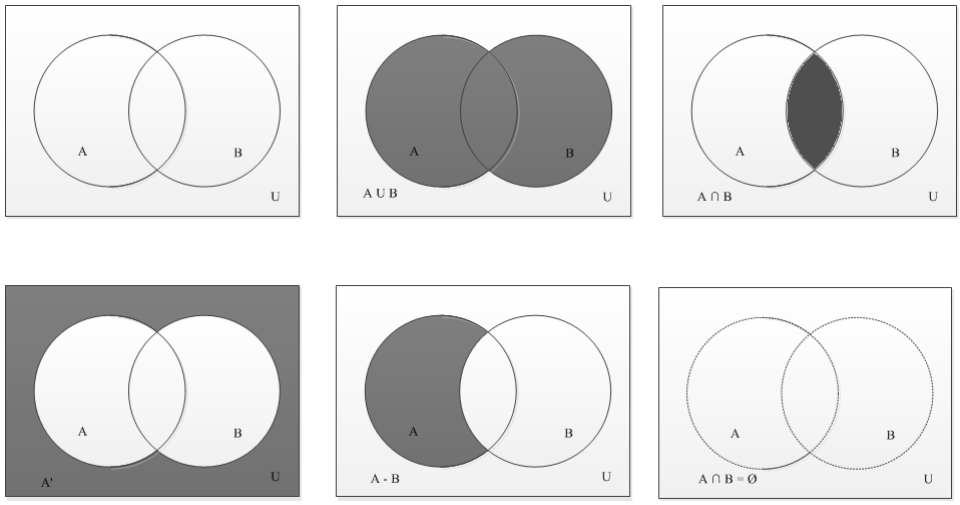


Figure 4.1: (a) Two sets of coordinates, A and B in 2D space. (b) Union of A and B. (c) Intersection of A and B. (d) Complement of A. (e) Difference between A and B. (c) A and B containing empty sets. Shaded areas indicate the members of the indicated set operations.

4.2.2 Morphological Operations

The basic preceding set operations and other set operations such as set reflection and translation are used extensively in morphology. The reflection of a set **A** is expressed in eq. 4.14 as:

$$\hat{\mathbf{A}} = \{w | w = -a, \text{for } a \in \mathbf{A}\} \tag{4.14}$$

where **A** is the set of pixels representing an object in an image and $\hat{\mathbf{A}}$ represents the coordinates (x,y) of **A** which have been replaced with $(-x,-y)$. The translation of a set **A** by point $z = (z_1, z_2)$ is expressed in eq. 4.15 as:

$$(\mathbf{A})_z = \{c | c = a + z, \text{for } a \in \mathbf{A} \tag{4.15}$$

where **A** is the set of pixels representing an object in an image and $(\mathbf{B})_z$ represents the coordinates (x,y) of **A** which have been replaced with $(x+z_1, y+z_2)$. Fig. 4.2 shows the reflection and translation of simple set **A**.

The structuring elements (SE) operations are formulated using set reflection and translation. When working with images, they are represented as rectangular arrays. This is achieved by appending the smallest possible number of background elements necessary to form the rectangular arrays [59].

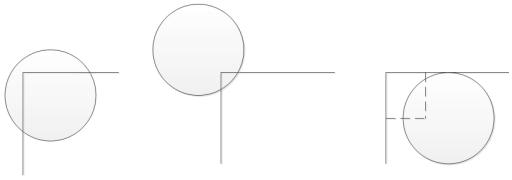


Figure 4.2: (a) A simple set **A**. (b) its reflection. (c) its translation by z.

Fig. 4.3(a) shows some examples of structuring elements and Fig. 4.3(b) the structuring elements represented as rectangular arrays.

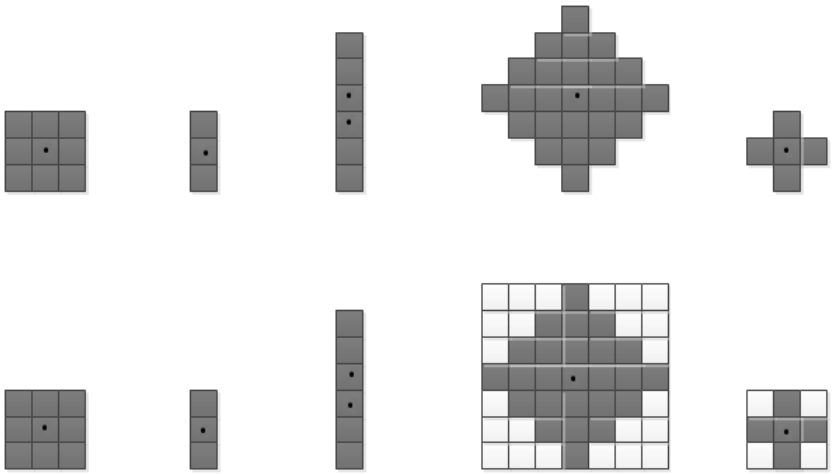


Figure 4.3: (a) Structuring elements. (b) Structuring elements represented as rectangular arrays

4.3 MORPHOLOGICAL OPERATIONS

The two primitive morphological operations, erosion and dilation, are discussed in the following sub-sections. These operations are fundamental to morphological processing as many of the other algorithms are based on these operations.

4.3.1 Binary Erosion

Erosion is the morphological transformation that combines two sets using the vector subtraction of set elements [58]. It is otherwise known as a shrinking or thinning operation where the elements of

a set (image objects) are strategically reduced. Given sets A and B in the Euclidean N-space, N^2, the erosion of A and B is defined in eq. 4.16 as:

$$\mathbf{A} \ominus \mathbf{B} = \{z|(\mathbf{B})_z \subseteq \mathbf{A}\} \tag{4.16}$$

that is, the erosion of A and B is the set of all points z such that **B** translated in z is contained in **A**. Alternatively, the erosion is defined in eq. 4.17 as:

$$\mathbf{A} \ominus \mathbf{B} = \{z|(\mathbf{B})_z \cap \mathbf{A}^c = \varnothing\} \tag{4.17}$$

where A^c is the complement of **A** and $\varnothing$ is the empty set. The erosion is simply defined as: set **B** has to be completely contained in **A** whilst not sharing any common elements with the background. The erosion process is illustrated in the following case example.

Given set **A** and structuring element (SE), **B**, a new set is created by running **B** over **A** so that the origin of **B** visits every element of **A**. The origin of SE is usually at its centre. At each element location of **A** where the origin of **B** is currently visiting; if all the elements of **B** (excluding the appended zeros) are completely contained in the same neighbourhood elements of **A**, then the location is marked/shaded as a member of the new set. Otherwise, the location is neither marked nor shaded as the member of the new set. Fig. 4.4 shows the illustration of the erosion case example. The elements of **A** are shaded as grey while the elements of **B** are shaded as dark blue.

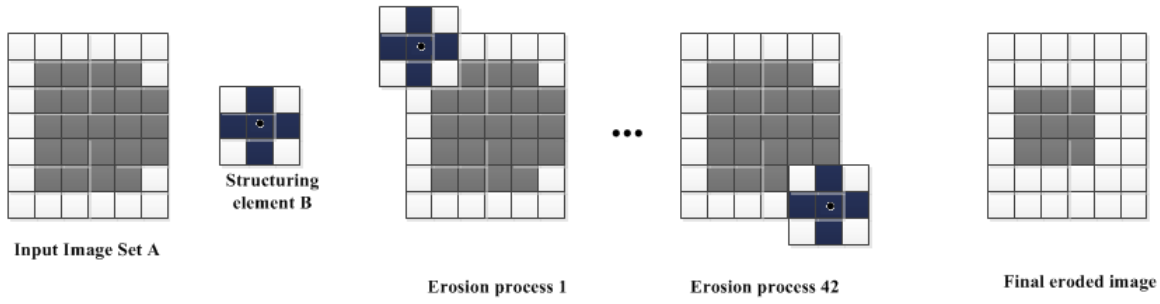


Figure 4.4: (a) A set (each shaded square is a member of the set). (b) A structuring element padded as a rectangular array. (c) First erosion process. (d) Last erosion process. (e) Set processed by the structuring element.

A morphological erosion process of a binary image is presented in Fig. 4.5. The image contains basic shapes including an upright rectangle, a diamond, a circle, a star and lines connecting to the border pads. All the elements of the SE are 1. The lines connecting to the border pads are removed by eroding the input image in Fig. 4.5(a) with square structuring element of size 15×15. Similarly, the star shape is removed using square SE of size 25×25. Eventually, all the shapes are removed using SE of size 50×50 except the circle which became a diamond shape. Fig. 4.5(b)-(d) show the eroded results.

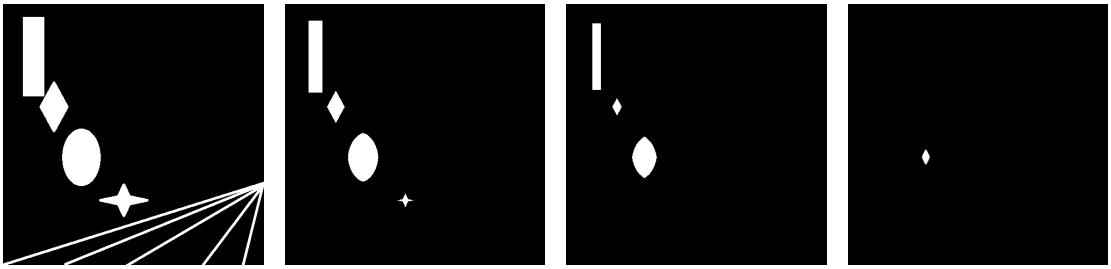


Figure 4.5: (a) Binary image. (b)-(d) Image eroded using square SE of sizes 15×15, 25×25, and 50×50 respectively.

As observed, the erosion process shrinks or thins objects in a binary image. The higher the SE size, the higher the shrinking/thinning of the binary image and eventually, all objects will be removed and the eroded output will only include background values.

4.3.2 Binary Dilation

Dilation is the dual morphological process of erosion. It is the morphological transformation which combines two sets using vector addition of set elements [58]. It is otherwise known as the growing or thickening operation because it tends to thicken objects in a binary image. The rate of the thickening is dependent on the size of the SE used, that is, a larger SE would result in larger thickening. Given sets A and B in the Euclidean N-space, N^2, the dilation of A and B is defined in eq. 4.18 as:

$$\mathbf{A} \oplus \mathbf{B} = \{z | (\hat{\mathbf{B}})_z \cap \mathbf{A} \neq \varnothing\} \tag{4.18}$$

Alternatively, the dilation is defined below in eq. 4.19; where it is the set of all displacements z, such that $\hat{\mathbf{B}}$ and $\mathbf{A}$ overlap by at least one element.

$$\mathbf{A} \oplus \mathbf{B} = \{z | (\hat{\mathbf{B}})_z \cap \mathbf{A} \neq \varnothing\} \tag{4.19}$$

Furthermore, the dilation operation is illustrated as follows:

Given set **A** and structuring element(SE), **B**; a new set is created by flipping (rotating) **B** about its origin and successfully displacing it so that it runs over **A** so that the origin of rotated **B** visits every element of **A**. The origin of SE is usually at its centre. At each element location of **A** where the origin of rotated **B** ($\hat{\mathbf{B}}$) is currently visiting, if $\hat{\mathbf{B}}$ and **A** overlap by at least one element, provided that $\hat{\mathbf{B}}$ is completely contained in **A**; then the location is marked/shaded as a member of the new set. Otherwise, the location is neither marked nor shaded as the member of the new set.

Fig. 4.6 shows the illustration of the dilation case example. The elements of **A** are shaded as grey while the elements of $\hat{\mathbf{B}}$ are shaded as dark blue.

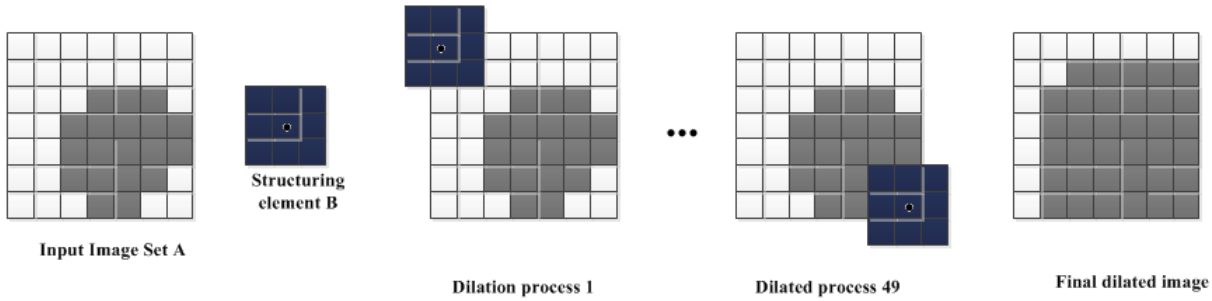


Figure 4.6: (a) A set (each shaded square is a member of the set). (b) A structuring element padded as a rectangular array. (c) First erosion process. (d) Last erosion process. (e) Set processed by the structuring element.

The dilation morphological process is implemented in Fig. 4.7(a) which shows a binary image with characters. Various SE sizes are implemented in order to thicken the characters. Fig. 4.7(b) - (d) show the result of the dilated images.

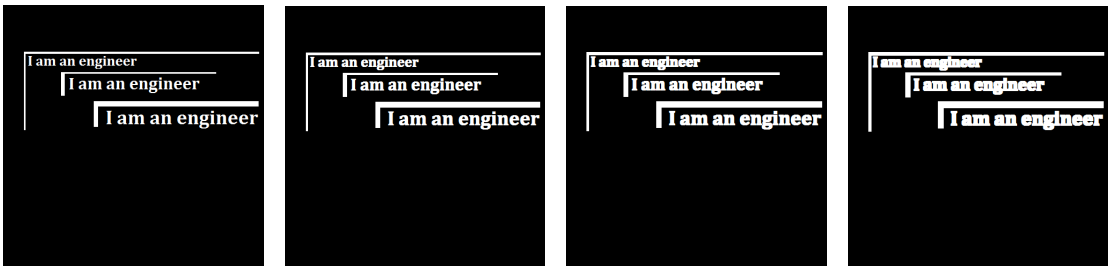


Figure 4.7: (a) Binary image. (b)-(d) Image dilated using square SE of sizes 2×2, 3×3, and 4×4 respectively.

It is observed that SE of size 2×2 was the optimal approach for thickening the characters. Higher SE size increases the thickness of the characters.

4.3.3 Binary Opening and Closing

Opening is a morphological process that generally smoothens the contour of an object, breaks narrow isthmuses and eliminates thin protrusions [59]. It is the dilation of the erosion of a set **A** by the structuring element **B**. It is denoted as $\mathbf{A} \circ \mathbf{B}$ and defined in eq. 4.20 as:

$$\mathbf{A} \circ \mathbf{B} = (\mathbf{A} \ominus \mathbf{B}) \oplus \mathbf{B} \tag{4.20}$$

Thus, the set is initially eroded, then the output of the eroded image is dilated.

Similarly, closing is the dual morphological process of opening. Although it also smoothens the sections of contours of an object, it opposes to opening, by eliminating small holes and filling gaps in the contour. It also generally fuses narrow breaks and long thin gulfs [59]. It is the erosion of the dilation of set **A** by the structuring element **B**. It is denoted as $\mathbf{A} \bullet \mathbf{B}$ and defined in eq. 4.21 as:

$$\mathbf{A} \circ \mathbf{B} = (\mathbf{A} \oplus \mathbf{B}) \ominus \mathbf{B} \tag{4.21}$$

In practice, dilations and erosions usually function in pairs in order to remove specified image details smaller than the structuring element without global geometric distortion [58]. Therefore, opening and closing morphological operations serve as noise removal operators because opening removes small objects while closing removes small holes present in the binary or greyscale image. One particular significance of dilations and erosions is their idempotent character. Therefore, these two operations

can be iteratively reapplied without further changes implemented on the previously transformed result. Other properties include increasing, extensive and translation invariant functions.

Opening and closing morphological operations can be used to construct filters similar to spatial filters. The objective is to remove noise whilst distorting (blurring) the image as little as possible. Fig. 4.8 illustrates the opening process of the morphological filtering where erosion is initially implemented. Fig. 4.8(a) shows the binary image corrupted with noise. Fig. 4.8(b) shows the structuring element used. Fig. 4.8(c) shows the implementation of erosion and Fig. 4.8(d) shows the implementation of dilation on the eroded results to obtain the opening set.

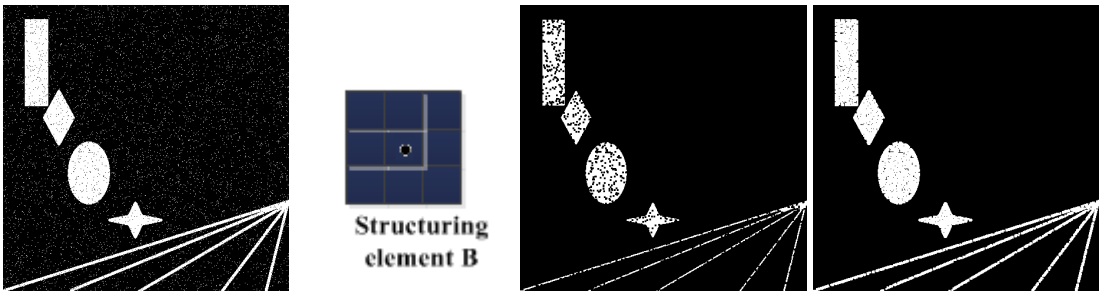


Figure 4.8: (a) Noisy image. (b) Structuring element. (c) Eroded image. (d) Opening of A.

It is observed that the background noise was completely eliminated in the erosion stage of the opening process but the noise elements contained within the shapes were increased. This increase is reduced or removed completely when the dilation stage of the opening process is implemented. Fig. 4.8(d) shows this result. Alternatively, Fig. 4.9 illustrates the closing process of the morphological filtering where dilation is initially implemented and the dilated results are subsequently eroded to obtain the closing set.

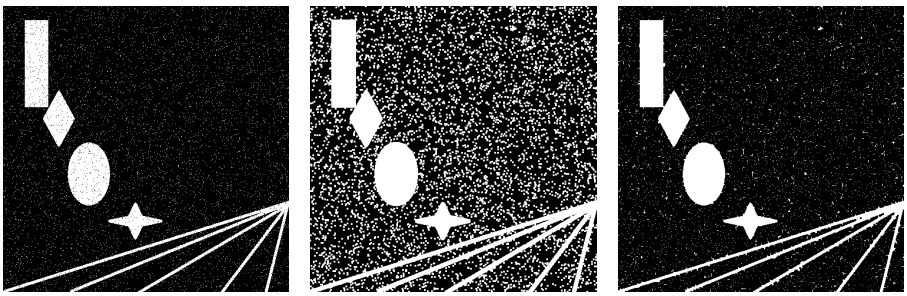


Figure 4.9: (a) Noisy image. (b) Dilated image. (d) Closing of A.

It is observed that the noise removal implementation is not successful due to initial implementation of the closing process rather than the opening process. Therefore, in order to implement morphological

filtering for noise removal, the opening process must be implemented first. Then any other required removal or gap filling can be implemented with the closing process. Fig. 4.10 illustrates the correct sequence of the opening and closing stages of morphological filtering.

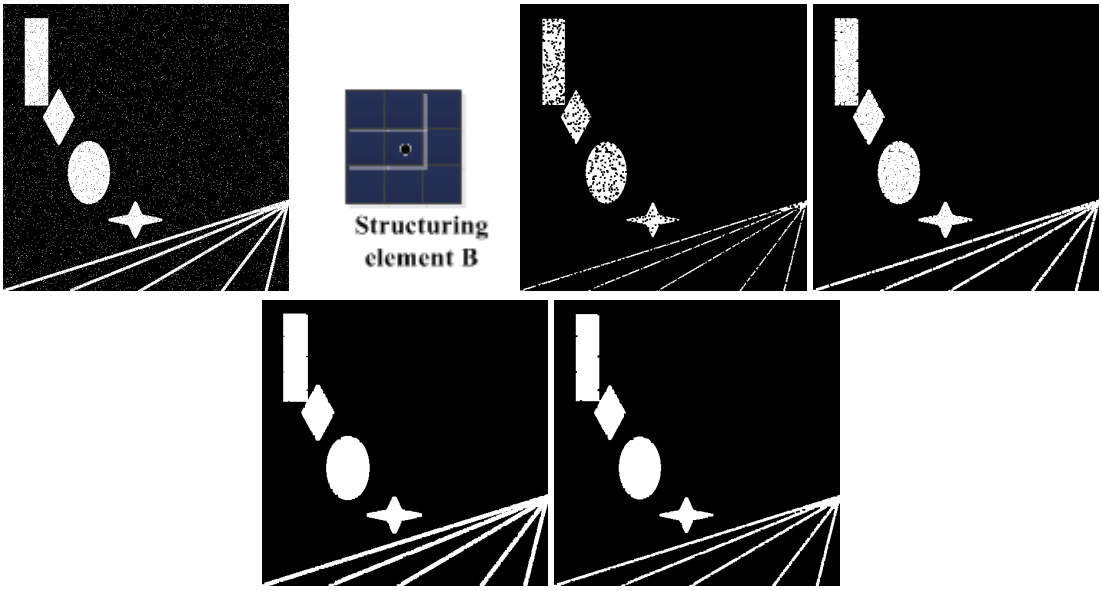


Figure 4.10: (a) Noisy image. (b) Structuring element. (c) Eroded image. (d) Opening of A. (e) Dilation of the opening. (f) Closing of the opening.

After the dilation-erosion process (repeated process in Fig. 4.8(c) and Fig. 4.8(d)), it was observed that new gaps were created between the shapes' ridges. Therefore, the closing process was implemented on the opened results. Firstly, the opened results are dilated to produce the results shown in Fig. 4.10(e) where most of the gaps were filled but the ridges were thickened. Hence, the implementation of the erosion stage in the closing process corrected the thickened ridges. Fig. 4.10(f) shows the final result that is remarkably clean of noise specks although there are some visible gaps in the print ridges.

CHAPTER 5 TRACKING USING PARTICLE FILTER ALGORITHM

"Nothing in life is to be feared,
it is only to be understood.
Now is the time to understand more,
so that we may fear less."

Marie Curie

5.1 CHAPTER OVERVIEW

Object tracking is considered as one of the important tasks in computer vision [52]. It is the process where moving objects are detected and the behaviour of the objects are recognised in a frame-to-frame analysis. Movements of a person (or persons) are observed and monitored over a period of time. Then, all the positions of the people are located and collected in every frame of video. This might serve as a challenging task because human beings have similar appearances so it is important to track multi-targets with state-of-the-art tracking techniques. Tracking is a key component in people counting because it provides the ability to monitor and estimate the location of the person. In this chapter, multi-target tracking approaches are explored and explained in detail.

5.2 STATE-OF-THE-ART TRACKING TECHNIQUES

There are three types of tracking methods: point, kernel and silhouette tracking methods [52]. A detailed overview is provided for four types of point and kernel tracking methods: mean-shift blob and KLT (Kanade-Lucas-Tomasi) tracking methods which fall under template kernel tracking; and Kalman and particle filters which fall under statistical point tracking[52].

Mean-shift algorithm is an algorithm that was originally developed for data analysis [62]. However, it was successfully adapted for image segmentation [63] and object tracking [53], [64]. The Mean-shift blob tracking algorithm is one of the popular tracking algorithms because it is simple to implement and computationally efficient [65]. It is explained as follows.

Given an input image that contains an object that needs to be tracked, each pixel (intensity) value of the image are denoted as sample points and they are weighted according to the following criteria. Each background pixel (excluding the object) is assigned to a low weight value while each foreground pixel of the object is assigned to a high weight value. These weights are then combined as sample weights in a local neighbourhood with a set of kernel weights in order to produce an offset [66]. The offset is used to track the centroid of the blob in the image.

The sample weights are examined in a local neighbourhood. The size of the neighbourhood is significantly dependent on the size of the mean-shift kernel. Therefore, the size selection is very important because, if the selected size is too large, then unwanted pixels (which contain background pixels) are included in the sample weights. Similarly, if the selected size is too small, necessary pixels (which do not contain enough foreground pixels) are not included in the sample weights.

Although, the mean-shift blob tracker is simple and efficient, its accuracy is largely dependent on fine-tuning the spatial, colour range bandwidths parameters [52] and the scale selection of the kernel.

KLT (Kanade-Lucas-Tomasi) tracker algorithm is the process of extracting the best features in an input image which are optimal for tracking. Then, the positions of the features are monitored over a pre-defined period of time. The algorithm highlights the importance of extracting the best features in an image so that these features can be accurately tracked. As a result, the features that need to be tracked must be relatively unique and be able to be compared with other points in the image [67].

The KLT tracker algorithm is explained as follows. The eigenvalues of the gradient matrix are computed. If the values of the two eigenvalues are larger than the value of a predefined threhold, then the features are selected [68].

Additionally, the features are correctly tracked by the implementation of an improved KLT tracker based on affine image changes [69]. The appearance of a feature is observed in the first and current

image frame. The dissimilarities between these frames are measured. If the dissimilarity is large, then the feature is not selected. This way, only the best features are selected for tracking.

Kalman filter is a mathematical process that predicts the position of a targeted object or objects in a current frame from a series of previous noisy measurements [50]. That is, the current state is estimated by using the estimated-previous state and the current measurements. If all noise is Gaussian-based, then it is regarded as an optimal linear estimator where the mean and covariance are used to predict and estimate the targeted object locations.

In the Kalman filter, the tracking system is represented with a state-space model. Then, the space of the filter is represented with two main variables: posteriori state estimate and posteriori error covariance matrix. The posterior state is estimated from collection of observations that are measured from the starting time until time k. Similarly, the posterior error covariance matrix is the measure of the estimated accuracy of state estimate from the starting time until time k.

Although the Kalman filter can be computed by using a single equation, it is often computed with two main equations: time update (predict) and measurement update equations [51]. The time update equation firstly computes both the current state and error covariance matrix estimates (expected values) at the current time t. These estimates are then regarded as priori estimates for time $t+1$. It is important to note that the estimates do not include the current observation information.

Then, for time $t+1$, the measurement update equation combines the priori (predicted) estimates with the observation information at the current time t. The updated estimate is then regarded as the posteriori state estimate.

In the case of object tracking, the video sequences are converted into frames. Given that the first frame is regarded as the current time (frame), the priori estimates are measured by computing the current state and error covariance matrix estimates at the current frame. Then, the posteriori state estimate is measured by combining the predicted estimates and the observation information at the second frame.

The observation information is very critical for the successful tracking of objects in the frames. The information is what each object in the frames is represented by. In the tracking system, each object is bounded with a bounding box. Therefore, the observation information includes:

- the *xy* coordinates of the object's location,
- the change in the *xy* coordinates of the object's location during subsequent frame (time) intervals,
- the width and height of the object's bounding rectangle, and
- the change in the width and height of the object's bounding rectangle during subsequent frame intervals

Kalman filtering is only an optimal solution when linear functions are observed and the state variables are normally distributed, that is, they have a Gaussian distribution [52]. However, for non-linear functions, it is not the best. Therefore, the extended Kalman filter (EKF) was implemented by [70]. The EKF linearises the non-linear functions by using Taylor series expansion. However, the state variables are still conditioned to be normally distributed. Therefore, when an object state cannot be assumed to be Gaussian and it is non-linear, the state estimation can be performed using particle filters [71].

Particle filters are also known as Sequential Monte Carlo methods. The particle filter is a model estimation technique that is used to implement recursive Bayesian filter by Monte Carlo sampling. It computes the estimation of the distribution of each target state rather than computing the estimation of each target state which is how it is implemented in the Kalman filtering process.

It is regarded as the bootstrap filter because the density of the target state vector is represented as a set of random particles. These are approximate probability density function of each target state. Furthermore, iterative weighting is applied when fast camera motion is observed [72]. Since particle filters are recursive Bayesian filters, they can also be implemented in two steps, namely, prediction and correction (update).

Therefore, with regards to object tracking, particle filters are employed to obtain the approximated probability distribution of each target state by computing the prediction and correction equations. In the prediction step, new sets of random particles (samples) are generated for each selected sample (obtained through probability distribution computation).

Subsequently, the weights corresponding to these samples are computed using the measurements in the correction step. Therefore, the position of the targeted object can be estimated using the newly-

obtained random samples.

Kalman and particle filters are primarily applicable to single object tracking. It is therefore a requirement to obtain a joint solution of data association and state estimation problems in order to track multiple persons in a frame-to-frame-analysis.

5.3 PARTICLE FILTER

Kalman filter is extensively used for object tracking in computer vision algorithms [52]. It has two limitations: the presented model is assumed to be linear and the state variables are assumed to be Gaussian - normally distributed. Therefore, if a non-linear model is presented with non-Gaussian state variables, the Kalman filter will fail (due to poor state estimations).

As a result, an extended Kalman filter (EKF) was proposed where the provided non-linear model is approximated into a linear model. However, the filter still assumes the state variables to be normally distributed. These assumptions can be successfully removed in particle filtering [52] [71]. Fig. 5.1 illustrates the result of extended Kalman filtering and particle filtering of a non-linear model.

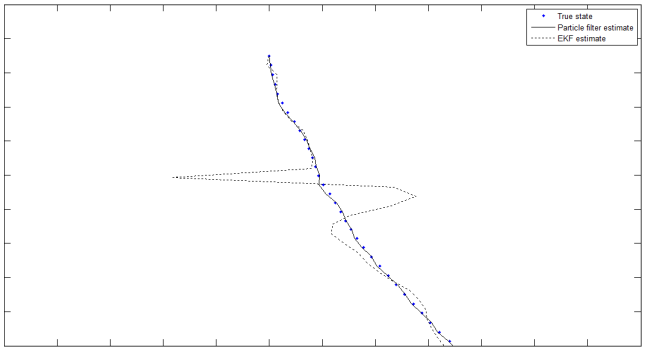


Figure 5.1: The result of extended Kalman filtering and particle filtering of a non-linear model (from observed data).

The particle filter employs a sequential Monte Carlo framework where the state density is represented by a set of random samples (particles). In object tracking, states are the location of the object that will be tracked in each frame. Therefore, the particle filter is concerned with tracking multiple (or single) target objects in an image. This is implemented by approximating the filtered posterior distribution by a set of weighted particles. The particles are weighted by computing the maximum likelihood value. Then, these particles are propagated according to the system (non-linear) model. Also, the particle

filter is known as the particle bootstrap filter because it follows a recursive approach.

Suppose a sequence of image frames is provided and it is required to track multiple targets in the image frames. The *xy* coordinates of each target location are obtained and stored as the state density for each frame. Therefore, for each frame, the state density is represented by a set of random samples with associated weights. The generation of the samples is explained as follows:

The state x_k is assumed to evolve according to the following model [73] in eq. 5.1

$$x_{k+1} = f_k(x_k, w_k) \tag{5.1}$$

where k is the frame time, f_k is the system transition function and w_k is the zero mean white-noise sequence, independent of past and current states. It is assumed that the model is non-linear, the noise is non-Gaussian and the probability density function (PDF) of w_k is known. At discrete times, measurements y_k is represented in eq. 5.2 as:

$$y_k = h_k(x_k, v_k) \tag{5.2}$$

where h_k is the measurement function and v_k is the measurement noise. Similarly, it is assumed that v_k is non-Gaussian and its PDF is known. Suppose that PDF $p(x_{k-1}|D_{k-1})$ is known at frame time $k-1$. That is, the location of the target is known at the initial frame. The PDF of the target state at frame time k is then represented in eq. 5.3 as:

$$p(x_k|D_{k-1}) = \int p(x_k|x_{k-1})\, p(x_{k-1}|D_{k-1})\, d_{x_{k-1}} \tag{5.3}$$

where $p(x_k|x_{k-1})$ is a Markov model, a probabilistic model of the state. It is represented in eq. 5.4 as:

$$p(x_k|x_{k-1}) = \int \delta\left(x_k - f_{k-1}(x_{k-1}, w_{k-1})\right)\, p(w_{k-1})\, d_{w_{k-1}} \tag{5.4}$$

and $p(x_k|D_k)$ is represented in eq. 5.5 as:

$$p(x_k|D_k) = \frac{p(x_k|D_k)\ p(x_k|D_{k-1})}{p(y_k|D_{k-1})} \tag{5.5}$$

Therefore, the set of random samples are obtained from $p(x_{k-1}|D_{k-1})$. The samples are represented in eq. 5.6 as:

$$\{x_{k-1}(i) : i = 1, ..., N\} \tag{5.6}$$

where k is the frame time, i is each sample and N is the total number of particles. It is important to note that the higher the number of particles, the better the accuracy of the target state estimation. Then, the samples are propagated and updated recursively so that these sets of values $\{x_{k-1}(i) : i = 1, ..., N\}$ are obtained which are approximately distributed as $p(x_k|D-k)$ [73]. The particle filter is implemented in two distinct stages: prediction and correction.

In the prediction stage, each of the N samples is passed through the non-linear model in order to obtain new samples x_k from the prior distribution at each frame time-step k. x_k is defined in eq. 5.1 but it is adapted into eq. 5.7 below for better understanding.

$$x_k = f_{k-1}\left(x_{k-1}(i), w_{k-1}(i)\right) \tag{5.7}$$

Once the measurements y_k are obtained, the weights are evaluated and normalized by resampling in the correction (update) stage. The weights are evaluated by computing the likelihood of each sample. The normalized weight w_i for each sample i is represented in eq. 5.8 as:

$$w_i = \frac{p\left(y_k|x_k(i)\right)}{\sum_{j=1}^{N} p\left(y_k|x_k(i)\right)} \tag{5.8}$$

Once the weights are successfully normalized, the effective number of samples are estimated. The effective samples $\hat{N}_{eff}$ are computed using eq. 5.9:

$$\hat{N}_{eff} = \frac{1}{\sum_{j=1}^{N}(w_i^2} \tag{5.9}$$

Finally, resampling is performed if the number of effective samples obtained is less than the given threshold value N_{thresh}. The resampling process is explained as follows. A random sample r is drawn from a uniform distribution and it is evaluated using eq. 5.10:

$$\sum_{j=0}^{M-1} w_i < r \leq \sum_{j=0}^{M} w_i \tag{5.10}$$

where $w_0 = 0$ is selected as the initial weight of the sample. Each state x_k is then determined by computing the mean of the effective samples, once the appropriate weight is obtained. It is important to note that this is a recursive process. Therefore, the two steps discussed above are performed recursively. The particle filter is implemented as sequential importance re-sampling (SIR). Particle filter is computationally more expensive than other tracking algorithms (Kalman filter, Mean-shift, etc) but it produces the highest accuracy in comparison. Algorithm 1 shows the pseudo-implementation of the SIR particle filtering algorithm.

Algorithm 1 SIR particle filter iteration

procedure SIR(k, X_k)
    Input Parameters:
    k = frame time-step
    X_k = k-th particle set
    for i = each iteration **do**
        Initialize the initial state for first frame x
        Set $\mathbf{X}_t = 0$
        Generate the set of random particles $\{x_{k-1}(i) : i = 1, ..., N\}$
        for $j = 1 - N$ each sample or particle **do**
            Pick the jth sample
            Draw $x_k^j = p(x_k|x_{k+1}$
            Calculate the normalized weight w_k^j
            Add (x_k^j, w_k^j) to $\mathbf{X}_k$
        end for
        for $i = 1 - m$ **do**
            Calculate the effective number of particles
            Perform resampling so that the appropriate weights w_k^i are obtained
            Select effective particle x_k^i
            Add x_k^i to $\mathbf{X}_k$
        end for
    end for
end procedure

CHAPTER 6 IMPLEMENTATION: APPROACH

"*It is the theory which decides*
what we can observe."

Albert Einstein

6.1 CHAPTER OVERVIEW

This chapter provides a detailed analysis of the approach followed for the proposed people counting algorithm which uses the novel background subtraction method known as the single-pixel method. The detailed analysis of the virtual-line direction-estimation method is also explained in this chapter where the directions in which the people are headed are estimated prior to the counting process.

Furthermore, the detailed analysis of the implemented techniques such as background subtraction method using frame differencing and single-pixel method, morphological processing, image smoothing, feature extraction and virtual-line direction-estimation method are presented.

6.2 IMPLEMENTATION

An input video feed or real-time video recording that shows active movements of people in an outdoor or indoor environment must be provided so that accurate functioning of the proposed algorithm can be achieved. One of the research objectives questions the robustness of people counting algorithms, so the proposed approach should be implementable on different scenarios. Examples of these scenarios are presented in Fig. 6.1.

Once input video feeds of active moments of people in outdoor or indoor environments are provided, the videos must be converted into image frames. After video-to-image conversion is implemented,

(a) Shopping mall entrance/exit

(b) Airport entrance/exit

(c) Wembley stadium

(d) PETS

Figure 6.1: Typical people counting scenes.

the following techniques are applied to the input frames: background subtraction method using frame differencing and single-pixel method, morphological processing, image smoothing, feature extraction and virtual-line direction-estimation method. These steps consolidate the approach taken for this study and are illustrated in Fig. 6.2.

6.3 VIDEO-TO-IMAGE CONVERSION

The input data can be provided in a video or image format. If data is provided in the video format, it will be required to convert the video into a MATLAB usable format so that the algorithm can be implemented over a frame-to-frame analysis. Otherwise, the algorithm can be implemented using the original input data if it is already provided in a MATLAB usable format.

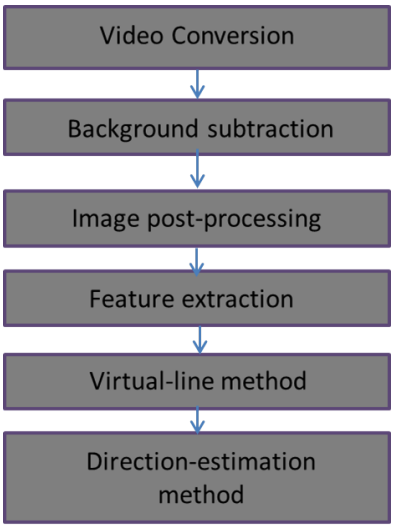


Figure 6.2: Illustration of the approach followed to implement robust people counting algorithm.

6.4 BACKGROUND SUBTRACTION

One of the most important and crucial step in people counting is background subtraction as presented in *Chapter 3*. Background subtraction consists of a process where a reference background model is firstly obtained, followed by a subtraction process between the background model and other input images, where the primary objective is to extract useful and necessary foreground pixels using a specified threshold value. In this study, an additional step, known as single-pixel method, is added to the background subtraction process as presented in Fig. 6.3.

6.5 MORPHOLOGICAL IMAGE PROCESSING

Two morphological processes namely, dilation and hole filling, will be performed on the input binary images. Morphological dilation is implemented so that the foreground objects (blobs) can be enlarged and visible, while morphological hole filling is performed so that all the broken gaps visible in the blobs can be repaired by filling the holes with foreground pixels.

6.6 SINGLE-PIXEL METHOD

The single-pixel method is the additional step that is included in the background subtraction process. It is the process where each object in the scene is represented with xy-coordinates. Therefore, xy center coordinates are obtained for each person in this step. The idea is to firstly represent each

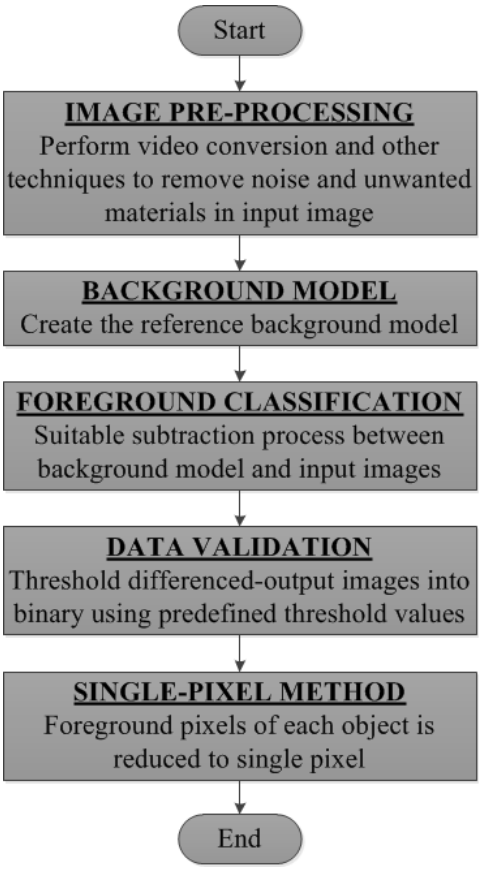


Figure 6.3: Illustration of the implementation of background subtraction including single-pixel method.

person in the scenes as blobs so that single-pixel method can be successfully achieved. Once the people in the scenes are bounded with bounding boxes, the dimensions and *xy* center coordinates of each bounding box of the people, as well as the current frame number, are collected and stored in a *csv* file. The *xy* center coordinates are regarded as the ground truth because they represent the actual location of each person in the scene over the frame-to-frame analysis.

The single-pixel method aims to answer two of the research questions defined in the study. The first objective questions the development of a people counting algorithm that is invariant to occlusions, illumination conditions, human pose and direction; while the second objective questions the development of a people counting algorithm with low complexity and high performance accuracy. The single-pixel process is divided into four steps, namely, blob-like shape analysis, median area value computation and single-pixel centroid *xy*-coordinates as shown in Fig. 6.4.

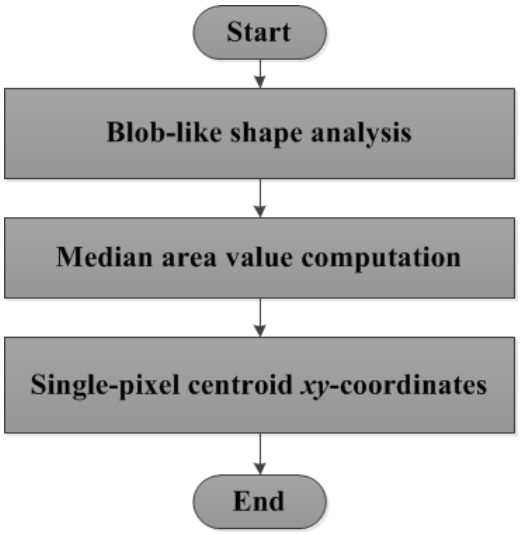


Figure 6.4: Illustration of the single-pixel method process.

6.6.1 Blob-like shape analysis

Foreground classified pixels are obtained using the background subtraction process and these pixels are displaced on output images over time t on a frame-to-frame analysis. Further image pre-processing techniques such as morphological processing and image smoothing can be implemented in order to remove extra unwanted pixels and noise still present in the output images. Once the blobs are obtained, they are bounded with bounding boxes.

A bounding box, also known as minimum bounding rectangle, is a rectangular model which contains measurements such as area, perimeter and boundary xy coordinates of the binary images. It is the smallest rectangle that can be created for each two-dimensional (2D) object that is found in a 2D (x, y) coordinate system. The boundary coordinates specifies the lower-left, lower-right, upper-left and upper-right corner xy coordinates of the bounding box.

6.6.2 Median area value computation

The general idea is to represent each person in the scene as blobs, then segregate the blobs using bounding boxes. Once this process is successfully implemented, the area pixel (intensity) value of each blob is calculated for each scene in the frame-to-frame analysis. Then, a median (average) area pixel value is calculated for each blob which is used as the deciding criteria on which objects to bound with bounding boxes. Suppose 50 frames are provided and the first detected person is denoted as the

first blob. Area pixel value is calculated for the blob in each frame so that an average area can be calculated over the 50 frames. The process is then repeated for the second blob, third blob, and so on, until the area pixel values for all the blobs have been calculated. The median area value **MAV** per scene t is defined in eq. 6.1 as:

$$\mathbf{MAV}_t = \frac{\sum_{i=1}^{O_N} A_{i,t}}{O_N} \text{ where } \{t = 1,2,3,...,N\} \tag{6.1}$$

where:

- $\mathbf{I}(x,y)$ denotes binary images,
- O_N is the total number of objects,
- $A_{i,t}$ is the area value per object i in frame t,
- O is the total number of frames, and
- t is the total number of frames (scenes) in the frame-to-frame analysis.

6.6.3 Single-pixel centroid *xy*-coordinates

Once the objects representing the people are bounded with bounding boxes, the centroid *xy*-coordinates of these objects are computed in the last step. These values are fed into the system in order to:

- count the number of people present in each scene,
- estimate the direction in which the people are headed and
- track the movements of the people using particle filtering tracking technique.

Algorithm 2 shows the pseudo-implementation of the single-pixel method.

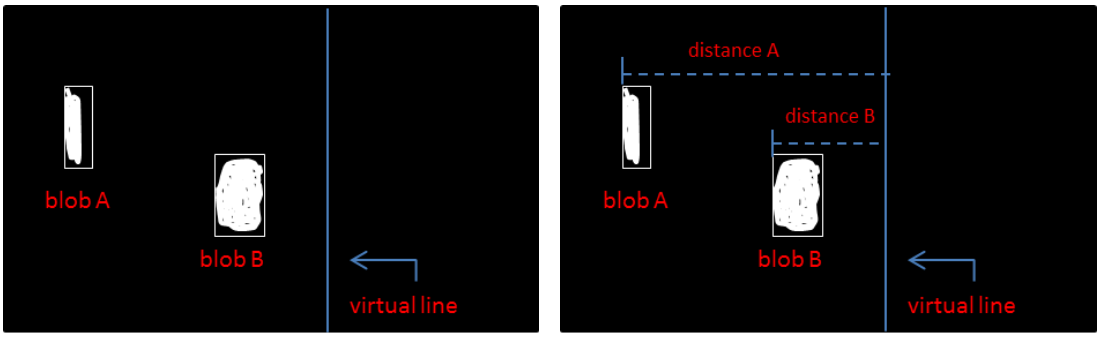
Algorithm 2 Single-pixel Method

procedure SINGLE-PIXEL(I, t, h, w)
    Input Parameters:
    $I(t)$ = pre-processed input images for all frames t where $i = \{1, ..., t\}$
    h, w: height and width of the input images
    for i = each frame **do**
        Read in the pre-processed (differenced and morphologically-processed) input image per frame i
        Locate all the objects o in each frame i:
        for j = each object **do**
            Compute the area $A_{i,j}$ of object o_j
        end for
        Compute the median area value MAV_i of all objects o in each frame i
        if vertical virtual-line **then**
            for $k = 1 : w$ **do**
                Compute $x(k_r = 2 * k - 1)$ to obtain only centroid x-coordinates in each frame i
            end for
        else if horizontal virtual-line **then**
            for $k = 1 : h$ **do**
                Compute $x(k_c = 2 * k - 1)$ to obtain only centroid x-coordinates in each frame i
            end for
        else
            otherwise nothing
        end if
    end for
end procedure

6.7 COUNTING: DISTANCE COMPUTATION

In this section, the counting method using distance computation is presented. It serves as a significant step in people counting algorithm because it requires actual counting of the people in the scenes.

In an ideal scenario, each person is represented by blobs where each blob is bounded with a bounding box. Then, the counting method can be implemented by counting all the boxes and returning the total number as the estimated number of people in the scene. Due to contributing factors such as occlusions, varying lighting conditions and cast shadows, it is not always possible to represent each person with a blob. As a result, a counting process is proposed where multiple people contained in blobs can be accurately counted using distance computation as well as camera angle and point of view. A case scenario presented in Fig. 6.5 is used to explain the process.



(a) **(b)**

Figure 6.5: Case scenario used to explain the counting process.

Two blobs are presented in Fig. 6.5a and denoted as blob A and blob B. Unknown to the viewer, each blob contains the same number of people. Suppose it is required to count the number of people in each blob by calculating the distance between each blob and the virtual line; once the distance is calculated, it will be observed that blob A is closer to the virtual line whilst blob B is further to the virtual line. However, it will be virtually impossible for the viewer to deduce that equal number of people is present in both blobs using the distance computations only. Therefore, another measure should be employed where a threshold value should be used to separate blobs in close proximities from blobs in far proximities. As a result, the blobs are classified according to their proximities as shown in Fig. 6.5b. With this, one is able to deduce that there are equal number of people in both blobs.

In summary, the proposed counting method is illustrated as follows. First, the distance computation is implemented where the proximity of each blob is computed by calculating the total distance between each blob and the virtual line. Using this, one is able to estimate the distance of the blob that is close or far to the virtual line. Then the blobs are classified according to short-distanced blobs or long-distanced blobs using a pre-defined threshold value. Short-distanced blobs are blobs that are closer to the virtual line or camera's point of view while long-distanced blobs are blobs observed at a further angle from the virtual line or camera's point of view.

The total distance **D** between each blob and virtual line is computed by implementing a subtraction process between the x-coordinates of the virtual line and x-center coordinates of each blob. If a vertical virtual-line is provided, the computation process of total distance $\mathbf{D}(x)$ is defined in eq. 6.2 as:

$$\mathbf{D}(x) = \mathbf{I}_t^v(x_1) - \mathbf{I}_t^{PersonZ}(x_c) \tag{6.2}$$

where:

- $\mathbf{I}_t^v(x_1)$ represents the x-coordinate of the virtual line v,
- x_1 represents the width of the virtual line,
- $\mathbf{I}_t^{PersonZ}(x_c)$ represents the x-centroid coordinates of each person,
- x_c represents the x-centroid coordinates, and
- Z is the detected person (represented as a blob) for each scene t.

If the horizontal virtual-line is provided, the computation process of total distance $\mathbf{D}(y)$ is defined in eq. 6.3 as:

$$\mathbf{D}(y) = \mathbf{I}_t^v(y_1) - \mathbf{I}_t^{PersonZ}(y_c) \tag{6.3}$$

where:

- $\mathbf{I}_t^v(y_1)$ represents the y-coordinate of the virtual line v,

- y_1 represents the height of the virtual line,
- $\mathbf{I}_t^{PersonZ}(y_c)$ represents the y-centroid coordinates of each person,
- y_c represents the y-centroid coordinates, and
- Z is the detected person (represented as a blob) for each scene t.

The difference between short-distanced and long-distanced blobs is computed by implementing both the median area value (**MAV**) computation process and distance computation process (**D**). Once all the distances of the blobs are calculated using either eq. 6.2 or eq. 6.3, the blobs are classified and grouped as either short-distanced or long-distanced blobs. The process of computing short-distanced blobs $\mathbf{SD}_{blobs}$ is represented in eq. 6.4 as:

$$\mathbf{SD}_{blobs} = \left\{ \begin{array}{ll} 1 & \text{if } \mathbf{D}(x) \text{ or } \mathbf{D}(y) < D_{threshold} \\ 0 & \text{if } \mathbf{D}(x) \text{ or } \mathbf{D}(y) \geq D_{threshold} \end{array} \right\} \tag{6.4}$$

where $D_{threshold}$ is the pre-defined threshold value. Similarly, the process of computing long-distanced blobs $\mathbf{LD}_{blobs}$ is represented in eq. 6.5 as:

$$\mathbf{LD}_{blobs} = \left\{ \begin{array}{ll} 1 & \text{if } \mathbf{D}(x) \text{ or } \mathbf{D}(y) \geq D_{threshold} \\ 0 & \text{if } \mathbf{D}(x) \text{ or } \mathbf{D}(y) < D_{threshold} \end{array} \right\} \tag{6.5}$$

The blobs are grouped according to their distance values and the average area value is computed for short-distanced and long-distanced blobs. If the vertical virtual-line is provided, the average value for short-distanced blobs $\mathbf{AV}_{short}^{v}$ is calculated using eq. 6.6:

$$\mathbf{AV}_{short}^{v} = \frac{\sum_{i=1}^{O_{N,short}} \mathbf{D}(x)}{O_{N,short}} \tag{6.6}$$

The average value for long-distanced blobs $\mathbf{AV}_{long}^{v}$ is calculated using eq. 6.7:

$$\mathbf{AV}_{long}^{v} = \frac{\sum_{i=1}^{O_{N,long}} \mathbf{D}(x)}{O_{N,long}} \tag{6.7}$$

If horizontal virtual-line is provided, the average value for short-distanced blobs $\mathbf{AV}^{h}_{short}$ is calculated using eq. 6.8:

$$\mathbf{AV}^{h}_{short} = \frac{\sum_{i=1}^{O_{N,short}} \mathbf{D}(y)}{O_{N,short}} \tag{6.8}$$

Similarly, the average value for long-distanced blobs $\mathbf{AV}^{h}_{long}$ is calculated using eq. 6.9:

$$\mathbf{AV}^{h}_{long} = \frac{\sum_{i=1}^{O_{N,long}} \mathbf{D}(y)}{O_{N,long}} \tag{6.9}$$

where:

- $O_{N,short}$ represents the total number of short-distanced blobs, and
- $O_{N,long}$ represents the total number of long-distanced blobs.

Finally, the process of accurately counting the number of people that are contained in each blob is illustrated as follows. If vertical virtual-line is provided, the process of counting the number of people in short-distanced and long-distanced blobs $\mathbf{C}^{v}$ is represented in eq. 6.10 as:

$$\mathbf{C}^{v} = \left\{ \begin{array}{ll} \frac{\mathbf{A}_{i,t}}{\mathbf{AV}^{v}_{short}} & \text{if } \mathbf{D}(x) < D_{threshold} \\ \frac{\mathbf{A}_{i,t}}{\mathbf{AV}^{v}_{long}} & \text{if } \mathbf{D}(x) \geq D_{threshold} \end{array} \right\} \tag{6.10}$$

where:

- $\mathbf{A}_{i,t}$ represents the area value of each blob, and
- $\mathbf{AV}^{v}$ represents the average value for either short-distanced or long-distanced blobs, if vertical virtual-line is provided.

Similarly, if horizontal virtual-line is provided, the process of counting the number of people in short-distanced and long-distanced blobs $\mathbf{C}^{h}$ is represented in eq. 6.11 as:

$$\mathbf{C}^h = \left\{ \begin{array}{ll} \dfrac{\mathbf{A}_{i,t}}{\mathbf{AV}^h_{short}} & \text{if } \mathbf{D}(y) < D_{threshold} \\ \dfrac{\mathbf{A}_{i,t}}{\mathbf{AV}^h_{long}} & \text{if } \mathbf{D}(y) \geq D_{threshold} \end{array} \right\} \tag{6.11}$$

where:

- $\mathbf{A}_{i,t}$ represents the area value of each blob, and
- $\mathbf{AV}^v$ represents the average value for either short-distanced or long-distanced blobs, if horizontal virtual-line is provided.

Algorithm 3 shows the pseudo-implementation of the virtual-line direction estimation method.

Algorithm 3 People counting process

procedure COUNT($I_t(x_p, y_q)$, N)
  Input Parameters:
  $I_t(x_p, y_q)$ pre-processed input images for all frames N where $t = \{1, 2, ..., N\}$
  p, q: width and height of the input images
  Choose either vertical v or horizontal h vertical-line
  for $i =$ each frame **do**
    Locate all the objects o in each frame i:
    for $j =$ each object **do**
      Collect centroid xy-coordinates of object o_j
      Calculate the area value $A_{i,t}$ for each object o_j
    end for
    Compute the average area value **MAV**
    for $j =$ each object **do**
      if Area value $A_{i,t} >$ average area value **MAV** **then**
        Calculate the distance **D** between each blob and virtual-line
        Compute the short-distanced $\mathbf{SD}_{blobs}$ and long-distanced $\mathbf{LD}_{blobs}$
        Calculate the average value for short-distanced blobs $\mathbf{AV}_{short}$
        Calculate the average value for long-distanced blobs $\mathbf{AV}_{long}$
        Compute the counting process **C**:
        if Distance **D** $<$ pre-defined threshold value $\mathbf{D}_{threshold}$ **then**
          Count the number of people contained in the short-distanced blobs
        else if Distance **D** $\geq$ pre-defined threshold value $\mathbf{D}_{threshold}$ **then**
          Count the number of people contained in the long-distanced blobs
        else
        end if
      end if
    end for
  end for

6.8 VIRTUAL-LINE DIRECTION ESTIMATION METHOD

In this section, virtual-line direction estimation method is presented where it is required to observe all the people in a frame-to-frame analysis over time t and determine the direction in which the people are headed. An entrance point is virtually created in the scene so that the people can be categorised as either going towards (entering) the entrance point or going away (leaving) from the entrance point of the scene. The virtual-line direction estimation method process is explained as follows:

A virtual-line is drawn as the entrance point of the scene; this line can be drawn vertically or horizontally. The virtual-line is formulated in eq. 6.12 as:

$$\mathbf{I}_t(x_p, y_q) = 1 \text{ where } \{t = 1,2,3,...,N\} \tag{6.12}$$

where:

- $\mathbf{I}(x,y)$ denotes binary images,
- N is the total number of frames,
- p is the total width and
- q is the total height of the binary image.

If a vertical virtual-line is required, the process is represented in eq. 6.13 as:

$$\mathbf{I}_t(x_1, y_q) = 1 \text{ where } \{t = 1,2,3,...,N\} \text{ and } \{y_q = y_1, y_2, ..., y_h\} \tag{6.13}$$

where a specific width is provided at x_1(i.e. $x_1 = 40$) and h is the height of the binary images. Similarly, the process of horizontal virtual-line is presented in eq. 6.14 as:

$$\mathbf{I}_t(x_p, y_1) = 1 \text{ where } \{t = 1,2,3,...,N\} \text{ and } \{x_p = x_1, x_2, ..., x_w\} \tag{6.14}$$

where a specific height is provided at y_1(i.e. $y_1 = 90$) and w is the width of the binary images.

Once the virtual-line is created, each person is observed and their location is collected over the frame-to-frame analysis using the single-pixel method. The process is represented in eq. 6.15 as:

$$\mathbf{I}_t^{PersonZ}(x_c, y_c) = \{\mathbf{I}_t^{Person1}(x_c, y_c), \mathbf{I}_t^{Person2}(x_c, y_c), \mathbf{I}_t^{Person3}(x_c, y_c), ..., \mathbf{I}_t^{PersonZ}(x_c, y_c)\} \tag{6.15}$$

where:

- (x_c, y_c) denotes the centroid coordinates and
- Z denotes the number of people in each scene t.

Then the centroid coordinates of the single pixels are compared with the coordinates of the virtual line. A subtraction process is implemented where the centroid coordinates of the single pixels are subtracted from the coordinates of the virtual line in order to provide output differences. A comparison check is then implemented on each output difference. If the output difference is less than 0, it is estimated that the person is heading towards the entrance point of the scene. Otherwise, it is estimated that the person is heading away from the entrance point of the scene.

If a vertical virtual-line is created, the subtraction process is executed between the x-coordinate of the virtual-line (x_1) and the x-centroid coordinates of the single pixels (x_c^p). If the person is headed towards the entrance point, the direction-estimation process $DE_{towards}$ is represented in eq. 6.16 as:

$$DE_{towards} = \left\{ \begin{array}{ll} 1 & \text{if } x_c^p - x_1 < 0 \\ 0 & \text{if } x_c^p - x_1 > 0 \end{array} \right\} \tag{6.16}$$

where:

- $DE_{towards}$ denotes the direction estimation process for people heading towards the entrance point, and
- $x_c^p = \{x_c^1, x_c^2, ..., x_c^{width}\}$ denotes the x-centroid coordinates of the singe pixels.

If the person is headed away from the entrance point, the direction-estimation process DE_{away} is represented in eq. 6.17 as:

$$DE_{away} = \left\{ \begin{array}{ll} 1 & \text{if } x_c^p - x_1 > 0 \\ 0 & \text{if } x_c^p - x_1 < 0 \end{array} \right\} \tag{6.17}$$

where:

- DE_{away} denotes the direction estimation process for people heading away from the entrance point and
- $x_c^p = \{x_c^1, x_c^2, ..., x_c^{width}\}$ denotes the x-centroid coordinates of the singe pixels.

Similarly, if a horizontal virtual-line is created, subtraction process is implemented between the y-coordinates of the virtual-line (y_1) and the y centroid coordinates of the single pixels (y_c^q). If the person is headed towards the entrance point, the direction-estimation process $DE_{towards}$ is represented in eq. 6.18 as:

$$DE_{towards} = \left\{ \begin{array}{ll} 1 & \text{if } y_c^p - y_1 < 0 \\ 0 & \text{if } y_c^p - y_1 > 0 \end{array} \right\} \tag{6.18}$$

where:

- $DE_{towards}$ denotes the direction estimation process for people heading towards the entrance point and
- $y_c^p = \{y_c^1, y_c^2, ..., y_c^{height}\}$ denotes the y-centroid coordinates of the singe pixels.

If the person is headed away from the entrance point, the direction-estimation process DE_{away} is represented in eq. 6.18 as:

$$DE_{away} = \left\{ \begin{array}{ll} 1 & \text{if } y_c^p - y_1 > 0 \\ 0 & \text{if } y_c^p - y_1 < 0 \end{array} \right\} \tag{6.19}$$

where:

- DE_{away} denotes the direction estimation process for people heading away from the entrance point and

- $y_c^p = \{y_c^1, y_c^2, ..., y_c^{height}\}$ denotes the y-centroid coordinates of the singe pixels.

As a result, if the person$'$s direction is towards the entrance point, the person is counted to be entering the specified area. If not, the person is counted as exiting the specified area. If the person is present in the scene but never gets to the entrance point, the person is counted but "unassigned". Algorithm 4 below shows the pseudo-implementation of the virtual-line direction estimation method.

Algorithm 4 Virtual-line Direction Estimation Method

procedure VLDSL($I_t(x_p, y_q), N$)
    Input Parameters:
    $I_t(x_p, y_q)$ pre-processed input images for all frames N where $t = \{1, 2, ..., N\}$
    p, q: width and height of the input images
    for $i =$ each frame **do**
        if vertical virtual-line **then**
            $I_t(x_1, y_q) = 1$ where $t = \{1, 2, ..., N\}$ and $y_q = \{y_1, y_2, ..., y_h\}$
            Locate all the objects o in each frame i:
            for $j =$ each object **do**
                Collect centroid xy-coordinates of object o_j
            end for
            Compute the direction-estimation process DE:
            if $x_c^p - x_1 < 0$ **then**
                Person is heading away from the entrance point of frame i
            else if $x_c^p - x_1 > 0$ **then**
                Person is heading towards the entrance point of frame i
            else
                Person is neither heading towards nor away from the entrance point of frame i
            end if
        else if horizontal virtual-line **then**
            $I_t(x_p, y_1) = 1$ where $t = \{1, 2, ..., N\}$ and $x_p = \{x_1, x_2, ..., x_w\}$
            Locate all the objects o in each frame i:

Algorithm 4 Virtual-line Direction Estimation Method (continued)

```
            for j = each object do
                Collect centroid xy-coordinates of object o_j
            end for
            Compute the direction-estimation process DE:
            if y_c^q − y_1 < 0 then
                Person is heading away from the entrance point of frame i
            else if y_c^q − y_1 > 0 then
                Person is heading towards the entrance point of frame i
            else
                Person is neither heading towards nor away from the entrance point of frame i
            end if
        else
        end if
    end for
end procedure
```

In summary, all the implementable techniques of the approach to robust people counting algorithm is consolidated into a work flow and illustrated in Fig. 6.6.

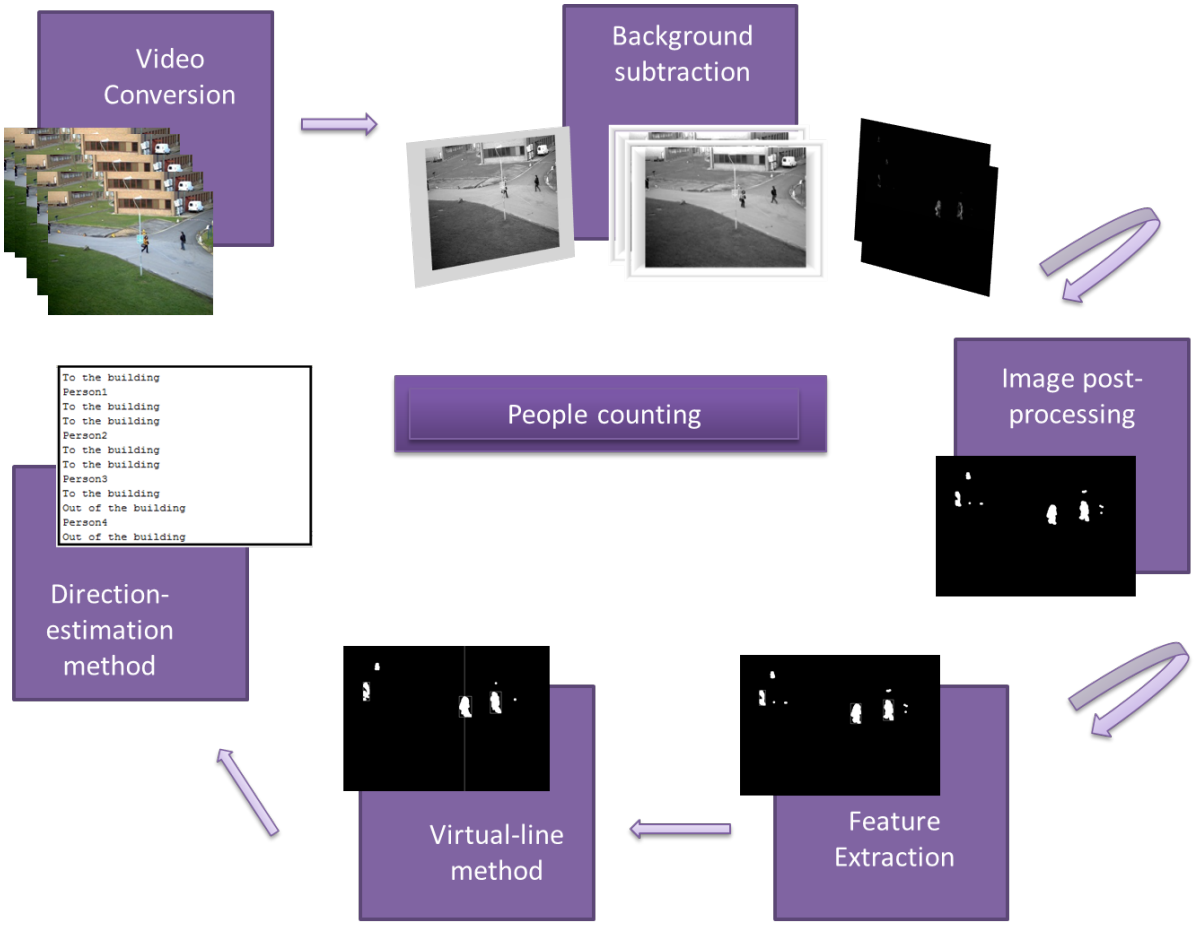


Figure 6.6: Workflow of the approach followed to implement robust people counting algorithm.

CHAPTER 7 EXPERIMENTAL RESULTS

"*No amount of experimentation*
can ever prove me right;
a single experiment can prove me wrong."

Albert Einstein

7.1 CHAPTER OVERVIEW

This chapter presents public outdoor and private indoor environments which are used as test input data. Each test environment is analysed and explained in detail. Then, the results are contextually evaluated.

7.2 INPUT DATA

In order to test the robustness of the approach, different test scenarios were obtained for input data. These scenarios include public-outdoor and private-indoor environments which are explained in detail as follows. A video sequence based on PETS dataset [74] is provided as an illustration of public outdoor environment while a video sequence based on EPFL dataset [75] is provided as an illustration of public indoor environment.

Input data are mostly provided in video formats. In some cases, an input video can be provided in a MATLAB unreadable format. This can be resolved by converting the input video into a readable and acceptable standard before video-to-image conversion is implemented. This is the process where the input video is converted into image frames and saved into output folders for later use (or observation).

7.2.1 Public outdoor environment

The public outdoor environment is represented with PETS dataset [74]. The PET datasets consist of multiple crowd scenarios ranging from simple to complex scenes where different levels of difficulties are observed over varying time periods. The scenarios were captured using various cameras.

The dataset framework consists of four datasets namely: *S*0 till *S*3. These are specifically designed for the implementation of different computer vision functions such as people tracking, people count and density estimation, multiple flow, event recognition and camera calibration [1]. The representation of each dataset is illustrated as follows:

1. *S*0 dataset: This dataset represents the background model of the system so it contains eight different views of the target area. The views are captured from eight cameras mounted in different locations of the target area. Fig. 7.1 shows the sample frames of the *S*0 dataset.

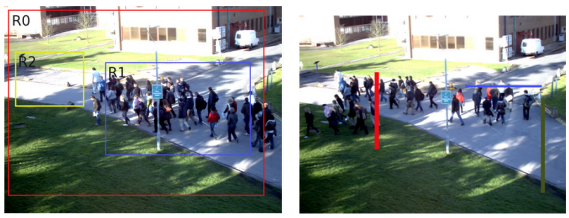


Figure 7.1: Sample frames of the *S*0 dataset [74].

2. *S*1 dataset: This dataset is designed for the implementation of people count and density estimation, therefore it contains images that are captured so that the number of people in the targeted area can be counted using density estimation. This is the process where the density of crowd is estimated by computing the maximum occupancy of groups of people. Fig. 7.2 shows sample frames of the *S*1 dataset.

Figure 7.2: Sample frames of the *S*1 dataset [74].

3. *S*2 dataset: This dataset is designed for the implementation of people tracking therefore it contains images that are captured so that all the individuals within the sequence can be tracked by estimating their locations. Fig. 7.3 shows the sample frames of the *S*2 dataset.

Figure 7.3: Sample frames of the *S*2 dataset [74].

4. *S*3 dataset: This dataset is designed for the implementation of multiple flow analysis and event recognition therefore it contains images where multiple flows can be observed, detected and estimated in each frame of the sequence. Also, crowd activities can be observed in event recognition.

Since *S*1 dataset is specifically designed for people counting, it is regarded as the best choice for the implementation of the proposed algorithm. As shown in Fig. 7.2, the *S*1 dataset was captured using eight cameras that were tagged as view 001 until view 008. Each of these views represented the angles and directions in which the cameras were mounted.

View 001 of *S*1 dataset was selected as the input data for public-outdoor environment. A network security camera at a rate of 7 frames per second was used to capture the dataset. This camera is an AXIS 223*M* 2.0 Megapixel that produces high-performance quality and exceptional image details. Also, it delivers higher resolution images when compared to standard analog camera. Each frame

measures 768 by 576 pixels [74].

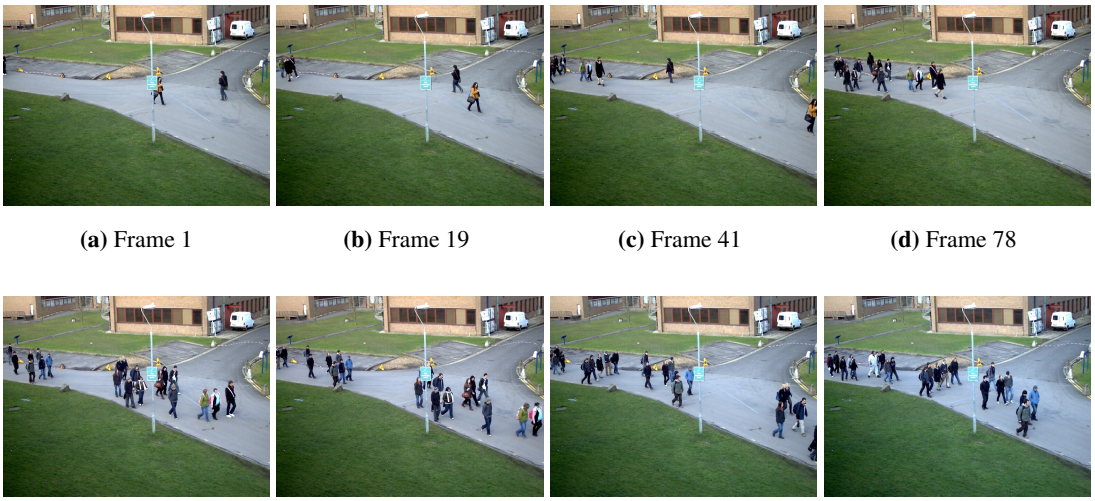
7.2.2 Private indoor environment

The private indoor environment is represented with EPFL dataset which contains scenarios that are specifically designed for people detection and tracking framework [75]. The video sequence that contains the training session of a local basketball team is used as input data in private indoor environment. The dataset of the basketball sequence was captured with four DV cameras at a rate of 25 frames per second. Then, the frames were encoded with Indeo 5 [75]. Fig. 7.4 show the sample frames of the indoor training session.

Figure 7.4: Sample frames of indoor basketball training session obtained for private indoor environment [75].

7.3 VIDEO-TO-IMAGE CONVERSION

Input data are collected from public-outdoor and private-indoor environment and used to implement the proposed approach. Fig. 7.5 illustrates some of the image frames after the video sequence has been converted. Frames 1, 19, 41, 78, 139, 158, 196 and 228 are observed in Fig. 7.5a - 7.5h respectively.

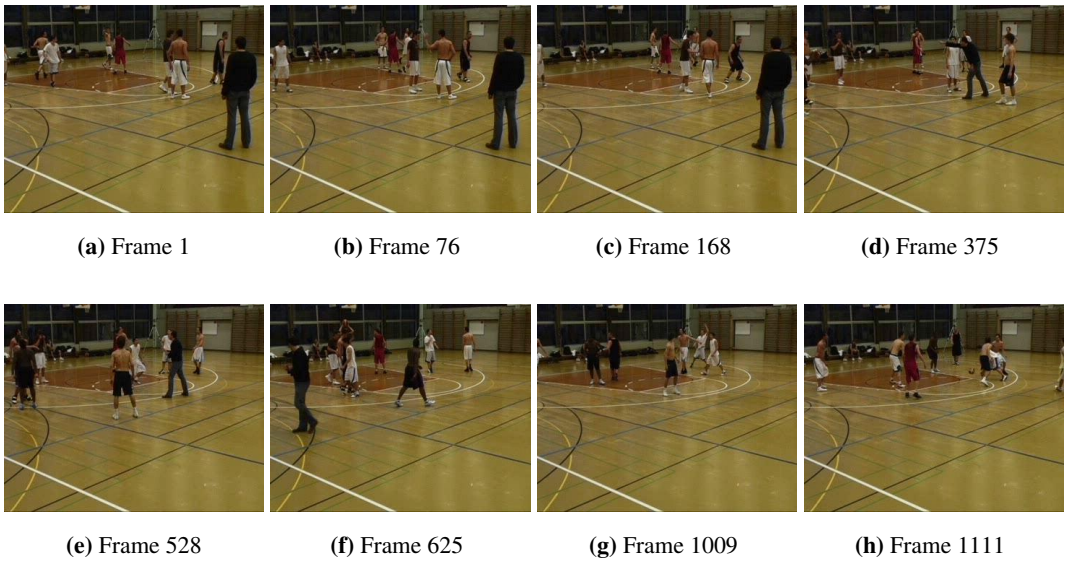


(a) Frame 1 **(b)** Frame 19 **(c)** Frame 41 **(d)** Frame 78

(e) Frame 139 **(f)** Frame 158 **(g)** Frame 196 **(h)** Frame 228

Figure 7.5: (a)-(h) Some of the image frames after video sequence of the moving people is converted into image frames.

In the private-indoor environment, the input video is converted to image frames and illustrated in Fig. 7.6. Frames 1, 76, 168, 375, 528, 625, 1009 and 1111 are observed in Fig. 7.5a - 7.5h respectively.



(a) Frame 1 **(b)** Frame 76 **(c)** Frame 168 **(d)** Frame 375

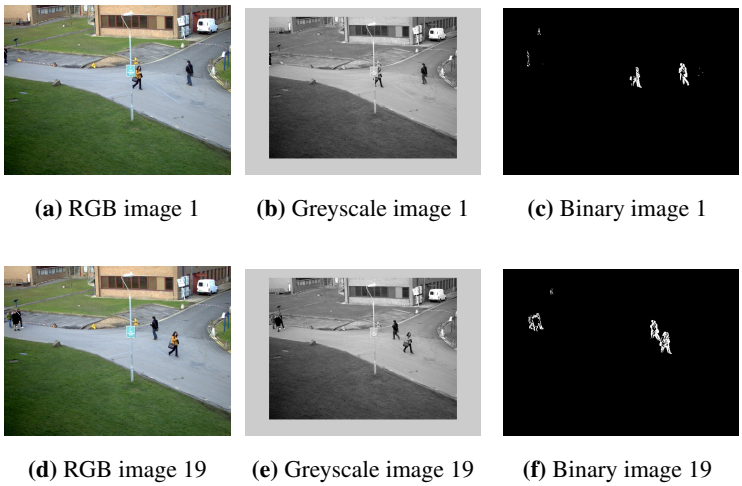
(e) Frame 528 **(f)** Frame 625 **(g)** Frame 1009 **(h)** Frame 1111

Figure 7.6: (a)-(h) Some of the image frames after video sequence of the moving people is converted into image frames.

7.4 IMAGE PRE-PROCESSING

The image preprocessing step is the process where unnecessary materials and noise are removed from the input images before major processing techniques are applied. The input images are provided in RGB format which have to be converted into greyscale images for easier input data functionality. The greyscale images are later converted to binary (black and white) images for simpler implementation of the algorithm.

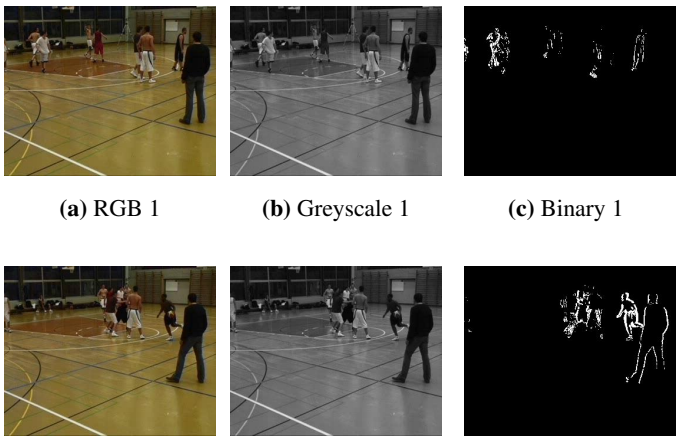
Fig. 7.7 and 7.8 illustrate the colour conversion processes of the images obtained from public-outdoor environment and private-indoor environment respectively. It can be observed that the conversion process is successfully implemented where the original colour images are converted into greyscale and binary images.



(a) RGB image 1 **(b)** Greyscale image 1 **(c)** Binary image 1

(d) RGB image 19 **(e)** Greyscale image 19 **(f)** Binary image 19

Figure 7.7: (a),(d) RGB input images converted to (b),(e) greyscale images and (c),(f) binary images for public-outdoor environment.



(a) RGB 1 **(b)** Greyscale 1 **(c)** Binary 1

(d) RGB 195 **(e)** Greyscale 195 **(f)** Binary 195

Figure 7.8: (a),(d) RGB input images converted to (b),(e) greyscale images and (c),(f) binary images for public-indoor environment.

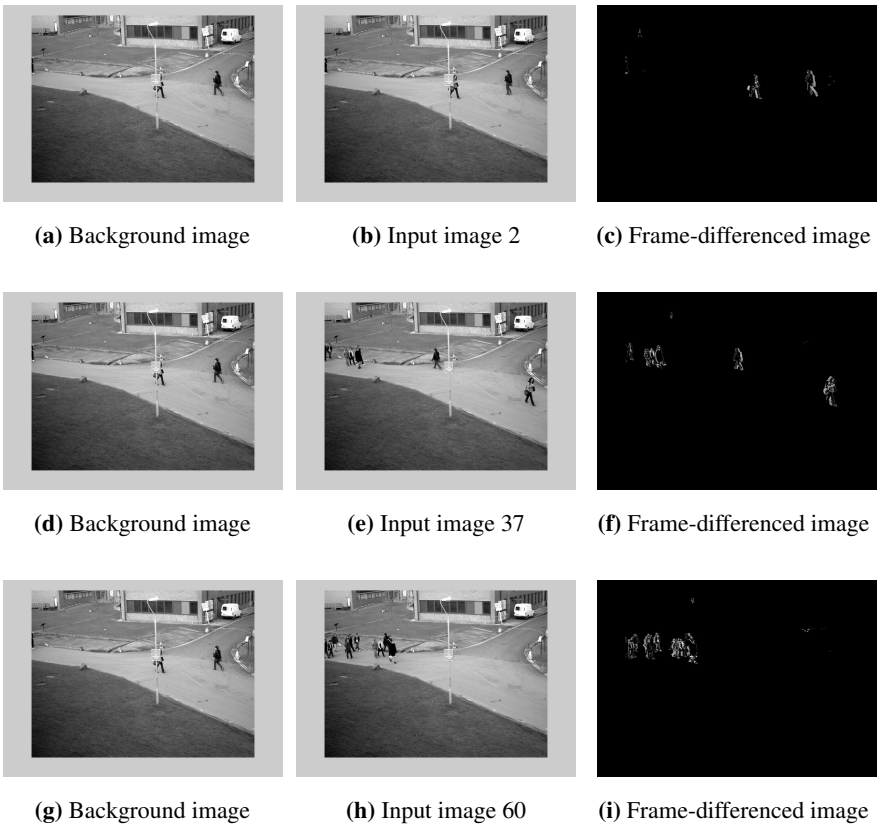
7.5 BACKGROUND SUBTRACTION

The frame differencing method is implemented for the background subtraction process. Two of the four major steps that constitute the background subtraction are addressed and implemented. These steps include the creation of a background reference model and foreground classification.

After the input images are converted to greyscale images, the initial greyscale image is assigned as the background reference model. The reference models for the indoor and outdoor environments are

presented in Fig. 7.9a and 7.11a.

A subtraction process is implemented where each remaining greyscale image is subtracted from the background image and the absolute difference is returned as the output frame-differenced image. Fig. 7.9 illustrates the background subtraction process for the public-outdoor environment using frame differencing method.



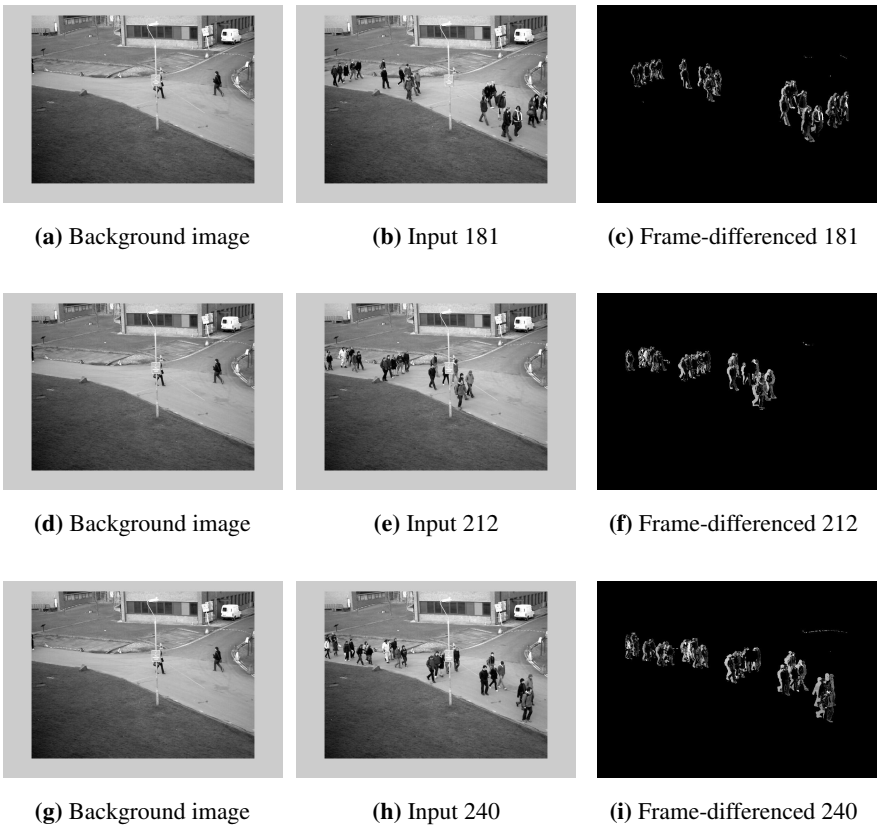
(a) Background image **(b)** Input image 2 **(c)** Frame-differenced image

(d) Background image **(e)** Input image 37 **(f)** Frame-differenced image

(g) Background image **(h)** Input image 60 **(i)** Frame-differenced image

Figure 7.9: (a)-(f) Some of the results obtained after background subtraction method has been implemented on the input image frames of the public-outdoor environment.

It can be observed that each individual is readily visible in the frame-differenced images so it can deduced that the background subtraction process was successfully implemented. Little or no occlusions are observed in the first few frames of the public-outdoor environment. The background subtraction process is implemented on more scene-complex frames and illustrated in Fig. 7.10.



(a) Background image **(b)** Input 181 **(c)** Frame-differenced 181

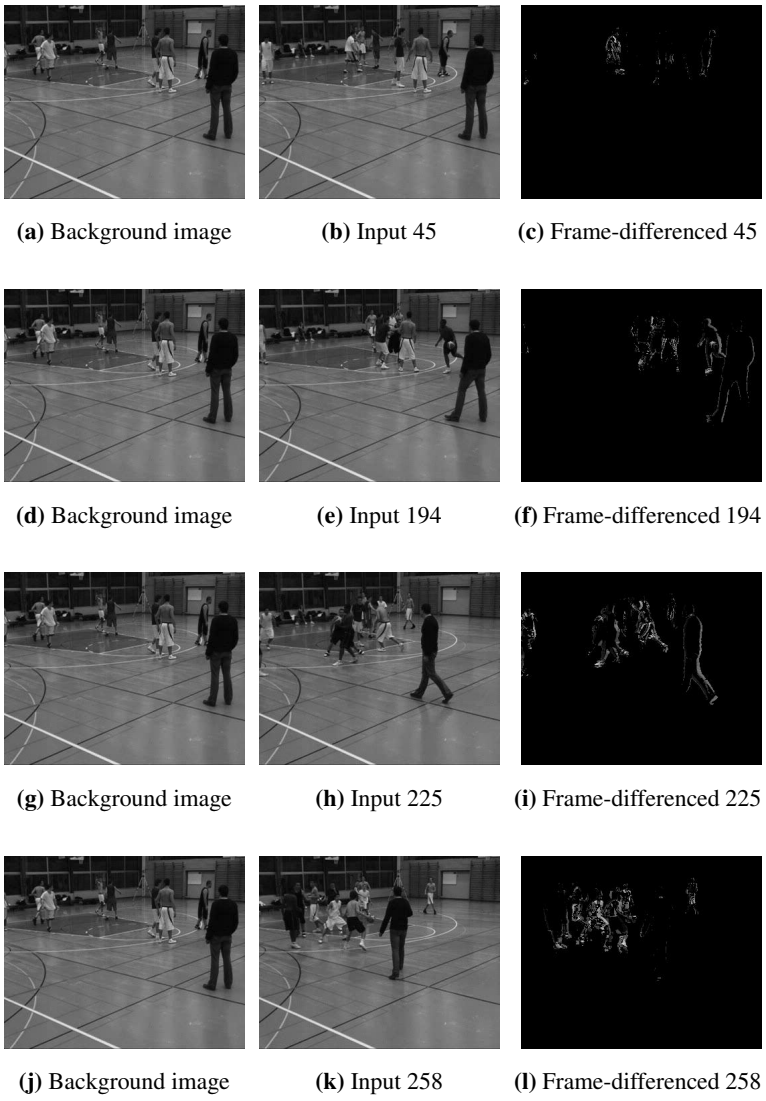
(d) Background image **(e)** Input 212 **(f)** Frame-differenced 212

(g) Background image **(h)** Input 240 **(i)** Frame-differenced 240

Figure 7.10: (a)-(f) Some of the results obtained after background subtraction method has been implemented on the more scene-complex frames of the public-outdoor environment.

It can be observed that the people are heavily occluded which highlights one of the significant problems in people counting. As a result, difficulties could arise when attempting to count each individual in the bounding box.

Fig. 7.11a - 7.11l illustrate the background subtraction process of the private-indoor environment using frame differencing method.



(a) Background image **(b)** Input 45 **(c)** Frame-differenced 45

(d) Background image **(e)** Input 194 **(f)** Frame-differenced 194

(g) Background image **(h)** Input 225 **(i)** Frame-differenced 225

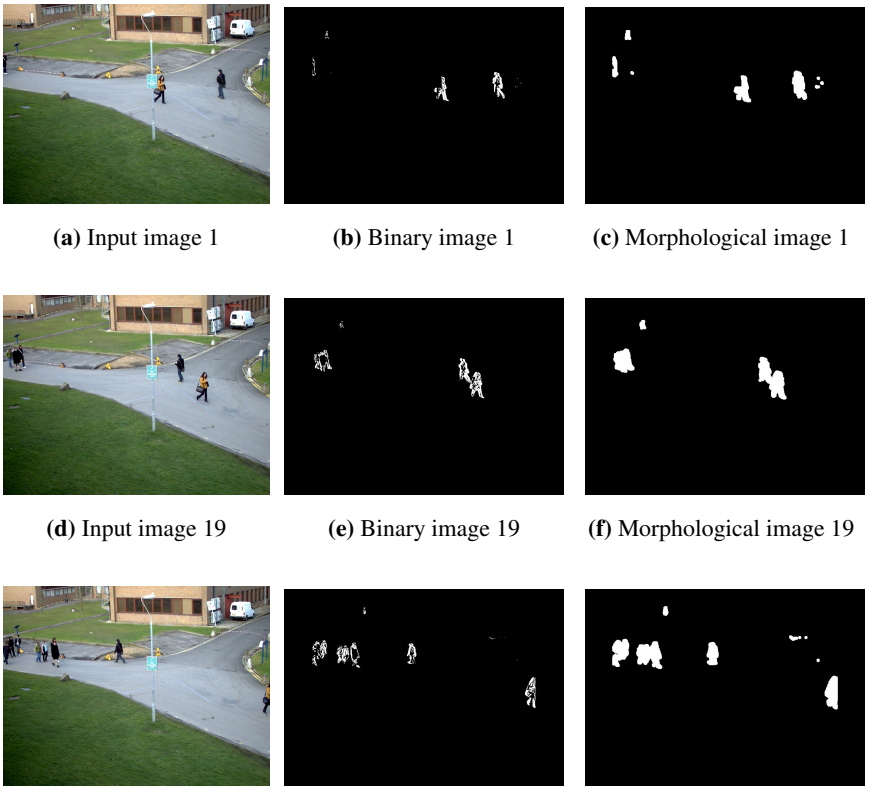
(j) Background image **(k)** Input 258 **(l)** Frame-differenced 258

Figure 7.11: (a)-(l) Some of the results obtained after background subtraction method has been implemented on the input image frames.

Similar results are obtained in the background subtraction process. It can be observed that each individual is clearly visible in the first few frames. But as scene complexities increase, clear visibility and distinction of each individual is not always successfully achieved in the frames.

7.6 MORPHOLOGICAL IMAGE PROCESSING

Fig. 7.12 illustrate the morphological process of the public-outdoor environment.



(a) Input image 1 **(b)** Binary image 1 **(c)** Morphological image 1

(d) Input image 19 **(e)** Binary image 19 **(f)** Morphological image 19

(g) Input image 42 **(h)** Binary image 42 **(i)** Morphological image 42

Figure 7.12: (a)-(i) Some of the results obtained after morphological image processing has been implemented on the input image frames.

It can be observed that the foreground pixels of the queried individuals are highlighted and clearly visible and some of the unnecessary pixels are removed from the images. Also, all the broken gaps visible in the blobs are filled with foreground pixels.

Fig. 7.13 illustrate the morphological process of the private-indoor environment.

(a) Input 1 **(b)** Binary 1 **(c)** Morphological 1

(d) Input 45 **(e)** Binary 45 **(f)** Morphological 45

(g) Input 225 **(h)** Binary 225 **(i)** Morphological 225

(j) Input 606 **(k)** Binary 606 **(l)** Morphological 606

Figure 7.13: (a)-(l) Some of the results obtained after morphological image processing has been implemented on the input image frames.

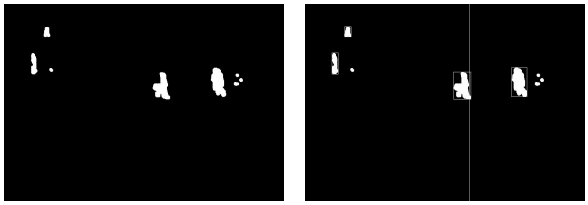
Similarly, it can be observed that the foreground pixels of the queried individuals are highlighted and clearly visible and some of the unnecessary pixels are removed from the images. In Fig. 7.13c and 7.13f, it can be observed that some of the broken gaps that are visible in the blobs are not completely filled with foreground pixels. This issue is only visible in few frames, otherwise the broken gaps are completely filled with foreground pixels as illustrated in Fig. 7.13i and 7.13l.

7.7 SINGLE-PIXEL METHOD

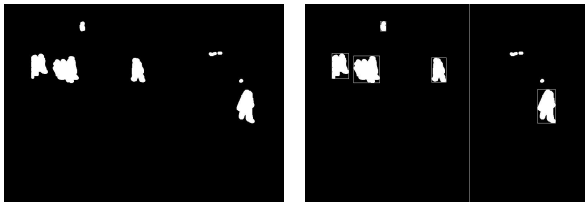
7.7.1 Bounding Box

Each person in the scene has to be bound with a bounding box so that the total number of people can be easily counted over the frame-to-frame analysis. The morphologically-processed images are regarded as input images so they are fed into the system and returned as bounded images where the people in the scene are bounded in individuals or groups.

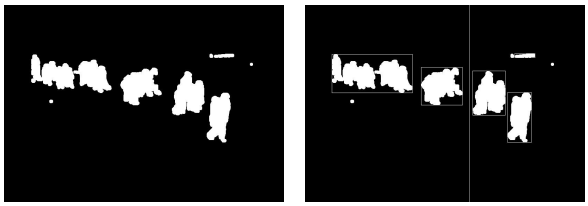
Additionally, a virtual line which represents the entrance point of the scene (horizontal or vertical line) can be created. The creation of the virtual line serves as a key step in the implementation of virtual-line direction-estimation method. For public-outdoor environment, vertical virtual-line is created. Some of the output images obtained after objects are bounded, are presented in Fig. 7.14a-7.14f.



(a) Morphological image 1 **(b)** Bounded image 1



(c) Morphological image 40 **(d)** Bounded image 40



(e) Morphological image 235 **(f)** Bounded image 235

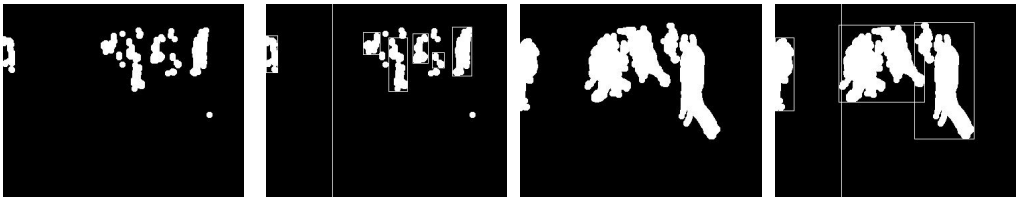
Figure 7.14: (a),(c),(e) Morphologically-processed images. (b),(d),(f) Output images where the people in the scene are bounded with bounding boxes (in individuals or groups).

The algorithm was implemented on 240 frames. It can be observed that the people in each scene over the frame-to-frame analysis are successfully detected and bounded with bounding boxes. In the first few frames, each bounding box contained one individual. But as the scene complexity increases, most of the bounding boxes contained two or more people.

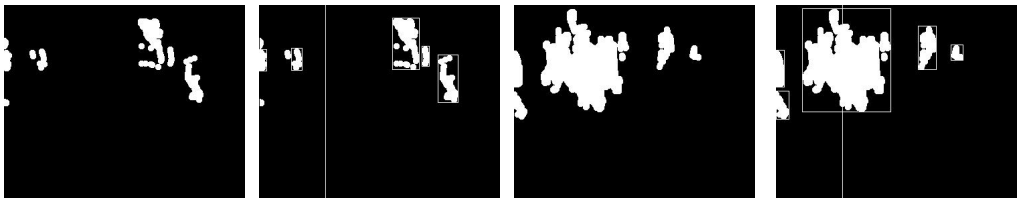
For private-indoor environment, vertical virtual-line is also created. The morphologically-processed images are fed into the single-pixel method in order to obtain bounded blobs. Initially, it was difficult to bound the blobs whose foreground pixels were included in the image borders. So morphological border cleaning was performed on the morphological-processed images before the blobs were bounded. This is the process where all foreground pixels on the image borders are converted to background pixels.

Some of the resulting output images are presented in Fig. 7.15a-7.15f. These images are obtained after the implementation of blob-shape analysis, bounding box and border cleaning processes in the single-pixel method.

(a) Morphological 1 **(b)** Bounded 1 **(c)** Morphological 45 **(d)** Bounded 45



(e) Morphological 150 **(f)** Bounded 150 **(g)** Morphological 225 **(h)** Bounded 225



(i) Morphological 362 **(j)** Bounded 362 **(k)** Morphological 606 **(l)** Bounded 606

Figure 7.15: (a),(c),(e),(g),(i),(k) Morphologically-processed images. (b),(d),(f),(h),(j),(l) Output images where the people in the scene are bounded with bounding boxes (in individuals or groups).

Most of the blobs in the scenes are successfully bounded with bounding boxes. The cases where the blobs are not bounded are observed because the broken gaps of the blobs were not successfully repaired in the morphological image processing step where dilation and hole-filling processes were implemented. The dimensions (upper-left, upper-right, lower-left and lower-right *xy* coordinates) of each bounding box are collected and stored in a *csv* file.

7.7.2 Median area value computation

The next step entails the computation of MAV. The process is illustrated as follows. Suppose bounded blobs are labelled as blob A, blob B, and so on. Furthermore, suppose all the blobs are observed in all frames. The area of blob A is calculated in each frame and the process is repeated for all frames. Then the median value of blob A, across all frames, is returned. This process is repeated for blob B and so on, until all the median values of all blobs have been calculated across all frames. In the public-outdoor environment, 240 frames are presented in the test data. Table 7.1 illustrates some of the median area values obtained for the public-outdoor environment.

	Median area values (MAV) for public-outdoor environment	
	Frame Number	**MAV**
	1	789.5
	40	1578.5
	235	4645.5

Table 7.1: Median area values obtained for some of the scenes of the frame-to-frame analysis of the input data.

Similarly, table 7.2 illustrates some of the median area values obtained for the private-indoor environment.

7.8 VIRTUAL-LINE DIRECTION-ESTIMATION METHOD

The directions in which the people are heading, are obtained using the virtual-line direction-estimation method. In each scene over the frame-to-frame analysis, the *xy* center coordinates of each individual are collected and stored in the single-pixel method. The coordinates obtained are compared with the *xy* center coordinates of the virtual-line. In the public-outdoor environment, ver-

	Median area values (MAV) for private-indoor environment	
	Frame Number	**MAV**
	1	736.5
	45	381.94
	150	411
	225	5362
	362	444
	606	2262

Table 7.2: Median area values obtained for some of the scenes of the frame-to-frame analysis of the input data.

tical virtual-line is created as the entrance point of the scene. If an individual is heading towards the entrance point, the direction $DE_{towards}$ of the individual is represented in eq. 7.1 as:

$$DE_{towards} = \left\{ \begin{array}{ll} 1 & \text{if } x_c^p - x_1 < 0 \\ 0 & \text{if } x_c^p - x_1 > 0 \end{array} \right\} \tag{7.1}$$

Similarly, if the individual is heading away from the entrance point, the direction DE_{away} of the individual is represented in eq. 7.2 as:

$$DE_{away} = \left\{ \begin{array}{ll} 1 & \text{if } x_c^p - x_1 > 0 \\ 0 & \text{if } x_c^p - x_1 < 0 \end{array} \right\} \tag{7.2}$$

where:

- $DE_{towards}$ denotes the direction estimation process for people heading towards the entrance point,
- DE_{away} denotes the direction estimation process for people heading away from the entrance point and
- $x_c^p = \{x_c^1, x_c^2, ..., x_c^{width}\}$ denotes the x-centroid coordinates of the single pixels.

In the public-indoor environment, horizontal virtual-line is created as the entrance point of the scene. If an individual is heading towards the entrance point, the direction $DE_{towards}$ of the individual is represented in eq. 7.3 as:

$$DE_{towards} = \left\{ \begin{array}{ll} 1 & \text{if } y_c^q - y_1 < 0 \\ 0 & \text{if } y_c^q - y_1 > 0 \end{array} \right\} \tag{7.3}$$

Similarly, if the individual is heading away from the entrance point, the direction DE_{away} of the individual is represented in eq. 7.4 as:

$$DE_{away} = \left\{ \begin{array}{ll} 1 & \text{if } y_c^q - y_1 > 0 \\ 0 & \text{if } y_c^q - y_1 < 0 \end{array} \right\} \tag{7.4}$$

where:

- $DE_{towards}$ denotes the direction estimation process for people heading towards the entrance point,
- DE_{away} denotes the direction estimation process for people heading away from the entrance point and
- $y_c^q = \{y_c^1, q_c^2, ..., y_c^{height}\}$ denotes the y-centroid coordinates of the single pixels.

7.9 COUNTING

The counting process is the final step in the implementation of people counting algorithm. It consists of counting and recording the number of people in each scene. It is implemented simultaneously with the virtual-line direction-estimation method where the direction of each individual is estimated for all the provided scenes in the frame-to-frame analysis; with the median area value computation process where the area pixel values are computed for all the blobs in each scene over the frame-to-frame analysis; and with distance computation where total distance of each blob is calculated from the location of each blob to the entrance point of the scene. The computational process is described as follows.

The *xy*-centroid coordinates of all the blobs in each scene, over the frame-to-frame analysis, are obtained in the single-pixel method and stored for later use. The area value $\mathbf{A}_{i,t}$ of each blob and the average area value **MAV** of all blobs are calculated in the median area value computation step for each scene, over the frame-to-frame analysis. These values are stored in excel spreadsheets for later use or observations.

The average area value is set as a deciding factor so that only the blobs whose area values fit between the range of the average area value are used. The blobs are regarded as targets. The blobs whose area values do not fit, are regarded as non-targets so they are discarded and disregarded in the implementation process. Also, a pre-defined threshold value is set to differentiate between short-distanced and long-distanced blobs.

Public-outdoor environment

For public-outdoor environment, the distance between the targeted blob and vertical virtual-line is computed by subtracting the *x*-centroid coordinate of the blob from the *x*-coordinate of the virtual-line. The output distance value is grouped into short-distanced or long-distanced blobs using the pre-defined threshold value. In this case, the pre-defined threshold value is set to 200. Therefore, all the distance values that are less than 200 are classified as short-distanced blobs. Otherwise, the distance values are classified as long-distanced blobs.

The average area values for short- and long-distanced blobs are calculated. In this case, the value obtained for the average area value for short-distanced blobs is 760, while the average area value obtained for long-distance blobs is 2210. The number of people that are bounded in the short-distanced blobs are estimated by dividing the area value of each blob with the short-distanced average area value. Similarly, the number of people that are bounded in the long-distanced blobs are estimated by dividing the area value of each blob with the long-distanced average area value.

Private-indoor environment

Similarly for private-indoor environment, the distance between the targeted blob and horizontal virtual-line is computed by subtracting the *x*-centroid coordinate of the blob from the *y*-coordinate of the virtual-line. The output distance value is grouped into short-distanced or long-distanced blobs using the pre-defined threshold value. In this case, the pre-defined threshold value is set to 200. Therefore, all the distance values that are less than 200 are classified as short-distanced blobs. Otherwise, the distance values are classified as long-distanced blobs.

The average area values for short- and long-distanced blobs are calculated. The value obtained for the average area value for short-distanced blobs is 220, while the average area value obtained for long-distanced blobs is 900. The number of people that are bounded in the short-distanced blobs are estimated by dividing the area value of each blob with the short-distanced average area value. Similarly, the number of people that are bounded in the long-distanced blobs are estimated by dividing the area value of each blob with the long-distanced average area value.

Fig. 7.16 provide some of the results obtained for public-outdoor and private-indoor environments when the counting process is implemented.

(a) Public Outdoor: Input Frame 226

(b) Public Outdoor: Output Frame 226

(c) Private Indoor:Input Frame 602

(d) Private Indoor: Output Frame 602

Figure 7.16: Results obtained for public-outdoor and private-indoor environments when counting process is implemented for people counting.

7.10 PERFORMANCE EVALUATION

The performance of the people counting technique is evaluated by comparing the ground truth (actual number of people per scene) to the estimated number of people using mean square error (MSE) computation [76]. MSE is quantified through the process where global difference is computed between input and output images, then an error percentage of the difference is returned. This is defined in eq. 7.5 as:

$$MSE = \frac{1}{N}\sum_{t=1}^{N}\left[\hat{\mathbf{C}}(t) - \mathbf{C}(t)\right]^2 \tag{7.5}$$

where:

- N is the total number of scenes over the frame-to-frame analysis,
- $\mathbf{C}(s)$ is the actual number of people per scene t and
- $\hat{\mathbf{C}}(t)$ is the output estimated output for the number of people per scene t.

Similarly, the root-mean square (RMS) can be computed. It is defined in eq. 7.6 as:

$$RMS = \sqrt{\frac{1}{N}\sum_{t=1}^{N}\left[\hat{\mathbf{C}}(t) - \mathbf{C}(t)\right]^2} \tag{7.6}$$

Depending on the dataset used for implementation, the ground truths may be provided. Otherwise, the ground truths can be obtained using an alternative process where the number of people are manually counted in each scene over the frame-to-frame analysis. The total number of people is collected and stored in a *csv* file for each scene.

Table 7.5 illustrates the results obtained when the number of people in each frame are summed up. The ground truth total of people is provided, as well as the estimated total number of people.

	Performance Accuracy for People Counting Process			
	Scenario	**No. of Frames**	**Ground truth total**	**Estimated total**
	Public Environment - PETS dataset	240	3935	3998
	Private Environment - EPFL dataset	160	1608	2114

Table 7.3: Total number of people in all frames.

Table 7.4 illustrates the results of the performance evaluation from the computation of the the root-mean square error.

	Performance Accuracy for People Counting Process		
	Scenario	**No. of Frames**	**RMS Error**
	Public Environment - PETS dataset	240	0.1581
	Private Environment - EPFL dataset	160	0.5153

Table 7.4: Performance results of RMS.

Results for public-outdoor environment

The number of people in each scene are manually counted. Then the total number of people in each scene are added to obtain the ground truth. A value of 3935 is obtained. Using the single-pixel method, the number of people in each scene are estimated. Similarly, the total number of people in each scene are added to obtain the estimated output of 3998 (including false positives) where over-counting is experienced. Over-counting is the process where the value of the estimated number of people is higher than the original number. In this case, a value of 62 was observed for over-counting, achieved by computing the difference between the ground truth and the estimated number.

The original number of people in each scene is compared to the estimated number. The results are plotted in a graph as illustrated in Fig. 7.17. It can be observed that the estimated number of people are similar to the ground truth.

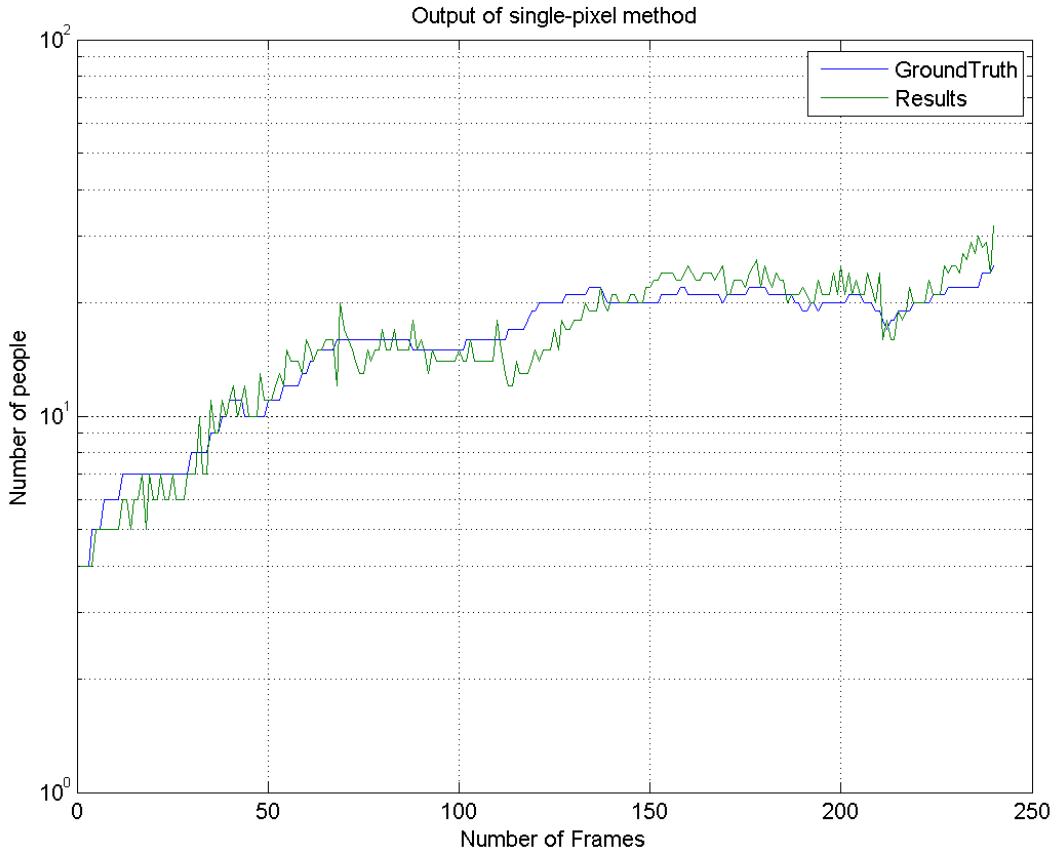


Figure 7.17: Plotted results of the comparison of original and estimated counting of people in a public environment using single pixel method.

Results for private-indoor environment

The same process of the public environment was repeated for the private environment, where the people were manually added in each frame in order to obtain the ground truth. A value of 1608 was obtained. The single-pixel method was tested on the private-indoor environment by estimating the number of people in each scene. A total of 2114 was obtained (including false positives) Over-counting was highly experienced in the private-indoor environment where the estimated number of people is higher than the original number of people. Over-counting was greatly due to the transition process between limited movements and regular movements of the same people in the scenes.

The transition process is where little or no movement is observed from a person in the first frame; then, regular movement is observed in the second frame. This produces significantly higher pixel count. As a result, over-counting is experienced which affects the accuracy of the counting process

of the single-pixel method.

Although a significantly low RMS value was obtained, the performance accuracy of the single-pixel method was affected by over-counting in the private-indoor environment. Fig. 7.18 illustrates the comparison between the original and estimated number of people in each scene in the private-indoor environment.

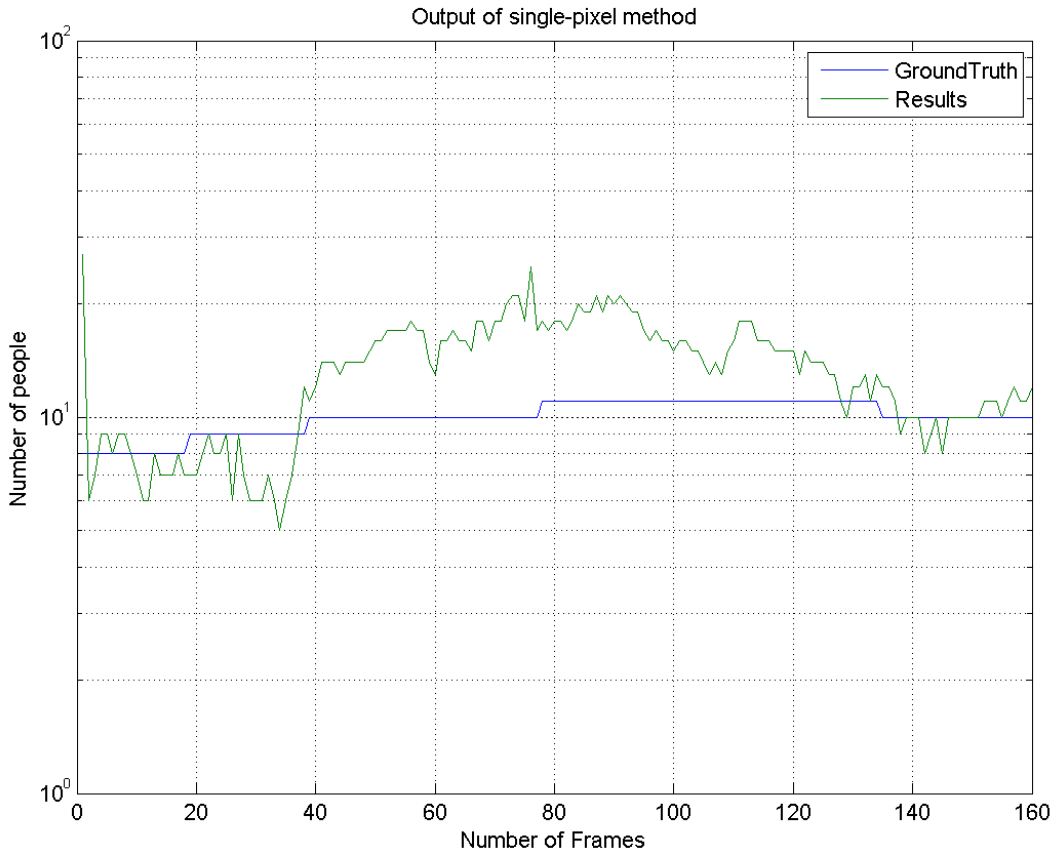
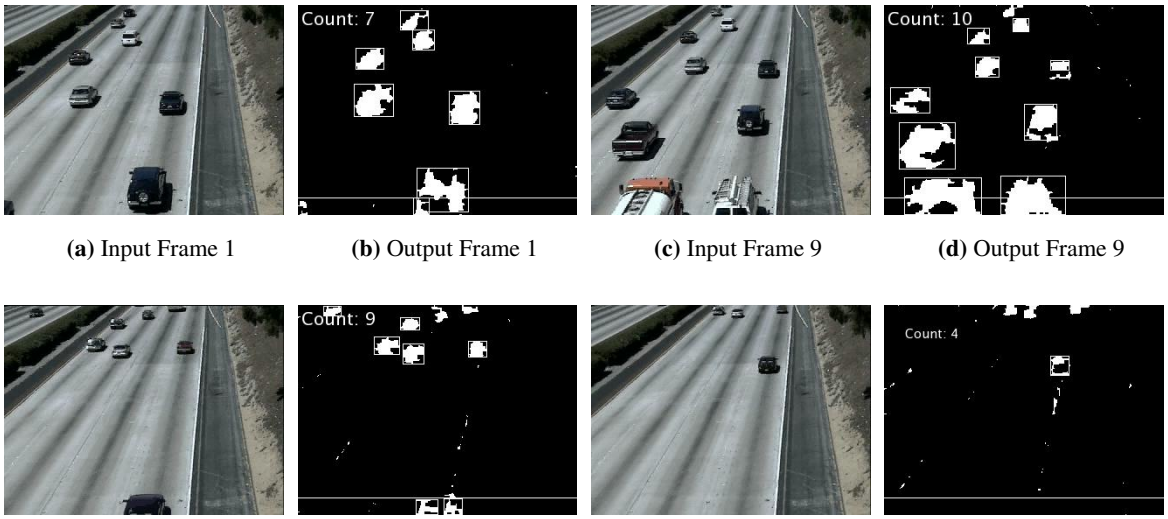


Figure 7.18: Plotted results of the comparison of original and estimated counting of people in a private environment using single pixel method.

7.11 VEHICLE COUNTING

Vehicle counting in traffic surveillance has attracted a lot of attention in intelligent transportation systems over the past years. Coupled with vision-based techniques, it plays a critical role in automated systems. In order to test for the robustness of the proposed approach, the algorithm was further tested on vehicle counting datasets with varying environmental conditions such as lighting, traffic flow, shadowing, daytime and night-time traffic surveillances. Similarly to people counting, vehicle counting is the process where vehicles in a specified area are counted and recorded. Fig. 7.19 - 7.21 present results of the proposed algorithm on the provided datasets.

Fig. 7.19a - 7.19h illustrate the results of the proposed algorithm for daytime traffic surveillance scene where low light and traffic flow are observed. The surveillance scene is captured from a highway.



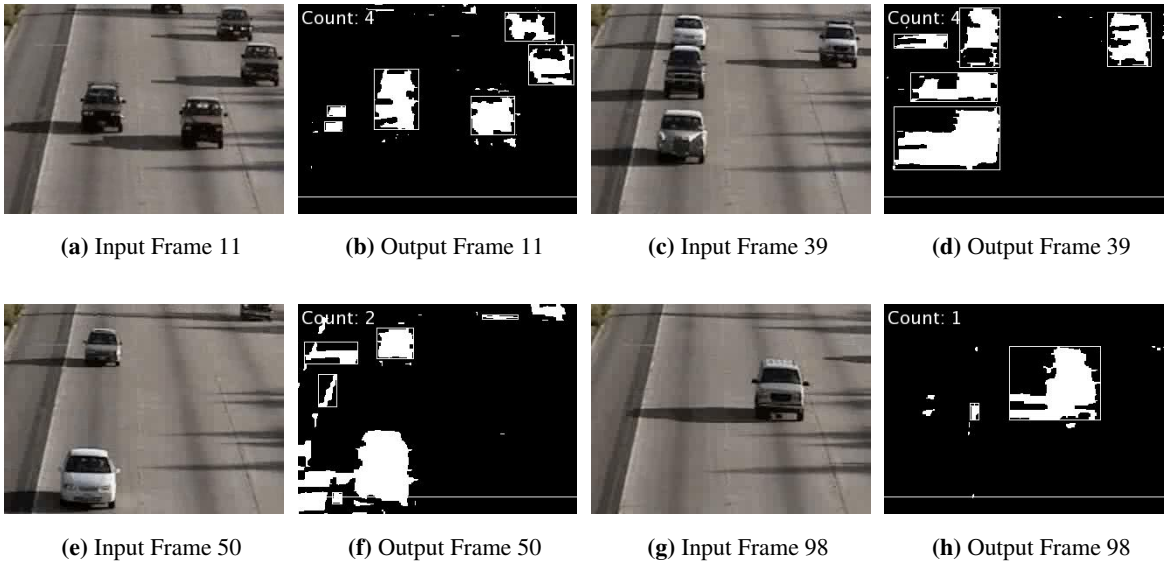
(a) Input Frame 1 **(b)** Output Frame 1 **(c)** Input Frame 9 **(d)** Output Frame 9

(e) Input Frame 73 **(f)** Output Frame 73 **(g)** Input Frame 95 **(h)** Output Frame 95

Figure 7.19: Results obtained after the implementation of the proposed algorithm on a highway traffic scene with low traffic flow.

Fig. 7.20a - 7.20h illustrate the results of the proposed algorithm for daytime traffic surveillance scene where heavy shadowing is observed due to the bright lighting conditions.

Fig. 7.21a - 7.21h illustrate the results of the proposed algorithm for vehicles captured at night. Also, it can be observed the vehicles are inside a tunnel.



(a) Input Frame 11 **(b)** Output Frame 11 **(c)** Input Frame 39 **(d)** Output Frame 39

(e) Input Frame 50 **(f)** Output Frame 50 **(g)** Input Frame 98 **(h)** Output Frame 98

Figure 7.20: Results obtained after proposed algorithm was implemented on daytime traffic surveillance under bright illuminations conditions where heavy shadowing is observed.

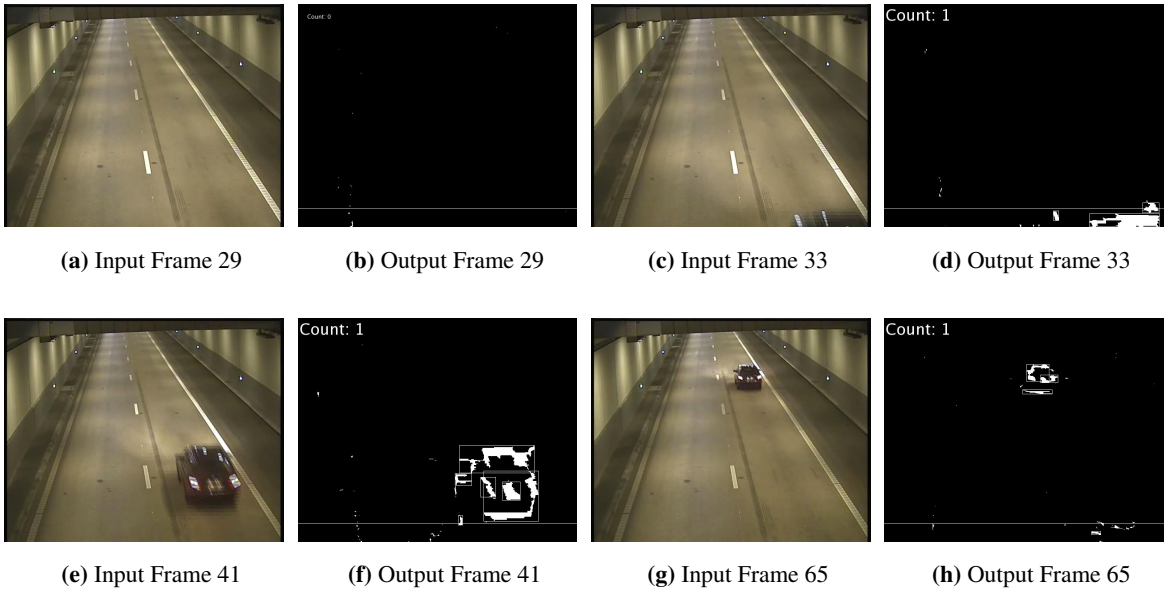
The vehicle counting process was evaluated using Jaccard coefficient [77] by comparing the ground truth with the estimated results. Jaccard coefficient is the process where comparison is implemented using true positives (TP), false positives (FP) and false negatives (FN). The process is defined in eq. 7.7 below.

$$J = \frac{TP}{TP+FP+FN} \tag{7.7}$$

where:

- True positives (TP) represent the number of vehicles that are correctly estimated,
- False positives (FP) represent the number of vehicles that are incorrectly estimated as other objects, and
- False negatives (FN) represent the number of objects that are incorrectly estimated as vehicles.

Tables 7.5 - 7.7 provide the performance results of vehicle counting using the proposed algorithm. Table 7.5 presents the Jaccard coefficient score for daytime traffic surveillance where low light and traffic flow are observed.



(a) Input Frame 29 **(b)** Output Frame 29 **(c)** Input Frame 33 **(d)** Output Frame 33

(e) Input Frame 41 **(f)** Output Frame 41 **(g)** Input Frame 65 **(h)** Output Frame 65

Figure 7.21: Results obtained after proposed algorithm was implemented on a night-time highway tunnel scene.

No. of Frames	100
Actual no. of Vehicles	824
Estimated no. of Vehicles	708
Counting Score	85.9%

Table 7.5: Performance results of vehicle counting using daytime highway traffic surveillance with low light and traffic flow.

Table 7.6 presents the Jaccard coefficient score for daytime traffic surveillance where bright lighting conditions and heavy shadowing are observed.

No. of Frames	100
Actual no. of Vehicles	349
Estimated no. of Vehicles	269
Counting Score	77.1%

Table 7.6: Performance results of vehicle counting process using daytime highway traffic surveillance with long, heavy shadows.

Table 7.7 presents the Jaccard coefficient score for night-time traffic surveillance. Vehicle headlights are present but no street lights are observed.

No. of Frames/ Time span	63
Actual no. of Vehicles	34
Estimated no. of Vehicles	38
Counting Score	89.5%

Table 7.7: Performance results of vehicle counting process using night-time highway tunnel surveillance with vehicle headlights but no street lights.

The robustness of the proposed algorithm is successfully achieved because significantly good results are observed when the algorithm is tested on different objects (people or vehicle) with varying environmental and weather conditions.

CHAPTER 8 CONCLUSION

"There is one thing even more vital to
science than intelligent methods;
and that is, the sincere desire to find out
the truth, whatever it may be."

Charles Pierce

The scope of the research was to address (and possibly solve) key problems that people counting algorithms are faced with. As a result, single-pixel and virtual-line direction-estimation methods were created in order to address and solve the challenging problems that people counting algorithms are currently faced with.

Four primary objectives were defined for this study: 1) the implementation of robust people counting algorithm that is not application specific; 2) that is invariant to occlusions, illumination conditions, human pose and direction; 3) that provides direction estimation of the people; 4) with low complexity and high performance accuracy. Therefore, this chapter discusses the extent to which these primary objectives have been satisfied, the results that have been achieved, the brief explanation on why the algorithm is not compared with existing counting algorithms and future work.

8.1 DISCUSSION 1: EXTENT TO WHICH PRIMARY OBJECTIVES HAVE BEEN SATISFIED

This section discusses the extent to which the four primary objectives have been satisfied in the study.

8.1.1 Development of people counting algorithm that is not application specific

Two test scenarios, namely, public-outdoor and private-indoor environments, were presented in *chapter 7.2* as input data for the implementation and testing of the single-pixel method. These environments accurately capture challenges (occlusions, lighting conditions, etc.) that people counting algorithms are faced with. For example, the lighting conditions in indoor scenarios might vary from outdoor scenarios so it was necessary to test for such conditions in order to ensure that the accuracy of the algorithm is not application specific.

As observed in the results in *chapter 7.9*, all the people in the scenes of private-indoor and public-outdoor environments were successfully counted. However, over-counting was commonly experienced in the private-indoor environment when limited or no movement was observed from the people in the scenes.

8.1.2 Development of people counting algorithm that is invariant to occlusions, illumination conditions, human pose and direction

The single pixel approach was developed as an extra step in the background subtraction method in order to produce a people counting algorithm that is invariant to occlusions, illumination conditions, human pose and direction. The steps followed in the single pixel process were presented in *chapter 6.6*.

As illustrated in the results in *chapter 7.7*, the steps in the single pixel process were not affected by the challenging conditions. All the useful features in the scenes from the private-indoor and public-outdoor environments were successfully extracted, morphologically processed, bounded into bounding boxes and counted. As a result, the research question posed from this objective was accurately addressed and satisfied.

8.1.3 Development of people counting algorithm that provides direction estimation of the people

A virtual-line direction-estimation method was developed in order to produce a people counting algorithm that provides direction estimation of the people. As presented in *chapter 6.8*, a vertical or horizontal virtual line was drawn as the entrance point of the scene. All the people in the scenes

from the private-indoor and public-outdoor environments were observed in order to determine the directions in which they were headed.

As illustrated in the results in *chapter 7.8*, the directions in which the people were headed were accurately determined. As a result, the research question posed from this objective was accurately addressed and satisfied.

8.1.4 Development of people counting algorithm with low complexity and high performance accuracy

The single-pixel and virtual-line direction-estimation methods were designed with low complexities. Also, high accuracies were observed in the performance evaluation presented in *chapter 7.10*.

8.2 DISCUSSION 2: CONCLUSION OF THE ACHIEVED RESULTS

This study was involved with the combination of various image processing techniques, creation of a virtual-line direction-estimation method, as well as creation of an extra step in background subtraction method known as single-pixel approach, in order to provide a fast and efficient people counting algorithm that is invariant to the significant restrictions and challenges whilst monitoring the trade-off between the complexity, speed and performance accuracy of the algorithm.

As observed and illustrated in *chapter 7*, satisfactory results were obtained from the performance evaluation of the people counting algorithm. However, over-counting was experienced in cases where limited or no movement was observed from the people in the scenes and other factors such as heavy occlusions, reduced the accuracy of the system. Also, the proposed algorithm was tested on vehicle counting where satisfactory results were obtained.

Finally, it can be concluded that the design and implementation of a fast and efficient people counting algorithm is possible using single-pixel and virtual-line direction-estimation methods because the performance the system was very satisfying and all of the desired objectives were successfully met.

8.3 DISCUSSION 3: COMPARISON TO EXISTING COUNTING METHODS

It should be noted that the proposed algorithm for people and vehicle counting is not comparatively evaluated with existing counting algorithms. This is primarily because the proposed algorithm is headed in a new direction thus difficult to compare with other algorithms. Also, the proposed algorithm is primarily concerned with robustness and invariance so that it can be implemented on different processes with varying environmental conditions; and invariant to all (or most) significant restrictions.

8.4 DISCUSSION 4: FUTURE RESEARCH

Single-pixel method was successfully developed and implemented but there is always room for improvement. Some aspects of the method can be optimized or the study can be extended to cover more aspects as listed below:

- Multi-target tracking can be appended to the system where the previous and current location of each person is monitored and recorded. However, the tracking system will increase the complexity. As a result, trade-off between the accuracy and complexity can be addressed. Tracking can be implemented using particle filter.

- Currently, the extracted useful features only contain (pixel intensities) internal characteristics of the images which becomes a challenge when large datasets are provided as input. As an optimization, external characteristics (boundaries) can be included in the features. These characteristics can be obtained from descriptors such as Fourier descriptors, regional descriptors, statistical moments, SURF, and so on.

- Finally, the performance of the system was reduced when limited or no movements were observed from the people in the scenes. Therefore, an approach where people's movements are invariant to the system, can be addressed and implemented in the future.

REFERENCES

[1] Ya-Li Hou, Grantham K. H. Pang, "People Counting and Human Detection in a Challenging Situation," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 41, no. 1, pp. 24-33, Jan. 2011.

[2] T. Zhao and R. Nevatia, "Tracking multiple humans in complex situations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1208-1221, Sep. 2004.

[3] T. Zhao, R. Nevatia, and B. Wu, "Segmentation and tracking of multiple humans in crowded environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 7, pp. 1198-1211, Jul. 2008.

[4] H. Xu, P. Lv and L. Meng, "A people counting system based on head-shoulder detection and tracking in surveillance video," *in International Conference on Computer Design and Applications*, 2010, pp. 394-398.

[5] Q. Ye, "A robust method for counting people in complex indoor spaces", *in 2nd Conference on Education Technology and Computer*, 2010, pp. 450-454.

[6] J. Li, L. Huang and C. Liu, "An efficient self-learning people counting system", *in 1st Asian Conference on Pattern Recognition*, 2011, pp. 125-129.

[7] D. Ryan, S. Denman, C. Fookes and S. Sridharan, "Crowd counting using group tracking and local features", *in 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2010, pp. 218-224.

[8] C. Gao, K. Huang and T. Tan, "People counting using combined feature", *in 3rd Chinese Conference on Intelligent Visual Surveillance*, 2011, pp. 81-84, Dec. 2011.

[9] L. Chen, J. Tao, Y. P. Tan, K. L. Chan, "People counting using iterative mean-shift fitting with symmetry measure", *in 6th International Conference on Information, Communications and Signal Processing*, 2007, pp. 1-4.

[10] J. Li, L. Huang and C. Liu, "Robust people counting in video surveillance: dataset and system", *in 8th IEEE International Conference on Advanced Video and Signal-Based Surveillance*, 2011, pp. 54-59.

[11] A. G. Vincente, I. B. Munoz, P. J. Molina and J. L. L. Galilea, "Embedded vision modules for tracking and counting people", *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 9, pp. 3004-3011, Sept. 2009.

[12] N. Li, J. Song, R. Y. Zhou and J. H. Gu, "A people counting system based on BP Neural Network", *in 4th International Conference on Fuzzy Systems and Knowledge Discovery*, 2007, pp. 283-287.

[13] J. Xing, H. Ai, L. Liu and S. Lao, "Robust crowd counting using direction flow", *in 8th International Conference on Image Processing*, 2011, pp. 2061-2064.

[14] Y. Cong, H. Gong, S. C. Zhu, and Y. Tang, "Flow mosaicking: Real-time pedestrian counting without scene-specific learning", *in IEEE International Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1093-1100.

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

[16] X. Wang, C. Wang and J. Yao,"A Heuristic Information Based System for People Counting ," *in 3rd International Conference on Intelligent Human-Machine Systems and Cybernetics*, 2011, pp. 22-26.

[17] Y. Benabbas, N. Ihaddadene, T. Yahiaoui, T. Urruty and C. Djeraba, "Spatio-Temporal Optical Flow Analysis for People Counting," *in 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2010, pp. 212-217.

[18] H. Ma, H. Lu and M. Zhang, "A Real-time Effective System for Tracking Passing People Using a Single Camera," *in 7th World Congress on Intelligent Control and Automation*, 2008,

pp. 6173-6177.

[19] X. Zhao, E. Delleandrea and L. Chen, "A People Counting System Based on Face Detection and Tracking in a Video," *in 6th IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2009, pp. 67-72.

[20] D. Merad, K. E. Aziz and N. Thome, "Fast People Counting Using Head Detection from Skeleton Graph," *in 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2010, pp. 233-240.

[21] M. Patzold, R. H. Evangelio and T. Sikora, "Counting People in Crowded Environments by Fusion of Shape and Motion Information," *in 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2010, pp. 157-164.

[22] J. Rittscher, P. H. Tu, and N. Krahnstoever, "Simultaneous estimation of segmentation and shape," *in IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005, pp. 486-493.

[23] X. Liu, P. H. Tu, J. Rittscher, A. Perera, and N. Krahnstoever, "Detecting and counting people in surveillance applications", *in IEEE Conference on Advanced Video and Signal Based Surveillance*, 2005, pp. 306-311.

[24] S. Velipasalar, Y. li Tian, and A. Hampapur, "Automatic counting of interacting people by using a single uncalibrated camera", *in IEEE International Conference on Multimedia and Expo*, 2006, pp. 1265-1268.

[25] O. Sidla, Y. Lypetskyy, N. Brandle, and S. Seer, "Pedestrian detection and tracking for counting applications in crowded situations", *in International Conference on Advanced Video and Signal-Based Surveillance*, 2006, pp. 70-75.

[26] D. Lan, V. Parameswaran, V. Ramesh and I. Zoghlami, "Fast Crowd Segmentation Using Shape Indexing", *in 11th IEEE International Conference on Computer Vision*, 2007, pp. 1-8.

[27] A. Albiol, I. Mora, and V. Naranjo, "Real-time high density people counter using morphological tools", *IEEE Transactions on Intelligent Transportation Systems*, vol. 2, no. 4, pp. 204-218, Dec. 2001.

[28] A. C. Davies, J. H. Yin and S. A. Velastin, "Crowd monitoring using image processing", *Electronics and Communication Engineering Journal*, vol. 7, no. 1, pp. 37-47, Feb. 1995.

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

[30] A. K. Jain and F. Farrokhnia, "Unsupervised Texture Segmentation Using Gabor Filters", *in Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 1990, pp. 14-19.

[31] A. Teuner, O. Pichler and B. J. Hosticka, "Unsupervised texture segmentation of images using tuned matched Gabor filters", *IEEE Transactions on Image Processing*, vol. 4, no. 6, pp. 863-870, Jun. 1995.

[32] D. Ryan, S. Denman, C. Fookes, S. Sridharan, "Crowd Counting Using Group Tracking and Local Features," *in 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2010, pp. 218-224.

[33] A. B. Chan, Z. S. J. Liang, and N. Vasconcelos, "Privacy Preserving Crowd Monitoring: Counting People without People Models or Tracking," *in IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1-7.

[34] P. Kilambi, E. Ribnick, A. Joshi, O. Masoud, and N. Papanikolopoulos, "Estimating Pedestrian Counts in Groups", *Computer Vision and Image Understanding*, vol. 110, no. 1, pp. 43-59, Apr. 2008.

[35] R. Cucchiara, C. Grana, M. Piccardi and A. Prati,"Detecting Moving Objects, Ghosts, and Shadows in Video Streams," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1337-1342, Oct. 2003.

[36] S. S. Cheung and C. Kamath, "Robust techniques for background subtraction in urban traffic video," *EURASIP Journal on Applied Signal Processing*, vol. 1, no. 1, pp. 2330-2340, Jan. 2005.

[37] B. Karasulu and S. Korukoglu,"Moving object detection and tracking by using annealed back-

ground subtraction method in videos: Performance optimization," *Expert Systems with Applications*, vol. 39, no. 1, pp. 33-43, Jan. 2012.

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

[39] I. Haritaoglu, D. Harwood and L. Davis,"W^4: Real-Time Surveillance of People and Their Activities," *in IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809-830, Aug. 2000.

[40] B. P. L. Lo and S. A. Velastin, "Automatic Congestion Detection System for Underground Platforms," *in Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing*, 2001, pp. 158-161.

[41] Q. Wu and B. Jeng, "Background Subtraction based on logarithmic intensities," *Pattern Recognition Letters*, vol. 23, no. 13, pp. 1529-1536, Nov. 2002.

[42] C. Wren, A. Azarbayejani, T. Darrell and A. Pentland,"Pfinder: real-time tracking of the human body," *in IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780-785, Jul. 1997.

[43] J. Gallego, M. Pard and G. Haro, "Enhanced foreground segmentation and tracking combining Bayesian background, shadow and foreground modeling," *Pattern Recognition Letters*, vol. 33, no. 12, pp. 1558-1568, Sept. 2012.

[44] Y. Benezeth, P. M. Jodoin, B. Emile, H. Laurent and C. Rosenberger,"Review and evaluation of commonly-implemented background subtraction algorithms," *in 19th International Conference on Pattern Recognition*, 2008, pp. 1-4.

[45] M. Piccardi, "Background subtraction techniques: a review," *in IEEE International Conference on Systems, Man and Cybernetics*, 2004, pp. 3099-3104.

[46] A. Elgammal, D. Harwood and L. Davis, "Non-parametric Model for Background Subtraction," *in Proceedings of the 6th European Conference on Computer Vision-Part II*, 2000, pp. 751-767.

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

[48] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," *in IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1999, pp. 246-252.

[49] X. Li,K. Wang, W. Wang and Y. Li, "A Multiple Object Tracking Method Using Kalman Filter," *in Proceedings of the 2010 IEEE International Conference on Information and Automation*, 2010, pp. 1862-1866.

[50] M. Szczodrak, P. Dalka and A. Czyzewski, "Performance evaluation of video object tracking algorithm in autonomous surveillance system," *in Proceedings of the 2nd International Conference on Information Technology*, 2010, pp. 31-34.

[51] G. Welch and G. Bishop, "An Introduction to Kalman Filter," Course 8, SIGGRAPH, Aug. 2001.

[52] A. Yilmaz, O. Javed and M. Shah, "Object Tracking: A Survey," *ACM Computing Surveys*, vol. 38, no. 4, pp. 1-45, Dec. 2006.

[53] D. Comaniciu, V. Ramesh and P. Meer, "Kernel-based Object Tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 2, pp. 564-577, Feb. 2002.

[54] D. Koller, J. Weber, and J. Malik, "Robust Multiple Car Tracking with Occlusion Reasoning," EECS Department, University of California, Berkeley, Technical Report No. UCB/CSD-93-780, Nov. 1993.

[55] K.-P. Karmann and A. Brandt, "Moving Object Recognition using and Adaptive Background Memory," *in Time-Varying Image Processing and Moving Object Recognition*, vol. 1, no. 2, pp. 289-307, 1990.

[56] N. Funk, "A Study of the Kalman Filter applied to Visual Tracking," University of Alberta, Project for CMPUT 652, Dec. 2003.

[57] B. Karasulu,"Review and evaluation of well-known methods for moving object detection and

tracking in videos," *Journal of Aeronautics and Space Technologies*, vol. 4, no. 4, pp. 11-22, Jul. 2010.

[58] R. Haralick, S.R. Sternberg and X. Zhuang, "Image Analysis Using Mathematical Morphology," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 4, 1987.

[59] R. C. Gonzalez and R. E. Woods, "*Digital Image Processing*," 3rd Edition, Prentice-Hall, 2008.

[60] E. R. Dougherty, "*An Introduction to Morphological Image Processing*," SPIE Optical Engineering Press, 1992.

[61] H. J. A. M. Heijmans, "Mathematical Morphology : A Modern Approach in Image Processing Based on Algebra and Geometry," *SIAM Review*, vol. 37, no. 1, pp. 1-36, 1995.

[62] F. Fukunaga and L. D. Hostetler, "The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition," *IEEE Transactions on Information Theory*, vol. 21, no. 1, pp. 32-40, Jan. 1975.

[63] D. Comaniciu and P. Meer, "Mean Shift : a Robust Approach toward Feature Space Analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603-619, May 2002.

[64] D. Comaniciu, V. Ramesh and P. Meer, "Real-Time Tracking of Non-Rigid Objects using Mean Shift," *in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2002, pp. 142-149.

[65] J. Ning, L. Zhang, D. Zhang and C. Wu, "Scale and Orientation Adaptive Mean Shift Tracking," *IET Computer Vision*, vol. 1, no. 1, 2011.

[66] R. Collins, "Mean-shift Blob Tracking through Scale Space," *Computer Vision and Pattern Recognition*, June 2003.

[67] G. Bradski and A. Kaehler, *Learning OpenCV*, O'Reilly, 2008.

[68] C. Tomasi and T. Kanade, "Detection and Tracking of Point Features," Carnegie Mellon University Technical Report, CMU-CS-91-132, Apr. 1991.

[69] J. Shi and C. Tomasi, "Good Features to Track," *in IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593âĂŞ600.

[70] Y. Bar-Shalom and T. Foreman, *Tracking and Data Association*, Academic Press, 1988.

[71] H. Tanizaki, "Non-gaussian state-space modelling of nonstationary time series," *Journal of America Statistics Association*, vol. 82, no. 1, pp. 1032-1063, 1987.

[72] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier and L. Van Gool, "Robust tracking-by-detection using a detector confidence particle filter," *in Proceedings of the 12th IEEE International Conference on Computer Vision*, 2009, pp. 1515-1522.

[73] N. J. Gordon, D. J. Salmond and A. F. M. Smith, "Novel approach to nonlinear/nonGaussian Bayesian state estimation," *IEEE Proceedings in Radar and Signal Processing*, vol. 140, no. 2, pp. 107-113, Apr. 1993.

[74] PETS 2010. (2009). PETS 2010 Benchmark Data [Online]. Available: http://pets2010.net/

[75] CVLAB - Computer Vision Laboratory. (2012). "EPFL" data set: Multi-camera Pedestrian Videos [Online]. Available: http://cvlab.epfl.ch/data/pom/

[76] R. Dosselmann and X. D. Yang, "Existing and Emerging Image Quality Metrics," *Canadian Conference on Electrical and Computer Engineering*, 2005, pp. 1906-1913.

[77] R. L. Rosin and E. Ioannidis, "Evaluation of global image thresholding for change detection," *Pattern Recognition Letters*, vol. 24, no. 14, pp. 2345-2356, Oct. 2003.