Technical University of Crete
Department of Electronics Engineering & Computer Engineering

Region-Based object Detection and Tracking

by

Panagiotis Paizakis
June 2009

Accepted on the recommendation of

Prof. Dr. Michalis Zervakis, Thesis Advisor

Prof. Dr. Nikolaos Sidiropoulos, Co-Examiner

Assoc. Prof. Dr. Euripides Petrakis, Co-Examiner

Abstract

Identifying moving objects is a critical task for many computer vision applications. It provides a classification of the pixel into either foreground or background. In this thesis, an algorithm is presented for segmentation of moving objects in image sequences. For each frame in the video sequence, an initial segmentation is performed. The result of this segmentation is a set of regions that completely cover the image. Then, each region is examined and classified either as moving object or as background. Thus, the problem of moving objects segmentation is transformed into a region classification. Each region in the initial partition must be either a part of moving object or part of the background. This classification rely on temporal information or on intensities differences between successive frames or rely on motion. A common approach used to achieve such classification is background removal. Two approaches for moving object detection have been implemented in this thesis, the Gaussian Mixture Model (GMM), a statistical model, and LOTS (Lehigh Omnidirectional Tracking System) algorithm, a region-based algorithm, too.

Temporal stability of the segmentation algorithm is established by incorporating a dynamic memory based on object tracking. An object tracker establishes the temporal correspondence of objects throughout the video sequence. The memory allows us to utilize the interdependencies between adjacent frames is a sequence, in order to increase the robustness of the segmentation algorithm.

Acknowledgments

This work has been carried out at the Computer Vision Laboratory at Technical University of Crete. I would like to thank everybody working there for their hospitality and the friendly atmosphere.

I would especially like to thank my supervisor Michalis Zervakis for introducing me to the field of computer vision and constantly discussing ideas and being extremely helpful and approachable.

I would also like to thank all my friends for supporting this work.

Paizakis Panagiotis 2009

Contents

3 Methodological Issues

4 Implementation Issues

5 Conclusions and future work

Bibliography

Chapter 1: Introduction

In this chapter we present the background and motivation of the thesis. The objectives and the structure of the thesis are also outlined.

1.1 Motivation

Understanding the motion of objects moving in a scene by the use of video is both a challenging scientific problem and a very fertile domain with many promising applications. Thus, it draws attention of several researches, and commercial companies. Our motivation is the study and the implementation of moving object detection methods.

Moving Object detection is the basic step for further analysis of video. It handles segmentation of moving object from stationary background objects. This not only creates a focus of attention for higher level processing but also decreases computation time considerably. Commonly used techniques for object detection are background subtraction and also statistical models. Due to environmental conditions such as illumination changes, shadows and waving tree branches in the wind object segmentation is a difficult and significant problem that needs to be handled well for a robust visual surveillance system. In our work we chose one algorithm for object detection with background subtraction (LOTS) and one statistical model (Gaussian Mixture Model).

The next step in the video analysis is tracking, which can be simply defined as the creation of temporal correspondence among detected objects

from frame to frame. This procedure provides temporal identification of the segmented regions and generates cohesive information about the objects in the monitored area such as trajectory, speed and direction. The output produced by tracking step is generally used for higher level activity analysis.

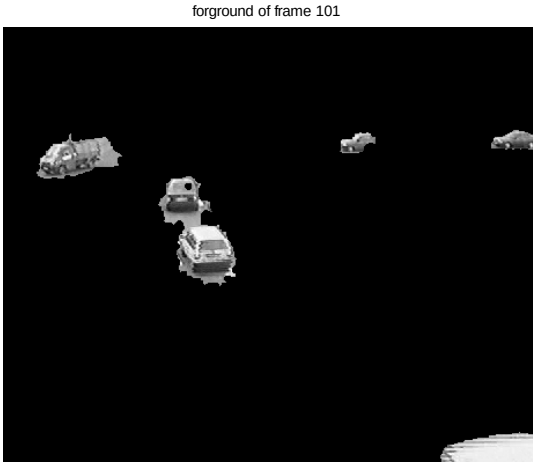


Figure 1.1 A binary background/foreground image.

1.2 Related Work on real time object detection and tracking

Background subtraction is particularly a commonly used technique for motion segmentation in static scenes. It attempts to detect moving regions by subtracting the current image pixel-by-pixel from a reference background image that is created by averaging images over time. The pixels where the difference is above a threshold are classified as foreground. The reference background is updated with new images over time to adapt to dynamic scene changes.

In [77] *Heikkila and Silven* a pixel at location (x, y) in the current image I_t is marked as foreground if

$$|I_t(x,y) - B_t(x,y)| > \tau$$

is satisfied where τ is a predefined threshold. The background image B_T is updated as follows

$$B_{t+1} = aI_t + (1-a))B_t$$

Toyama et al. [9] propose a three component system for background maintenance, (Wallflower algorithm): the *pixel-level* component which performs Wiener filtering, the *region-level* component, fills in homogenous regions of foreground objects and the *frame-level* component for sudden, global changes. Two auto-regressive background models are used, along with a background threshold.

Halevi and Weinshall [27] present an approach to the tracking of very non rigid patterns of motion, such as water flowing down a stream. The algorithm based on a "disturbance map", which is obtained by linearly subtracting the temporal average of the previous frames from the new frame. Every local motion creates a disturbance having the form of a wave, with a "head" at the present position of the motion and a historical "tail" that indicates the previous locations of that motion. The algorithm is very fast and can be performed in real-time.

Wren et al. [10], Pfinder models the background using a single Gaussian distribution and uses a multi-class **statistical model** for the tracked object; uses a simple scheme, where background pixels are modelled by a single value and foreground pixels are modeled by a mean and covariance, which are updated recursively.

Haritaoglu et al. [4] propose a real time visual surveillance system, *W4*, for detecting and tracking multiple people and monitoring their activities in an outdoor environment. The system can identify and segment the objects that are carried by people and can track both objects and people separately. The W4 system uses a **statistical background model** where each pixel is represented with its minimum (M) and maximum (N) intensity values and

maximum intensity difference (D) between any consecutive frames observed during initial training period where the scene contains no moving objects. A pixel in the current frame I_t is classified as foreground if it satisfies:

$$|M(x,y) - I_t(x,y)| > D(x,y) \text{ or } |N(x,y) - I_t(x,y)| > D(x,y)$$

The statistics of the background pixels that belong to the non-moving regions of current frame are updated with new image data.

Many researchers have tried various approaches for object tracking. Nature of the technique used largely depends on the application domain. Some of the research work done in the field of object tracking includes:

A. Gyaourova, C. Kamath, S. and C. Cheung have studied the block matching technique for object tracking in traffic scenes. A motionless airborne camera is used for video capturing. They have discussed the block matching technique for different resolutions and complexities [73].

Yoav Rosenberg and Michael Werman explains an object tracking algorithm using moving camera. The algorithm is based on domain knowledge and motion modelling. Displacement of each point is assigned a discreet probability distribution matrix. Based on the model, image registration step is carried out. The registered image is then compared with the background to track the moving object [74].

A. Turolla, L. Marchesotti and C.S. Regazzoni discuss the camera model consisting of multiple cameras. They use object features gathered from two or more cameras situated at different locations. These features are then combined for location estimation in video surveillance systems [75].

One simple feature based object tracking method is explained by *Yiwei Wang, John Doherty and Robet Van Dyck [76].* The method first segments the image into foreground and background to find objects of interest. Then four types of features are gathered for each object of interest. Then for each consecutive frame the changes in features are calculated for various possible directions of movement. The one that satisfies certain threshold conditions is selected as the position of the object in the next frame.

1.3 Objectives

The main objective of this thesis is to propose a method for background modeling that incorporates the detection and tracking. Other objectives are to review and compare existing methods for background modeling, as well as to implement selected algorithms on Matlab. The methods for background modeling that will be reviewed have been preselected.

1.4 Outline

To accomplish the main objective the thesis has been outlined as follows. The thesis is divided into five chapters, introduction, moving object detection and tracking, methodological issues, implementation issues and future work.

Chapter 1 provides an introduction to the thesis. The background and motivation to the thesis are given and the objectives are outlined.

Chapter 2 introduces the concept of adaptive background models for video sequences and describes five methods stemming from related to background subtraction. Additional it introduces the next step in our video analysis, object tracking, which can simplify defined as the creation of temporal correspondence among detected objects from frame to frame.

Wallflower algorithm [9] is a three component system for background maintenance: *the pixel-level* component performs Wiener filtering to make probabilistic predictions of the expected background; *the region level* component fills in homogenous regions of foreground objects; and the *frame-level* component detects sudden, global changes. Using *Halevy* algorithm [27] we track non rigid patterns of motion, such as water flowing down a stream. The algorithm is based on a 'disturbance map' which is obtained by linearly subtracting the temporal average of the previous frames from the new frame. The algorithm is very fast and can be performed in real time. The third

algorithm *LOTS* ([5], [7]), uses two background images and two per-pixel thresholds. The two background models periodic changes such as moving trees. The per-pixel threshold image can treat each pixel differently, allowing the detector to be robust to localized noise in low-size image regions. The per pixel threshold evolves according to a pixel label provided by a quasi Connected Components analysis (QCC). In *W4* method [4], the background scene is modelled by representing each pixel by three values; minimum intensity (Min), maximum intensity (Max) and the maximum intensity difference (D) between consecutives during the training period. The last method *SGM (Single Gaussian Model)* [11] assumes that each pixel is a realization of a random variable with a Gaussian distribution. Furthermore, we study Mixture Models. Stauffer and Grimson [1] use Mixture Models to represent the statistics of the scene. This method is more complex and robust for real-time background modelling.

Four approaches of object tracking are distinguished: model-based, active contour-based, region-based and feature-based [72].On *Model-based approach* when a shape model of the tracked object is available, it can be fitted to the images in the sequence. This approach is useful for tracking rigid-body objects. On *active contour-based approach*, the outline of an object is known and it can be tracked using active contours or snakes. The contour is adapted to the image data using energy minimization. On *Region-based approach*, when a moving object is segmented, a region of pixels assigned to the object. This region can be tracked using approaches like cross-correlation. Many times a moving object corresponds to one or several tracked regions. To solve this problem, a combination of several regions to one object is perfomed at a higher level of abstraction. *Feature-based approach*, extract features from the image and track them, e.g, tracking of line segments and corners [50] and motion of the centroid [47]

Chapter 3 examines two methods to the main objective of the thesis, a method for background modeling and moving object detection. The main

focus is the final algorithm and its implementation. We study analytically both Gaussian Mixture Model [1] and LOTS algorithm ([5], [7]). An overview of Object Tracking algorithm that we have implemented is presented analytically. Additionally, we applied a shadow elimination algorithm based on gradient features at output frames of Gaussian Mixture Model Its theory with its critical steps are presented at the last unit of this chapter.

Chapter 4 A presentation of the typical results with an accompanying discussion is given. We adduce input-frames, output-frames and the background-frames for each method. We line up the object tracking results for the two methods and we make a compare among them. The shadow elimination results for two different frames are presented. The chapter ends with some conclusions.

Chapter 5 Future work.

Chapter 2: Moving Object Detection and Tracking

This chapter introduces the concept of adaptive background models for video sequences and describes methods for background modelling using background subtraction and statistical approaches. We line up analytically two methods for moving object detection that we implemented. The first method is a more complex method and uses statistical approaches for moving object detection (Gaussian Mixture Model). Focus is given on understanding the underlying theory of the method. The second algorithm, LOTS, is a moving object detection and tracking algorithm that based on background subtraction. We present for moving object detection, methods based on background subtraction and on statistical approaches that used widely with a brief explanation and analysis. Besides, we are being related on object tracking methods.

2.1 Introduction

In computer vision a background model refers to an estimated image or the statistics of the background of a scene which an image or video sequence depicts. In object tracking from video sequences, i.e. tracking people, cars, etc., the background model plays a crucial role in separating the foreground from the background.

The simplest form of background model is perhaps taking an image of the scene when no objects are present and then using that image as the background model. The foreground can be determined by frame differencing, i.e. comparing each pixel in the currently sampled frame to the background image and if the difference is below some threshold, the pixel is classified as background. Such a solution may be sufficient in a controlled environment, but in an arbitrary environment such as outdoor scenes, light conditions will vary over time. Also, it may be either difficult or impossible to be able to take an image of the scene without any objects present. It is therefore highly desirable to have a background model that adapts to the scene regardless of its initial state.

This thesis focuses on adaptive background models that can be maintained in real-time. In some literature the adaptive methods explained here are referred to as recursive techniques, since the current background model is recursively updated in each iteration.

2.2 Background Subtraction Techniques

Background subtraction is a used class of techniques for segmenting out objects of interest in a scene. Here we present some published techniques for background subtraction and analyses them with respect to three important attributes:

- Foreground detection
- Background maintenance
- Post-processing

Background subtraction involves comparing an observed image with an estimate of the image if it contained no objects of interest. The areas of the image plane where there is a significant difference between the observed and estimated images indicate the location of the objects of interest. The name

'background subtraction' comes from the simple technique of subtracting the observed image from the estimated image and thresholding the result to generate the object of interest.

Our problems here are how the object areas are distinguished from the background; how the background is maintained over time; and how the segmented object areas are post processed to reject false positives, etc. From several algorithms, some conclusions are drawn about what features are important in an algorithm.

2.2.1 Wallflower

In [12], two auto-regressive background models are used:

$$B_t = -\sum_{k=1}^{P} a_k B_{t-k} \qquad [2.1]$$

$$\hat{I}_t = -\sum_{k=1}^{p} a_k I_{t-k} \qquad [2.2]$$

along with a background threshold

$$\varepsilon\left(e_t^2\right) = \varepsilon\left(B_t^2\right) + \sum_{k=1}^{p} a_k \varepsilon\left(B_t B_{t-k}\right) \qquad [2.3]$$

$$\tau = 4\sqrt{\varepsilon\left(e_t^2\right)} \qquad [2.4]$$

Pixels are marked as background if

$$\left|I_t - B_t\right| < \tau \quad \text{and} \quad \left|I_t - \hat{I}_t\right| < \tau \qquad [2.5]$$

The coefficients α_k are updated each frame time from the sample covariances of the observed background values. In the implementation, the last 50 values are used to estimate 30 parameters. If more than 70% of the image is classified as foreground, the model is abandoned and replaced with a "back-up" model.

2.2.2 Halevy

In [4], the background is updated by

$$B_{t+1} = aS(I_t) + (1-a)B_t \qquad [2.6]$$

at all pixels, where S(It) is a smoothed version of It. Foreground pixels are identified by tracking the maxima of $S(I_t\text{-}B_t)$, as opposed to thresholding. They use α= [0.3……0.5] and rely on the streaking effect to help in determining correspondence between frames. They also note that $(1-a)^t < 0.1$gives an indication of the number of frames t needed for the background to settle down after initialisation.

2.2.3 LOTS

In [1], three background models are simultaneously kept, a primary, a secondary, and an old background. They are updated as follows:

1. The primary background is updated as

$$B_{t+1} = aI_t + (1-a)B_t \qquad [2.7]$$

if the pixel is not marked as foreground, and is updated as

$$B_{t+1} = \beta I_t + (1-\beta)B_t \quad [2.8]$$

if the pixel is marked as foreground. In the above, α was selected from within the range [0.0000610351….0.25], with the default value α= 0.0078125, and β = 0.25α.

2. The secondary background is updated as

$$B_{t+1} = aI_t + (1-a)B_t \quad [2.9]$$

at pixels where the incoming image is not significantly different from the current value of the secondary background, where α is as for the primary background. At pixels where there is a significant difference, the secondary background is updated by

$$B_{t+1} = I_t \quad [2.10]$$

3. The old background is a copy of the incoming image from 9000 to 18000 frames ago.

Foreground detection is based on adaptive thresholding with hystersis, with spatially varying thresholds. Several corrections are applied:

1. Small foreground regions are rejected.
2. The number of pixels above threshold in the current frame is compared to the number in the previous frame. A signifcant change is interpreted as a rapid lighting change. In response the global threshold is temporarily increased.
3. The pixel values in each foreground region are compared to those in the corresponding parts of the primary and secondary backgrounds, after scaling to match the mean intensity. These eliminate artifacts

due to local lighting changes and stationary foreground objects, respectively.

2.2.4 W4

In [5, 6, 7], a pixel is marked as foreground if

$$|M - I_t| > D \text{ or } |N - I_t| > D \quad [2.11]$$

where the (per pixel) parameters M, N, and D represent the minimum, maximum, and largest interframe absolute difference observable in the background scene. These parameters are initially estimated from the first few seconds of video and are periodically updated for those parts of the scene not containing foreground objects.

The resulting foreground “image" is eroded to eliminate 1-pixel thick noise, then connected component labelled and small regions rejected. Finally, the remaining regions are dilated and then eroded.

If we set M, N as background images and we use the total background of the area, then this operation will accentuate new objects. Otherwise, if we use previous frames then the operator will emphasize on differences from the previous state of the image. This scheme can also work with LOTS and the other background subtraction algorithms.

2.2.5 Single Gaussian Model (SGM)

In this method the information is collected in a vector $[Y,U,V]^T$, which defines the intensity and color of each pixel. The mean $\mu(x,y)$ and covariance $\Sigma(x,y)$ of each pixel can be recursively updated as follows

$$\mu^t(x,y) = (1-a)\mu^{t-1}(x,y) + aI^t(x,y) \quad [2.12]$$

$$\Sigma^t(x,y) = (1-a)\Sigma^{t-1}(x,y) + a\left(I^t(x,y) - \mu^t(x,y)\right)\left(I^t(x,y) - \mu^t(x,y)\right)^T \qquad [2.13]$$

where $I(x,y)$is the pixel of the current frame in YUV color space, α is a constant.

After updating the background, the SGM performs a binary classification of the pixels into foreground or background nad tries to cluster foreground pixels into blobs. Pixels in the current frame are compared with the background by measuring the log likelihood in color space. Thus, individual pixels are assigned either to the background region or a foreground region

$$I(x,y) = -\frac{1}{2}\left(I^t(x,y) - \mu^t(x,y)\right)^T \left(\Sigma^{-1}\right)^t \left(I^t(x,y) - \mu^t(x,y)\right) - \frac{1}{2}\ln\left|\Sigma^t\right| - \frac{m}{2}\ln(2\pi) \qquad [2.14]$$

where $I^t(x,y)$ is a vector $(Y,U,V)^T$ defined for each pixel in the current image, $\mu^t(x,y)$ is the pixel vector in the background image B. If a small likelihood is computed using (2.14), the pixel is classified as active. Otherwise, it is classified as background.

2.2.6 Mixture Models

The method for background modelling explored here was introduced by Stauffer and Grimson in [1]. Stauffer and Grimson noted that the computational performance of computers at the time, 1999, had reached a level where more complex and robust methods for real-time background modelling could be considered. Their approach is to use mixture models to represent the statistics of the scene. In contrast to approximate median filtering, using mixture models allows for a multi-modal background model

which can be very useful in removing repetitive motion, e.g. shimmering water, leaves on a branch, or a swaying flag.

2.3 Shadow detection and light change detection

The algorithms described above for motion detection perform well on indoor and outdoor environments and have been used for real-time surveillance for years. However, most of these algorithms are susceptible to both local (e.g. shadows and highlights) and global illumination changes (e.g. sun being covered/uncovered by clouds). Shadows cause the motion detection methods fail in segmenting only the moving objects.

Moving shadows need careful consideration in the development of robust dynamic scene analysis systems. Moving shadow detection is critical for accurate object detection in video streams since shadow points are often misclassified as object points, causing errors in segmentation and tracking. Many algorithms have been proposed in the literature that deals with shadows. Here we present a comprehensive survey of moving shadow detection approaches. We organize contributions reported in the literature in four classes two of them are statistical and two are deterministic.

Detection and tracking of moving objects is at the core of many applications dealing with image sequences. One of the main challenges in these applications is identifying shadows which objects cast and which move along with them in the scene. Shadows cause serious problems while segmenting and extracting moving objects due to the misclassification of shadow points as foreground. Shadows can cause object merging, object shape distortion, and even object losses (due to the shadow cast over another object). The difficulties associated with shadow detection arise since shadows and objects share two important visual features. First, shadow points are detectable as foreground points since they typically differ significantly from the background. Second, shadows have the same motion as the objects

casting them. For this reason, the shadow identification is critical both for still images and for image sequences (video) and has become an active research area, especially in the recent past. It should be noted that, while the main concepts utilized for shadow analysis in still and video images are similar, typically, the purpose behind shadow extraction is somewhat different. In the case of still images, shadows are often analyzed and exploited to infer geometric properties of the objects causing the shadow ("shape from shadow" approaches) as well as to enhance object localization and measurements.

2.3.1 Taxonomy of shadow detection algorithms

We have organized the various algorithms in a two-layer taxonomy. The first layer classification considers whether the decision process introduces and exploits uncertainty. **Deterministic approaches** use an on/off decision process, whereas **statistical approaches** use probabilistic functions to describe the class membership. Introducing uncertainty to the class membership assignment can reduce noise sensitivity. In the statistical methods the parameter selection is a critical issue. Furthermore we divide the statistical approaches in **parametric** and **nonparametric** methods. The deterministic class can be further subdivided. Sub classification can be based on whether the on/off decision can be supported by model-based knowledge or not. Choosing a model-based approach undoubtedly achieves the best results, but is, most of the time, too complex and time consuming compared to the nonmodel-based.

The types of features are extracted from three domains: spectral, spatial, and temporal. Approaches can exploit differently spectral features, i.e., using grey level or colour information. Some approaches improve results by using spatial information working at a region level or at a frame level instead of pixel level. This is a classification similar to that used in [15] for

the background maintenance algorithms. Finally, some methods exploit temporal redundancy to integrate and improve results.

2.4 Object Tracking

Object tracking is an important task within the field of computer vision. The proliferation of high-powered computers, the availability of high quality and inexpensive video cameras, and the increasing need for automated video analysis has generated a great deal of interest in object tracking algorithms.

The aim of object tracking is to establish a correspondence between objects or object parts in consecutive frames and to extract temporal information about objects such as trajectory, speed, direction etc.

2.4.1 Introduction

There are three key steps in video analysis: detection of interesting moving objects, tracking of such objects from frame to frame, and analysis of object tracks to recognize their behavior. Therefore, the use of object tracking is pertinent in the tasks of:

- Motion-based recognition, that is human identification based on gait, automatic object detection, etc;
- Automated surveillance, that is monitoring a scene to detect suspicious activities or unlikely events;
- Video indexing, that is, automatic annotation and retrieval of the videos in multimedia databases;
- Human-computer interaction, that is, gesture recognition, eye gaze tracking for data input to computers, etc;
- Traffic monitoring, that is, real-time gathering of traffic statistics to direct traffic flow.

- Vehicle navigation that is, video-based path planning and obstacle avoidance capabilities.

In its simplest form, tracking can be defined as the problem of estimating the trajectory of an object in the image plane as it moves around the scene. In other words, a tracker assigns consistent labels to the tracked objects in different frames of a video.

Additionally, depending on the tracking domain, a tracker can also provide object-centric information, such as orientation, area, or shape of an object. Tracking objects can be complex due to:

—loss of information caused by projection of the 3D world on a 2D image,

—noise in images,

—complex object motion,

—nonrigid or articulated nature of objects,

—partial and full object occlusions,

—complex object shapes,

—scene illumination changes, and

—real-time processing requirements.

One can simplify tracking by imposing constraints on the motion and/or appearance of objects. For example, almost all tracking algorithms assume that the object motion is smooth with no abrupt changes. One can further constrain the object motion to be of constant velocity or constant acceleration based on a priori information. Prior knowledge about the number and the size of objects, or the object appearance and shape, can also be used to simplify the problem.

Numerous approaches for object tracking have been proposed. These primarily differ from each other based on the way they approach the following questions: Which object representation is suitable for tracking? Which image features should be used? How should the motion, appearance, and shape of the object be modelled? The answers to these questions depend

on the context/environment in which the tracking is performed and the end use for which the tracking information is being sought. A large number of tracking methods have been proposed which attempt to answer these questions for a variety of scenarios.

2.4.2 Approaches of Object tracking

Four approaches for object tracking are generally distinguished [14]: model-based, active contour-based, region-based and feature-based. Model-based techniques usually require object classification before tracking; other approaches can be used either before or after object classification. Feature the tracked features is needed.

2.4.2.1 Model-based

When a shape model of the tracked object is available, it can be fitted to the images in the sequence. This gives the position and motion of the tracked object and at the same time an estimate of the object pose. This approach is very useful for tracking rigid-body objects such as robots [16] and cars ([17], [18], [19], [20]. However, fitting a model to image data is computationally expensive.

With respect to humans, body parts like the face [21], head ([22], [23], [24]) and hands ([25], [26]) are often tracked. See Figure 3.1(a) for an example of a head model. However, this is performed mainly in structured scenes where only one or two moving objects are in the camera view. Generally, many pixels are required on the moving objects. For full human body tracking ([28],[29]) the demands on scene composition and number of object pixels are even more strict, for example only people walking parallel to the image plane are considered [30]. Often, several cameras are used to create a 3D scene reconstruction [12], see Figure 2.2(b).

2.4.2.2 Active contour-based

After segmentation, the outline of an object is known. The outline can be tracked using active contours or snakes ([13], [15]), see Figure 2.1-2.2 for an example. Using energy minimization, the contour is adapted to the image data. Besides tracking of the object, an accurate object contour description is now available each frame.

At high computational cost, the active contour approach is able to describe arbitrary shapes, as long as the smoothness constraint is satisfied. However, for this approach initialization and association between frames is a problem, for example when objects form groups and split again. Another disadvantage of active contours is that tracking is based on the most deformable part of the object, its contour.

To overcome the limitation of association, the active contour algorithm can be combined with an algorithm describing the colour of the object [37].

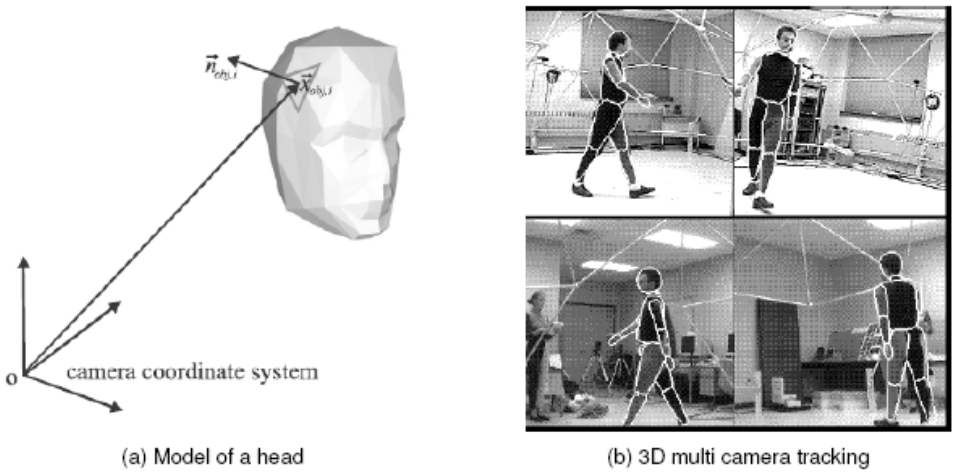


Figure 2.1 Two model-based object Tracking examples

Figure 2.2 Example of active contour-based tracking

2.4.2.3 Region-based

When a moving object is segmented, a region of pixels assigned to the object is available. This region can be tracked using approaches like cross-correlation. The location of the region in the next frame is to be determined. A moving object usually corresponds to one [29] or several [35] tracked regions. Combination of several regions to one object is then performed at a higher level of abstraction.

Several techniques are available for modelling and tracking image regions. The regions are often modelled using a probability density distribution of their colour. This distribution can be described using a colour histogram ([31],[34]) , or a mixture of Gaussian kernels ([40],[36]). Instead of using one 3D probability density distribution, separate distributions for each of the colours can be used [43].

Probability density distributions of the colour are relatively invariant to changes in object orientation, scale, partial occlusion, viewing position and object deformation [39]. This makes them particularly interesting for tracking nonrigid objects such as humans. However, the distributions capture

only the colours in an image and do not include any spatial correlation information. Therefore, they have limited discriminative power. A colour correlogram, on the other hand, is a cooccurrence matrix that gives the probability that a pixel at a distance d from a given pixel of colour $\tilde{c}_i$ is of colour $\tilde{c}_j$. This way spatial information in the form of distance to pixels of a certain colour is introduced [34].

Other approaches taking spatial information into account are using many small regions [35] and using the time average per-pixel colour [46]. Instead of choosing one colour space, automatic selection of most discriminative features can be used [41]. This adapts the colour space used by comparing the specific tracked region with the local background, leading to more precise object segmentation. However, when pixels are misclassified and consequently used for updating the wrong model, this solution will become unstable.

Considering the low number of pixels in each tracked region, histograms will become quite sparse. On the other hand, it also is not easy to estimate the parameters of a Gaussian mixture model from only a few data points, especially when also the number of kernels is unknown. From the point of view of computational complexity, the use of a histogram approach is most affordable because template matching can be performed very efficiently.

Considering that a probability density function is available with these techniques, it is unfortunate that object segmentation is often based on a static threshold. Calculated probabilities could be used in a probabilistic foreground/background classification algorithm.

Moving objects can also be modelled using a fixed or parameterized shape, like in the mean-shift approach [44], and the particle filter ([42],[45]). Disadvantage of such techniques is that they are unable to describe an arbitrary shape, changing between subsequent frames.

2.4.2.4 Feature-based

Feature-based object tracking approaches are similar to region-based approaches. Instead of tracking the entire region, feature-based approaches extract features from the image and track these. Examples are tracking of line segments and corners [50] and motion of the centroid [47]. Features such as moment invariant functions and aspect ratio [49] could also be used.

Such approaches can be implemented very efficiently and are theoretically able to handle partial occlusions. They are used frequently for traffic surveillance. Reasons they are not used often in general surveillance applications include: low recognition rate of features due to nonlinear perspective transformation and the stability of dealing with occlusions is generally poor [14].

2.4.2.5 Wrap-up

For surveillance applications, model-based approaches are computationally complex and often the amount of pixels required for an accurate fit of the model is not available. Feature-based approaches should also not be used, because of the low recognition rate of features and problems with occlusion.

Active contour-based approaches have the advantage of generating an accurate object contour. This can be a major advantage for applications requiring pose estimation for example. However, these approaches are generally more computational complex than region-based approaches, and require object detection by another algorithm. Occlusion and robustness over many frames can pose problems, as tracking is based on the most unstable part of the object. If computational complexity is not an issue, the combination of region-based tracking and active contour-based tracking is optimal, in other cases region-based tracking is a good choice.

The region-based approach is most popular for surveillance applications. It is computationally fast compared to approaches based on active contours and object models, and as region-based methods use the entire region, they are more stable and can better cope with occlusions than feature-based approaches.

2.4.3 Applications of Object Tracking

In the last few years, a new application domain has emerged in computer vision. This domain works on the analysis of images involving humans, covering, among others, issues like, hand gesture recognition, lip tracking and whole-body tracking [48]. The tracking of human motion could be put in a general framework of human motion analysis [51]. A step of motion analysis involving human body parts may precede the tracking-phase, providing to it some low-level information (e.g. body part segmentation or joint detection and identification) that may be useful during the tracking-phase. Finally, a higher-level task of recognizing human activities may follow a successful tracking stage, completing the procedure of human motion analysis. There are many interesting and promising applications in this area. For a summary, see Table 2.1 [48].

More specifically, human motion analysis can help in the development of advanced social interfaces, where computer-generated characters may interact with the user in a more friendly way, using human-like behaviours [52].

General Domain	Specific Area
Advanced user interfaces	social interfaces Sign-language translation Gesture driven control Signaling in high-noise enviroments (airports,factories)
Virtual reality	Interactive virtual worlds Games Virtual studios Character animation Teleconferencing (e.g. film, advertising, home-use)
Motion analysis	Content-based indexing of sports video footage Personalized training in golf, tennis, etc. Choreography of dance and ballet Clinical studies of orthopedic patients
Smart surveillance systems	Access control Parking lots Supermarkets, departments stores Vending machines,ATMs Traffic

Table 2.1 Applications of Object Tracking

Furthermore, a speech-guided interface can use computer vision, either in order to detect the presence of a user and commence an interaction, either in order to recognize a user, distinguish multiple users and guide the dialogue in a more proper way, or finally in order to enable a more robust recognition of speech in the presence of acoustic noise [56]. Other interesting applications in this domain are sign-language interpretation [53], gesture-driven control for people with disabilities [54] and signalling in high-noise environments, such as airports and factories.

The development of interactive virtual worlds is also relevant to the above application domain. The tracking of the human body may help in the creation of a human presence in a virtual space, whereas the tracking of hand gestures may be useful in finding a natural way to interact with virtual objects. Other applications in the domain of virtual reality are games [60], virtual studios and character animation

Moreover, visual-based human motion analysis can be applied in personalized training systems for various activities, like sports and dance. It can also help the clinical research in medical branches, like orthopaedics. Another possible application could be the content-based indexing of sports video footage, that would decrease the browsing-effort through a large data set, for example in a query like "give me all the cases of action X of the player Y" [48].

Another important application domain is that of "smart" surveillance. Applications may range from detection of human presence and motion to face recognition for the purpose of access control or the observation of human actions and suspicious behaviours. These applications are useful in areas such as parking lots, airports, department stores or traffic management systems. Of course the matter of privacy rights must be taken into account in these cases [48].

Especially in traffic management systems, it is usually desirable to track, apart from humans, other objects as well, for example, vehicles, obstacles and traffic signs. The goal is usually the maintenance of a secure distance of the pedestrian or vehicle from static or moving obstacles and the observance of traffic laws [58]. Furthermore, visual systems that track only vehicles are also useful in applications such as the measurement of traffic flow or the computation of parameters like the average vehicle speed and spatial occupancy [58].

Visual tracking is also applicable in areas where the human motion is not involved in a direct way like in the previous applications, or is not

involved at all. Medicine is one of them and relevant application is the tracking of biological structures in MR images [59].

Robotic applications are another domain. These can include mobile robot navigation [61], machine-learning [62] and visual servo [63].

Finally, tracking techniques are often applied in the area of model-based coding in order to accomplish low bit-rate video compression [55].

Chapter 3: Methodological Issues

This chapter gives details to approaches used for moving object detection and object tracking. The Gaussian Mixture Model [1] and LOTS algorithm ([5],[7]) are common approach in the literature. Gaussian Mixture Model is presented in detail in section 3.2 and LOTS method is presented in detail in section 3.3. The object tracking algorithm is presented in section 3.4.

3.1 Introduction

Background subtraction is a convenient and effective method for detecting moving foreground objects in the scene. A reliable background image is important for foreground segmentation. The pixel-based background subtraction method basically involves subtraction of a considered image from a reference image.

Although background subtracting approach is simple, it may be impractical in some real applications because backgrounds can change over time in some cases. Lighting can change the background subtly or the camera position may drift. An alternative approach is to find a way of adapting the background slowly such that changing background can be characterized in real-time. Such an approach is called adaptive background mixture models.

In our work, we implemented Stauffer and Grimson's algorithm [1] for background modelling together with a shadow elimination algorithm. The Gaussian mixture model representation of the scene statistics has proven to

be very flexible and reasonably efficient when implemented. Additionally we implemented the tracking algorithm LOTS (Lehigh Omni-directional Tracking System) ([5],[7]).

3.2 Gaussian Mixture Method

Stauffer and Grimson [1] presented a novel adaptive online background mixture model that can robustly deal with lighting changes, repetitive motions, clutter, introducing or removing objects from the scene and slowly moving objects. Their motivation was that a unimodal background model could not handle image acquisition noise, light change and multiple surfaces for a particular pixel at the same time. Thus, they used a mixture of Gaussian distributions to represent each pixel in the model. Due to its promising features, we implemented and integrated this model in our visual surveillance system.

3.2.1 Mixture Models

The method for background modeling was introduced by Stauffer and Grimson in [1]. Their approach is to use mixture models to represent the statistics of the scene. Mixture models allows a multi-modal background model which can be very useful in removing repetitive motion, e.g. leaves on a branch, a swaying flag or shimmering water. The method is quite flexible and here we present the mathematical theory.

The pixel value measured by the camera sensor is the radiance emitted from the surface point of first object to intersect that pixel's optical ray. In a dynamically changing scene, with moving objects, the observed pixel value depends on the surface of the possible intersecting objects as well as noise introduced by the camera.

Using mathematical terms we can view the sampling process as the sampling of a random process **X** where each sampled value is generated by the surface of some object. The sampled value of **X** is an observation of random variables $\mathbf{S_k}$, with K indicating which object was observed. Since only one of the objects can intersect the pixel's optical ray at a time except object edges, the underlying events, that object k is observed, are disjoint. Therefore, the density function of **X** can be modeled as linear combination of the density functions of the objects that generated the sample values. Such a model is called a mixture model and defines a mixture density as

$$p(x|\Theta) = \sum_{k=1}^{K} \omega_k \, p_k(x|\theta_k) \tag{3.1}$$

where x is a sample of **X** and Θ is the parameter vector describing p, where $\Theta = \{\omega_1,.....,\omega_k,\theta_1,.....,\theta_k\}$, θ_k in turn is a parameter vector describing p_k. Additionally, the mixing densities ω_k are in the range [0,1] and $\sum_{k=1}^{K} \omega_k = 1$ $p(x|\theta_k)$ are called component density functions and are normalized so that,

$$\int_{Dx} p_k(x|\Theta_k)dx = 1, \text{ for } \; k \in [1,....,K] \tag{3.2}$$

3.2.1.1 Theoretical Derivation of Update Procedure

The problem is given a data set of finite size of independently drawn samples of **X,** how to choose the parameter vector Θ. Maximum likelihood estimation is a common approach.

As **X** denote a random variable with probability density function $p(x|\Theta)$ and $X = \{x_1,.....,x_N\}$ denote a data set of N independently drawn samples of **X**. The likelihood function is then

$$L(\Theta|X) = \prod_{i=1}^{N} P(x_i|\Theta) \tag{3.3}$$

The optimal parameter vector is found by maximizing the likelihood function with respect to Θ,

$$\Theta^* = \arg\max_{\Theta} L(\Theta|X) \tag{3.4}$$

3.2.1.2 Expectation-Maximization Algorithm

The expectation-maximization algorithm [64], or EM, is a general iterative method for finding the maximum-likelihood estimation of the parameters of a given data set's governing distribution when that data set is incomplete. A data set is said to be incomplete when the governing distribution either has unknown, or hidden, parameters or missing values.

Let X be an incomplete data set that is known. By assuming the existence of a complete data set Z=(X,Y) a joint density function can be specified,

$$p(z|\Theta) = p(x,y|\Theta) = p(x|\Theta)p(y|x,\Theta) \tag{3.5}$$

where y denotes the unknown data of Z. Using this density function a new likelihood function, called the complete data likelihood function, can be defined as

$$L(\Theta|Z) = L(\Theta|x,y) = \prod_{i=1}^{N} p(x_i|\Theta)p(y_i|x_i,\Theta) \tag{3.6}$$

The complete data likelihood function becomes a function of the variable y, by letting Θ be fix. That is

$$L(\Theta|X,Y) = /\Theta \ \ fix / = l(Y|X,\Theta) \tag{3.7}$$

Assuming, Y is an instance of a random variable **Y,** we can calculate the expected value of log $l(\mathbf{Y}|X,\Theta)$. The EM algorithm uses a given data set of parameters estimates Θ_{t-1}, to evaluate the marginal density of **Y** given X and uses this function to calculate the expected value of the complete data log-likelihood function; this step is known as the expectation step or E-step.

Since Θ is fix, the expected value will be a deterministic function of Θ. At [5] Dempster define function Q as

$$Q\left(\Theta^{t},\Theta^{t-1}\right)=\mathrm{E}\left[\log l\left(\mathrm{Y}\middle|\mathrm{X},\Theta^{t}\right)\middle|\mathrm{X},\Theta^{t-1}\right] \tag{3.8}$$

where $\Theta^{t}=\Theta$ and the right hand side is evaluated by

$$E\left[\log l\left(\mathrm{Y}\middle|\mathrm{X},\Theta^{t}\right)\middle|\mathrm{X},\Theta^{t-1}\right]=\int_{y\in Dy}\log l\left(y\middle|\mathrm{X},\Theta^{t}\right)p\left(y\middle|\mathrm{X},\Theta^{t-1}\right)dy \tag{3.9}$$

The final step of the EM algorithm is to maximize Q with respect to Θ^{t} , that is

$$\Theta^{*}=\arg\max_{\Theta}Q\left(\Theta^{t},\Theta^{t-1}\right) \tag{3.10}$$

This step of the algorithm is known as the maximization step or M step. The EM algorithm uses Θ^{*} as Θ^{t-1} in (3.10) and the two steps of the algorithm are repeated. Each iteration of the algorithm is guaranteed to increase the likelihood function until a local maxima is reached.

3.2.1.3 Gaussian Mixture Models using EM algorithm

A common approach to the problem of determining maximum likelihood parameter estimates for mixture models is to use the EM algorithm. We consider a random variable **X** represented by a mixture model consisting of K underlying random processes each with their own probability density function. The mixture density is then

$$p\left(x|\Theta\right)=\sum_{k=1}^{K}\omega_{k}\,p_{k}\left(x|\theta_{k}\right) \tag{3.11}$$

Let $X=\{x_{1},.....,x_{N}\}$ denote a data set of N independently drawn samples of **X**. since the samples are independently drawn and identically distributed by $p\left(x|\Theta\right)$, the log-likelihood function becomes

$$\log L(\Theta|X) = \log \prod_{i=1}^{N} p(x_i|\Theta) = \sum_{i=1}^{N} \log p(x_i|\Theta) = \sum_{i=1}^{N} \log \sum_{k=1}^{k} \omega_k p_k(x_i|\theta_k) \qquad (3.12)$$

We can simplify (4.12) if it is known which process generated each sample, by assuming that the underlying processes are disjoint. Let $k_i \in \{1,.....,K\}$ be a variable that is known, indicating this information. This expression reduces to

$$\log L(\Theta|X) = \sum_{k=1}^{N} \log \omega_{k_i} p_{k_i}(x_i|\theta_{k_i}) \qquad (3.13)$$

Assume that the data set X is incomplete in the sense that an accompanying unobserved data set indicating which process generated each sample value exists. Call this missing data set $K = \{k_1,.....,K_N\}$ and let Z=(X,K) denote the complete data set. If K is known the complete data log-likelihood becomes

$$\log L(\Theta|X,K) = \sum_{i=1}^{N} \log \omega_{k_i} p_{k_i}(x_i|\theta_{k_i}) \qquad (3.14)$$

K is unknown and we assume that K is an instance of a random variable **K** and we proceed with the expectation step of the algorithm. All that is needed is an expression for the marginal density of **K** given X. In this case a sample k of **K** will be a vector of N elements. Using Bayes rule it is easy to find an expression for the conditional probability of k_i given x_i and a parameter vector Θ^{t-1}

$$p(k_i|x_i,\Theta^{t-1}) = \frac{\omega_{k_i} p_{k_i}(x_i|\theta_{k_i}^{t-1})}{\sum_{k=1}^{K} \omega_k p_k(x_i|\theta_k^{t-1})} \qquad (3.15)$$

Since the samples are independently drawn, the marginal density is

$$p(K|X,\Theta^{t-1}) = \prod_{i=1}^{N} p(k_i|x_i,\Theta^{t-1}) \qquad (3.16)$$

Equation (3.8) takes the form

$$Q\left(\Theta^t,\Theta^{t-1}\right)=E\left[\log l\left(K\middle|X,\Theta^t\right)\middle|X,\Theta^{t-1}\right]=.....=$$
$$=\sum_{k=1}^{N}\sum_{i=1}^{N}\log\left(\omega_k p_k\left(x_i\middle|\theta_k^t\right)\right)p\left(k\middle|x_i,\Theta^{t-1}\right) \tag{3.17}$$

With the exception of determining ω_k, performing the M-step is a matter of choosing component density functions and differentiating $Q\left(\Theta^t,\Theta^{t-1}\right)$ with respect to the various parameters and setting them to 0. It is not necessary to specify component density functions in order to determine ω_k. Using d-dimensional Gaussian with parameter vectors $\theta_k=\left\{\mu_k,\Sigma_k\right\}$,

$$p_k\left(x\middle|\mu_k,\Sigma_k\right)=\frac{1}{\left(2\pi\right)^{d/2}\left|\Sigma_k\right|^{1/2}}e^{-\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)} \tag{3.18}$$

gives as results the expressions

$$\omega_k^t=\frac{1}{N}\sum_{i=1}^{N}p\left(k\middle|x_i,\Theta^{t-1}\right) \tag{3.19}$$

$$\mu_k^t=\frac{\sum_{i=1}^{N}x_i\,p\left(k\middle|x_i,\Theta^{t-1}\right)}{\sum_{i=1}^{N}p\left(k\middle|x_i,\Theta^{t-1}\right)} \tag{3.20}$$

$$\sum\nolimits_k^t=\frac{\sum_{i=1}^{N}p\left(k\middle|x_i,\Theta^{t-1}\right)\left(x_i-\mu_k^t\right)\left(x_i-\mu_k^t\right)^T}{\sum_{i=1}^{N}p\left(k\middle|x_i,\Theta^{t-1}\right)} \tag{3.21}$$

3.2.1.4 Stauffer and Grimsons algorithm

We understood how the EM algorithm is applied to Gaussian Mixtures Models and we can move on to look at the algorithm Stauffer and Grimson [1].

Since we expect the scene to dynamically change over time, the pixel process **X** can not be considered a stationary process over a long period of time. By specifying N only a finite history of samples will be used to determine the parameter vector Θ. Equations (3.19)-(3.21) needs to be evaluated for every pixel in every video frame. Since this is a computationally cumbersome operation when N is big, a trick that simplifies the operation is to compute the averages recursively.

By evaluating the expressions to N+1 the mixing densities in (3.19) evaluated to N+1 are

$$\omega_k^{N+1} = (1-a)\omega_k^N + a \cdot p\left(k \middle| x_{N+1}, \Theta^{t-1}\right) \tag{3.22}$$

where $a = \frac{1}{N+1}$. Using the same technique to (4.20)

$$\mu_k^{N+1} = (1-\rho_k)\mu_k^N + \rho_k x_{N+1} \tag{3.23}$$

where

$$\rho_k = \frac{a \cdot p\left(k \middle| x_{N+1}, \Theta^{t-1}\right)}{\omega_k^{N+1}} \tag{3.24}$$

Finally (3.21) yields

$$\sum_k^{N+1} = (1-\rho_k)\sum_k^N + \rho_k \left(x_{N+1} - \mu_k^{N+1}\right)\left(x_{N+1} - \mu_k^{N+1}\right)^T \tag{3.25}$$

3.2.1.5 Experiments

Firstly, in this algorithm we have to estimate which of the K underlying distributions generated the current pixel value x_t and update its parameter. In paper [1], Stauffer and Grimson define a match as pixel value falling within λ=2.5 standard deviations of a distribution, which can be interpreted in terms of Mahalanobis distance as

$$d_k^2 = (x_t - \mu_k)^T \sum_k^{-1} (x_t - \mu_k) \tag{3.26}$$

$$M_k = \begin{cases} 1 & if \;\; d_k < \lambda \\ 0 & otherwise \end{cases} \tag{3.27}$$

Equation (3.27) is essentially an approximation of $p\left(k \middle| x_{N+1}, \Theta^{t-1}\right)$. In practice $p\left(k \middle| x_{N+1}, \Theta^{t-1}\right)$ is approximately 1 for distributions with a mean close to the generated pixel value and approximately 0 otherwise.
Furthermore, in [1] Stauffer and Grimson use a single scalar to represent their covariance matrix, that is $\sum_k \sigma_k^2 \mathbf{I}$, where I is the identity matrix.

If several distributions match, the highest peaking distribution is chosen by choosing the k which maximizes $\frac{\omega_k}{\|\Sigma_k\|}$. Its parameters are then updated using (3.22)-(3.23)-(3.25) and the mixture densities are renormalized so that they sum up to one.

When calculating the new parameters, the only computationally bothersome operation is the evaluation of ρ_k. The approximations that can be made is $\rho_k \approx \frac{a}{\omega_k^{N+1}}$ or $\rho_k = a$.

If no distributions can be considered a match, the lowest peaking distribution is replaced by a new distribution, using the current pixel value as its mean, a high covariance matrix and a low mixing density. This is how new objects become part of the background.

In [1] Stauffer and Grimson maintain the distributions ordered with respect $\frac{\omega}{\sigma_k}$. The first B distributions to have a combined mixing density greater than a threshold T are chosen as the background model,

$$B = \arg\min_b \left(\sum_{k=1}^{b} \omega_k > T \right) \tag{3.28}$$

The binary foreground image is then estimated by calculating the Mahalanobian distance for the B first distribution. If the pixel can't be considered a match, it is deemed foreground.

3.2.2 Gaussian Mixture Model

In this model, the values of an individual pixel (e. g. scalars for gray values or vectors for color images) over time is considered as a "pixel process" and the recent history of each pixel, {X1, . . . ,Xt}, is modeled by a mixture of K Gaussian distributions. The probability of observing current pixel value then becomes:

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} * \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{3.29}$$

where $w_{i,t}$ is an estimate of the weight (what portion of the data is accounted for this Gaussian) of the ith Gaussian $G_{i,t}$ in the mixture at time t, $\mu_{i,t}$ is the

mean value of $G_{i,t}$ and $\Sigma_{i,t}$ is the covariance matrix of $G_{i,t}$ and η is a Gaussian probability density function:

$$\eta(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t-\mu_t)^T \Sigma^{-1}(X_t-\mu_t)} \tag{3.30}$$

Decision on K depends on the available memory and computational power. Also, the covariance matrix is assumed to be of the following form for computational efficiency:

$$\sum_{k,t} a_k^2 I \tag{3.31}$$

which assumes that red, green, blue color components are independent and have the same variance.

The procedure for detecting foreground pixels is as follows. At the beginning of the system, the K Gaussian distributions for a pixel are initialized with predefined mean, high variance and low prior weight. When a new pixel is observed in the image sequence, to determine its type, its RGB vector is checked against the K Gaussians, until a match is found. A match is defined as a pixel value within (L=2.5) standard deviation of a distribution.

$$|X_t - \mu_{i,t-1}| \leq L\sigma_{i,t-1} \tag{3.32}$$

Next, the prior weights of the K distributions at time t, $w_{k,t}$, are updated as follows:

$$\omega_{k,t} = (1-a)\omega_{k,t-1} + \alpha M_{k,t} \tag{3.33}$$

Where a is the learning rate and $M_{k,t}$ is 1 for the matching Gaussian distribution and 0 for the remaining distributions. After this step the prior weights of the distributions are normalized and the parameters of the matching Gaussian are updated with the new observation as follows:

$$\mu_t = (1-\rho)\mu_{t-1} + \rho(X_t) \tag{3.34}$$

$$\sigma_t^2 = (1-\rho)\sigma_{t-1}^2 + \rho(X_t - \mu_t)^T (X_t - \mu_t) \tag{3.35}$$

where

$$\rho = \alpha\eta(X_t | \mu_k, \sigma_k) \tag{3.36}$$

If no match is found for the new observed pixel, the Gaussian distribution with the least probability is replace with a new distribution with the current pixel value as its mean value, an initially high variance and low prior weight.

In order to detect the type (foreground or background) of the new pixel, the K Gaussian distributions are sorted by the value of ω/σ. This ordered list of distributions reflect the most probable backgrounds from top to bottom since by Equation 3.33 background pixel processes make the corresponding Gaussian distribution have larger prior weight and less variance. Then the first B distributions are chosen as the background model, where

$$B = \arg\min_b \left(\sum_{k=1}^{b} \omega_k > T \right) \tag{3.37}$$

and T is the minimum portion of the pixel data that should be accounted for by the background. If a small value is chosen for T, the background is generally unimodal. Figure 3.1 shows the flowchart of Gaussian Mixture Model for moving object Detection.

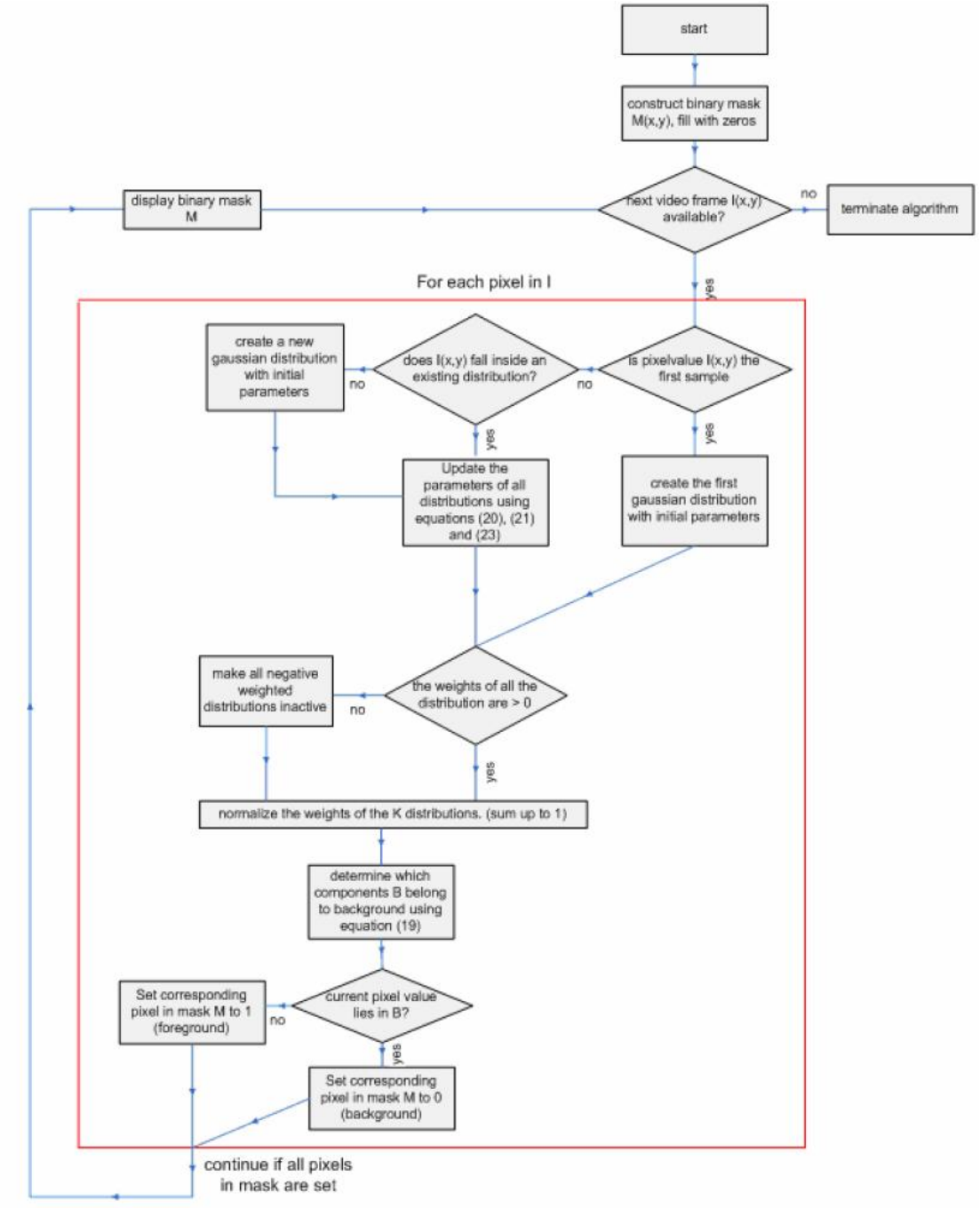


Figure 3.1 Flowchart of Gaussian Mixture Implementation (statistic approach)

3.2.3 Analysis of parameter values

The Gaussian mixture models are a type of density models which are composed of a number of components. These functions can be used to model the colours of objects or backgrounds in a scene. Adaptive Gaussian distributions are applicable for modelling changes, especially when related to fast moving objects.

Threshold T is to define the fraction between background distribution and foreground distribution. This value is based on the background scene and the number of components in the Gaussian Mixture Model. A small value of T (e.g T=0.1), will lead to a situation in which not all background distribution is covered; a large T value (T=0.9) will lead to a situation in which the foreground distribution is merging with the background distribution. In our thesis we use T=0.9 and change it observing different results.

K denotes the number of components in a Gaussian mixture model. For simple indoor scenes, a small value of K is sufficient, e.g K=2. For outdoor complex scenes, a larger K is needed, usually 3, 4, 5. In our thesis we use K=4 and change it observing different results.

There are two learning rates defined in [1]: one is the predefined learning rate $\boldsymbol{a}$, the other is the calculated learning rate $\boldsymbol{\rho}$, which is used as a second filter in [1]. But using ρ as a second learning rate is not helpful. If we assume that the computation time of using one learning rate α is m seconds, the computation time of using two learning rates a and ρ was greater than 2m seconds. For computation reasons, we used the same learning rate, $\rho = a$.

How to assign a reasonable value to a will depend on the given background scenery. A slowly changing background scene needs a small learning rate; a fast changing background scene needs a larger learning rate.

Learning rate a show the speed of update. In our thesis we use different values of a ($a = 0.01, 0.9, 0.2$ etc) with best results using $a = 0.01$.

There is an initialization procedure when starting the surveillance system. Assigning different initial values in this procedure will affect the extraction of foreground regions. There are two values that need initial consideration: mean and standard deviation. Regarding the mean value, from our testing sequences we conclude that assigning either a very large value or a very small value can be considered to be of benefit. In our thesis for initial mean we use $\mu = 0.05$.

In the initialization procedure, we assign the value $\sigma_o = 10$ to the standard deviation based in our experiments. For standard deviation equal to zero, many background pixels are misclassified as foreground region. In general, using very small value of the standard deviation causes that background pixels classified as foreground distribution.

3.3 LOTS algorithm

This algorithm operates on grey scale images. It uses two background images and two per-pixel thresholds ([5],[7]). The two backgrounds model periodic changes. The per-pixel threshold image can treat each pixel differently, allowing the detector to be robust to localized noise in low-size image regions. The per-pixel threshold evolves according to a pixel label provided by a Quasi Connected Components analysis (QCC).

3.3.1 Algorithm

The steps of the algorithm are:

I. Background Modeling.

We presume a two background model, the primary background $B_p^t(\phi)$ and the secondary background $B_s^t(\phi)$, where ϕ is the pixel index. The pixel intensity value is $I^t(\phi)$. We presume the input at time t-1 was closest to the primary model $B_p^{t-1}(\phi)$ and if that is not true we swap the pixels between the two background images. We define the difference images as

$$D_p^t(\phi) = I^t(\phi) - B_p^t(\phi) \qquad (3.38)$$
$$D_s^t(\phi) = I^t(\phi) - B_s^t(\phi)$$

and we define variable $q \in \{p, s\}$ as the index with smaller difference D^t and $\bar{q}$ as the remaining index. We allow for some process to label the pixel ϕ as being in the target set T or in the non-target set N. We update the background as

$$B_q^{t+1}(\phi) = \begin{cases} [1-\alpha']B_q^t(\phi) + a'I^t(\phi) & \phi \in \mathrm{T}^t \\ [1-a]B_q^t(\phi) + aI^t(\phi) & \phi \in \mathrm{N}^t \end{cases} \qquad (3.39)$$

where a' smaller than a. In our algorithm we used as $a = 0.2$ and $a' = 0.02$. The other background model is not updated,

$$B_{\bar{q}}^{t-1}(\phi) = B_{\bar{q}}^t(\phi) \qquad (3.40)$$

The motivation of equation 3.39 is to support temporal changes in lighting. Furthermore the blending of a moving target with the background process produces a 'beneficial ghost' of the target's path. The use of $a' < a$ allows the system to more slowly adapt in target regions, limiting how quickly a target will be blended with the background.

Lots does not update the background images every frame. It is updated every 64 frames and it reduces the cost. If the background updated each frame, it became the most computationally expensive component of the system, larger than the operations of subtraction and thresholding.

II. Grouping: Quasi-Connected Components (QCC)

After change detection is applied, most systems form regions by collecting connected pixels. Many systems augment their connected components with morphological processing.

In this section is presented an approach which combines **grouping with the thresholding** into a process called quasi-connected components (QCC).

A main problem for any pixel-level change detection technique is the setting of the threshold for deciding what a significant change is. If one chooses a high threshold, to maintain a small false alarm then the miss detection rate is increased. On the other hand, the lower threshold needed for low miss detection rate and a high false alarm rate. In our algorithm we use *thresholding-with-hysteresis* (TWH). The idea is to have two thresholds, a high threshold (T_h) and low threshold (T_L). Regions are defined by connected components pixels above the low threshold where the region also contains a given fraction of its pixels above the high threshold. TWH fills gaps between high-confidence regions in a more meaningful way. A problem is that with a low threshold near zero, gaps will occur because parts of targets can match the background exactly. A technique that can fill across small gaps is the *quasi-connected components* that combine TWH with gap filling and connected component labeling. The process insures that each pixel in a quasi-connected region is “connected” to a given number of pixels above the high threshold, even if the pixel is within a gap.

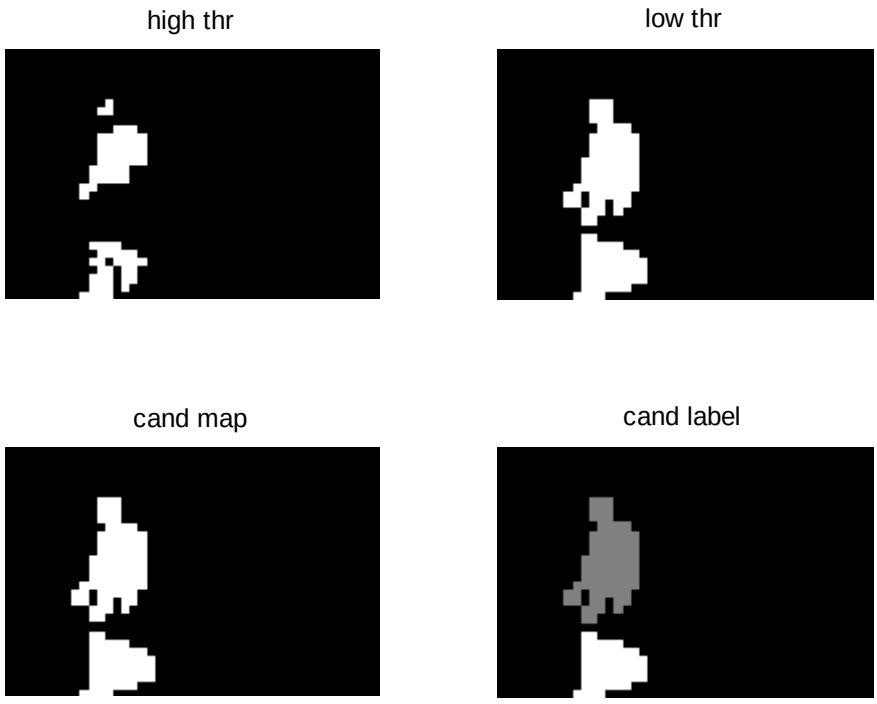


Figure 3.2 Example showing the high threshold image, the low threshold image, the candidate map and the candidate labeling of regions

In the figure 3.2 is illustrated the images that extracted using Lots algorithm. In the left upon image is illustrated the image that contains pixels above the high threshold value, the right upon image is created by the pixels that are above the low threshold. In the bottom left image is illustrated the blended image from the merge of low threshold and high threshold images. In the bottom right image we present the candidate labeling of regions that corresponds to moving objects.

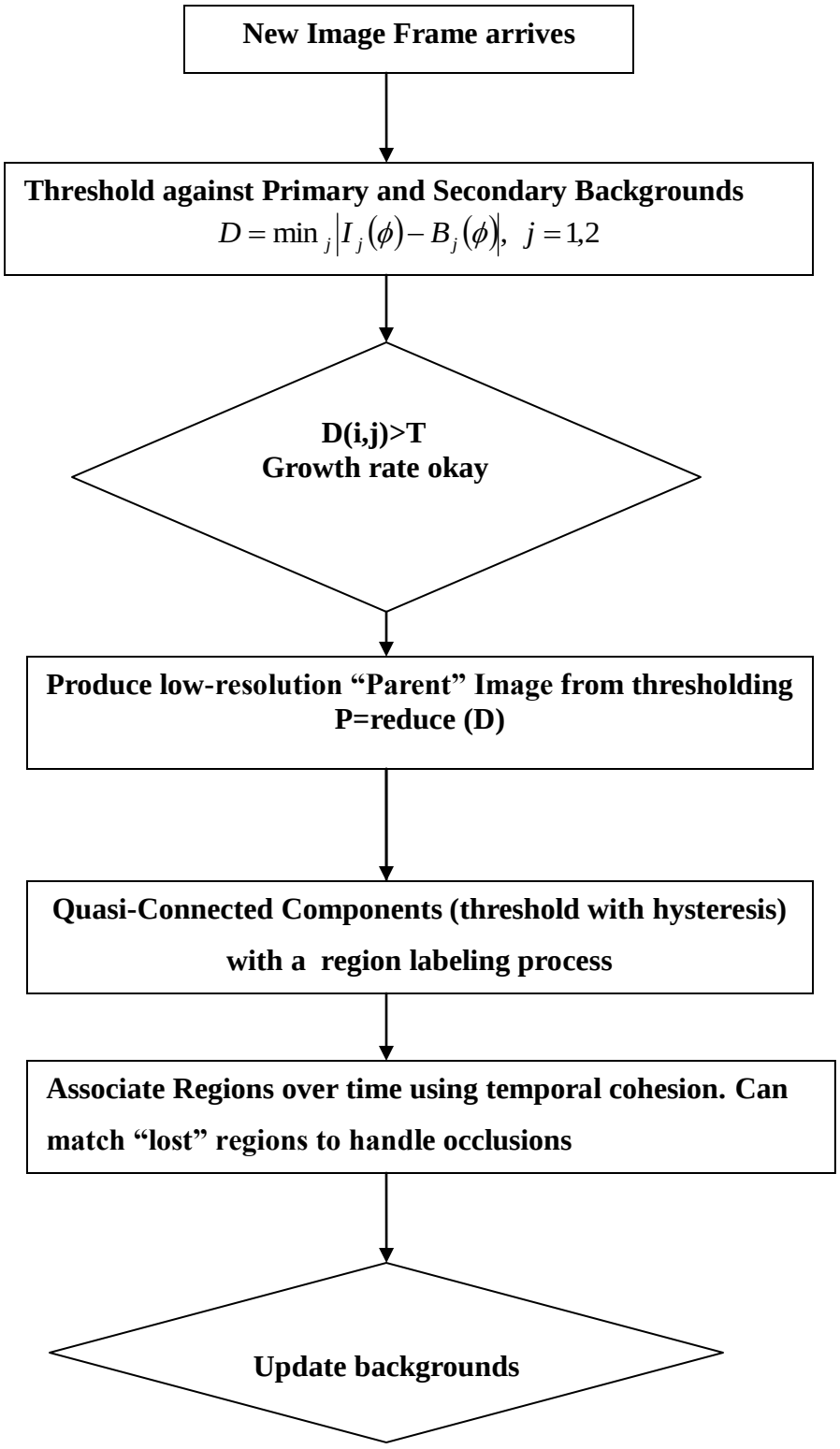


Figure.3.3 Lots Algorithm

The quasi-connected algorithm gathers information about the number of change pixels above the high threshold in an image block of the difference image and stores it as an image value in a lower resolution image on which connected component analysis is performed. An example of this is illustrated in Figure 3.4, where Figure 3.4a represents the high/low threshold image where H and L denote the high and low threshold pixels found in the difference image respectively. In this example, the parent image represents the downsampling of the original image by a factor of 2 in both the horizontal and vertical directions. The numbers shown in the image of Figure 3.4b represents the number of high threshold pixels detected in each 2x2 image block of Figure 3.4a and likewise for Figure 3.4c except it represents the number of low threshold pixels in each image block. Connected component analysis is performed on a parent image computed, given that Figure 3.4b represents P_H and Figure 3.4c represents P_L

$$P = P_H \cup P_L$$

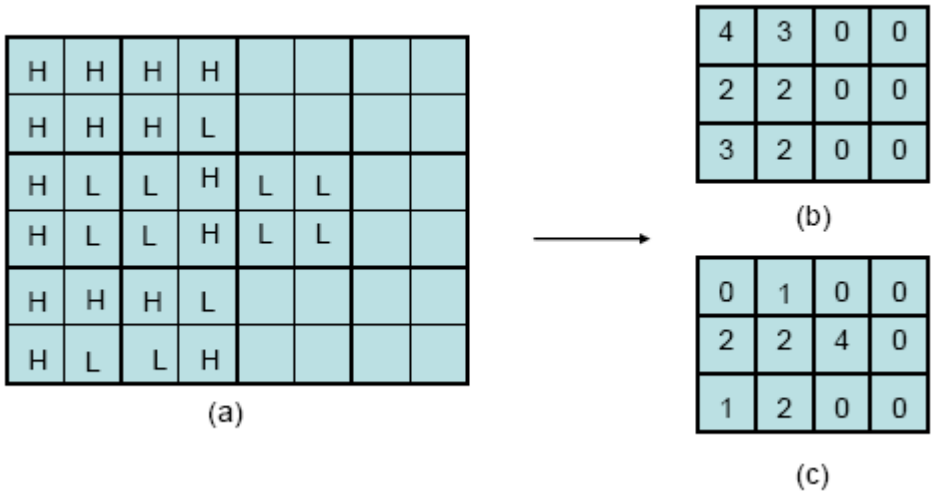


Figure 3.4 (a) Low-High Threshold Image (b) High Threshold Parent Image (c) Low Threshold Parent Image

QCC approach is presented in figure 3.5. During the detection phase, the system builds a lower resolution image of the pixels above threshold (the 24x24 image is compressed down to the smaller 6x6 image). This is called the parent image, where each parent pixel has multiple associated pixels that contribute to it. The value of each pixel in this parent image is a count of how many of its associated children (high resolution) pixels were above the low threshold and how many were above the high threshold. The count for exceeding the low threshold is in the low order word; the count for exceeding the high threshold is in the high order word. Since the resolution is reduced by a factor of four in each direction. The low order and high order words of the parent image contain values between zero and sixteen.

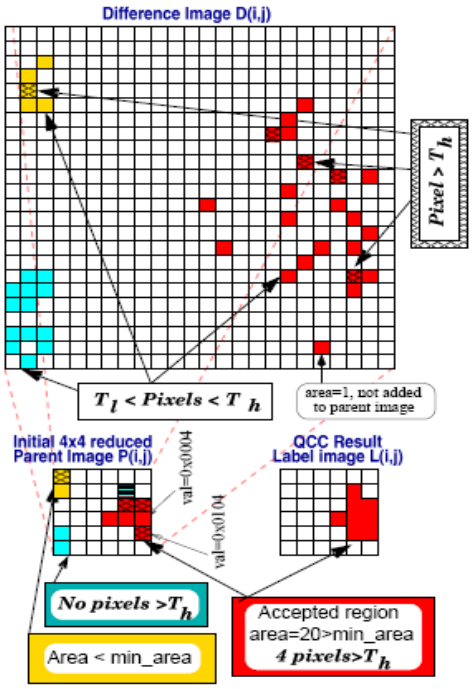


Figure 3.5 Example showing thresholding-with-hysteresis, quasi-connected components and area thresholding processing [5].

In figure 3.5 the shaded pixels are above the low threshold and the pixels that are both shaded and patterned are above low and high thresholds. Connected components are not computed in the high resolution image but only in the low-resolution image. A low resolution image pixel with a count of one is ignored when forming the parent image. The setting of low threshold is the sum of the dynamic threshold procedure and the global threshold that adjusted by the user. The high threshold is currently set at a constant either 4 higher than the low threshold.

The early version of LOTS simply required a region to have at least one pixel above the high threshold. Because the probability of some noise pixels being above the high threshold increases with the number of pixels in the regions, we changed the system to have the number of pixels required to be above high threshold increase to ceil (1/128 A), where A is the high resolution area of a region.

3.3.2 Parameterization for the experiments

We use median image of the entire sequence as Primary Background Image and we initialize the Secondary Background Image as *Secondary Background Image= Primary Background Image*. In our work, without loss of generality, we presume the input at time t-1 was closest to the primary background model B_p. For the thresholds we use for low-threshold T_L=0.1 and for high-threshold T_H=0.4). The detection and labelling step is performed on every frame after the initialization. In our work in the experiments, we used $a = 0.2$ and $a' = \frac{a}{4}$ as proposed in [5]. In our experiments, we used images of 320x240 pixels.

3.4 Object Tracking Method

We used an object level tracking algorithm in our system. We track objects as a whole from frame to frame. The information extracted by this level of tracking is adequate for most of the smart surveillance applications [2]. Our method makes use of the object features such size, centre of mass, bounding box which are extracted in previous steps to establish a matching between objects in consecutive frames.

3.4.1 Correspondence-based object matching

The first step on our object tracking algorithm is matching the objects (O_p's) in previous image (I_{n-1}) to the new objects (O_i's) detected in current image (I_n).

For each previous object, O_p, we iterate over new objects and first check whether a new object O_i in the new objects list is close to O_p or not. The criterion for closeness is defined as the distance between the center of mass points of these two objects (O_p and O_i) being smaller than a pre-defined constant. This check is inspired by the fact that the displacement of an object between consecutive images should be small. We consider that two objects with center of mass points c_p and c_i are close to each other if the following is satisfied:

$$Dist(c_p, c_i) < \tau \qquad [3.41]$$

where Dist() function is defines as the Euclidean distance between two points, which is:

$$Dist(c_p, c_i) = \sqrt{\left(x_{C_p} - x_{C_i}\right)^2 + \left(y_{C_p} - y_{C_i}\right)^2} \quad [3.42]$$

Since every two objects that are close to each other within a threshold are not necessarily a successful match, in the next step we check the similarity of these two objects to improve correct matching. The criterion for similarity comparison is the size ratio of the objects. This check is motivated by the fact that objects do not grow or shrink too much between consecutive frames. Thus, two objects are classified as similar if they satisfy the following:

$$\frac{s_p}{s_i} < \mu \text{ or } \frac{s_i}{s_p} < \mu \quad [3.43]$$

where s_i is the size of object O_i and μ is a pre-defined threshold. Checking the objects for size is especially useful if an object in the previous frame splits into a large and a very small region due to inaccurate segmentation. This check eliminates the chance of matching a big region to a small region.

After the second step we check further whether the object O_p has already a match/correspondence or not. If the object O_p has a prior-correspondence O_k, we perform additional steps to resolve the correspondence conflict.

In resolving a matching conflict we compare the correspondences of objects O_i and O_k to O_p. By comparing the correspondence of O_i and O_p with the correspondence of O_k and O_p, we try to decide which one of O_i and O_k is the correct match to object O_p. The correspondences are compared by using the distance between the center of mass points of O_p and O_i, and let d_{pk} be the distance between center of mass points O_p and O_k. The correspondence is resolved in favor of O_k if $d_{pk}<d_{pi}$, otherwise resolution is in favor of O_i.

In establishing a matching between previous object and new objects five different match cases can occur.

I. One to one: The previous object O_p is matched with a single new object O_i. The features of O_p are updated with incoming information from O_i.
II. One to many: The previous object O_p is matched with more than one new object. The conflicting case is resolved by distance-based correspondence comparison and it reduces to case 1.
III. One to none: In this case the previous object O_p is not matched to any new object. This case occurs if an object disappears from the scene or if the object is occluded by other objects. In case of an occlusion, the object is preserved until the detection of the corresponding occlusion split. Otherwise, this object is deleted from the previous objects list.
IV. None to one: Here the new object O_i is not matched to any of the existing objects. This case occurs if a new object enters into the scene or occluded objects split.
V. Many to one: This is the case where new object O_i is matched with more than one previous object. This conflicting case is resolved by distance-based correspondence comparison and it reduces to case 1.

3.5 Shadow Detection and Elimination

The algorithms described above for motion detection perform well on indoor and outdoor environments and have been used for real-time surveillance for years. However, most of these algorithms are susceptible to both local (e.g. shadows and highlights) and global illumination changes (e.g. sun being covered/uncovered by clouds). Shadows cause the motion detection methods fail in segmenting only the moving objects.

Moving shadows need careful consideration in the development of robust dynamic scene analysis systems. Moving shadow detection is critical for accurate object detection in video streams since shadow points are often misclassified as object points, causing errors in segmentation and tracking. Many algorithms have been proposed in the literature that deals with shadows.

Detection and tracking of moving objects is at the core of many applications dealing with image sequences. One of the main challenges in these applications is identifying shadows which objects cast and which move along with them in the scene. Shadows cause serious problems while segmenting and extracting moving objects due to the misclassification of shadow points as foreground. Shadows can cause object merging, object shape distortion, and even object losses (due to the shadow cast over another object). The difficulties associated with shadow detection arise since shadows and objects share two important visual features. First, shadow points are detectable as foreground points since they typically differ significantly from the background. Second, shadows have the same motion as the objects casting them. For this reason, the shadow identification is critical both for still images and for image sequences (video) and has become an active research area, especially in the recent past. It should be noted that, while the main concepts utilized for shadow analysis in still and video images are similar, typically, the purpose behind shadow extraction is somewhat different. In the case of still images, shadows are often analyzed and exploited to infer geometric properties of the objects causing the shadow ("shape from shadow" approaches) as well as to enhance object localization and measurements.

Here we propose the shadow elimination method SEGB. In this method, moving foregrounds are first segmented from background using a background subtraction technique. For all moving pixels, our approach SEGB using gradient feature, detect shadow pixels. This method is based on the

observation that shadow regions present some textural characteristics in each frame of the video as in the corresponding adaptive background model. It is important that gradient feature is robust to illumination changes.

3.5.1 Shadow Elimination Based on Gradient Feature (SEGB)

In this method [72] we use gradient feature which can well represent texture information and is robust to illumination changes. Our approach is to get the gradient image of moving foreground and the relevant background. Gradient information of moving foreground includes gradient of moving vehicles and gradient of moving shadows. Gradient information of relevant background includes gradient of only background.

Gradient of moving vehicles is different to gradient of relevant background and gradient of moving shadows is similar to that of relevant background. As result the difference of the two gradient images will reserve more gradient information at the moving vehicles areas and remove most of the shadow gradient at shadow region. Figure 3.6 shows the simple four gradient operators.

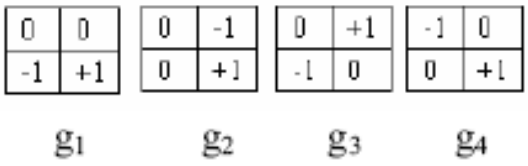


Figure 3.6 Gradient operators

The position meaning of each gradient operator is shown in table 3.1

(x-1,y-1)	(x,y-1)
(x-1,y)	(x,y)

Table 3.1 The four position of operator

The above operators consider the vertical, horizontal and diagonal edge. To calculate the gradient information we get the grey image of the result of moving foreground and then using the above four gradient operators, the gradient information of pixel at coordinate (x,y) be calculated by

$$G(x) = \begin{cases} 255, \ if \ \sum_{i=1}^{4} |g_i(x,y)| > 255 \\ \sum_{i=1}^{4} |g_i(x,y)| \end{cases} \tag{3.44}$$

The gradient image of moving foreground blobs and relevant background is calculated by formula 3.47. The difference image of the above two gradient images shows that most gradient of moving shadow has eliminated. Then binary the result image to remove noise.

For shadow elimination we have three main steps:

1. Finding and detecting the moving objects (moving foreground) in the frame.
2. Finding the relevant background of the moving foreground.
3. Applying the gradient on both foreground image and background image and the difference of both of them is a shadow less moving object.

In the first part of the algorithm, the detection of moving objects was done using Gaussian Mixture Model. Though there are few discrepancies in the detection of moving object it works quite well.

In the second part, the calculation of the background of moving object is required. We find out what is lying underneath and behind the moving object. We pick up the 1st picture from the input frames because it contains no moving object and we store this image in matrix RGB1. Than we start taking the output pictures and store them in matrix RGB2. The image stored

in RGB2 contains only the moving object and rest of the image is black. The black parts of the image have the quality if they are added to another colored image than the result will be colored image. Then we make the moving object black in RGB2 matrix and all of the rest part pf the image is than made white through frame processing. Next we add RGB1 and RGB2 and store it in another matrix which contains the background of the moving object and every other part is turned white. Finally we make the white part of the image to black through frame processing.

In the third part we find the gradient of moving foreground and the gradient of relevant background and we take the difference of the 2 gradient images. The result is the shadow eliminated frames. Figure 3.7 presents the flowchart of the code.

Main ( )

➔gmm ( )

➔segm_frame ( )

➔tracking ( )

➔shadow_elimination_main ( )

➔rbackgrnd ( )

➔shadow_elimination ( )

➔gradient ( )

Fig. 3.7 Flowchart of the shadow elimination code

Chapter 4: Implementation Issues

4.1 Test Application and System

We tested the computational performance and detection quality of two different object detection algorithms, Gaussian Mixture Model [1] and LOTS algorithm ([5],[7]). We used sample indoor and outdoor video clips. We used as input images 3 different video sequences on both algorithms. We used two video sequences (Dtneu_winter.avi and Norwayhighway.avi) from highways with more than one moving object. Besides we tested our algorithms at a video sequence with less complex background; the moving object is one or two men and the background is simpler. We implemented the Gaussian Mixture Model and the LOTS algorithm using Matlab 2007b, RADtools for video analysis on Microsoft Windows XP professional operating system on a computer Intel core Duo and 2048 MB of RAM.

4.2 Object Detection and Tracking

We tested the computational performance and detection quality of two different object detection algorithms adaptive background mixture models [1] and LOTS (Lehigh Omnidirectional Tracking System) ([5],[7]). The time performance analysis, which is the per-frame processing time of these algorithms for an image size of 320x240 pixels, is shown in Table 4.1

Detection Algorithm	Average time to process a frame
Adaptive Background Mixture Model	12 msec
LOTS algorithm	15 msec

Table 4.1 Performance of object detection algorithms

4.3 Moving Foreground Detection using Gaussian Mixture Model

Figure 4.1 shows the flow chart of the Gaussian mixture model process. Five different video sequences tested using Gaussian Mixture Model and the results are illustrated in Figures 4.2 to 4.6. Figures 4.2 and 4.3 are being taken from two different highways and the parameters that have been used are illustrated in table 4.2. We used video sequences of 100 image frames for each one. We commented that we get satisfactory results.

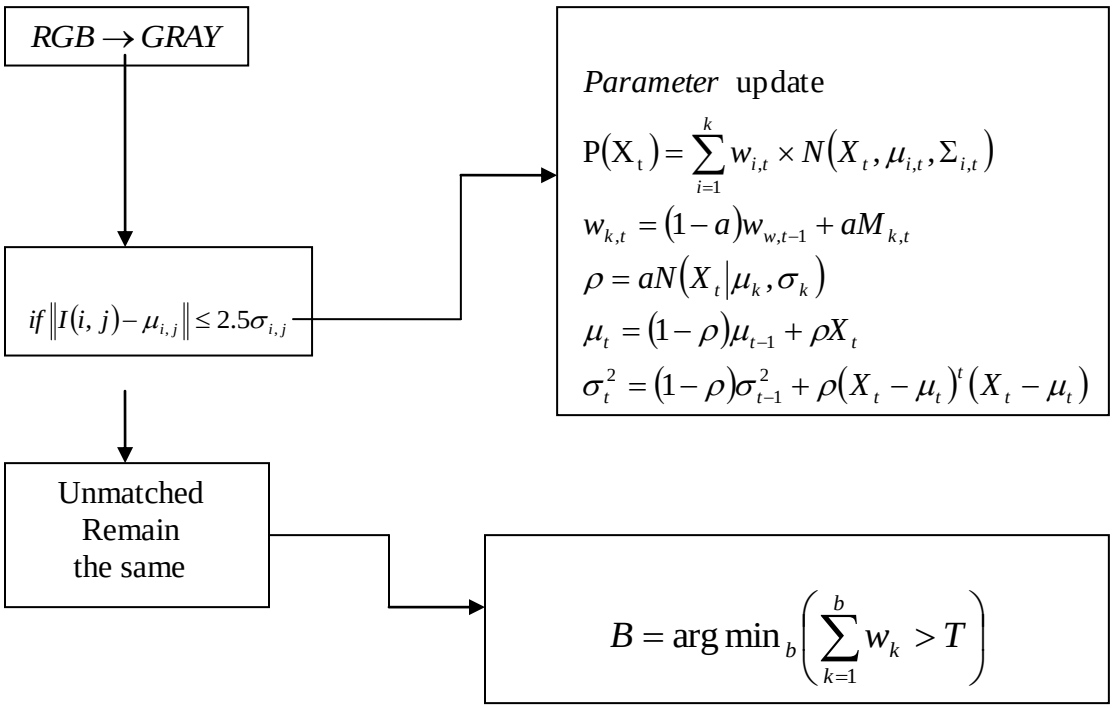
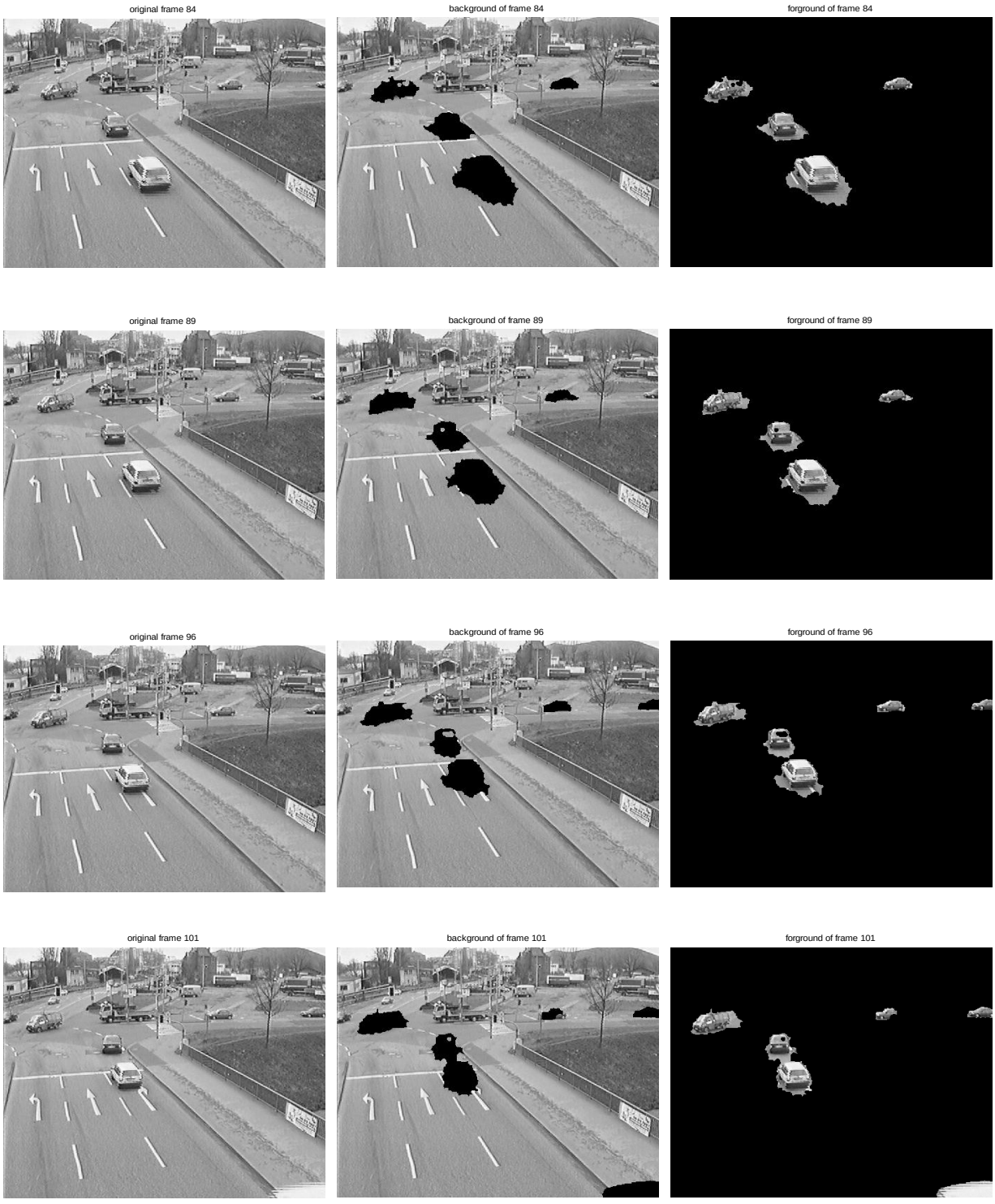
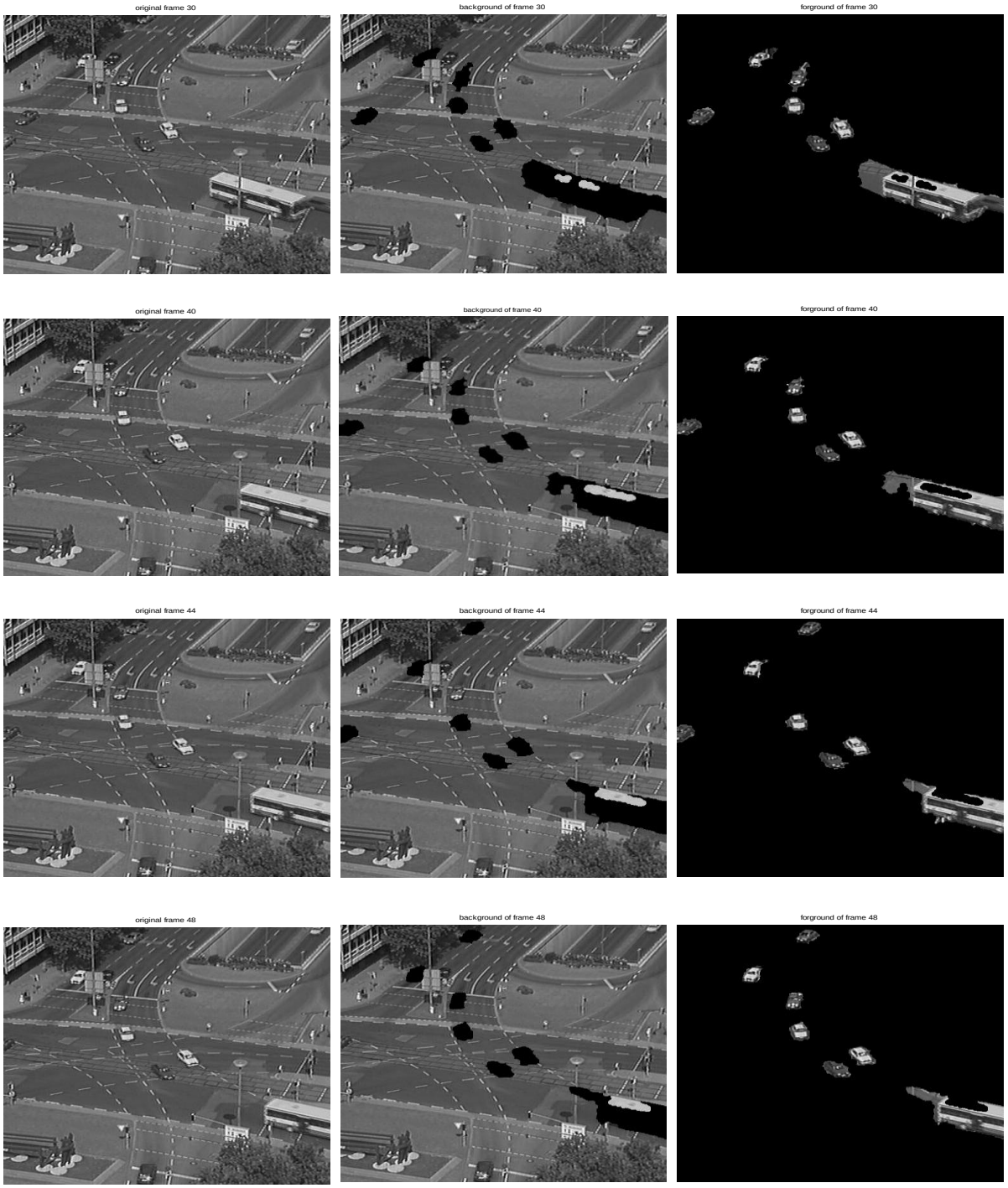


Figure 4.1 The flow chart of the Gaussian mixture model process



Original **Background** **Moving Foreground**

Figure 4.2 Results obtained when running the complete algorithm on a video sequence recorded in Norway. The parameters used are given in table 4.2



Original **Background** **Moving Foreground**

Figure 4.3 Results obtained when running the complete algorithm on a video sequence recorded in Norway (Norway-highway.avi). The parameters were used are given in table 4.2

The algorithm detects the most moving objects. Without dispute, there are objects that have not been detected and the reason is for these objects there is not motion during image sequences. Furthermore, we comment that in many of the moving objects there are shadows. Moreover, it is very important the fact that the algorithm detects greatly the moving objects that are appeared in the scene after some frames and we can notice it in figures 4.2 and 4.3

In figure 4.3, we used a video sequence recorded at an intersection in Norway with more than 10 moving objects as we can see from the input image (original image). The current frame is shown to the left, the background frame is shown at the middle and the segmented moving foreground is shown to the right. A complete list of the parameters used is given in table 4.2.

In figures 4.4 and 4.5 we make a comparison among the same image frames using different parameters at each occasion. In figure 4.4 we use as learning rate $\alpha=0.01$ and $K=4$ and in figure 4.5 we use as learning rate $\alpha=0.9$, and the same number of Gaussian components, $K=4$. Comparing the correspondences image frames from figures 4.4-4.5 we notice that in figure 4.5 the algorithm detects more moving objects with less shadow. The disadvantage is that in some occasions the algorithm does not detect the whole moving object but a part of it.

Lastly, in figure 4.6 (Twomen.avi) we tested the Gaussian Mixture Model at a video sequences with less complex background. There are only two moving objects with a stationary background. The results are adequately but we notice that in the first frames there is noise but at the last frames we take clearly the moving objects and without shadows.

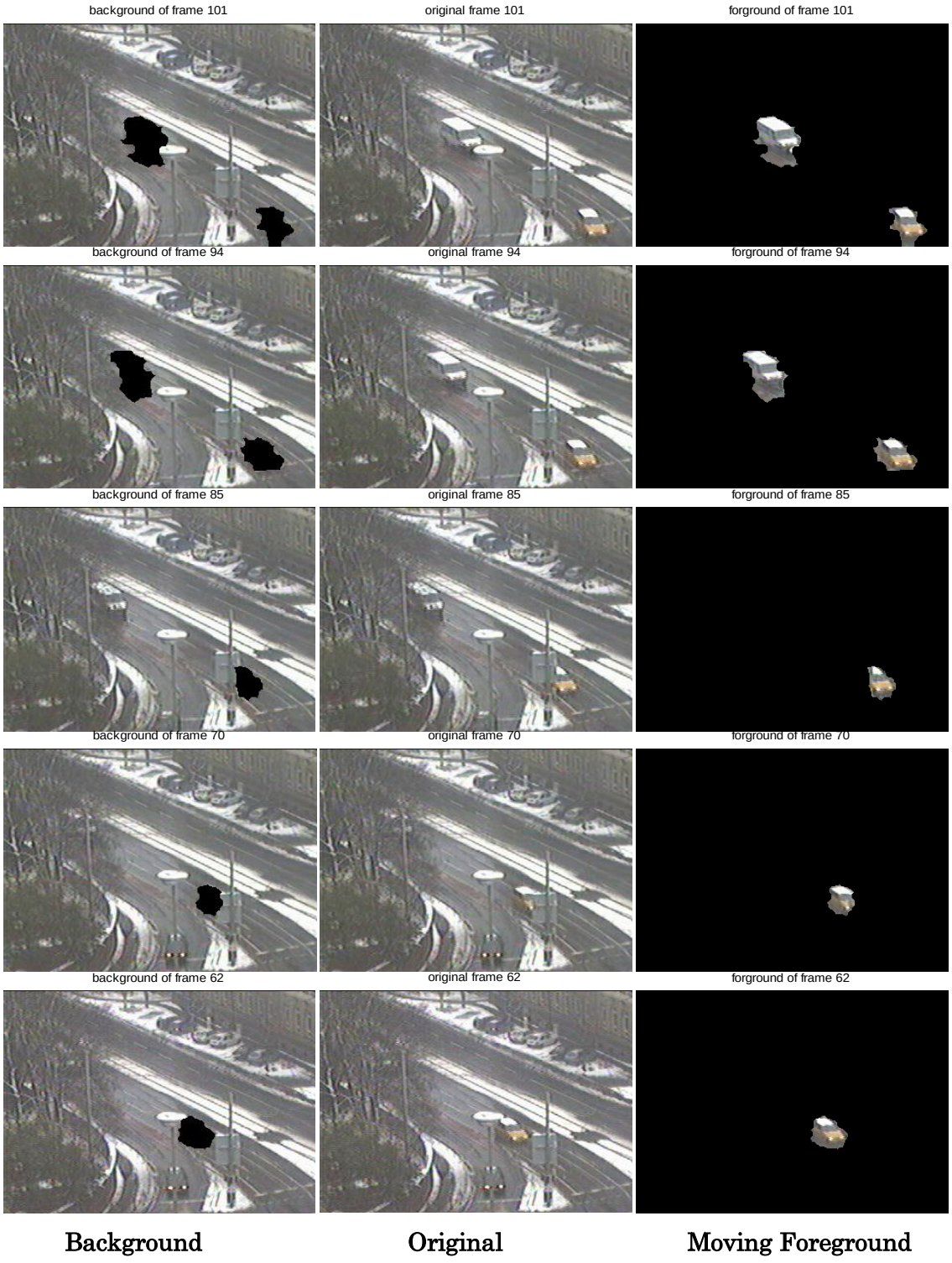


Fig.4.4 Dtneu_Winter avi **α=0.01**, K=4 From Left to right: Background-Original.-Moving Foreground (dtneu_winter.avi)

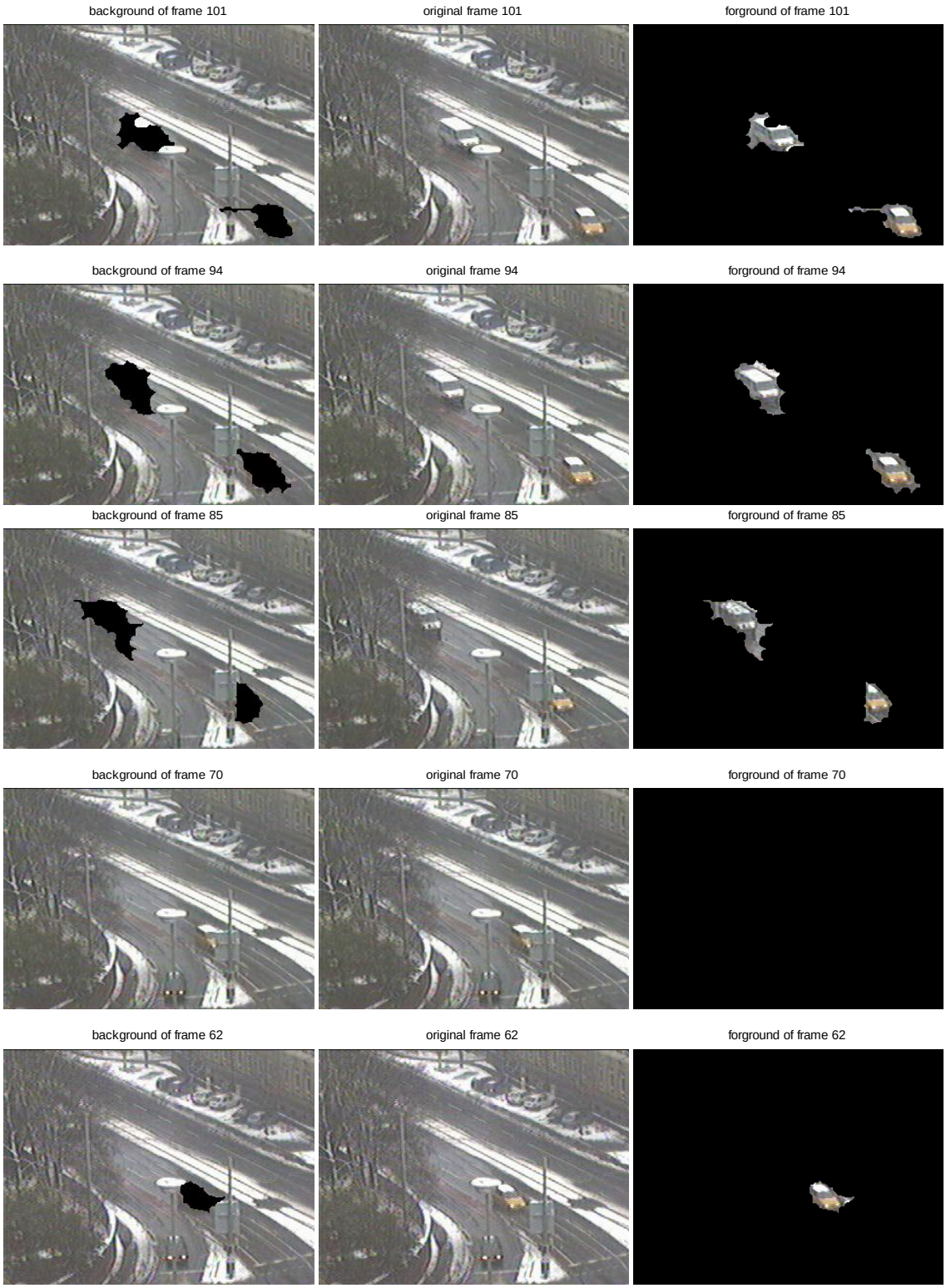


Fig. 4.5 Dtneu_Winter avi **α=0.9**, K=4 From Left to right: Background-Original-Moving Foreground (dtneu_winter.avi)

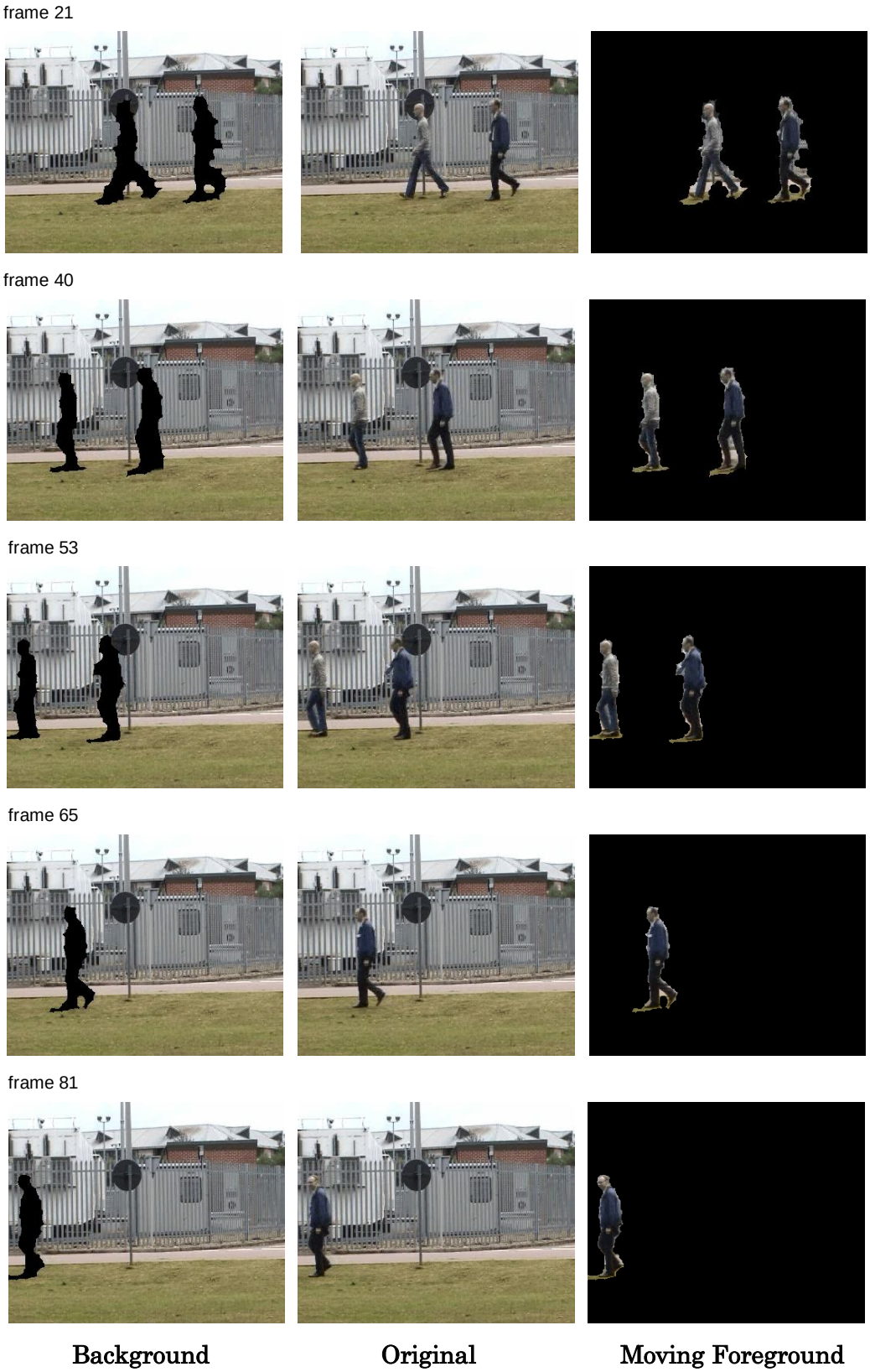
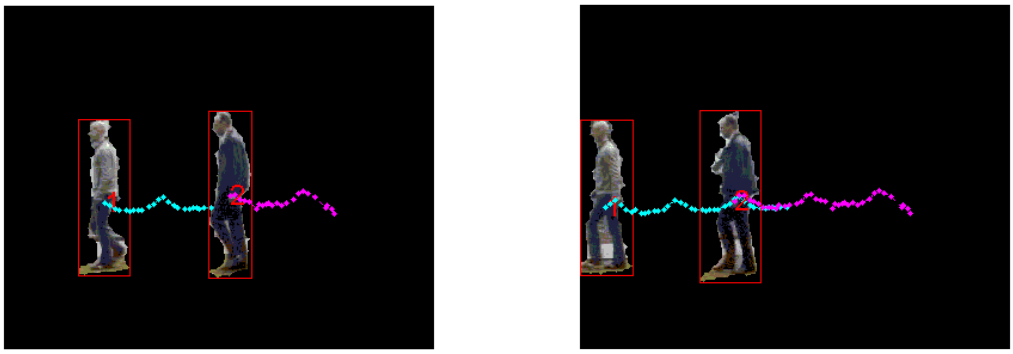
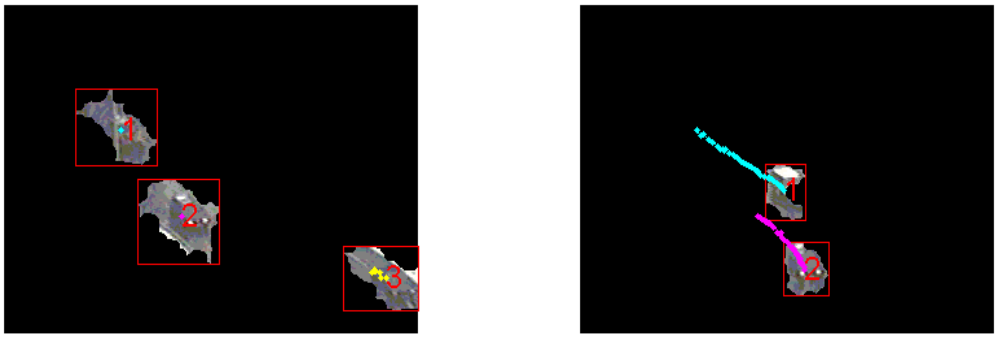


Fig.4.6 Twomen avi **α=0.9**, **K=4** From Left to right: Background-Original-Foreground (Twomen.avi)



(a) Twomen.avi

(b) Oneman.avi



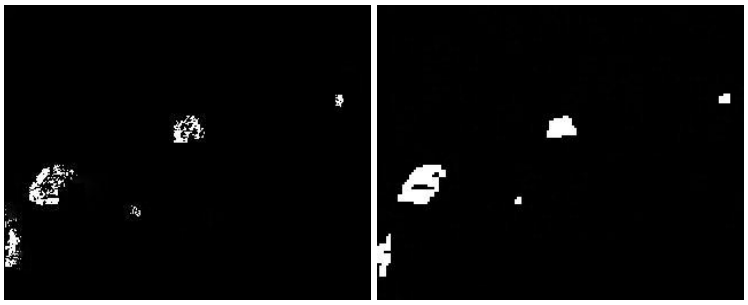
(c) dtneu_winter.avi

Fig 4.7 Object Tracking Results for Gaussian Mixture Model for (a) TwoMen avi, (b) OneMan.avi (c) Dtneu_Winter.avi

4.3.1 Shadow Elimination Based on Gradient Feature (SEBG)

After moving foreground detection using Gaussian Mixture Model we apply a shadow elimination approach based on gradient feature as we explained above. Figure 4.8 shows an example of shadow elimination using gradient feature.

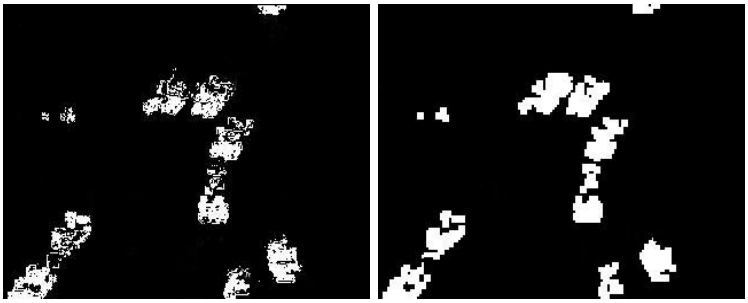
(a) Input image (b) Segmented image



(c) Gradient image (d) Segmented image

Fig.4.8 Results of shadow elimination using SEGB

(a) Input image (b) Segmented image



(c) Gradient image (d) Segmented image

Fig.4.9 Results of shadow elimination using SEGB (Norway-highway.avi)

Figures 4.8 and 4.9 shows the results of our method at two different frames of two different video sequences. Figure (a) is the input image after background subtraction process, figure (b) is the moving object after moving object detection process, figure (c) is the gradient image of moving object and (d) is the segmented image of gradient image.

4.3.2 Results and Discussion

We implemented the method reported in [1], "a statistical adaptive Gaussian Mixture model for background subtraction". We only chose the distribution with the highest weight ω/σ as the background pixel value, instead of using the T criterion before mentioned. In the experiment the variables are parameters α and K.

For the following tests, we used video sequences from European intersections in Norway as input to the complete algorithm. Table 4.2 shows the parameter value that used to generate figures 4.3 and 4.4.The current image is shown to the left, the background image is shown in the middle and the segmented foreground is shown to the right.

Background Modelling
K=4
α=0.001
λ=2.5
T=0.9
μ=0.01

Table 4.2 Parameter values used to generate figures 4.2 and 4.3

The advantage is that there are only two parameters that need to be defined in advance, and they do not need to be changed during sequence processing. Also it is a stable and robust method. It works very well for fast moving objects in complex environments. On the other hand there are many disadvantages using Gaussian Mixture Model.

1. The main disadvantage is that while an object is moving very slowly, it will be treated as part of the background, or just detected based on differences between the current frame and previous frames, and the overlapping regions of the moving object cannot be detected as foreground.
2. While testing a large moving object, holes left at the overlapping regions. This occurs because a slowly moving object has a small variance, which will match the background model, and as a result the slowly moving object was absorbed by the background.
3. The assign of initial values to these parameters (α, T, K) affects the accuracy of background subtraction.
4. In case that shadows are foreground, of the surface was covered by shadows a significant amount of the time, a Gaussian representing those pixels values may be significant enough to be considered background.
5. Furthermore, when an object enters the scene it is not well detected during a few frames since the Gaussian models have to adapt to this case.
6. Lastly, when a moving object stops, the MGM starts to split the region until it disappears, becoming part of the background.

It is very important for shadow elimination method the process of moving object detection. We used Gaussian Mixture Model for moving object detection and our algorithm work well though there are few discrepancies. If our moving object detection algorithm had better results it is unexceptionably that we would get better results using shadow elimination method.

Furthermore is necessary to say that we can get better results for moving objects with more edge features. If moving objects have less edge information this method will not effective.

4.4 Moving Foreground Detection using Lehigh Omnidirectional Tracking System (LOTS)

4.4.1 Results and discussion

This algorithm ([5],[7]) operates on grey scale images. It uses two background images and two per-pixel thresholds. The two backgrounds model periodic changes. The per-pixel threshold image can treat each pixel differently, allowing the detector to be robust to localized noise in low-size image regions. The per-pixel threshold evolves according to a pixel label provided by a Quasi Connected Components analysis (QCC).

Three test sequences were used for this study. Each video sequences has 100 frames and 100 images were used to build the background model. The resolution for each frame is 320 x 240 pixels and 24bit. In figure 4.11 we test a video sequence with two moving objects (Twomen.avi) using as T_L=0.1 and T_H=0.4 with satisfactory results. On the left of each figure is illustrated the original image, in the middle the moving object and on the right the background that we have used. In figures 4.11 and 4.12 we use image sequences from the *Oneman.avi* using different Low Threshold T_L in each image sequence. In the video sequence in figure 4.11 we use T_L=0.1 and in

figure 4.12 we use T_L=0.2 and we obtained better results using T_L=0.1. In the third video sequence were used again two different low threshold T_L, T_L=0.1 for video sequence in figure 4.14 and T_L=0.2 in figure 4.15 obtaining better results using again T_L=0.1. We observe that at each video sequence, it is very important to use the right values for low threshold and for high threshold for satisfactory results. It is important to say that for the sake of computational burden, LOTS does not update the background image in every single frame. In our algorithm we update the background every frame.

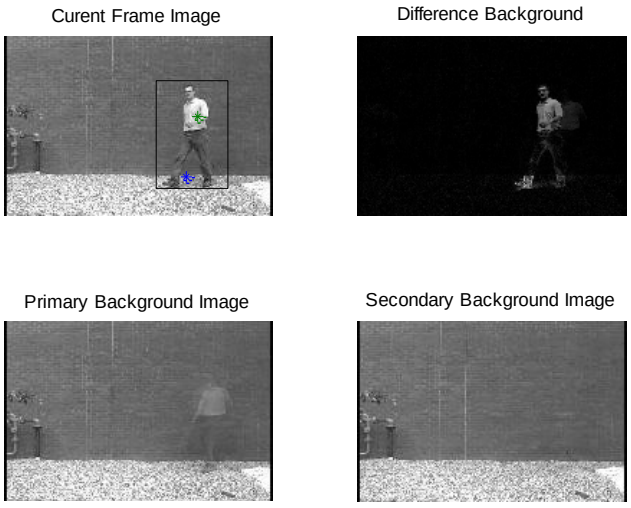
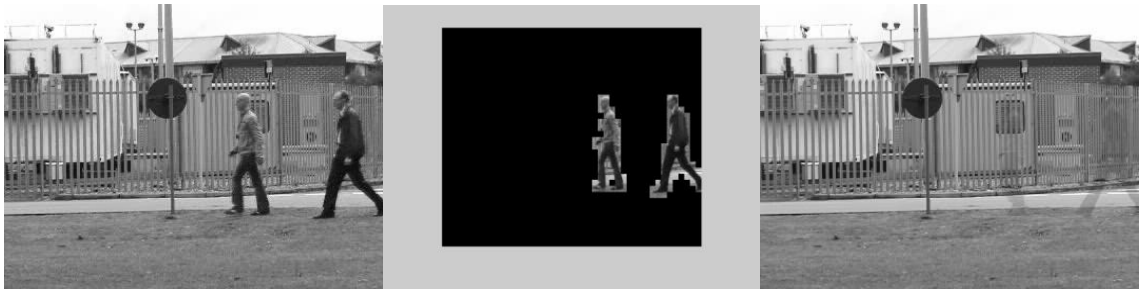


Figure 4.10 Output Frames of Lots algorithm

Figure 4.10 shows the output frames using Lots algorithm. The image in the left top row shows the current frame image with the moving object, the right top the difference image and the images in the bottom left and bottom right shows the two backgrounds, the primary background and the secondary background.



Frame 2 Original Moving Object Primary Background

Frame 10 Original Moving Object Primary Background

Frame 20 Original Moving Object Primary Background

Frame 30 Original Moving Object Primary Background

Figure 4.11 Results obtained when running the complete algorithm on a video sequence using T_L=0.1 and T_H=0.4 (Twomen.avi)

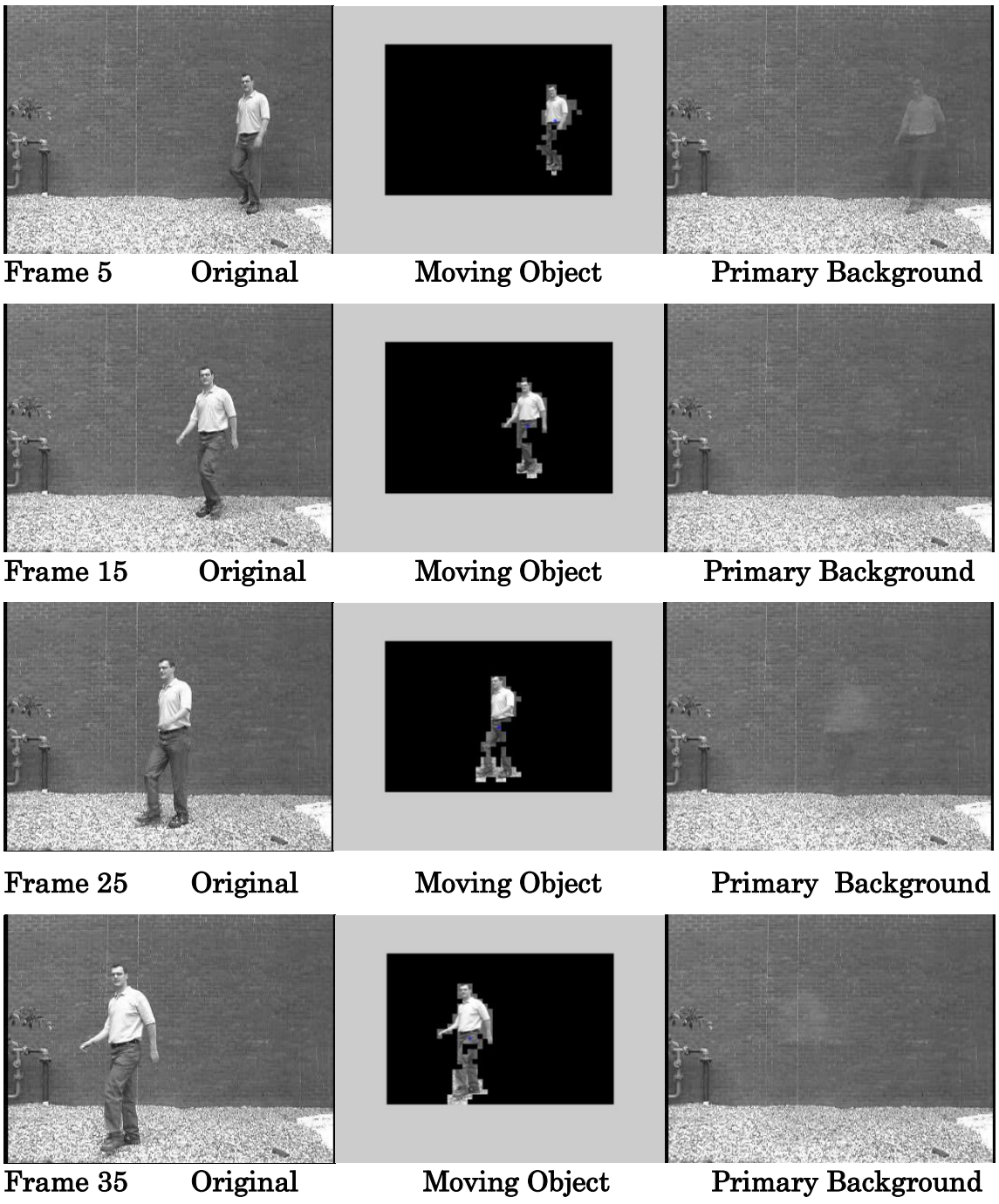


Figure 4.12 Results obtained when running the complete algorithm on a video sequence using T_L=0.1 and T_H=0.4 (Oneman.avi)

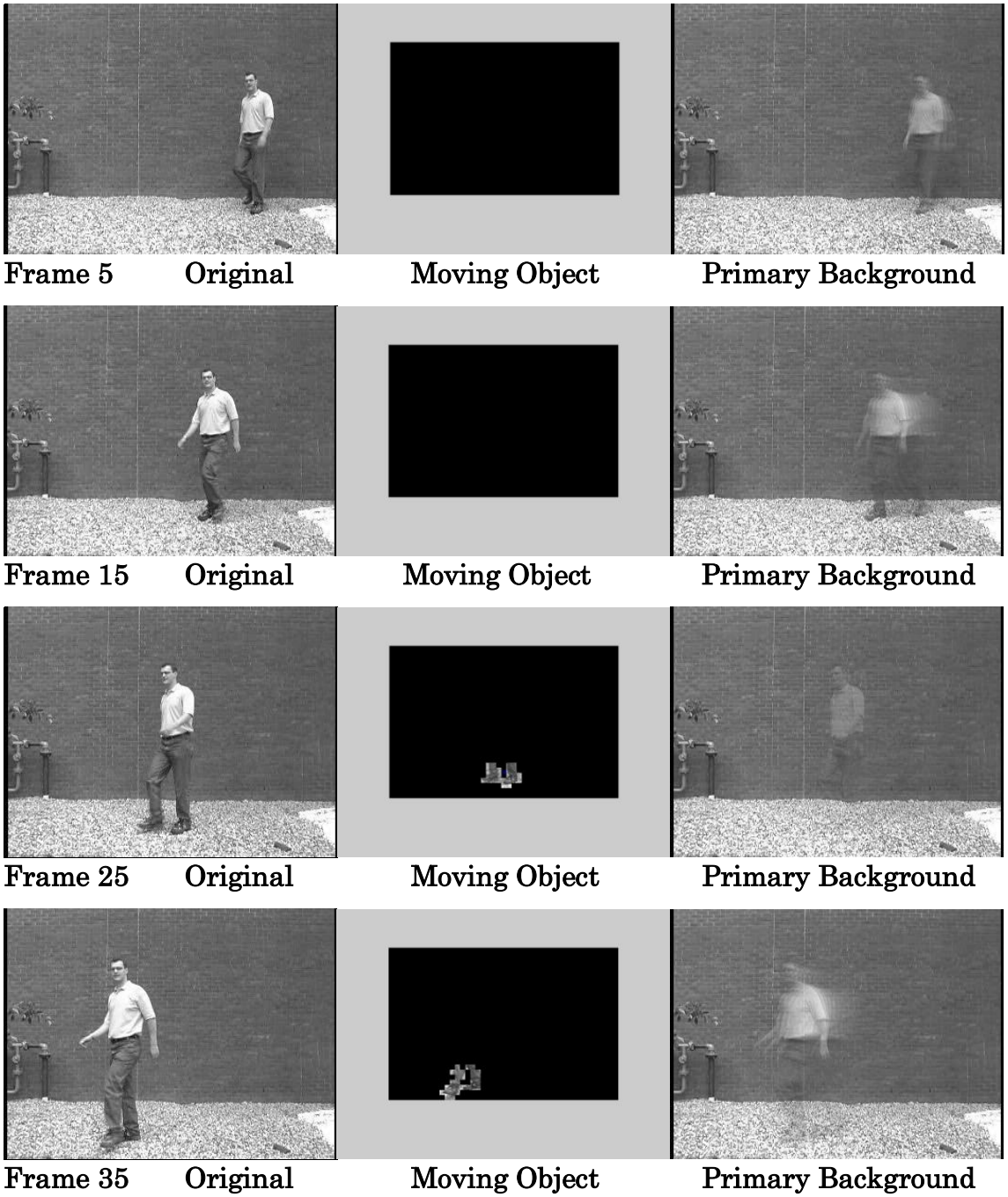


Figure 4.13 Results obtained when running the complete algorithm on a video sequence using T_L=0.2 and T_H=0.4 (Oneman.avi)

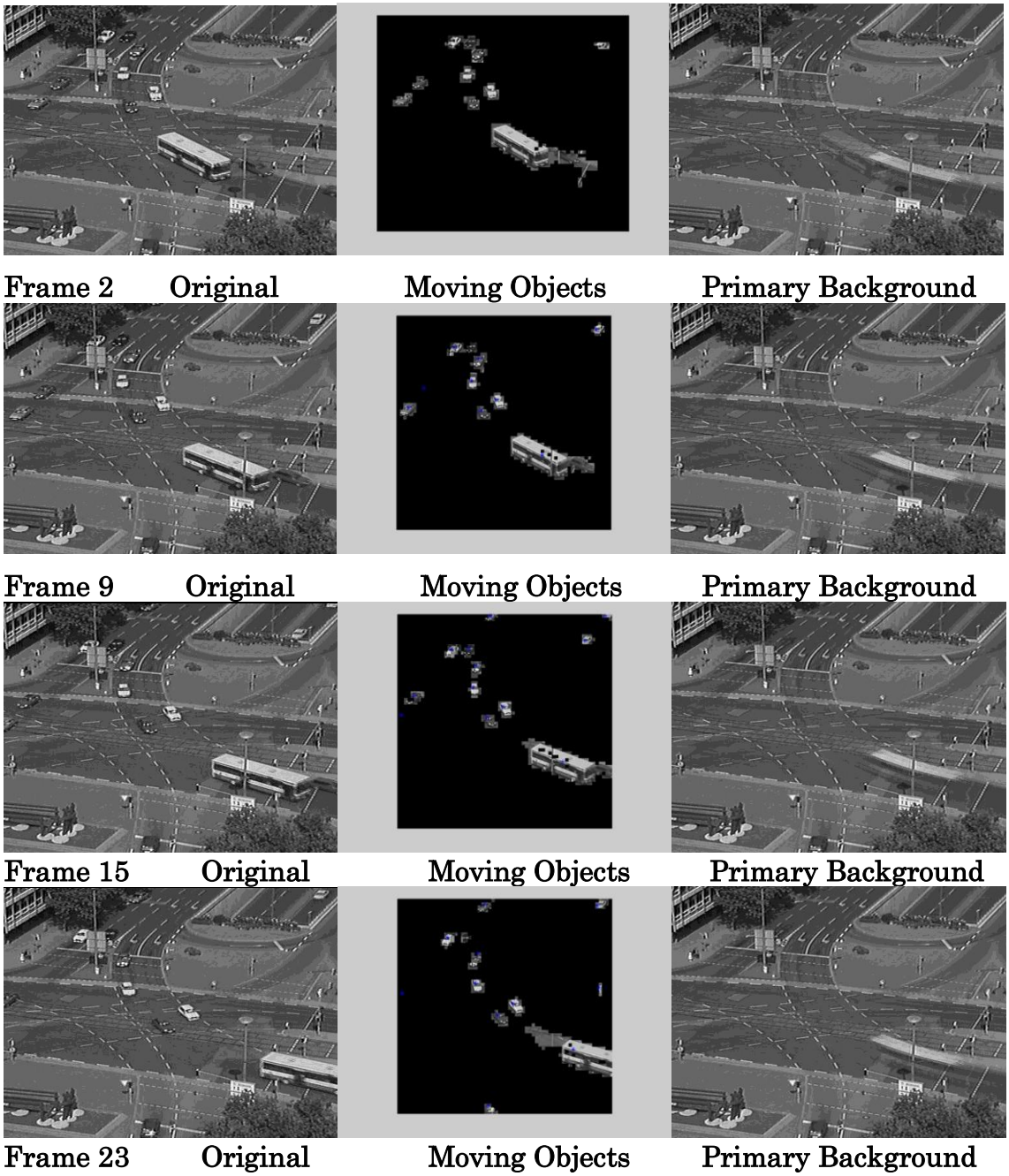


Figure 4.14 Results obtained when running the complete algorithm on a video sequence using T_L=0.1 and T_H=0.4 (Norway-highway.avi)

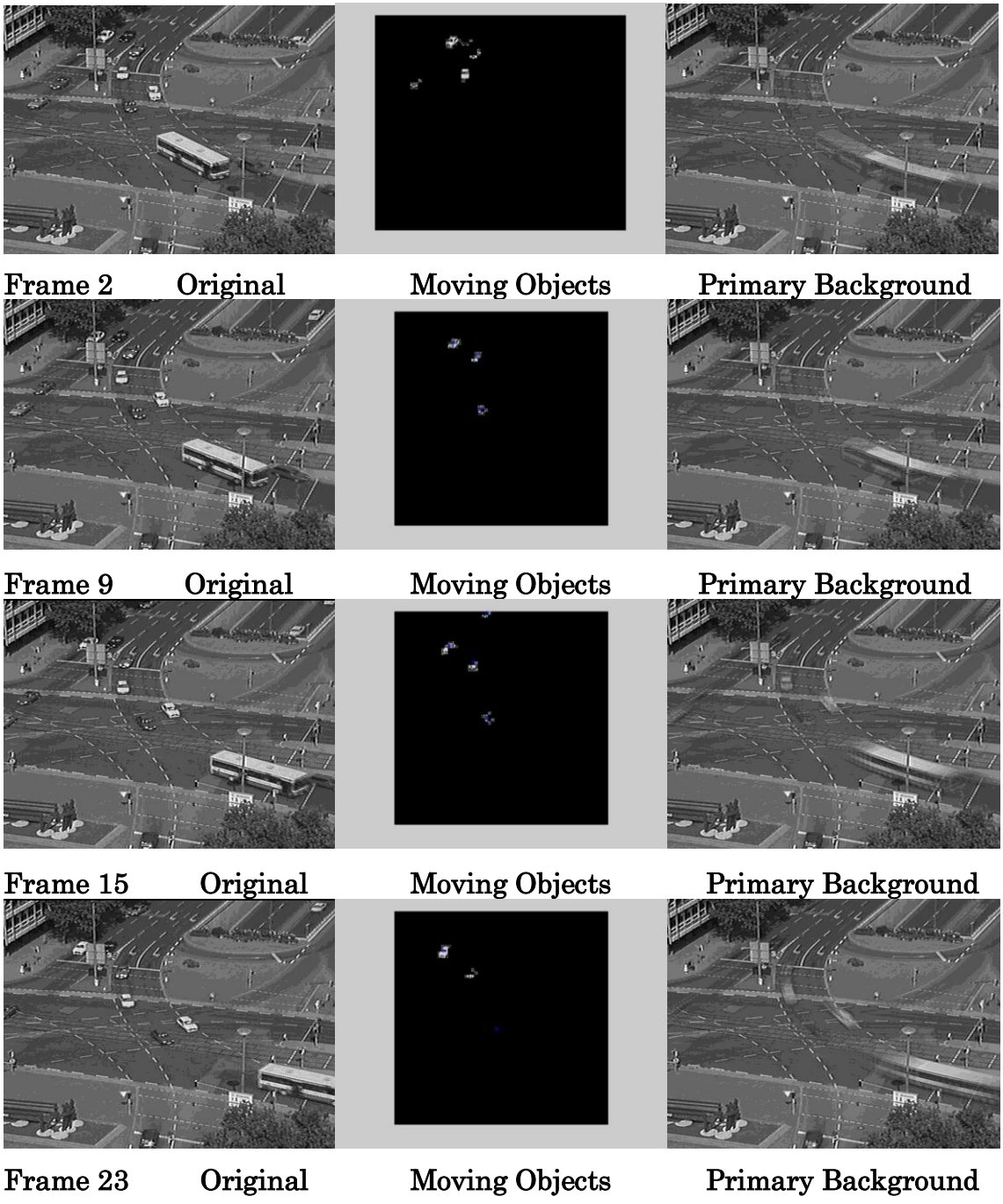


Figure 4.15 Results obtained when running the complete algorithm on a video sequence using T_L=0.2 and T_H=0.4 (Norway-highway.avi)

In figures 4.16, 4.17 and 4.18 the tracking system is assumed to be able to extract and label almost all moving objects corrected.

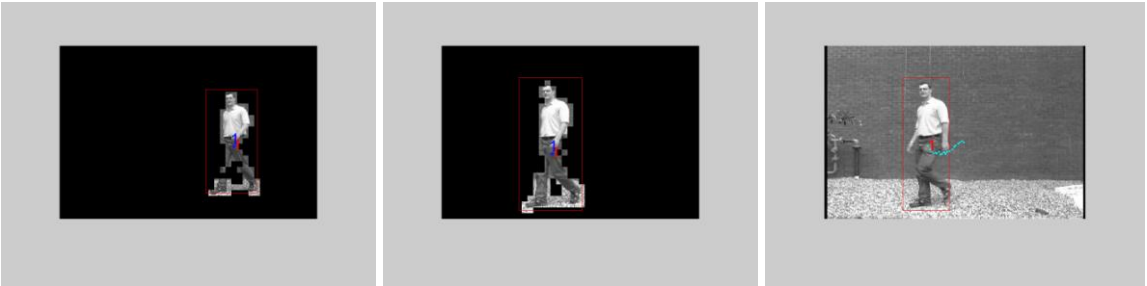


Figure 4.16 Object Tracking results for the 'One-Man' sequence using Lots approach

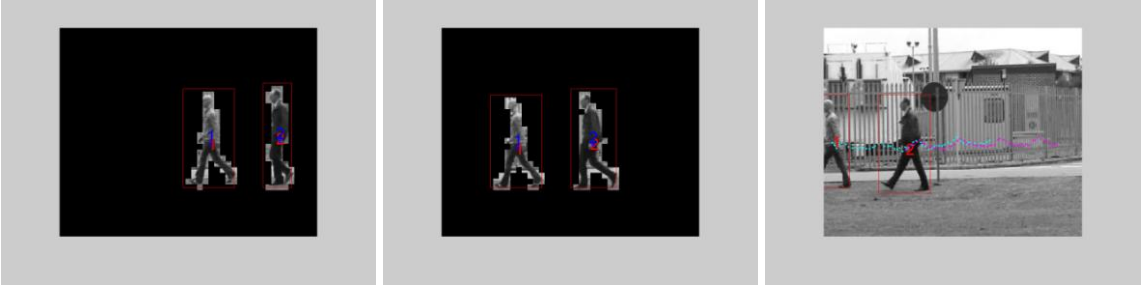


Figure 4.17 Object Tracking results for the 'Two-Men' sequence using Lots approach.

Figure 4.18 Object Tracking results for the 'dtneu_winter' sequence using Lots approach.

4.5 Comparisons

In figures 4.13-4.15 we used for LOTS tests as T_L=0.2 and as T_H=0.4 and we get worse results because the lower number of pixels that are above the low threshold value.

In figures 4.19-4.20-4.21 we compare the original frame and the results that are being obtained by moving object detections algorithms LOTS and GMM. We notice in figure 4.21 that using GMM we get better results than LOTS. The contours of moving objects are more exact using GMM than using LOTS because LOTS as part of the algorithm uses lower resolution image to detect object, so the contour won't be exact.

Fig.4.19 Orig. Image-Lots approach-Gaussian Mixture approach (Dtneu_winter.avi)

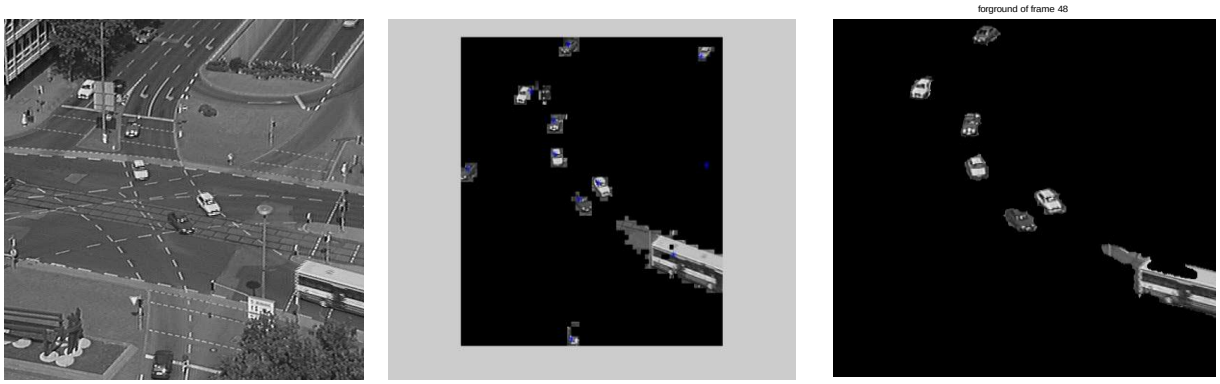


Fig.4.20 Orig. image-Lots approach-Gaussian Mix. approach (Norwayhighway.avi)

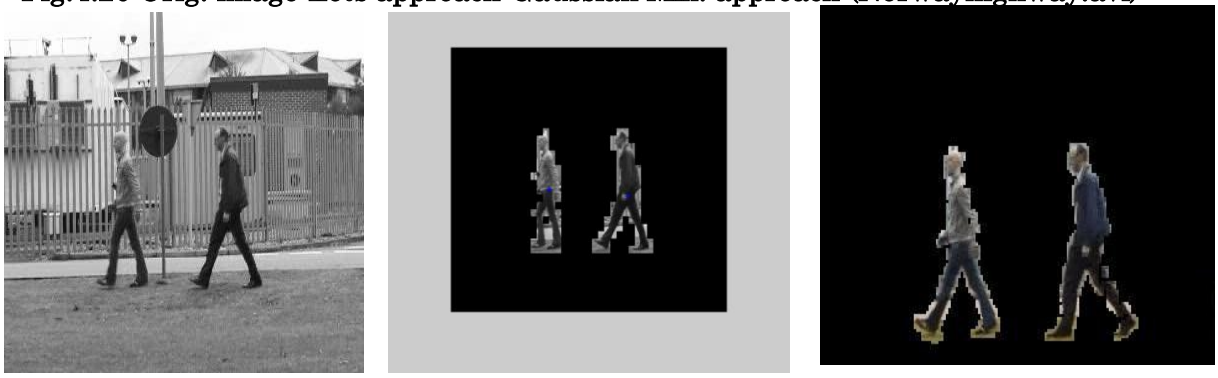


Fig.4.21 Orig. image-Lots approach-Gaussian Mix. approach (Twomen.avi)

We get better results using Gaussian Mixture Model than LOTS algorithm but we must mention that there are shadows both using Gaussian Mixture Model and Lots algorithm. For both algorithms, the parameters that we used is a very important point.

In the first image (Dtneu_winter.avi), we get better result using LOTS than Gaussian Mixture Model algorithm, and we notice that there are shadows on both moving objects algorithms. On the other two video sequences (Norway-highway.avi, TwoMen.avi), we get very good results.

A main point for both algorithms, LOTS and Gaussian Mixture Model, is the computational time. The average time to process a frame is about 12msec for Gaussian Mixture Model and about 15msec for Lots algorithm.

Comparing the Object Tracking results for LOTS and GMM algorithms, we assume that the results are satisfactory for both of them. Figure 4.15 illustrate the different positions of the man using a bounding box for LOTS algorithm, in figure 4.16 is illustrated the different positions for two moving objects (two men) using a bounding box and figure 4.17 illustrates the different positions for more than two moving objects (cars). Figure 4.7 (a) shows the object tracking results using Gaussian Mixture Model for Oneman.avi, 4.7(b) for Twomen.avi and 4.7(c) for Dtneu_winter.avi. Segmentation is a key step since it influences the performance of the other modules e.g., object tracking.

Our algorithm for object tracking has some problems. For example, when a new object arrives or the problem of the occlusion for some frames. Furthermore, the cross (merge) of two or more moving objects is one more important problem for both algorithms. Besides, one more problem with the current method for moving object tracking is the partial object occlusion. The moving object may be divided to 2-3 objects and it causes negatives results on finding the path for each moving object.

.

Chapter 5: Conclusions and future work

In this thesis we presented a set of methods for background modelling. We implemented two different object detection algorithms, Gaussian Mixture Model algorithm and LOTS algorithm. No object detection algorithm is perfect, so is our method. In short, the methods we presented for 'smart' visual surveillance show promising results and can be both used as part of a real-time surveillance system or utilized as a base for more advanced research such as activity analysis in video. Using mixture models provides a flexible and powerful method for background modelling and the shadow detection algorithm is helpful.

Beside the various contributions in the present thesis, the complete framework for intelligent video analysis is still non perfect. Many improvements could be introduced at several levels.

The generic framework for intelligent video analysis as presented in Chapter still is uncompleted. We have explored in our research some stages of this framework and not all the stages. The exploration of the other stages (action recognition, semantic description, personal identification and fusion of multiple cameras) makes the application range wider. Thus, we can consider more advanced applications based on fusion of multiple sensors as well as a recognition system for controlling high security areas.

Bibliography

[1] Chris Stauffer and W. Eric L. Grimson. *Adaptive background mixture models for real–time tracking*. In **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, pages II: 246–252, 1999.

[2] Richard A. Redner and Homer F. Walker. *Mixture densities, maximum likelihood and the EM algorithm.* **SIAM Review**, 26(2):195–239, 1984.

[3] Andrea Prati, Ivana Mikic, Mohan M. Trivedi, and Rita Cucchiara. *Detecting moving shadows: Formulation, algorithms and evaluation.* **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 25(7):918–923, 2003.

[4] I. Haritaoglu, D. Harwood, and L. S. Davis, "*W4: Who? when? where? what? a real time system for detecting and tracking people*," **in IEEE International Conference on Automatic Face and Gesture Recognition,** April 1998, pp. 222–227.

[5] T. Boult, R. Micheals, X. Gao, and M. Eckmann, "*Into the woods: Visual surveillance of non-cooperative camouflaged targets in complex outdoor settings*," **in Proceedings of the IEEE**, October 2001, pp. 1382–1402.

[6] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. **Prentice Hall**, 2002.

[7] T. Boult, R. Micheals, X. Gao, W. Y. P. Lewis, C. Power, and A. Erkan, "*Frame- rate omnidirectional surveillance and tracking of camouflaged and occluded targets*," **in Second IEEE International Workshop onVisual Surveillance**, 1999, pp. 48–55.

[8] I. Haritaoglu, R. Cutler, D. Harwood and L. S. Davis: Backpack*: Detection of people carrying objects using silhouettes in*: **International Conference on Computer Vision** (1999) pp. 102-107.

[9] K. Toyama, J. Krumm, B. Brumitt and B. Meyers: *Wallflower: Principlesand practice of background maintenance* in: **International Conference on Computer Vision** (1999) pp. 255-261.

[10] C. Wren, A. Azabayejani, T. Darrell and A. Pentland: Pfinder: *Real-time tracking of the human body* **IEEE Transactions on Pattern Analysis and Machine Intelligence**19 (1997) 780-785.

[11] Alan M. McIvor, *Background Subtraction Techniques*, **Reveal Ltd POBox 128-221,** Remuera, Auckland, New Zealand.

[12] Dariu M. Gavrila. *Vision–based 3–D tracking of humans in action*. **PhD thesis**, University of Maryland, 1996.

[13] Tat H. Nguyen, Marcel Worring, Rein van den Boomgaard, and Arnold W.M. Smeulders. *Tracking non-parameterized object contours in video.***IEEE Transactions on Image Processing**, 11(9):1081–1091, 2002.

[14] Weiming Hu, Tien-Niu Tan, Liang Wang, and Steven J. Maybank. *A survey on visual surveillance of object motion and behaviors.* **IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews,** 34(3):334– 352, 2004b.

[15] Adam M. Baumberg*. Learning Deformable Models for Tracking Human Motion*. **PhD thesis**, University of Leeds, 1995.

[16] Kemal B. Yesin and Bradley J. Nelson. *Robust cad model based visual tracking for 3d microassembly using image space potentials*. In **Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)**, pages II:1868–1873, 2004.

[17] Marie-Pierre Dubuisson and Anil K. Jain. *2d matching of 3d moving objects in color outdoor scenes.* In **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, pages 887–891, 1994.

[18] Warren F. Gardner and Daryl T. Lawton. *Interactive model-based vehicle tracking*. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 18(11):1115–1121, 1996.

[19] Holger Leuck and Hans-Hellmut Nagel *Model-based initialisation of vehicle tracking: Dependency on illumination.* **In Proceedings of the IEEE International Conference on Computer Vision (ICCV)**, pages I:309–314, 2001.

[20] Tie-Niu Tan, G.D. Sullivan, and Keith D. Baker. *Model-based localisation and recognition of road vehicles.* **International Journal on Computer Vision**, 27(1):5–25, 1998.

[21] Douglas Decarlo and Dimitris Metaxas. *Optical flow constraints on deformable models with applications to face tracking*. **International Journal on Computer Vision**, 38(2):99–127, 2000.

[22] J.A. Paterson and Andrew W. Fitzgibbon. *3d head tracking using non-linear optimization.* In **Proceedings of the Brittisch Machine Vision Conference.**

[23] Ye Zhang and Chandra Kambhamettu. *3d head tracking under partial occlusion*. **Pattern Recognition**, 35(7):1545–1557, 2002.

[24] Zoran Zivkovic and Ferdinand van der Heijden. *A stabilized adaptive appearance changes model for 3d head tracking*. In **Proceedings of theIEEE Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS)**, pages 175–181, 2001.

[25] Shan Lu, Dimitris N. Metaxas, Dimitris Samaras, and John Oliensis. *Using multiple cues for hand tracking and model refinement.* In **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, pages 443–450, 2003.

[26] Bjoern Stenger, Paulo R.S. Mendonc¸a, and Roberto Cipolla. *Model based 3D tracking of an articulated hand.* In **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, pagesII:310–315, 2001.

[27] G. Halevy and D.Weinshall: *Motion of disturbances: Detection and tracking of multibody non-rigid motion* **Machine Vision and Applications 11** (1999) 122-137, 1999.

[28] Maylor K. Leung and Yee-Hong Yang. First sight: *A human body outline labeling system*. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 17(4):359–377, 1995.

[29] Christopher R. Wren, Ali Azarbayejani, Trevor Darrell, and AlexPentland. *Pfinder: – time tracking of the human body.* **IEEE Transactions on Pattern** *Real* **Analysis and Machine Intelligence**, 19(7):780–785, 1997.

[30] A. Geurtz. *Model-based shape estimation*. **PhD thesis**, Department of. Electrical Engineering, Polytechnic Institute of Lausanne, 1993.

[31] Johnson I. Agbinya and David Rees. *Multi-object tracking in video*. **Real-Time Imaging**, 5 (5):295–304, 1999.

[32] Jacinto Nascimento and Jorge Marques. *Performance evaluation of object detection algorithms for video surveillance*, **Multimedia, IEEE Transactions on Volume 8, Issue 4,** Aug. 2006 Page(s):761 – 774, 2006.

[33] Christopher Rasmussen and Gregory D. Hager. *Probabilistic data association methods for tracking complex visual objects.* **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 23(6):560–576, 2001.

[34] M.B. Capellades, David S. Doermann, Daniel DeMenthon, and RamaChellappa. *An appearance based approach for human and object tracking*. In **Proceedings of the IEEE International Conference on Image Processing (ICIP)**, pages 85–88, Maryland University College, 2003.

[35] Stephen J. McKenna, Sumer Jabri, Zoran Duric, Azriel Rosenfeld, andHarry Wechsler. *Tracking groups of people*. **Computer Vision and Image Understanding**, 80(1):42–56, 2000.

[36] Yogesh Raja, Stephen J. McKenna, and Shaogang Gong. *Segmentation and tracking using color mixture models.* In **Asian Conference on Computer Vision (ACCV)**, pages 607– 614, 1998.

[37] Michael Isard and Andrew Blake. ICONDENSATION: *Unifying low–level and high–level tracking in a stochastic framework.* In **Proceedings of the European Conference on Computer Vision (ECCV)**, pages I:893–908, University of Oxford, Oxford, 1998.

[38] Qi Zang and Reinhard Klette, *Evaluation of an Adaptive Composite Gaussian Model in Video Surveillance*, pages 165-172, **Computer Science Department of the University of Auckland**, 2003.

[39] Michael J. Swain and Dana H. Ballard. *Indexing via color histograms*. In **Defense Advanced Research Projects Agency (DARPA)**, pages 623–630, 1990.

[40] Stephen J. McKenna, Yogesh Raja, and Shaogang Gong. *Object tracking using adaptive color mixture models.* In **Asian Conference on Computer Vision (ACCV)**, pages 615–622, 1998.

[41] Hwann-Tzong Chen, Tyng-Luh Liu, and Chiou-Shann Fuh. *Probabilistic tracking with adaptive feature selection*. In **Proceedings of the IEEE International Conference on Pattern recognition (ICPR)**, pages I:736–739, National Taiwan University, 2004.

[42] Katja Nummiaro, Esther Koller-Meier, and Luc J. Van Gool. *Color features for tracking nonrigid objects.* **Chinese Journal of Automation**, 29(3):345–355, 2003.

[43] Guillaume Gasser, Nathaniel Bird, Osama Masoud, and Nikolaos Papanikolopoulos. *Human activities monitoring at bus stops*. In **Proceedings of the IEEE International Conference on Pattern recognition (ICPR)**, pages I: 90–95, Dept. of Comput. Science. & Engineering, Minnesota University, Minneapolis, USA, 2004.

[44] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. *Real–time tracking of non– rigid objects using mean shift.* In **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, pages 142–151, 2000.

[45] Patrick P´ erez, Carine Hue, Jaco Vermaak, and Michel Gangnet. *Color–based probabilistic tracking.* In **Proceedings of the European Conference on Computer Vision (ECCV)**, pages I:661–675, Hilton Head Island, SC, USA, 2002.

[46] Rita Cucchiara, Costantino Grana, G. Tardini, and R. Vezzani. *Probabilistic people tracking for occlusion handling.* In **Proceedings of the IEEE International Conference on Pattern recognition (ICPR)**, pages I:132–135, Modena University, Italy, 2004.

[47] Ramprasard Polana and Randal C. Nelson. *Low level recognition of human motion*. In **Proceedings of the IEEE Workshop on Nonrigid and Articulated Motion (WNAM)**, pages 77–82, Austin, TX, USA, 1994.

[48] D. M. Garvila. *The Visual Analysis of Human Movement: A Survey*. **In Computer Vision and Image Understanding, Vol. 73, No. 1**, pp. 82-98, Ulm, Germany, January 1999.

[49] Paul J. Withagen, Klamer Schutte, Albert M. Vossepoel, and Marcel G.J. Breuers. *Automatic classification of ships from infrared (flir) images*. In **Proceedings of the SPIE International Symposium on Aerospace/Defense Sensing, Simulation, and Controls AeroSense**, volume 3720, pages 180–187, Orlando, 1999.

[50] B. Coifman, David J. Beymer, Phil McLauchlan, and Jitendra Malik. *Areal-time computer vision system for vehicle tracking and traffic surveillance*. **Transportation Research–Part C**, 6(4):271–288, 1998.

[51] J.K. Aggarwall and Q.Cai, *Human Motion Analysis: A Review*. **In Computer Vision and Image Understanding, Vol. 73**, No. 3, pp. 428-440, Dept. of Electr. & Comput. Eng., Texas Univ. March 1999.

[52] M. Turk, *Visual Interaction with Lifelike Characters*. **Proceedings of the Second International Conference** on Automatic Face and Gesture Recognition, Killington, VT, pp. 368-373, Killington, VT, USA, October 1996.

[53] B. Dorner, *Hand shape identification and tracking for sign language interpretation*. In Looking at People, **International Joint Conference on Artificial Intelligence,** Chambery, 1993.

[54] M. Betke, J. Gips and P. Fleming, *The Camera Mouse: Visual Tracking of Body Features to Provide Computer Access for People With SevereDisabilities*. **IEEE Transactions on Neural Networks and Rehabilitation Engineering,** Vol. 10, No. 1, March 2002.

[55] K. Aizawa and T. Huang, *Model-based image coding: Advanced video coding techniques for very low bit-rate applications*. **Proc. IEEE**, Vol 83, No. 2,pp. 259–271, 1995.

[56] R. Kaucic, B. Dalton and A. Blake, *Real-Time Lip Tracking for Audio-Visual Speech Recognition Applications*. **Proceedings of the European Conference on Computer Vision, Cambridge**, UK, pp. 376-387, 1996.

[57] M. Takatoo, T. Kitamura, Y. Okuyama, Y. Kobayashi, K Kikuchi, H.Nakanishi, and T. Shibata, *Traffic flow measuring system using image processing*. **In Proceedings of SPIE,** pp. 172-180, Kawasaki, Japan, 1990.

[58] C. E. Smith, C. A. Richards, S. A. Brandt and N. P. Papanikolopoulos, *Visual Tracking for Intelligent Vehicle-Highway Systems*. **IEEE Transactions on Vehicular Technology,** Vol. 45, No. 4, pp. 744-758, November 1996.

[59] D. Rueckert, P. Burger, S. M. Forbat, R. D. Mohiaddin, and G. Z. Yang,*Automatic Tracking of the Aorta in Cardiovascular MR Images Using Deformable Models*. **IEEE Transactions on Medical Imaging**, Vol. 16, No. 5, pp. 581-590, October 1997.

[60] W. Freeman, K. Tanaka, J. Ohta, and K. Kyuma, *Computer vision for computer games*. **In Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition**, Killington, pp. 100–105, 1996.

[61] A. Davison, *Mobile Robot Navigation Using Active Vision*. **PhD thesis, University of Oxford,** 1998.

[62] D.C. Bentivegna, A. Ude, C.G. Atkeson, and G. Cheng, *Humanoid robot learning and game playing using PC-based vision*. **In Proc. IEEE/RSJ 2002 Int. Conf. on Intelligent Robots and Systems, Laussane, Suisse,**Vol. 3, pp. 2449–2454, 2002.

[63] S. Hutchinson, G.D. Hagar, and P.I. Corke, *A tutorial on visual servo control*. **IEEE Transactions on Robotics and Automation,** Vol. 12, No. 5, pp. 651–670, October 1996.

[64] A. P. Dempster, N. M. Laird, and D. B. Rubin. *Maximum likelihood from incomplete data via the EM algorithm.* **Journal of the Royal Statistical Society. Series B (Methodological)**, 39(1):1–38, 1977.

[65] R. J. Oliveira, P. Canotilho Ribeiro, J. dos Santos Salvador Marques, and J. M.Lemos. *A video system for urban surveillance: Function integration and evaluation*. In **International Workshop on Image Analysis for Multimedia Interactive Systems,** Lisboa, Portugal, 2004.

[66] Alper Yilmaz, Omar Javed, Mubarak Shah. *Object Tracking: "A Survey"* **ACM Computing Surveys,** Vol.38, No.4 Article 13, Publication date: December 2006

[67] John Wood." *Statistical Background Models with Shadow Detection for Video Based Tracking,* **Technical Report**, 2007.

[68] Yigithan Dedeoglu: "*Moving Object Detection, Tracking and Classification for smart video surveillance*", **Thesis**, 2004.

[69] Qi Zang and Reinhard Klette: "*Parameter Analysis for Mixture of Gaussians Model"*, **Department of Computer Science**, Tamaki Campus, The University of Auckland , New Zealand, 2002

[70] Tun-Yu Chiang, Wilson Lau: "*Segmentation of Vehicles in Traffic Video*", **Project Report**, Stanford University, 2002

[71]Jacinto C.Nascimento, Jorge S.Marques: "*Novel Metrics for PerformanceEvaluation of Object Detection Algorithms*", **IEEE Transactions on Multimedia**, (TMM):761-774, Lisboa Portugal, 2006.

[72] Hong Liu, Jintao Li, Qun Liu and Yueliang Qian: "*Shadow Elimination in Traffic Video Segmentation*", **Institute of Computing technology, Chinese Academy of Sciences**, Beijing, China, 2007.

[73] A. Gyaourova, C. Kamath, S. and C. Cheung - *Block matching object Tracking* - **LLNL Technical report**, October 2003.

[74] Y. Rosenberg and M. Werman – *Real-Time Object Tracking from a Moving Video Camera: A software approach on PC* **Applications of Computer Vision,** 1998. WACV '98. Proceedings.

[75] A. Turolla, L. Marchesotti and C.S. Regazzoni – *Multicamera object Tracking in video surveillance applications*.

[76] Y. Wang, J. Doherty and R. Van Dyck – *Moving Object Tracking in video Proc. **Conference on Information Sciences and Systems***, **Princeton**, NJ, March 2000.

[77] J.Heikkila and O.Silven. *A real time system for monitoring of cyclists and pedestrians*. **In proc. of second IEEE Workshop on visual Surveillance**, pages 74-81, Fort Collins, Colorado, June 1999.