國立交通大學
資訊工程學系

碩 士 論 文

以物件為基礎監控視訊內容之追蹤與摘要

Object-Based Video Tracking and Abstraction on Surveillance Videos

研 究 生：郭慧冰

指導教授：李素瑛 教授

中 華 民 國 九 十 三 年 六 月

以物件為基礎監控視訊內容之追蹤與摘要

Object-Based Video Tracking and Abstraction on Surveillance Videos

研 究 生：郭慧冰　　Student：Hui-Ping Kuo

指導教授：李素瑛 教授　　Advisor：Prof. Suh-Yin Lee

國 立 交 通 大 學

資 訊 工 程 學 系

碩 士 論 文

A Thesis Submitted to
Institute of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of
Master
in
Computer Science and Information Engineering

June 2004

Hsinchu, Taiwan, Republic of China

中華民國九十三年六月

以物件為基礎監控視訊內容之追蹤與摘要

研究生：郭慧冰　　指導教授：李素瑛

國立交通大學資訊工程學系

摘要

影像內容摘要是藉由擷取影像資料之中的重要涵義來精簡地表示影像資料之重要的內容。而在監控視訊，我們可以用以物件為基礎之視訊摘要來表達物件所發生的事件與其代表重要的意義。因此，在視訊監控系統如智慧型運輸系統，物件為基礎之視訊摘要可以用來對物體所發生之重要事件發出警訊。此外，視訊內容摘要也可以幫助我們從監控視訊影像的資料庫中取得與管理重要的內容。在這篇論文中，我們提出了一個以物件為基礎之監控視訊之影像內容追蹤與摘要的系統。首先，我們使用以背景作為參考之動態物件切割演算法把運動物體從背景中分割出來。接著，我們使用一個簡單但有效的動態物件追蹤演算法得到運動物件的軌跡與特徵。最後，我們設計了一個藉由選取代表著重要影像內容的動態物體來產生影像內容摘要的演算法。我們用此演算法來產生以物件為基礎之影像內容摘要。

我們藉由展示出我們所選取到具有代表性意義與事件的物件來證明我們提出的系統的成效。根據所提出的系統架構，我們實作了一個能夠線上發出警訊之即時影像監控

系統。在這個系統中，我們可以藉由所產生的內容摘要即時地對重要的事件發出警訊並且同時對監控影像中的物體做即時的追蹤。我們測試了一些不同類型的監控視訊影像的片段，而實驗結果證明，我們的系統在物件內容之追蹤與在重要物件內容之摘要都得到滿意的結果。

Object-Based Video Tracking and Abstraction on Surveillance Videos

Student: Hui-Ping Kuo Advisor: Prof. Suh-Yin Lee

Institute of Computer Science and Information Engineering
National Chiao Tung University

Abstract

Video abstractions represent the video contents concisely by extracting the important semantics in the video. Important events and semantic meanings of moving objects in the surveillance videos can be represented using the object-based video abstractions. Therefore, the object-based video abstractions can be used to alarm important events in the surveillance and monitoring systems like the intelligent transportation system (ITS). Besides, the video abstractions can also help retrieving and managing important contents from the surveillance video database. In this thesis, we propose an object-based video tracking and abstraction system. First, we use a background-registration segmentation algorithm to segment the moving objects. Then, a simple but effective tracking algorithm is introduced to extract the object trajectories and object features. Finally, an abstraction algorithm is applied to generate object-based abstraction by selecting key objects which contain important semantic contents.

We will reveal the performance of the tracking and abstraction algorithms by showing the results of extracting objects with representative features and events such as object appearance/disappearance, objects occlusion/split and changes in motion. We will present a real-time tracking system with on-line alarming to demonstrate the implemented system. In

this system, important object events will be alarmed using the generated object-based abstraction on-line while the surveillance video is being tracked in real-time. We test our system with several surveillance video sequences and the experimental results prove that we can get satisfactory results in both the object-based tracking and abstraction.

Acknowledgement

I sincerely appreciate the guidance and the encouragement of my advisor, Prof. Suh-Yin Lee. She encouraged me in exploiting research topics freely and enthusiastically helped me. Without her, I cannot complete this thesis.

Besides, I would like to extend my thanks to the lab mates in the Information System Laboratory, especially Mr. Duan-Yu Chen and Mr. Ming-Ho Hsiao. They gave me a lot of suggestions and shared their experience.

Finally, I want to express my appreciation to my parents for their support. They gave me the opportunity to have good education. This thesis is dedicated to them.

Table of Contents

List of Figures

Lists of Equations

List of Tables

Chapter 1

Introduction

1.1 Motivation

In recent years, the video technology has advanced surprisingly. Because of the maturity of the digital video processing techniques and the compression standards, applications of digital video were widely adopted, for example, the video surveillance systems and stream media applications. However, with the traditional video encoding standard such as MPEG1 and MPEG2, it is difficult to manage and retrieve the important content or content of interest efficiently when the amount of video data repository is very huge. Thus, it is essential to manage the video content to enhance the value and the usability the produced video data.

Fortunately, in the MPEG 4 encoding standard, a new feature Video Object Plane (VOP) was introduced. The concept of VOP is that video sequences can be encoded into separate bitstreams according to the contents of the video, such as moving object and background. Thus, we can segment video frames into some semantically meaningful video objects and encode these video objects separately. This feature allows us to manage and retrieve the object-level information easily and thus provides higher level semantics and interactivity. Especially in the surveillance videos, because the moving objects are the most important contents, the benefits of extracting object-level information can be fully exploited.

To represent the important contents in the video concisely, the video abstraction is usually used. The video abstraction helps us to retrieve the important contents in the video. In the traditional methods, the abstraction is usually generated by extracting key frames with representative features. Since the moving objects are important in the surveillance videos, we can use object-based abstraction and the key frames can be extracted by extracting significant

key objects. Because the abstraction is object-based, object-level semantics or events can be represented.

The object-based video abstraction can be further applied to the surveillance systems such as the Intelligent Transportation System (ITS). The abstraction can be used to alarm object events and important contents.

In this thesis, we will present an object-based video tracking and abstraction system on surveillance video. The general object events such as appearances, occlusions and the changes in motion will be detected. With the domain knowledge, these general object events can be further extended and deployed in many surveillance applications. For example, the occlusion of objects can indicate the car accident and the changes in the object motion can be used to detect whether there is illegal driving. A real-time tracking system with on-line general events alarming will be implemented in this thesis.

1.2 Organization

The rest of the paper is organized as follows. Chapter 2 introduces the background and the related work of the video object segmentation, tracking and abstraction. Chapter 3 presents our proposed algorithms for video tracking and abstraction. Chapter 4 shows the architecture of the system and the experimental results. We will make a conclusion in Chapter 5.

Chapter 2

Background

In some applications in Intelligent Transportation System (ITS) such as traffic surveillance and monitoring, the goal is to monitor the moving objects in the surveillance environment. Thus, video object segmentation is required to first extract the moving objects. After that, video object tracking and abstraction processes are required to track the object trajectory and extract the significant key objects which we may be interested in. In this chapter, we will introduce the related works of the video object segmentation, video object tracking and video abstraction. The details of the related works of video objects segmentation, video object tracking and video abstraction will be introduced in Section 2.1, 2.2 and 2.3, respectively.

2.1 Video Object Segmentation

Object segmentation is the first step toward the object-level abstraction and is the task to find a mask indicating the shape and the position of the moving object. There are many researches in the literature of object segmentation. Generally, segmentation algorithms can be classified into two categories, the homogeneity based methods and the change detection based methods. The homogeneity based algorithms [1-4] segment moving objects based on the homogeneity of their color, texture or motion information. Pixels with some similar features are first grouped into small regions, and these regions are then grouped into objects with some other features. This kind [1-4] of algorithms can provide precise object masks; however, the watershed algorithm for the boundary decision is a computational expensive process. Also, the motion estimation process to compute the precise motion vectors for clustering small regions also takes a lot of time. Thus, this kind of algorithm is not a good choice for systems

that have real-time requirements.

The other category of segmentation algorithms is the change detection based algorithms. This kind of algorithms [5-7] segment objects by taking difference between the current input frame and a reference image, and then a threshold is chosen to decide a difference mask indicating the shape and the position of the moving objects. Traditionally, the previous input frame is chosen as the reference image and this is quite simple and efficient. However, there are some well-known drawbacks [8]. First, when the speed of the moving object is not consistent, it becomes impossible to indicate the position using the difference image and thus miss or false alarm in segmentation is unavoidable. Second, the uncovered background is another problem in traditional change-detection algorithms because the uncovered background regions that are covered by objects in previous frame may be considered as changed. Although, uncovered background can be detected and removed when the motion information is taken into consideration the computation of motion estimation is expensive and greatly lower the efficiency of the change-detection algorithms.

Recently, some change detection algorithms [8-11], [15], [27] use a reference background image to segment moving objects. The reference background image is acquired beforehand or by some means to update dynamically and contains the still background without any objects. The change-detection algorithms with registered background effectively solve the problems of uncovered background and inconsistent object speed effectively. Besides, they are efficient and can meet the real-time requirement. In our proposed system, we will adopt the change-detection based algorithm with registered background to segment moving objects.

2.2 Video Object Tracking

Video object tracking is an important and frequently discussed research topic. Its objective is to match the detected objects in the current frame to the corresponding objects

detected previously. The tracked position and shape of objects can be used for the generation of the VOPs in MPEG-4 for some objects or to form the object trajectories for later object-based analysis and abstraction. Thus, video object tracking plays an important role in systems to extract MPEG-4 VOPs or object-based description and abstraction in MPEG-7.

The object tracking algorithms first take the detected object masks from the segmentation algorithms as input data, and try to match the objects detected earlier using some features such as the position, shape and color. For example in [12], Oberti et al. used the shape of the object corners to track video objects. Some tracking algorithms also take motion information into consideration. For example, Kim et al. use the direction of the motion and the variation of the speed to compute smoothness feature as the matching criteria [13]. Chen used the motion as the constraint to find matching objects [14]. Some other algorithms [15-17] adopted Kalman Filtering. It is a linear estimation process that estimates the current value and updates the prediction recursively, to estimate and track the position of the objects.

The precision of the prediction involves two errors: the process error and the measurement error. Because sometimes there are errors in the segmentation process due to the cluttered scene, the object masks would not be very precise and hence the measurement errors would be large. Besides, some abrupt movements of the objects such as waving of hands will make the process error large. The prediction error may not converge quickly if both the process error and measurement errors are large. Thus, it may be difficult to match correctly due to the uncertainty of the prediction. In this thesis, instead of using Kalman Filtering, our algorithm uses the motion information as feature, which is efficient and effective. The occlusion and the split of objects can also be handled in our tracking algorithm. The trajectories of the objects as well as the event of occlusion and split will be stored for the object abstraction in the later stage.

2.3 Video Abstraction

The video abstraction is to represent the large amount of video data in a compact form and is generated by selecting the key frames which are representative of the features and the contents of the video. Traditionally, the key frame extraction (KFE) algorithms [18-21] are frame-based and the image features are used. For example, the video sequence is first decomposed into scenes and the scenes are then decomposed into shots. The key frames are then extracted from each shot to represent the features of that shot. The image features like color and the motion intensity are usually used as the criteria to select key frames.

In the surveillance video, because the background is stationary and the moving objects are the most important contents in the video, we can choose the moving objects with important features or significant events to represent the video contents. However, the traditional KFE algorithms are not applicable because they use only low-level images features and the generated abstractions are lack of object-level semantics. In [22] and [23], Kim et al. proposed a system that selects key objects for video abstraction using the shape information of the moving objects. They tried to detect the changes in the shapes by computing moment invariant moments [12] to capture the actions of the moving objects. However, it is difficult to detect the changes in the object shapes for the rigid moving objects such as the vehicles. In our proposed abstraction algorithm, we will try to select key objects based on the characteristics of the object trajectories and the object events detected.

Other object-based KFE algorithms proposed are in [24] and [25]. In [24], the number of the intra-coded macroblocks (I-MBs) to the total number of the encoded macroblocks in the VOP used as the criteria of the key object selection; while [25] detected significant changes in the shape of the VOP in the MPEG-4 compressed domain. Unlike the previous works, which used pre-segmented VOPs, we propose a system is which aims to track video objects in real-time and select key objects to generate video abstractions on-line.

Chapter 3

Algorithm for Object-Based Video Tracking and Abstraction on Surveillance Videos

In this chapter, we will present the proposed system for video object tracking and abstraction. The whole system is composed of three main modules: video object segmentation, video object tracking and video abstraction. In section 3.1, we will give an overview of the whole system In section 3.2, we will present our video object segmentation algorithm. In section 3.3, we will present the video object tracking algorithm to track the moving objects detected. In section 3.4, the video abstraction algorithm will be presented.

3.1 System Overview

Our proposed system contains three modules, which are the video object segmentation module, the object tracking module and the video abstraction module. The surveillance video data are first captured and input to the video object segmentation module. The object segmentation algorithm segments the moving objects from the background and generates the object masks which indicating the position and the shape of the moving objects. The segmented object masks are then input to the video object tracking module. The object tracking algorithm matches the input object masks to those objects which have been input previously. Also, the occlusion and the splitting of the objects are detected and are reasoned in the object tracking stage. In the video abstraction module, video abstraction will be generated by selecting those frames with semantically meaningful objects contained.

3.2 Video Object Segmentation

In our object-based tracking and abstraction system, the first step is to segment the moving objects as precisely as possible. The object segmentation algorithm directly takes the

raw video data as input to segment the moving objects in the surveillance video sequence and extracts the object masks to indicate the presence of the moving objects. In the surveillance video, since the position of the camera is always fixed and the background is stationary, so the simplest way to segment moving objects is to use the change detection based method. Because when comparing a frame to a background image, it is straightforward to consider the regions that change significantly as moving objects. Thus, selecting background image for the change detection based algorithm as reference can effectively achieve our goal.

However, besides the moving objects that we are interested in, there are other types of changing regions that may be miss-classified in the segmentation process. One of which is the camera noises. The camera noises are the white noise of the camera and are usually small. The other type is often called 'ghost'. The ghost is the changing region that appears and then disappears quickly without steady motion and is usually bigger than the camera noise. The ghost effect is usually resulted from the waving of tree leaves and regional lighting effect. In order to obtain accurate object masks, these annoying changing regions should be filtered out.

In our segmentation algorithm, we adopt the change detection based algorithm with background registration and the filtering process of noises and ghosts is also applied. The whole process of our segmentation algorithm is shown in Fig. 1.

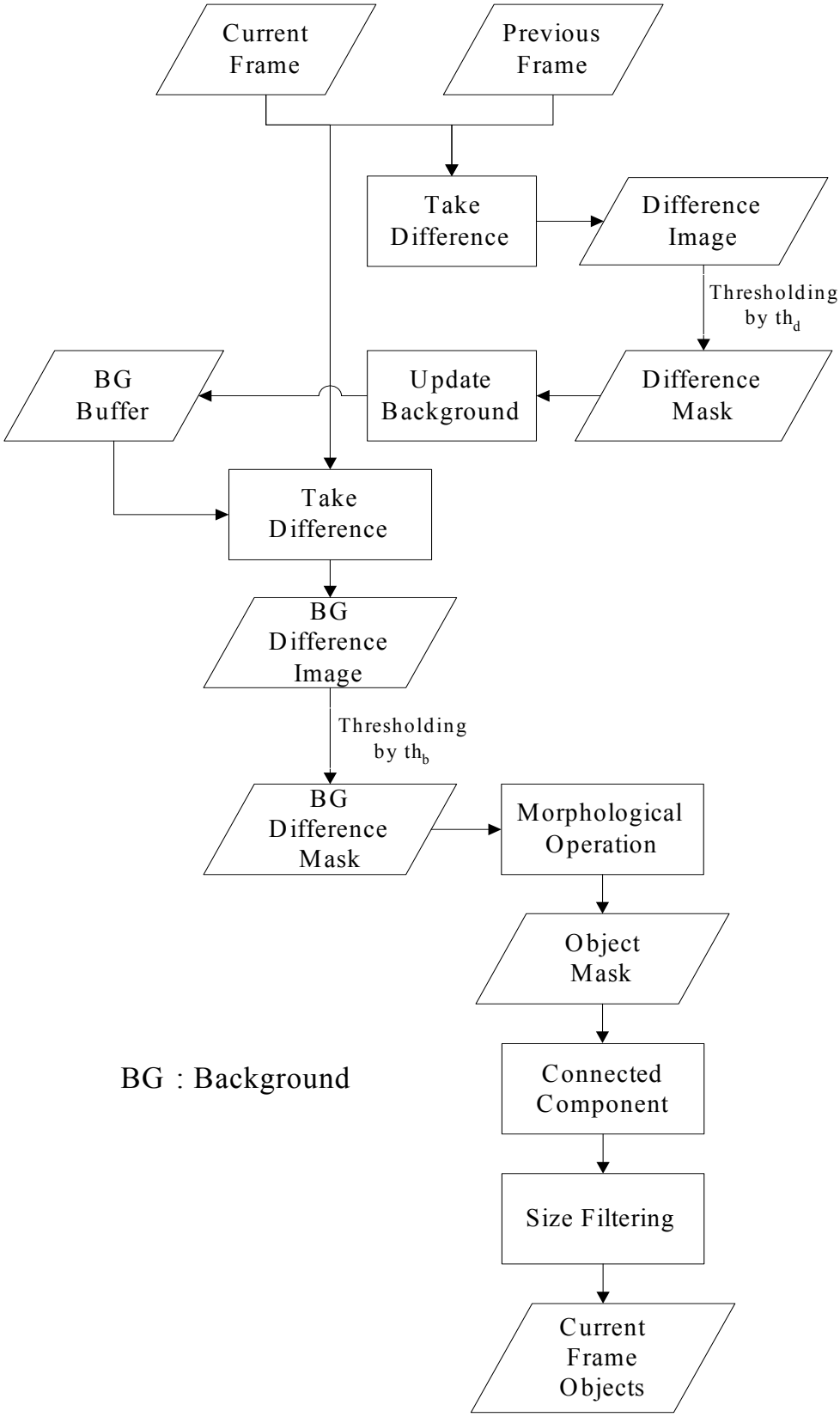


Fig. 1. Segmentation process diagram

3.2.1 Inter-frame Differencing

In the segmentation algorithm, the first step is to compute the inter-frame difference image (DI) between the current input frame and the previous frame. The distance value in the DI shows how strong a pixel changes in two consecutive frames and the possibility that a pixel will be considered as changing. Because the human eyes are more sensitive to luminance than to chrominance, we only take difference value on the luminance channel. After taking threshold TH_d on the difference image, we can obtain a difference mask (DM) that indicates the changed regions between two consecutive frames. The computations of DI and DM are shown in Eq. (1) and Eq. (2), where the $C^Y(i,j)$ and $P^Y(i,j)$ denotes the pixel value in current frame and previous frame in the luminance channel respectively. The DN will be used to update the background image in the next step.

$$DI(i,j) = \left| C^Y(i,j) - P^Y(i,j) \right| \tag{1}$$

$$DM(i,j) = \begin{cases} 0 & if \quad DI(i,j) > TH_d \\ 255 & if \quad DI(i,j) \le TH_d \end{cases} \tag{2}$$

3.2.2 Dynamic Threshold Decision

In order to make our segmentation algorithm adapt to various kinds of environments and video contents, the threshold TH_d for deciding the DM cannot be fixed and should be selected adaptively. In many researches [8], [10], [26-27], the values in the difference image can be modeled by a mixture of two distributions. And thus, finding the threshold corresponds to finding the two distribution functions that approximate the histogram of the difference values. Traditionally, the valley between two peaks is found and is chosen as the threshold dividing two distributions. However, in the real case as shown in Fig. 2, the histogram fluctuates heavily and it is difficult to find a threshold just by finding a valley. In [26], Wu et al. suggested that the histogram can be converted to a monotonic increasing histogram by accumulating the original histogram values, as shown in Fig. 3. In the cumulative histogram,

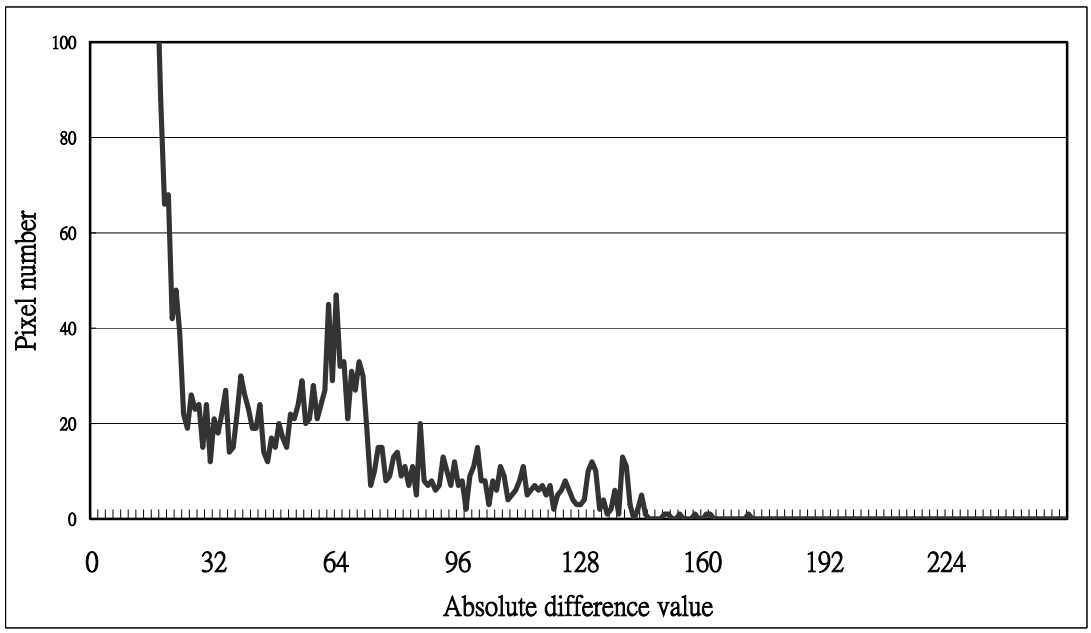


Fig.2. The histogram of a difference image

the problem of finding a threshold can be simplified to finding an intermediate point such that two straight lines which are interpolated by the start point, end point and the intermediate point can best approximate the cumulative histogram. Instead of using the ratio histogram in [26], we simply use the difference since the computational cost is much more expensive for the ratio histogram.

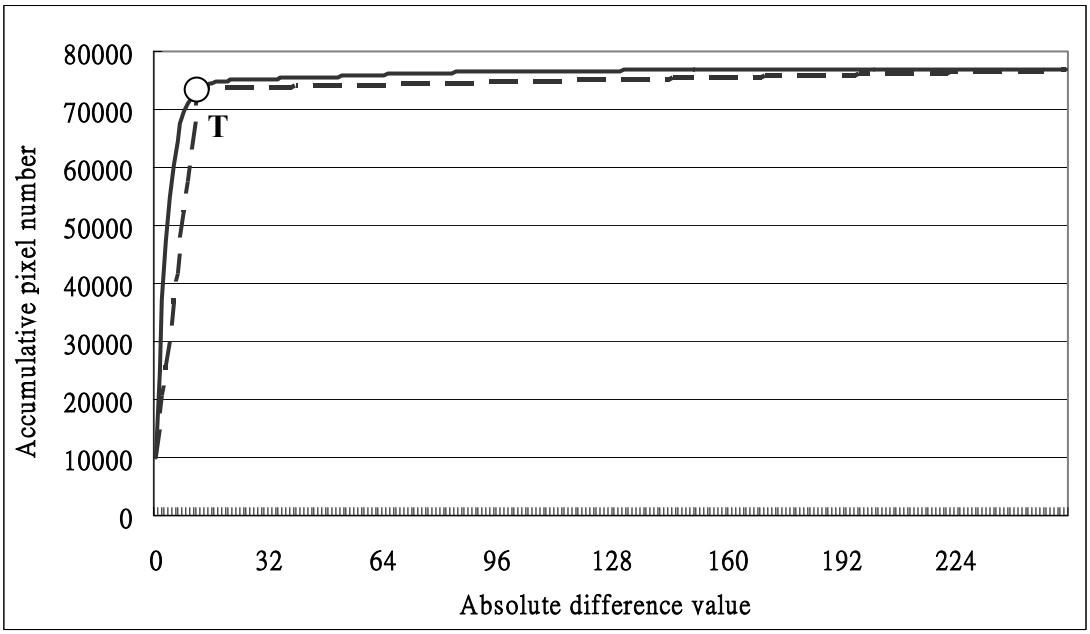


Fig. 3. The cumulative histogram and the interpolated lines

3.2.3 Background Buffer Update

The next step in the segmentation algorithm is to update the background image (BI). In the background registration method [8-11], [15], [27], because the performance of the segmentation result relies on the correctness of the background dramatically, we need a robust method to retrieve and maintain the background image. The simplest way to obtain the background image is to capture the background beforehand. However, the background image may change slightly and gradually because the luminance may vary with time. In our algorithm, we dynamically update the background buffer using the difference mask. With the difference mask, the regions that are marked as 'changed' will not be updated to avoid distortion. Every time when a new difference mask is computed, the background image buffer at current time *t* (BI_t) at position (i,j) is updated using the equation in Eq. (3) and Eq. (4). In the equations, the k(i,j) is the bias factor of the pixel (I,j) which accelerate the speed of background update. The symbol α is the weighting factor used in the update function Eq. (4).

$$k(i,j) = \begin{cases} 1 & if \;\; C^Y(i,j) > BI^Y_{t-1}(i,j) \\ -1 & if \;\; C^Y(i,j) \le BI^Y_{t-1}(i,j \end{cases} \tag{3}$$

$$\mathrm{BI}^Y_t(i,j) = \begin{cases} BI^Y_{t-1}(i,j) & if \;\; DM(i,j) = 255 \\ \alpha \cdot C^Y(i,j) + (1-\alpha) \cdot BI^Y_{t-1}(i,j) + k(i,j) & if \;\; DM(i,j) = 0 \end{cases} \tag{4}$$

When the system starts up, the background buffer is empty and a period of time for the background buffer initialization is required. With our background update equation, the initialization can be completed in a few frames. After the initialization process, we update the background buffer every 30 frames since background color does not change frequently. The gradual variation can quickly be updated to background buffer. Even if there is a sudden lighting variation when the clouds are dispersed and the sun is revealed, the update equation can also catch up the variation in a short period of time.

3.2.4 Background Differencing

After we obtain a background buffer, we can segment the moving objects from the

background. Unlike the way adopted in computing the difference image, we use the luminance and chrominance channels together instead of using luminance only. Because some moving objects look quite different compared to the background in their color, but the difference between the current frame and the background is almost zero when the chrominance information is discarded. In order to extract accurate object masks and not to miss any important moving objects, the chrominance information must be considered. Because the importance of the chrominance channels depends on the intensity of the luminance channel, we design a difference score function to evaluate the difference in YUV color space. Denote the different score as DS, the equations Eq. (5) through Eq. (8) show how to compute the different score. For a pixel (i,j), we first get the strongest luminance intensity among the current frame and the background image. Then, we decide the weighting factor of the chrominance based on the luminance intensity. Because the valid range of the luminance channel after conversion is from 16 to 235, we can subtract the luminance value from 16 and divide it into 11 levels, which are from 0.0 to 1.0. After that, we can use the weighting sum equation Eq. (7) to compute the color distance in the YUV color space.

$$M(i,j) = \max(C^Y(i,j), BI_t^Y(i,j)) \tag{5}$$

$$w = floor\left(\frac{M(i,j)-16}{20}\right)/10 \tag{6}$$

$$DS(C(i,j), BI(i,j)) = \frac{\left|C^Y(i,j) - BI^Y(i,j)\right| + w\cdot\left|C^U(i,j) - BI^U(i,j)\right| + w\cdot\left|C^V(i,j) - BI^V(i,j)\right|}{1+(2\cdot w)} \tag{7}$$

$$BDI(i,j) = DS\,(C(i,j), BI(i,j)) \tag{8}$$

$$BDM(i,j) = \begin{cases} 0 & if\ \ BDI(i,j) > TH_b \\ 255 & if\ \ BDI(i,j) \le TH_b \end{cases} \tag{9}$$

Sometimes an object enters a frame and then keeps stays in the same position on the xy plane. We call such kind of object as ‘stopped object’ since the motions in both the x and y

direction are almost zero. Because the algorithm updates the inputted frame to the background for the unchanged regions, the color of the stopped objects will be updated to background buffer when they stop too long. In this case, object regions will be false alarmed because the background has been wrongly updated. Although this problem can be solved by lengthening the interval of background updates, the time needed to adapt to the luminance variations is also be lengthened. And thus it is a tradeoff and both of the cases must be taken into consideration.

After we finish computing the background difference image using the difference score function, another dynamically selected threshold TH_d is applied to get the background difference mask, as shown in Eq. (9). The background difference mask extracted here indicates the moving object regions compared to the reference background image. However, the background difference mask contains a lot of noises and the object boundaries are not smooth. Thus, further filtering is required.

3.2.5 Morphological Operation

To smooth the object boundaries and remove the noises, two kinds of morphological operations are frequently used [8], [13], [23]. The closing operation is first used to fill the black holes inside the object masks and the opening operation is then used to remove the small noises that do not belong to the moving object. In our algorithm, the structure element of size 7x7 and 5x5 are selected for closing and opening operations respectively. In most of the cases, the smaller camera noises can be successfully filtered. However, larger regions caused by ghost effect are hard to remove out. Although larger structuring element may help, the computation cost will also be more expensive. And thus, instead of using larger structuring element, we will filter out these ghost regions in the video object tracking algorithm with temporal and spatial filtering.

After the morphological operations, the object mask is smoothened and indicates the

shapes and the positions of all the moving objects in the current frame. The individual object in the object mask is then extracted in the next process.

3.2.6 Connected Component Labeling Algorithm and Size Filtering

The tracking algorithm gets the extracted object mask as the input. However, the object mask simply indicates the positions and the shapes of all the moving regions without separate information. Thus, each individual object in the object mask must be extracted and assigned an identifying label. The connected component algorithm is a frequently used algorithm [8], [29] to achieve this work. For every pixel, it first examines the neighboring pixels and assigns that pixel a label. After that, pixels with the same label or equivalent labels are clustered together to form an isolated object.

Because there are some large noise and ghost regions which are hard to be completely removed out, the size filtering must be performed after the labeling process. The size filtering process filters out those regions which are smaller than a predefined threshold. The objects that are not filtered out are called the object-of-interests and these objects will be tracked in the tracking algorithm.

3.3 Video Object Tracking

The second module in our system is the video object tracking module. Although the object-level information can be extracted via the segmentation of video objects, the higher level object semantics can only be extracted from the object trajectories. Thus the object tacking process is the key role toward the semantics analysis and video abstraction. Our tracking algorithm gets the extracted object masks as the input and tracks all the objects to get the object trajectories. The tracking algorithm can be divided into several sub-modules and will be presented later. The diagram is shown in Fig. 4.

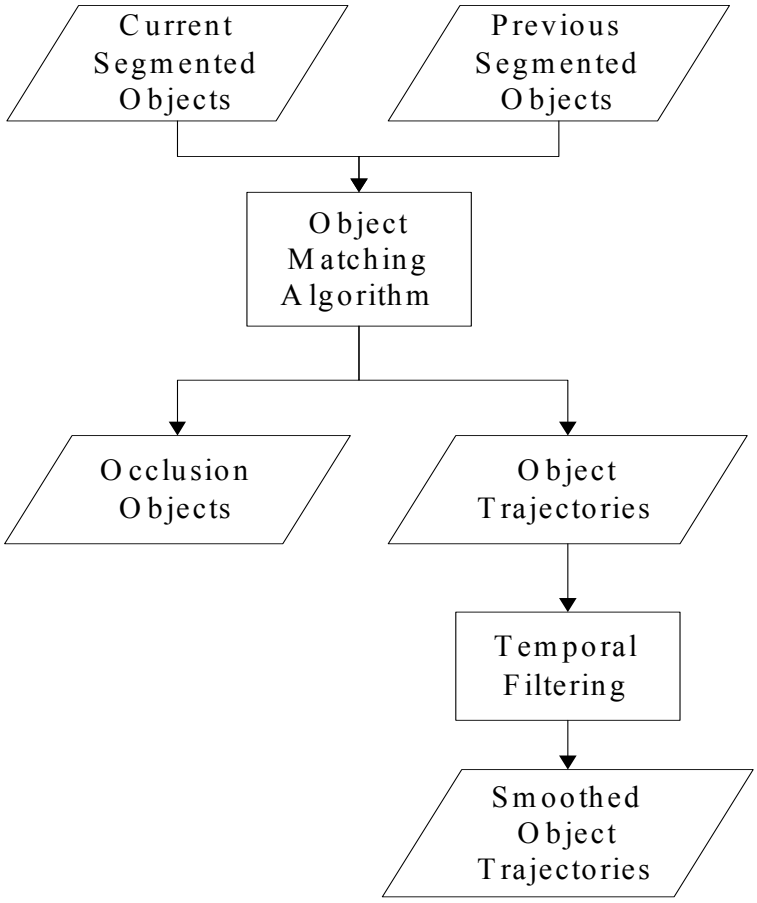


Fig. 4. The process diagram of the tracking algorithm

In the tracking algorithm, the object information, such as mass center and motion, is first gathered for each detected object. We use a simple but effective matching function based on the motion and will be presented in details later.

Based on the observation, the occlusion can happen inside the camera view or outside

the camera view. For the first condition, we can observe the occlusion of the objects. For the second condition, the objects are occluded when entering the camera view. According to Jung [15], we define the first condition as EXPLICIT OCCLUSION and the second condition as IMPLICIT OCCLUSION. The occlusion events are detected and the objects after splitting are tracked and matched using the motion information.

After the matching of the objects, the object trajectories are smoothened in the temporal filtering process. The smoothened trajectories are then input to the video abstraction algorithm to generate abstractions.

3.3.1 Matching Function

The object tracking algorithm tracks the object trajectories by matching the current video objects to the previously tracked video objects. In the literature of object tracking, some algorithms [15-17] adopt the Kalman Filtering to estimate and track the objects. It is appealing because it recursively estimates the object states and updates the predictions. In the ideal situation, when the moving paths are very smooth and the object masks are very accurate, the prediction error converges quickly because both the measurement error and the process error are small. However, the detected object boundaries may contain some errors due to the clutter scenes in the real environments, and the measurement errors thus become large. In addition, the path of a moving object is not as smooth as we expected. For example, if we connect the mass centers of a walking person, the connected path looks like zigzag rather than a straight line because all the actions such as waving of hands and striding affect the mass centers significantly. Under this condition that both the measurement error and the process error are high, the prediction error may not converge quickly. Thus, it is difficult to track and handle some complicated conditions like object occlusions due to the uncertainty of the estimation.

Due to these reasons, we utilize the motion information to match objects in our object tracking algorithm. We use a simple but effective motion distance evaluating function to compute the motion distance for objects matching. Let us denote the *ith* segmented video object at time *t* as VO_t^i. Suppose there are n current video objects and m previous objects, which belong to m tracked trajectories. We can know that all the m previous objects VO_{t-1}^i have been tracked at time t-1 and thus the motion vectors are known at time t and denoted as MV_{t-1}^i. Besides, the mass center of the n current objects and m previous objects are also known, which are denoted as XY_{t-1}^i and XY_t^i respectively. The computations of motion distance function (MVD) are shown in the equations Eq. (10) and Eq. (11). In the equations,

the mv(i, j) is the object motion vector between current object *i* and previous object *j* and the Θ is the included angle between mv(i,j) and MV_{t-1}^{j}.

$$\mathrm{mv}(i,j) = \left(XY_t^{\,i} - XY_{t-1}^{\,j}\right) \tag{10}$$

$$MVD\left(VO_t^{\,i}, VO_{t-1}^{\,j}\right) = \sqrt{\left|MV_{t-1}^{\,j}\right|^2 + \left|\left(\mathrm{mv}(i,j)\right)\right|^2 - 2\cdot\left|MV_{t-1}^{\,j}\right|\cdot\left|\mathrm{mv}(i,j)\right|\cdot\cos(\theta)} \tag{11}$$

The motion vector distance function takes both the position and the moving direction into evaluation. The motion vector distance function can be further explained in its geometric meaning. As shown in Fig. 5, the geometric meaning of the MVD is the length of differencing motion vector. And the equation shows that both the position and the moving direction are evaluated in the equation.

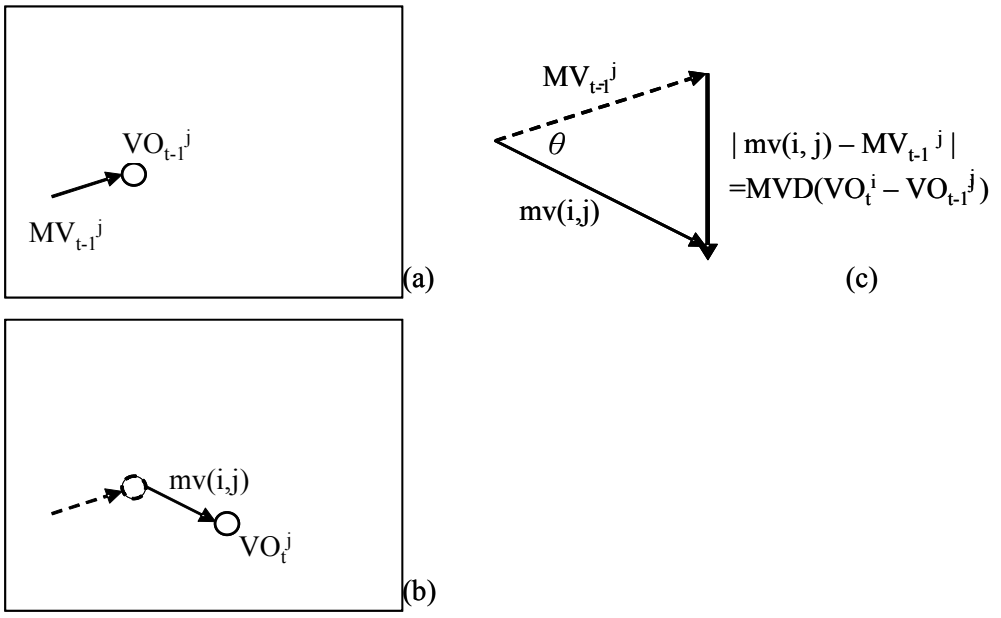


Fig. 5. (a) video object j in the time t-1 and the tracked motion vector; (b) video object in time t, the mv(i,j) is computed using Eq. (9); (c)motion vector distance

To match the n current objects to the m previous objects, the matching function first builds up an m by n table and computes the motion vector distance in each table entry. The matching function then picks up the entry that the motion vector distance value is the minimum. If the minimum value does not exceed a predefined threshold $TH_{match,}$ the

corresponding current object and previous object in that selected entry are matched. Note that the matching function is a 1-to-1 function, the matched current object and previous object cannot be matched again. The matching function runs iteratively until the selected motion vector distance exceeds the threshold or either the current objects or the previous objects are

```
create a MVD n by m table T for the m previous objects and n current objects
COsize = n;
POsize = m;

for(i=0; i<n; i++)
{
      for(j=0; j<m; j++)
      {
            T = MVD( VO_t^i, VO_t-1^j );
      }
}

while(COsize >0 && POLsize >0)
{
      min_Value = the minimum MVD value in T;
      min_Entry = the entry T[x,y] that has minimum MVD value;
      if(min >= TH_match) break;

      match VO_t^x to VO_t-1^y;
      COsize --;
      POsize --;
      Delete the row T[x,*] in T;
      Delete the column T[*,y] in T;
}
end of matching algorithm;
```

Fig. 6. Pseudo code of the matching function

all matched. Fig. 6 shows the pseudo code of the matching function.

3.3.2 Object States

Before entering the object matching algorithm, we must introduce the object states associated with the tracked objects. In the tracking process, because there are several object events such as the appearance, the disappearance, the occlusion and the split, we need additional flags to indicate the current state of an object. Therefore, we define two object state flags, the OCC_STATE and the OBJ_STATE. The OBJ_STATE indicates the object condition in its life cycle. The OBJ_STATE has three states: NORMAL, DYING and DEAD. Fig. 7 shows the state transition graph. The life cycle of an object starts when the object first appears

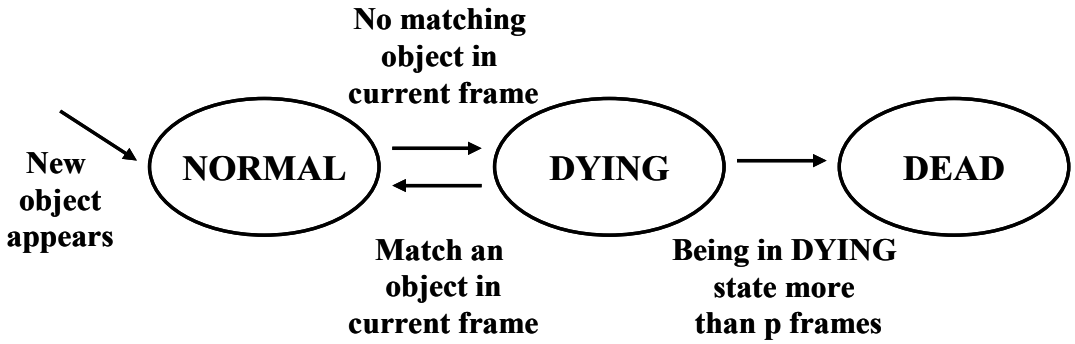


Fig. 7. The state transition diagram of the OBJ_STATE

in the frame and then the state goes to the NORMAL state. Because sometimes when the scene is clutter or the moving object is small, the object may be missed or be filtered out in the segmentation process and the moving object may disappear temporally. In traditional approaches, the original object may be considered disappeared and a new object entry will be created under this condition. However, in our human's perception, there should be only one object. Therefore, instead of considering the temporally disappeared object dead directly, we let the object go to the DYING state first. When the object in the DYING state cannot find a match in the next p frames (say three), we will think the object is really disappeared and let the object go to the DEAD state. On the contrary, the object goes back to the NORMAL state when a good match is found before the time limit.

The other state flag is the OCC_STATE, which indicates the relationship to other objects. The OCC_STATE has three states: NORMAL, COLLISION and OCCLUSION. The state transition diagram is shown in Fig. 8. When a new object appears, the OCC_STATE goes to

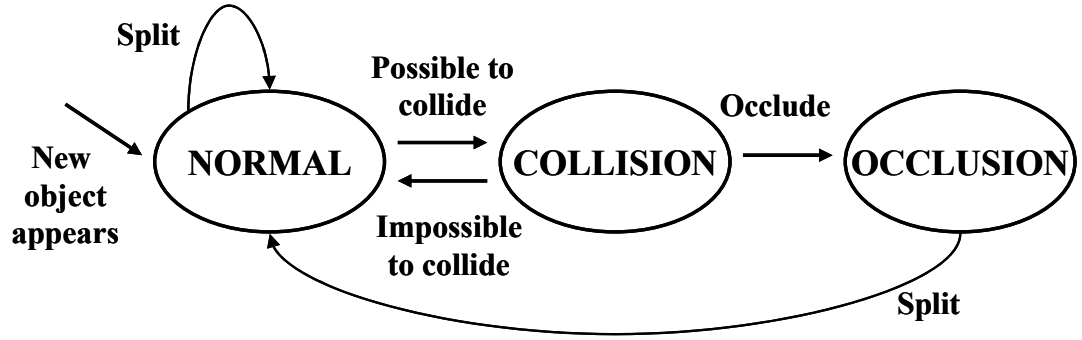


Fig. 8. The state transition diagram of the OCC_STATE

the NORMAL state. During the tracking process, each time when a new object is matched, our algorithm examines the possibility that whether this object will collide to other objects or not in the next few frames by estimating the object position. If it is possible to collide, then the object goes to the COLLISION state. When an object in previous frame is left unmatched after the matching process, the COLLISION state can be used to judge if an occlusion occurs or an object disappears since both of the cases will lead to failure in matching a previous object to a current object. If it is in the case that the object occludes the other objects, then it goes to the OCCLUSION state. On the other hand, if the object in the COLLISION state will not collide with any other objects, it goes back to the NORMAL state. Similarly, for the current object that fails to match any previous objects, the OCCLUSION state can be used to judge if the objects with explicit occlusion split to two or if a new object appears. Once the occluded object splits, the state goes back to the NORMAL state. However, in the case of implicit occlusion, the objects occlude outside the camera view and the occlusion event cannot be observed. Therefore, the implicitly occluded objects are not in the OCCLUSION state and the OCC_STATE remains unchanged when the objects split.

With these indicating states, the tracking algorithm can handle complicated conditions without ambiguity. In the next section, we will show how the object matching algorithm

utilizes these states to match objects.

3.3.3 Objects Matching Algorithm

The object matching algorithm in the sub-module in Fig. 4 tries to find matches for the objects using the matching function. Because our algorithm tries to handle various conditions, simply matching the objects detected in current frame to the objects detected in the previous frame is not enough. In our matching algorithm, the process is divided into several stages. For short, the objects detected in the current frame and the objects detected in the previous frame are denoted as CurrObj and PrevObj respectively, and the diagram is shown in Fig. 9.

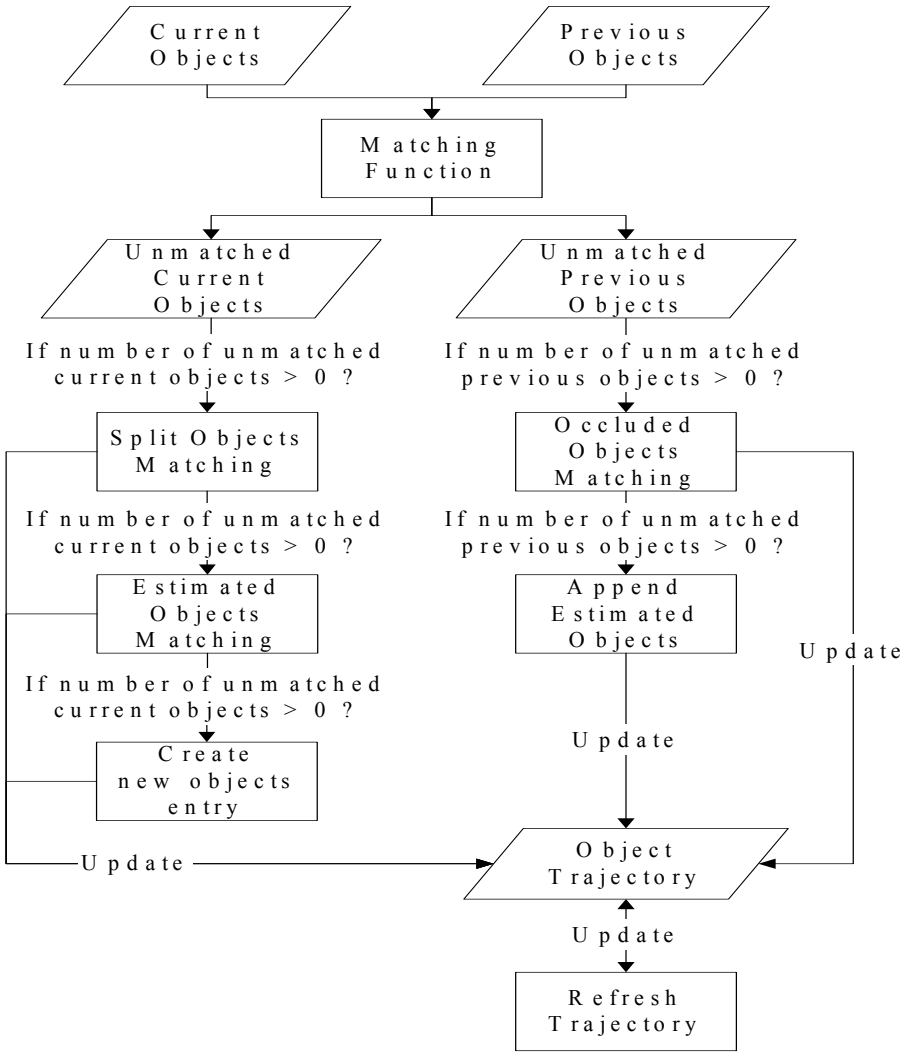


Fig. 9. The process of the whole matching algorithm

In the matching algorithm, the objects detected in current frame and the objects detected in previous frame are taken into the matching function to find a best match. In our algorithm, we use the object trajectory to stores the tracked objects in each frame for each object entry.

So, the current objects that found matches here are appended to their object trajectories respectively. Because the matching function terminates when no more good matches could be found, there could be some current objects and previous objects left unmatched and they are stored in the CurrObjRest and PrevObjRest respectively. The current objects that left unmatched may be resulted from the appearance of new objects, the split of occluded objects or temporally disappeared objects revealed. Similarly, the previous objects that left unmatched may be resulted from the occlusion or the disappearance of the objects. Sub-modules designed to handled the events and condition will be presented in detail. The whole process of the matching algorithm will be presented after these sub-modules are presented.

3.3.3.1 Occluded Objects Matching

For those previous objects that left unmatched, our matching algorithm first tries to find if there is any objects occlusion events. As mentioned earlier, since both the conditions of occlusion and disappearance of objects will leave the previous objects unmatched, the COLLISION state must be used to judge if there is an occlusion event. Fig. 10 illustrates the relationship of occluded objects and Fig. 11 shows the diagram of occlusion objects matching process. Assume that the video objects VO_{t-1}^{1} and VO_{t-1}^{2} in time t-1 may collide with each other in the future, so both the objects are in the COLLISION state and we define that these two objects are in the same 'collision group'. In addition, we also assume that the two objects occlude at time t and thus only one isolated object is detected. As described earlier, the current

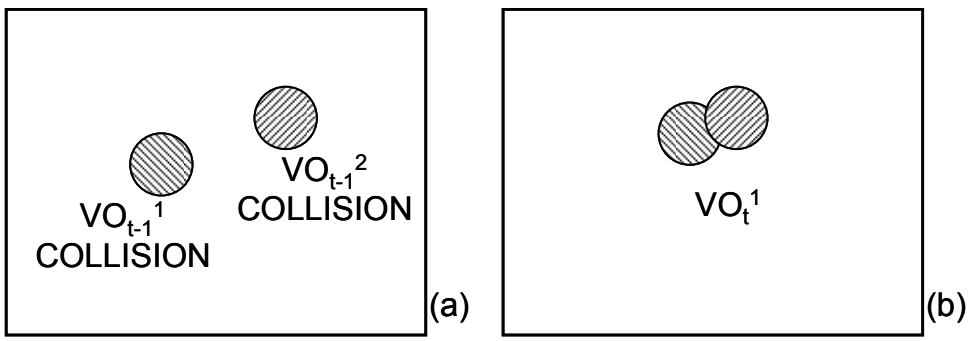


Fig. 10. The relationship of occlusion objects;
(a) Before occlusion; (b) after occlusion

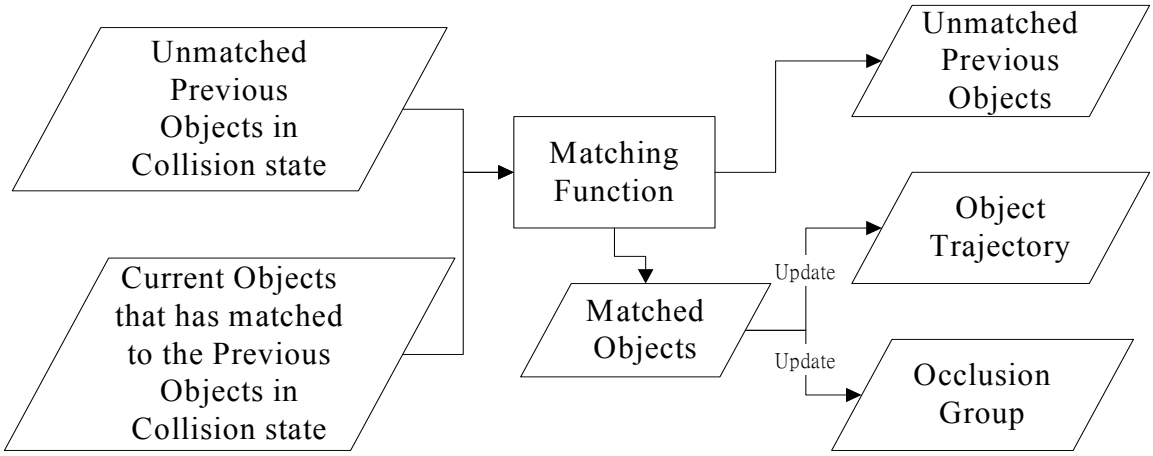


Fig. 11. The occluded objects matching process

objects and the previous objects are first taken into the matching function and thus one of the objects in t-1, says VO_{t-1}^{2}, is matched to the VO_{t}^{1} at time t. After the 1-to-1 matching function, the video object VO_{t-1}^{1} is left unmatched. To handle the occlusion events, our algorithm first checks the possibility of objects occlusion by examining the COLLISION state of the unmatched previous object, for example the VO_{t-1}^{1} in Fig. 10, and the previous object in COLLISION state is taken into the matching function. Then, the current object that has being matched to the previous object which is also in the 'collision group' is also taken into the matching function. If a match is successfully found, it implies that there is indeed an occlusion since the current object can match to the previous objects in COLLISION state and it satisfies the real situation of the objects occlusion.

Once the event of the objects occlusion is detected, both the occluded objects go to the OCCLUSION state and they share the same object trajectory until they split into two. We define that these occluded objects are in the same 'occlusion group'. Note that because the individual motion is required to track the each object when the occluded objects split, our algorithm keeps estimating the individual trajectory when the objects are occluded. The tracked objects are appended to respective object trajectories.

3.3.3.2 Split Objects Matching

The split objects matching process handles the event of object split and matches the split objects. The relationship of the split objects and the diagram of the process are shown in Fig. 12 and Fig. 13. Assume the object VO_{t-1}^{1} in time t-1 is merged from two objects and they split

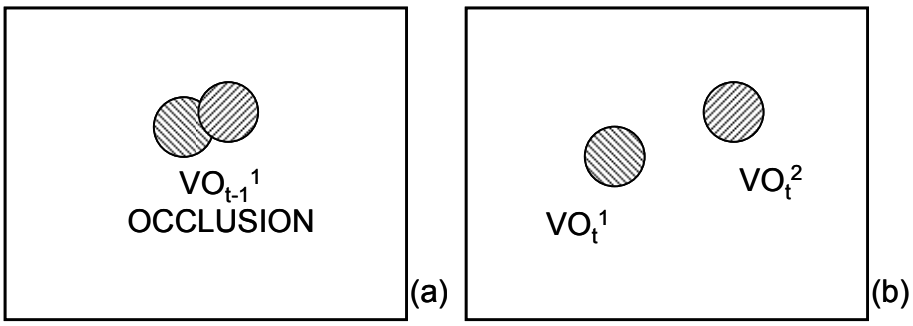


Fig. 12. The relationship of split objects;
(a) Before splitting; (b) after splitting

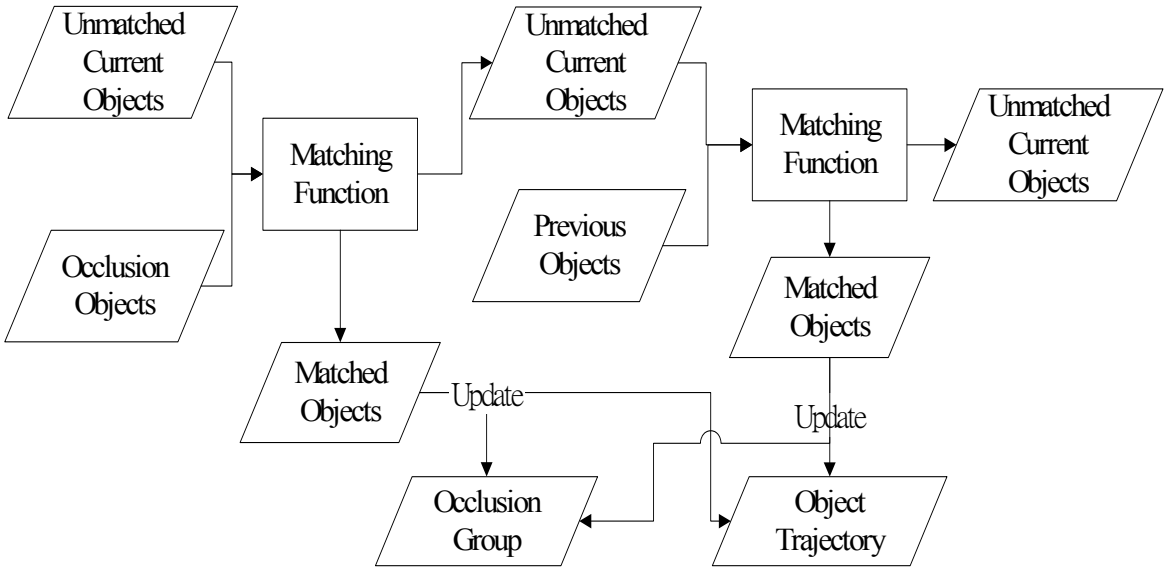


Fig. 13. The split object matching process

into two objects, VO_t^1 and VO_t^2, at time t. According to the matching function performed on current and previous objects, the previous object VO_{t-1}^{1} is matched to one of the current objects, for example VO_t^2. Therefore, the previous object VO_t^1 is left unmatched. Remember that there are explicit occlusion and implicit occlusion. Therefore, the conditions of split event become more complex. Because we cannot judge the possibility of split event simply with the OCCLUSION state, we need to divide the split object matching process into two steps.

In the first step, we try to detect the split events from explicitly occluded objects. First,

we find all the objects which are in the OCCLUSION state from the object trajectory lists and the objects that have not been matched to current objects are picked. Take Fig. 12 for example. Although there is only one previous object, there is another object trajectory in OCCLUSION state. Then, the unmatched current objects and the objects we picked are taken into the matching function. Note that because the estimated motion for each individual object is used here. If a good match is found, it implies some occluded objects now split because the previously occluded objects now match to two objects individually. In this case, the split objects go back to the NORMAL state and the occlusion group for the occluded objects is deleted. Then the tracked current objects are appended to their respective object trajectories.

If the number of unmatched current objects is not zero, the second step is performed. In the second step, we try to detect the split events from implicitly occluded objects. The process is quite similar. However, instead of finding objects in OCCLUSION state from the object trajectories, all the previous objects are used for matching here since we cannot find any OCCLUSION flag in implicitly occluded objects. If a good match is found, the implication is that one previous object matches to two current objects, which means the split event of implicitly occluded object. In this case, our algorithm creates a new object trajectory for the object that splits out and the tracked objects before splitting are duplicated.

3.3.3.3 Estimated Objects Matching

Because we do not think that the object is dead soon after it disappears, we append an estimated object to its object trajectory for later matching process. We use the object information in the past few frames to predict the position and the motion of the estimated object. When the temporally disappeared object now reveals again in the current frame, therefore, we must pick up the estimated objects for matching. Fig. 14 shows the situations that an estimated object is used and Fig. 15 shows the diagram of the estimated object matching process.

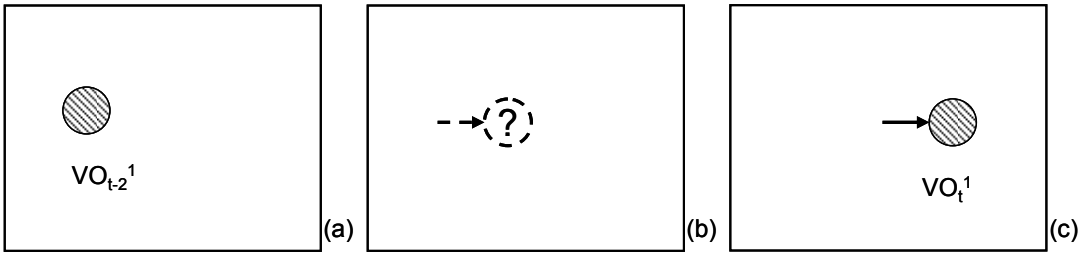


Fig. 14. The condition that estimated object is appended (a) Time t-2; (b) Time t-1, the object disappears, and an estimated object is appended ;(c) Time t, the object reveals again and is going to be match to the estimated object

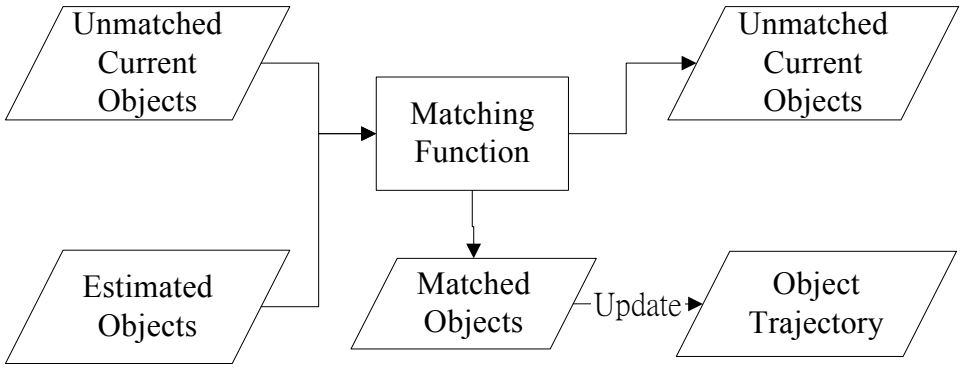


Fig. 15. The estimated objects matching process

This matching process first finds all the object trajectories in the DYING state and picks up the estimated objects from these object trajectories. Then, these estimated objects and the unmatched current objects are taken into the matching function. If an estimated object successfully matches to an unmatched current object, the current object is appended to the object trajectory of that estimated object and the OBJ_STATE goes back from the DYING state to the NORMAL state.

After all the sub-matching modules are presented, we now illustrate the process of the matching algorithm. The current objects and the previous objects are taken into the matching function, and the matching function terminates when no more good match can be found. If there are any events such as appearance, disappearance splitting and occlusion of objects, some current objects and previous objects will be left unmatched. As shown in Fig. 9, for the unmatched previous objects, first the matching algorithm performs the occluded objects matching process to check whether there are any objects occlusion events and tries to find

matches. If there are still any previous objects that cannot find a match, we think that there are objects disappeared. For the object trajectories of these unmatched previous objects, we make the OBJ_STATE go to the DYING state and estimated objects are appended.

For the unmatched current objects, first it goes to the split objects matching process to check whether there are any object split events and try to find matches. The current objects that still cannot find matches will then go to the estimated objects matching process. After the estimated objects matching process, we will consider the rest of current objects as new objects and new entries for the object trajectories will be created.

Finally, the matching algorithm goes to the refresh trajectory function. In this function, the objects in DYING state are first examined. If the object stays in the DYING state too long, we will consider that the object is really disappeared for\ever and let it go to the DEAD state. After that, the motion vector of each moving object is re-computed using the position of the newly tracked object position. Finally, based on the tracked object positions and the computed motion vectors, the function examines if any two objects may collide with each other in the near future. Each time after the matching algorithm finishes matching and processing all the objects, the tracking algorithm will pass the object to temporal filtering process.

3.3.4 Temporal Filtering

The temporal filtering process here is designed to filter out the ghost effect. Because ghost usually appears and disappears very quickly, we can use the temporal filtering to filter out the ghost objects. In our algorithm, we will not think a detected and tracked object valid unless it survives more than a time period. In other words, an object that goes to DEAD too soon after it appears will be filtered out and excluded from the key objects selection process in the video abstraction algorithm.

Another objective of the temporal filtering process is to smooth the object motion. In the trajectory of a non-rigid object like a walking person, because the connected path of the mass center fluctuates like zigzags, the precision of the analysis of moving direction is seriously affected. Therefore, the paths of the moving objects need to be smoothed in the temporal filtering process. Fig. 16 shows the zigzag-liked moving path and the smoothed moving path. The solid lines represent the true motion vector by connecting the mass centers and the direction of the dash line represents the moving direction after temporal filtering.

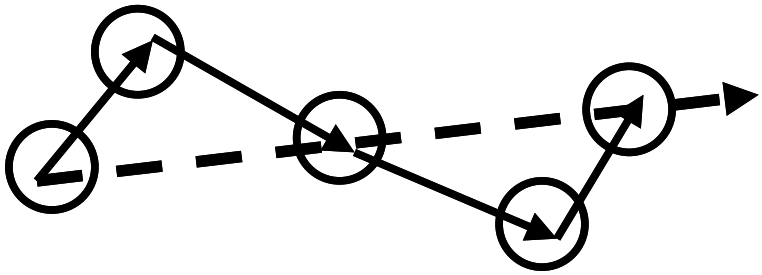


Fig. 16. The path of the mass center of a walking person

3.4 Video Abstraction Algorithm

The last module in the system is the video object abstraction module. The video abstraction is generated using the video abstraction algorithm by selecting the key frames with meaningful semantics. Because the moving objects are the most important parts in surveillance videos, the selection of important key frames is equivalent to the selection of important key objects. Therefore, in our video abstraction algorithm, we will analyze the tracked object trajectories and detect object events to extract representative key objects.

Although the best and the most representative key objects of an object trajectory can be selected after the life cycle of that object is terminated, this kind of approach is not suitable for a real-time tracking system like ours. In order to achieve on-line alarming on real-time tracking system, the key objects must be selected near real-time, which means the delay must be bounded and very small. Therefore, every time a new frame comes in, our algorithm examines the current tracked object in each trajectory and selects it as a new key object if it is representative enough for its trajectory.

One of the criteria for key object selection is based on the object events which are representative for some object states or objects relationships at some time instant. Such events may raise our human's interests. There are some important object events in general domains, such as appearance and disappearance. Besides, the motions and the positions of the object may also be used as the selection criteria. For example, we may have interests and pay more attention when a new object appears or the moving direction of the object changes because they can represent significant events. Therefore, the analysis of the object trajectories to extract this specific information is required.

The diagram of the abstraction algorithm is shown in Fig. 17. There are three modules in this algorithm. The algorithm takes the object trajectories generated in the video object tracking algorithm as input. The abstraction will be generated by selecting the frames with

key objects and output to clients.

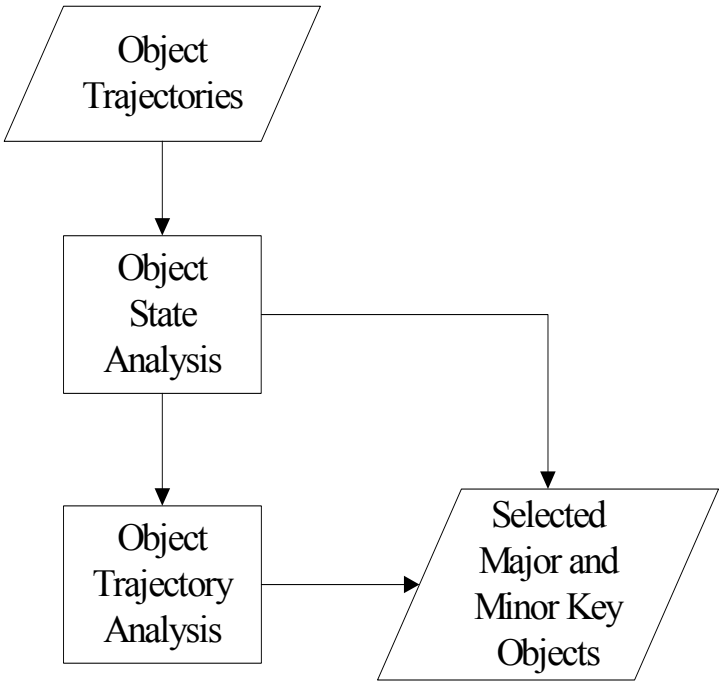


Fig. 17. The abstraction algorithm

3.4.1 Object State Analysis

The object state analysis process detects the general object events such as appearance, disappearance, occlusion and split of objects. Because we have handled and detected these events for object matching in the tracking algorithm, we can directly capture these events by examining the state transition of OBJ_STATE and the OCC_STATE of the objects. The only exception is that we do not directly extract the event when an object appears because the temporal filtering is applied to filter out the ghosts. Therefore, the events of object appearance will only be captured when the object survives for a period of time after it appears.

3.4.2 Object Trajectory Analysis

The object trajectory analysis process tries to analyze the trajectories to find the featured objects as key objects. The featured objects are representative for the changes in the moving speed and direction, the position in the frame view or the object size. Every time when new an object is tracked, our algorithm compares the motion, position and size of that object to those object features of the previously selected key object. To evaluate the motion difference of current object and the previously selected key objects, the motion vector distance function described in section 3.3.2 is used. However, to avoid the zigzag-like paths for non-rigid objects to affect the analysis of motion direction, the motion vector after temporal filtering is used. Fig. 18 shows the analysis process.

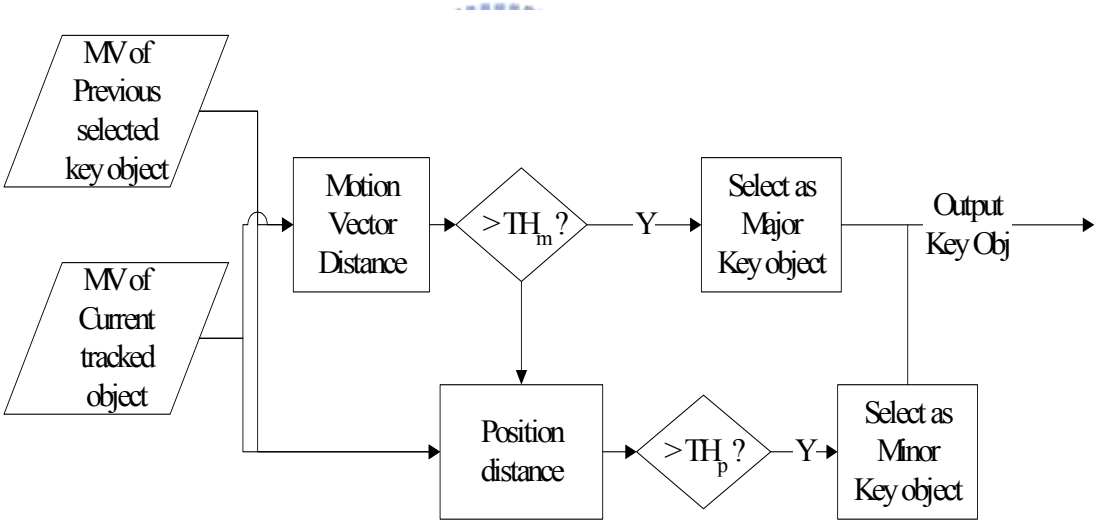


Fig. 18. The trajectory analysis and key object selection process

3.4.3 Video Abstraction with Selected Key Objects

After the object event detection process and the object trajectory analysis process, the abstraction can be output using the selected key objects. In our algorithm, we define two types of key objects: major key objects and minor key objects. The major key objects represent important event and are always exported. On the contrary, the minor key objects are less important and are exported only when there is no other key object exported recently. All the

key objects selected in object state analysis process are major key objects. Besides, the objects which change in motion significantly are also selected as major key objects. The key objects which are selected using the position as the criterion are minor key objects. Fig. 19 shows how the algorithm selects the key objects to export.

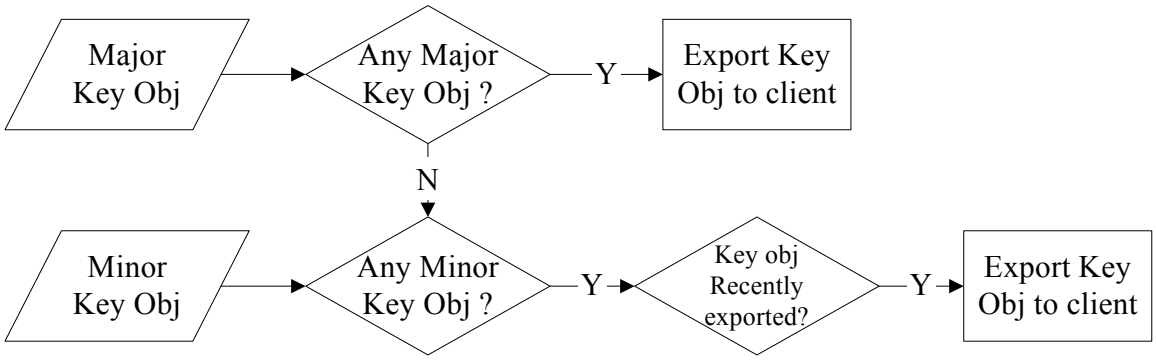


Fig. 19. The key objects exporting process

Chapter 4

System Architecture and Experiment Result

In this chapter, we will present the system for object-based video tracking and abstraction. In the section 4.1, we will first show an overview of the system architecture. In the sections 4.2 to 4.4, the experiment results of each module will be represented. The implemented system will be presented in the section 4.5.

4.1 System Architecture Overview

In this thesis, we implemented an object-based tracking and abstraction system on surveillance videos. The raw video data captured lively are input to our system and the process of object segmentation, object tracking and video abstraction are performed on-the-fly. The abstraction is used for on-line alarming at the client while the surveillance video can be monitored simultaneously. Except some predefined thresholds, all the initializations are done automatically without manual interactions. Fig. 20 shows the overview of the system.

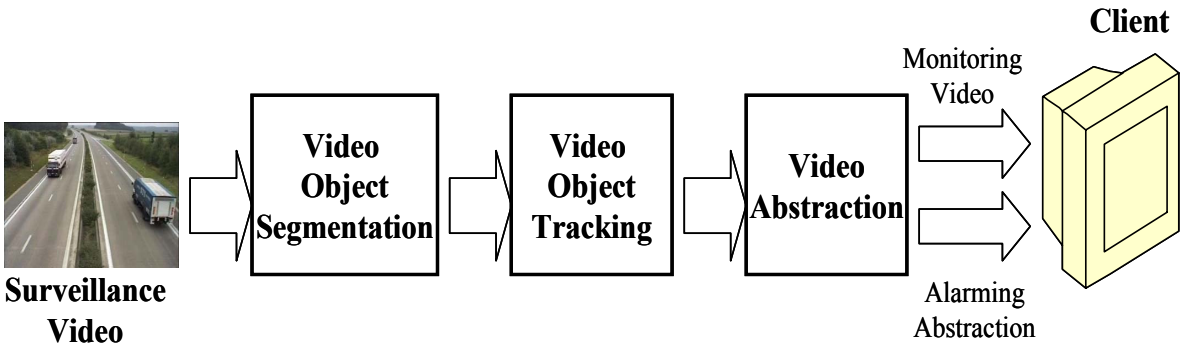


Fig. 20. System architecture overview

4.2 Experimental Results of the Video Object Segmentation

In the segmentation algorithm, we use first 40 frames for background initialization before we start segmentation. In the morphological operations, the size of structuring element for closing operation is 7 by 7 and the size for opening operation is 5 by 5. The size threshold used for size filtering is 450 pixels.

Fig. 21 presents the segmentation results of the ETRI_B clip at frame NO.95. Fig. 21(a) shows the original image and Fig. 21(b) shows the performance of using the luminance and chrominance together to segment video objects. For reference, Fig. 21(c) shows the segmentation result that only luminance channel is used. The results show that combining the luminance and the chrominance to segment object can improve the segmentation results a lot.

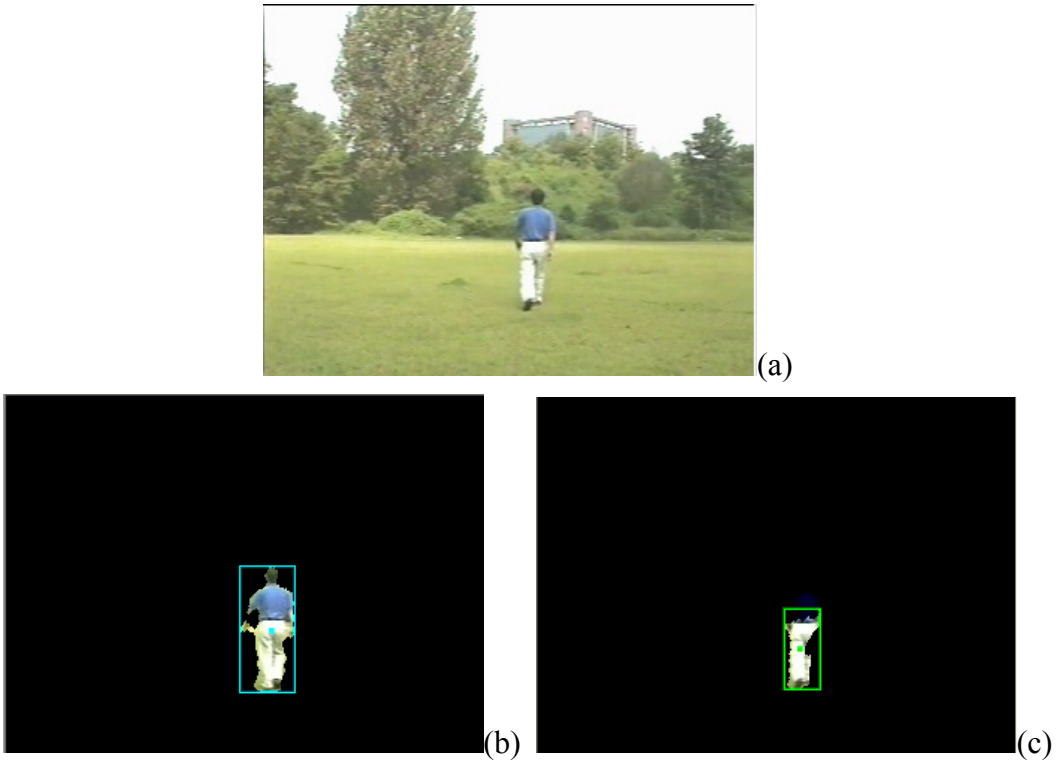
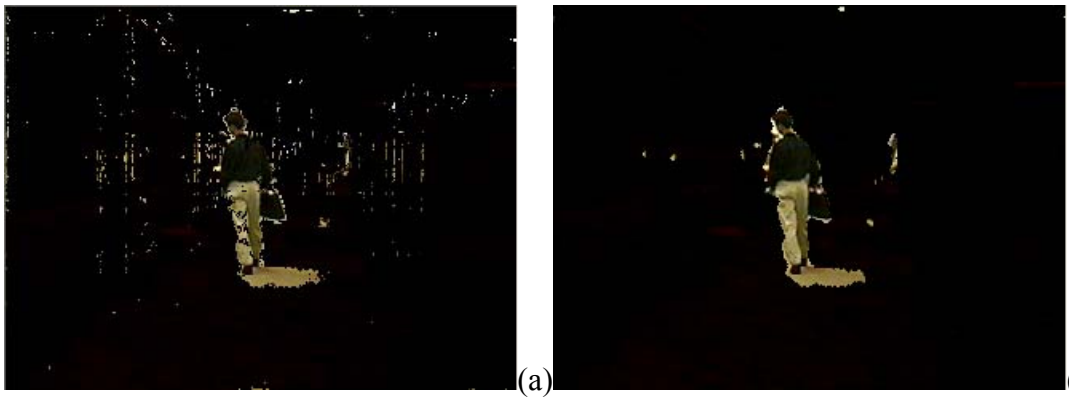


Fig. 21. (a) original image; (b) segmentation result that luminance and chrominance channel are used; (c) segmentation results that only luminance channel is used

Fig. 22 shows the result of the clip “hall monitor” after applying the morphological operation. The small noises are removed and the black holes inside the objects are filled. The

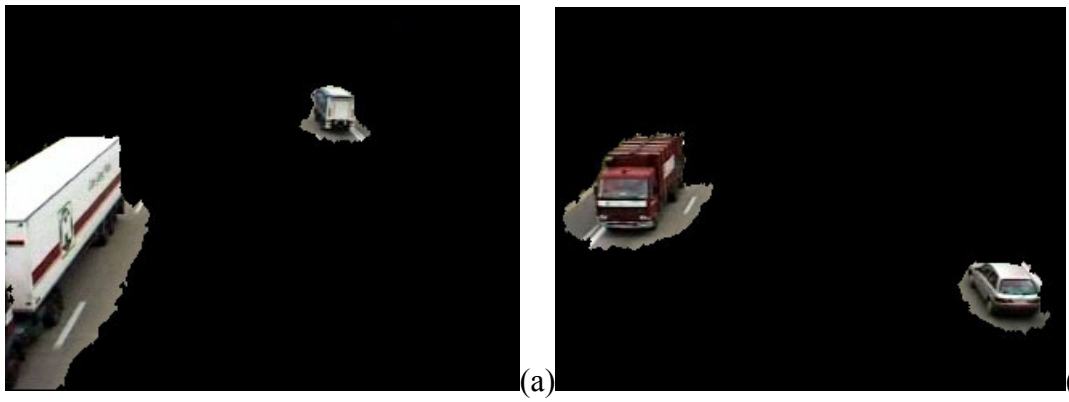
bigger noise regions left in Fig. 22 (b) will be filtered out in the size filtering process.



(a) (b)

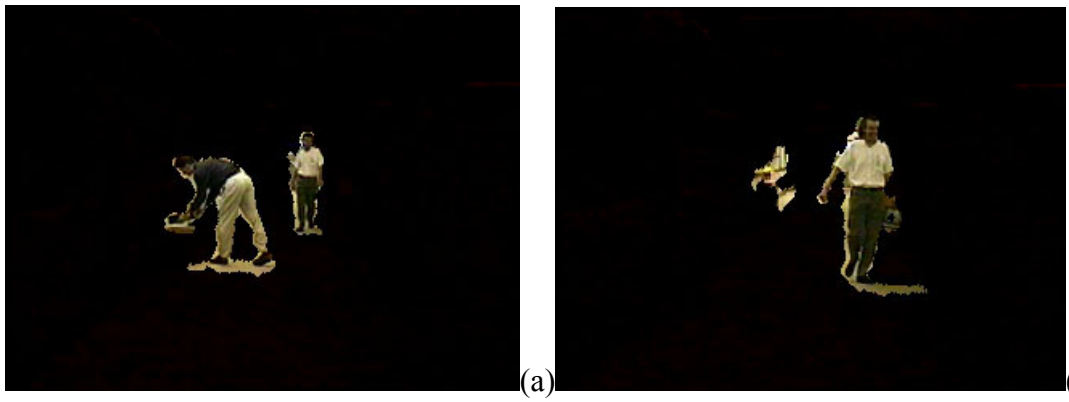
Fig. 22. (a) the segment result before the morphological operation; (b) the result after the morphological operation

Figs.23 through Fig. 27 show some results of the segmentation algorithm and the tested video sequences.



(a) (b)

Fig. 23. Segmentation results of the clip speedway at frame (a) #585; (b) #673;



(a) (b)

Fig. 24. Segmentation results of the clip hall monitor at frame (a) #114; (b) #273;

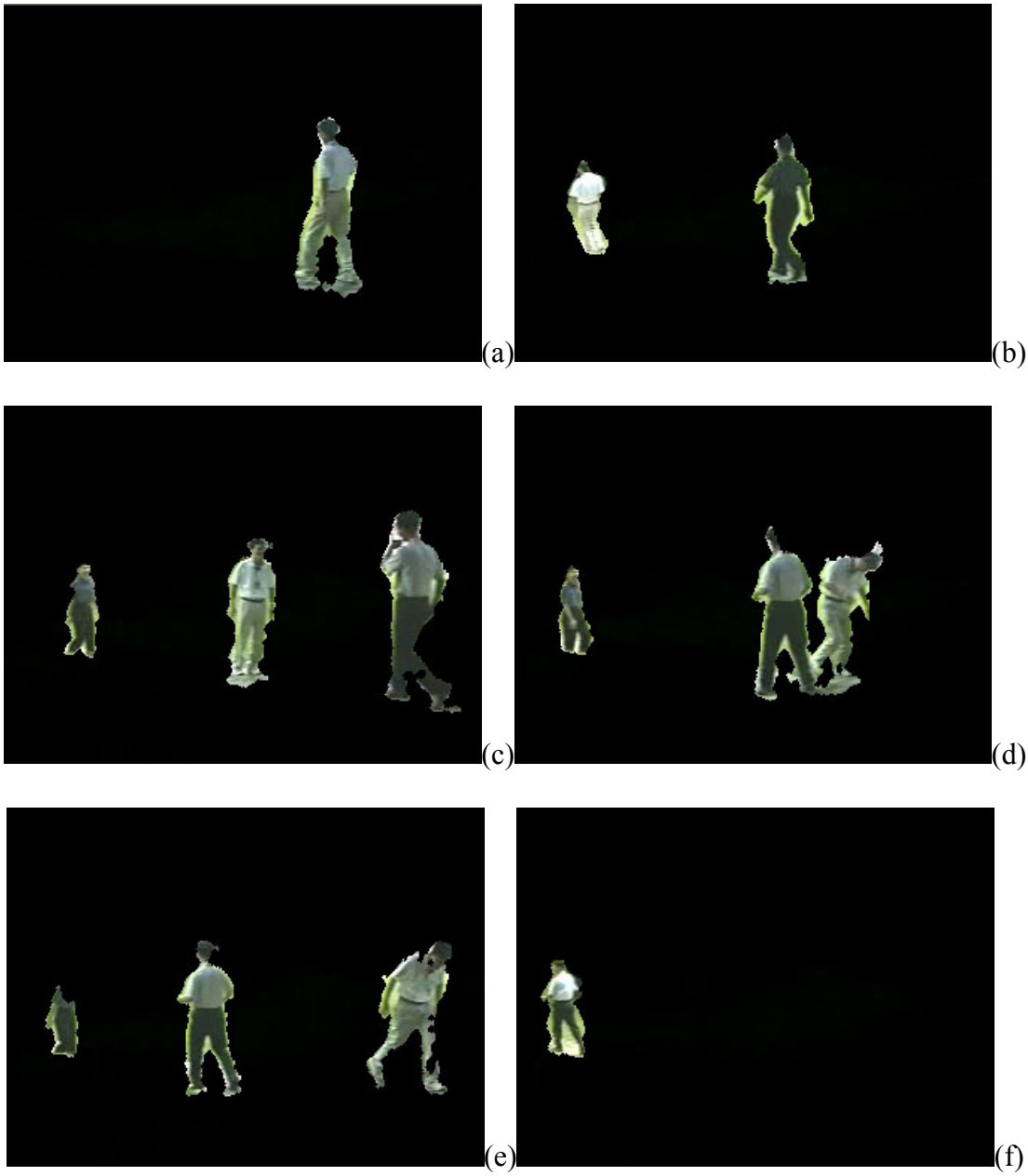


Fig. 25. Segmentation results of the clip ETRI_A a at frame
(a) #95; (b) #120; (c) #360; (d) #470; (e) #567; (f) #617;

Fig. 26. Segmentation results of the clip ETRI_B at frame
(a) #168; (b) #378; (c) #1045; (d) #1267; (e) #1477; (f) #1969; (g) #2511; (h) #2753

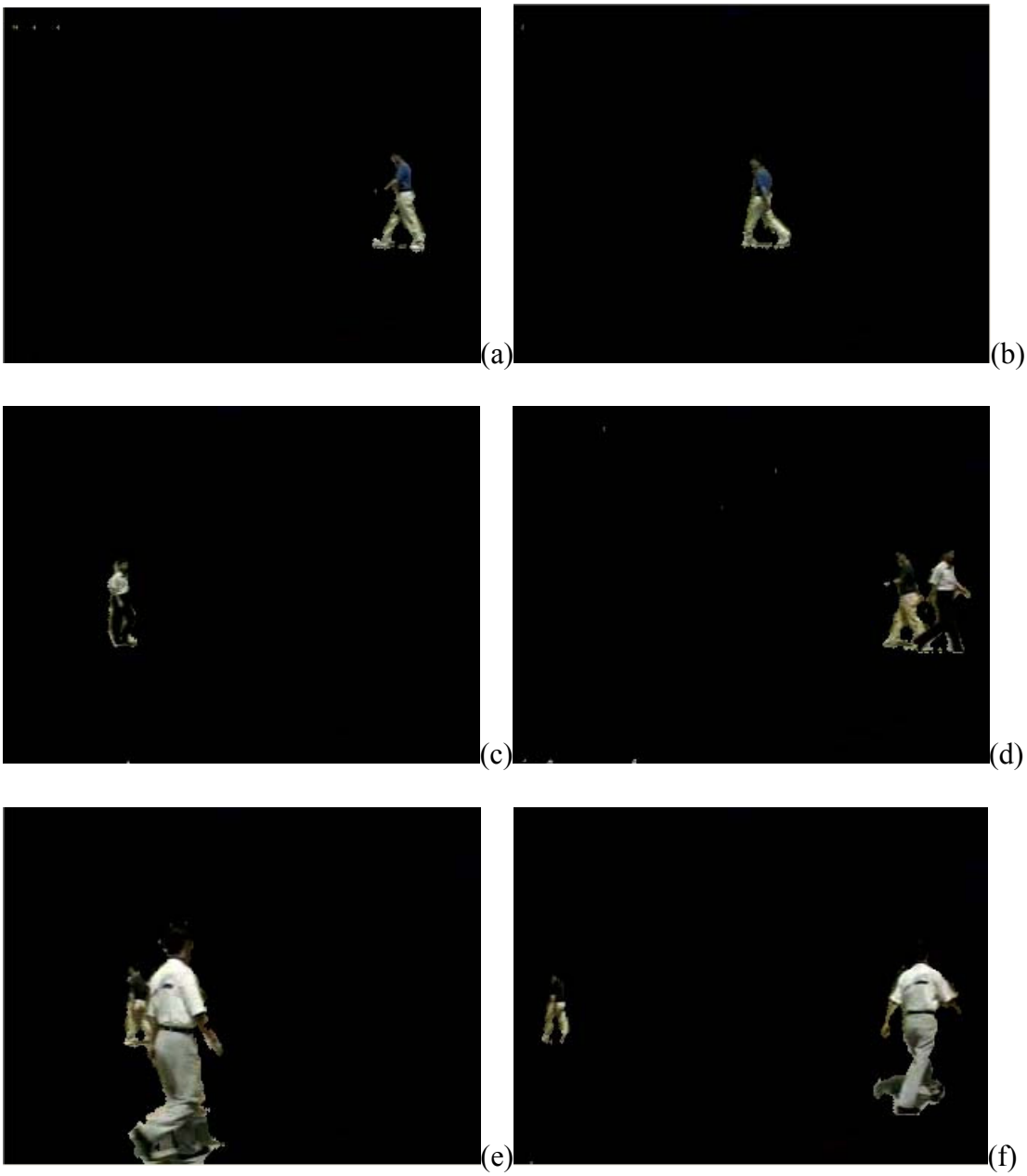


Fig. 27. Segmentation results of the clip ETRI_C at frame
(a) #95; (b) #120; (c) #360; (d) #470; (e) #567; (f) #617;

The results show that most of the noise regions are successfully filtered out. However, the ghost regions in Fig. 24(b) and Fig. 26(b) are not removed out because the size of these regions exceeds the filtering threshold.

In the video clip of Fig. 26, because the sun light varies dramatically, large-scale of regions which are directly illuminated and the color of these regions thus change significantly. Due to this reason, many objects detected in this video clip are false alarmed.

The other false alarmed region in Fig. 24(b) is resulted from the stopped object. Although the person wearing white pants in Fig. 24(a) does not completely stop, the motions in both the x and y direction are almost zero and thus the color of the object region is updated to background. Because the color of the background is distorted and is different from the real background color, the region is false alarmed.

The table 1 is the statistics of the segmentation result. Four video clips of different environments and contents are tested. The first column is the total number of ground truth objects in all the frames of the video clip. The moving regions that can be clearly distinguished are selected as ground truth. The second column is the total number of the object detected after morphological operation and size filtering. The precision and the recall

Table 1. Statistics of segmentation result

Sequence name	Ground Truth	Detected	False alarm	Miss	Precision	Recall
Hall Monitor	463	487	25	1	94.86%	99.78%
ETRI_A	1365	1401	39	3	97.21%	99.78%
ETRI_C	526	513	6	19	98.83%	96.38%
Speedway	410	354	0	56	100%	86.34%

$$precision = \frac{hits}{hits + false\_alarm} \tag{12}$$

$$recall = \frac{hits}{hits + miss} \tag{13}$$

are defined in the Eq. (12) and Eq. (13) respectively.

The false alarms are mainly due to the stopped object and the lighting variation. The recall rate drops in the speedway sequence because the vehicles are very small when they are far away. Therefore, these vehicles are filtered out although they are detected after the morphological operation. However, these filtered out small objects would not affect the result of video abstraction since they are too small and contain little semantics.

4.3 Experiment Results of the Video Object Tracking

In the tracking algorithm, the threshold TH_{match} used in the matching function is heuristically set to 50. The window size used in the temporal filtering is set to 5.

Fig. 28 through Fig. 30 show the results of video object tracking algorithm. In order to check the result easily, objects which belong to the same trajectory are marked with an identifying label manually.

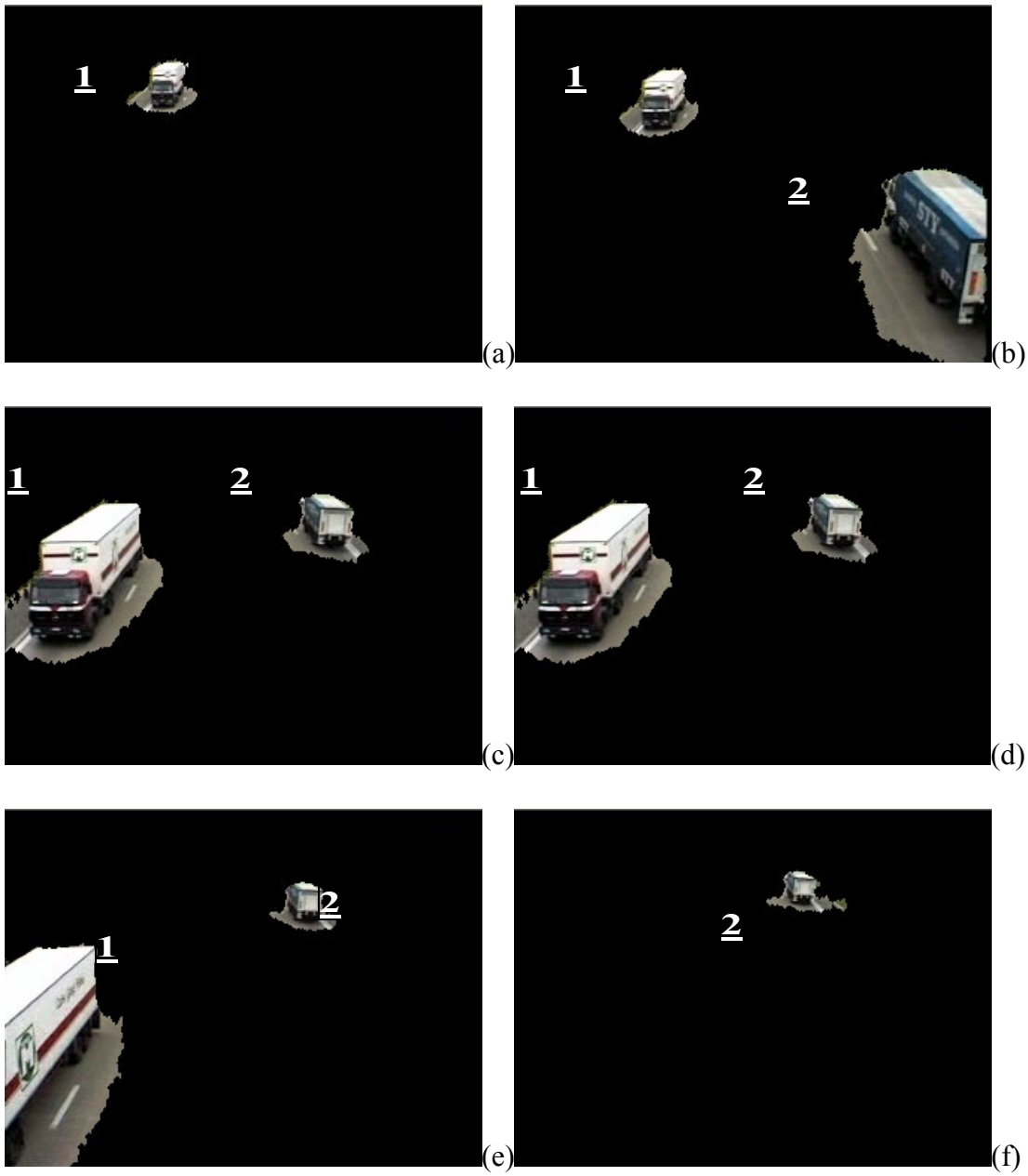
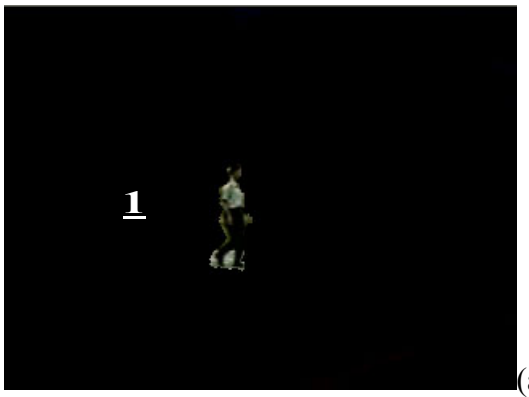
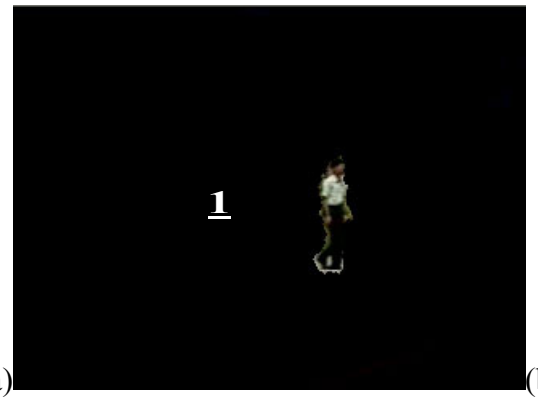
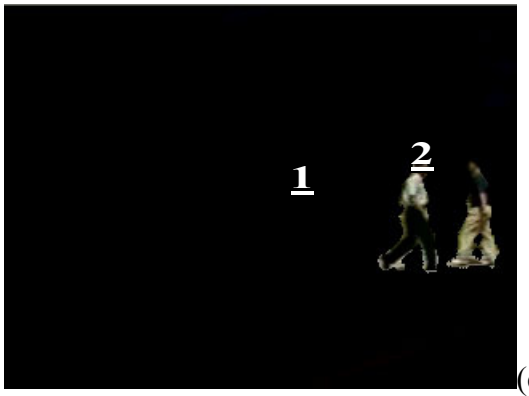
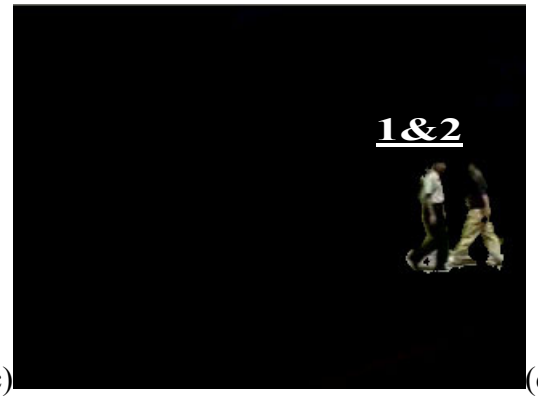


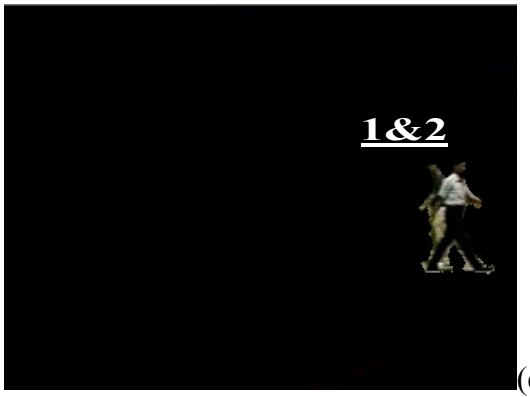
Fig. 28. Tracking results of the speedway sequence
(a) #530; (b) #545; (c) #560; (d) #575; (e) #590; (f) #605;



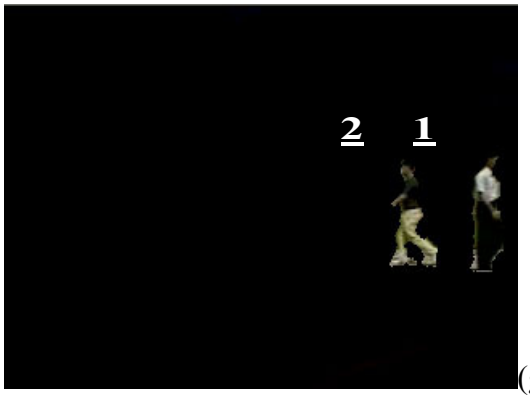
(a)


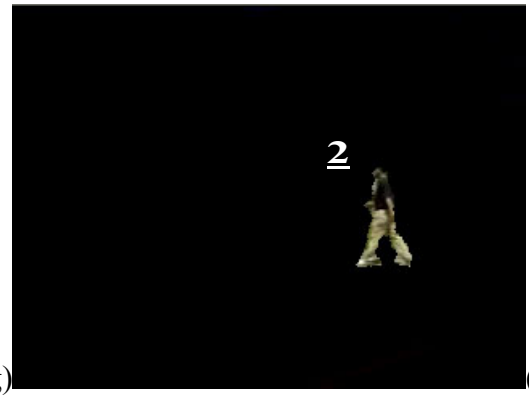
(b)


(c)


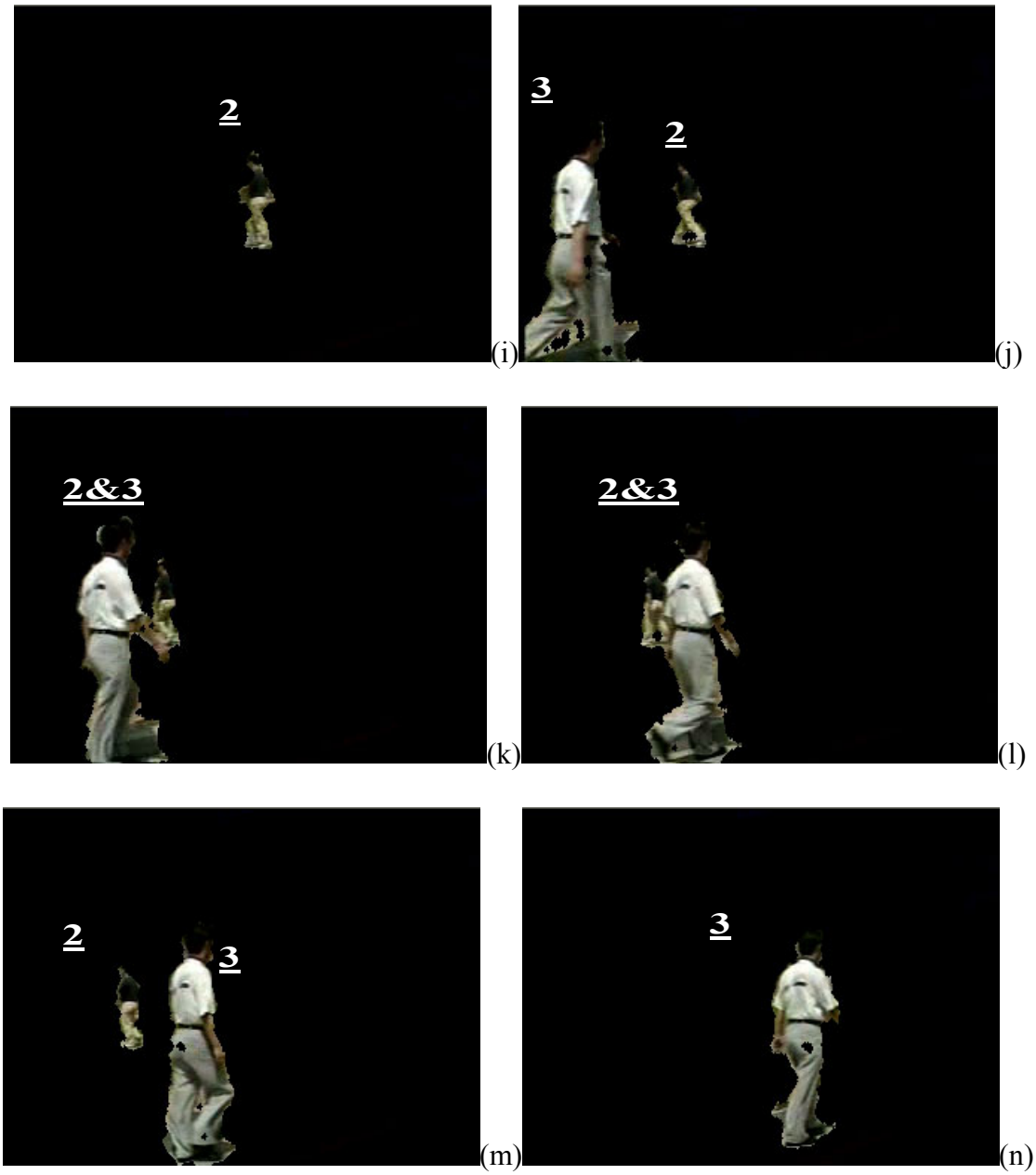
(d)


(e)


(f)


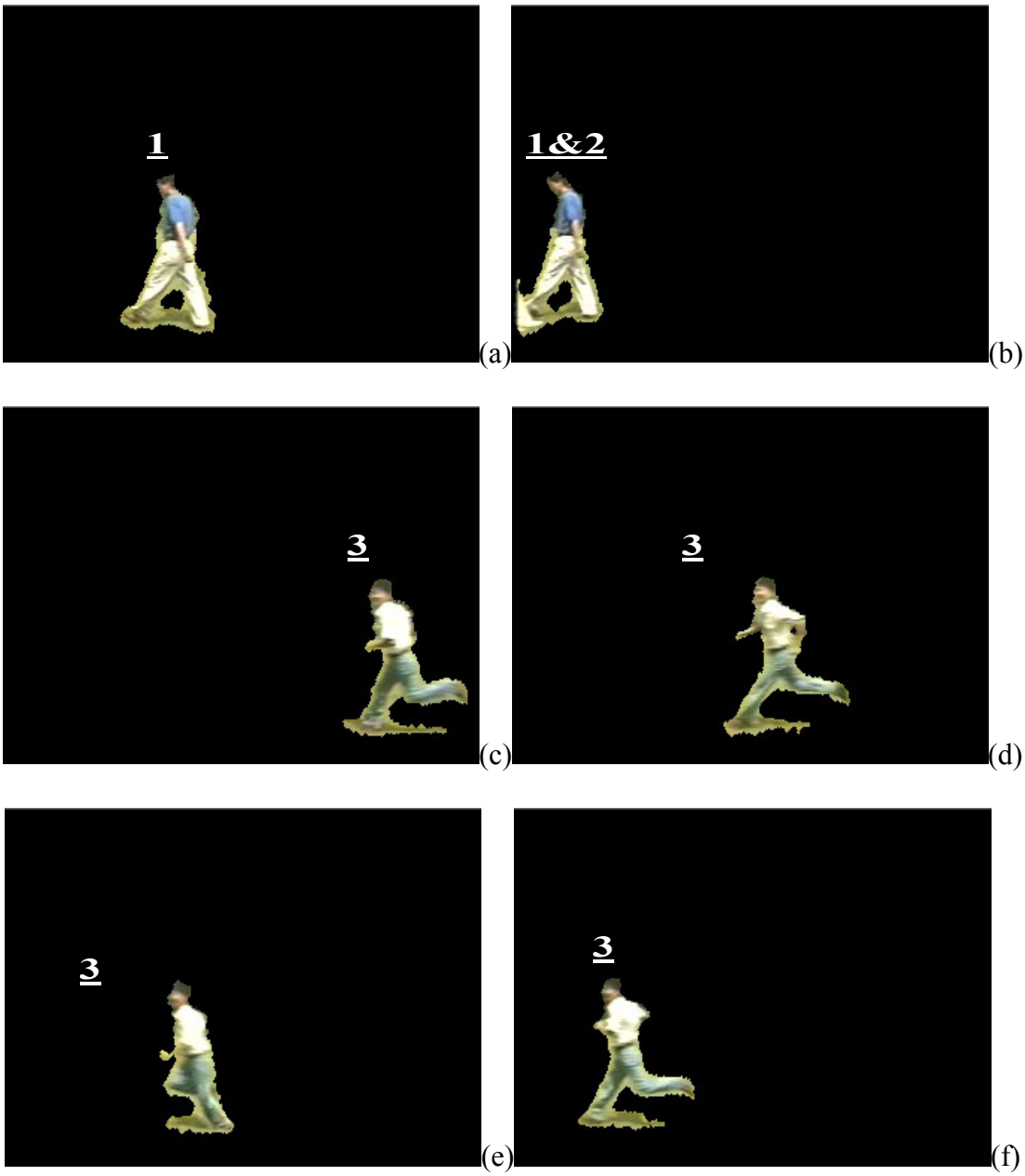
(g)


(h)



(i) (j) (k) (l) (m) (n)

Fig. 29. Tracking results of the ETRI_C sequence
(a) #420; (b) #450; (c) #475; (d) #478; (e) #486; (f) #491; (g) #494;
(h) #505; (i) #540; (j) #570; (k) #576; (l) #586; (m) #589; (n) #605;



(a) (b) (c) (d) (e) (f)

Fig. 30. Tracking results of the ETRI_B sequence
(a) #2190; (b) #2227; (c) #2500; (d) #2508; (e) #2518; (f) #2513;

The results show that the tracking algorithm successfully tracks the video objects and detects the occlusion events. Our algorithm also matches the split objects to the objects before occlusion correctly, as shown in Fig. 29. Besides, objects moving in different speed, for example the person who walks slowly (obj 1) and the person who runs quickly (obj 3) in Fig. 30, are all successfully tracked. The table 2 shows the statistics of the detecting and tracking of occlusion and split events. The results show that all the objects before and after the

occlusions are matched perfectly. The only failure in detecting occlusion events happens in the sequence ETRI_B, which is shown in Fig. 30(a) and Fig. 30(b). Because the object 2 is occluded by the object 1 in the first frame when it enters the camera view, it is impossible to detect the occlusion under such condition.

Table 2. Statistics of the tracking and detecting of occlusion events

Sequence name	Number of occlusion events occurred	Number of occlusion events detected	Number of the matching failures after the split
ETRI_A	3	3	0
ETRI_B	4	3	0
ETRI_C	2	2	0

The statistics in table 3 show the result of the tracking algorithm. Because the sequence "ETRI_C" is too long, we only took the first 3000 frames. Note that when a person is going to walk behind the tree, we consider that the trajectory before he is covered and the trajectory after he is uncovered are two different trajectories since that we can observe the object indeed disappeared for a while. We can see that many several ghost regions can be filtered out with the temporal filter. The false alarms are mainly due to the stopped object effect and the light variation which keeps changing severely. The only missed object is the object which is occluded at the first frame it appears and thus fails to detect the object.

Table 3. Statistics of the tracked trajectories

Sequence name	Ground truth	Tracked	Tracked after temporal filtering	False alarmed trajectory	Missed
speedway	6	6	6	0	0
Hall monitor	2	7	3	1	0
ETRI_A	10	23	15	5	0
ETRI_B	16	38	22	7	1
ETRI_C	22	37	24	2	0

The results can show that our algorithm can robustly track almost all the trajectories and reason the occlusion and split events. Although some false alarms exist, the ghost regions caused by the lighting effect and the stopped object effect can also be filtered effectively. The robustness of our tracking results can be used to extract key objects for abstraction later.

4.4 Experiment Results of the Video Abstraction

In the abstraction algorithm, the thresholds TH_m for the MVD function to decide whether there is significant change in motion is set to 2 pixels. And the TH_p used to decide whether the spatial distance is large enough is set to 80 pixels. The interval for selecting minor key objects is 60 frames or a video sequence has 30fps.

Fig. 31 and Fig. 32 show the selected key objects for the detected occlusion and split events. The objects in continuous frames are listed and the selected key objects for the specific event are marked using a rectangle.

(a) (b) (c) (d) (e) (f)

Fig. 31. Selected key objects for the detected occlusion event

(a) (b) (c) (d) (e) (f)

Fig. 32. Selected key objects for the detected split event

Fig. 33 shows the selected key objects for the 33rd object in the sequence "ETRI_B". The person first walks into the frame (a) and slightly changes the direction (b). After a period of time, because the distance of the object positions in (b) and (c) are big enough, the object in (c) is also selected as key object. After a while, he starts to rush and the key objects are selected in (d) and (e). Finally, the object in (f) is disappearing and is selected as key object.

Fig. 34 shows the 30th object in the sequence "ETRI_B". Because the person keeps jumping in the camera view and the movements are very heavy, thus it is selected as key objects. Fig. 35 through Fig. 37 show parts of the generated abstraction of the video sequence

of “ETRI_C”, “hall monitor” and “speedway”.

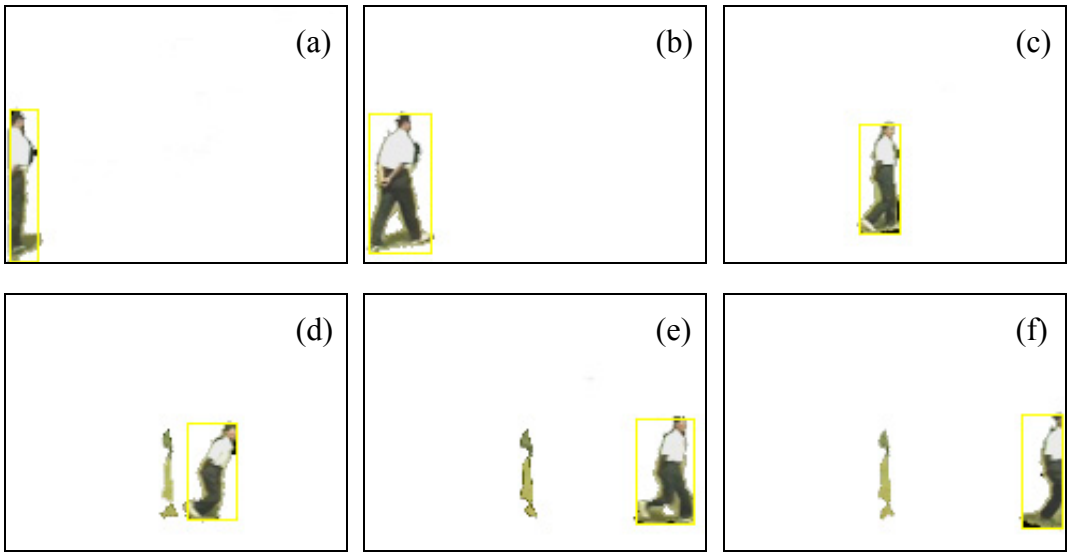


Fig. 33. Selected key objects for the ETRI_B sequence
(a) object appears; (b) change in motion; (c) change in position;
(d) change in motion; (e) change in motion; (f) object is disappearing

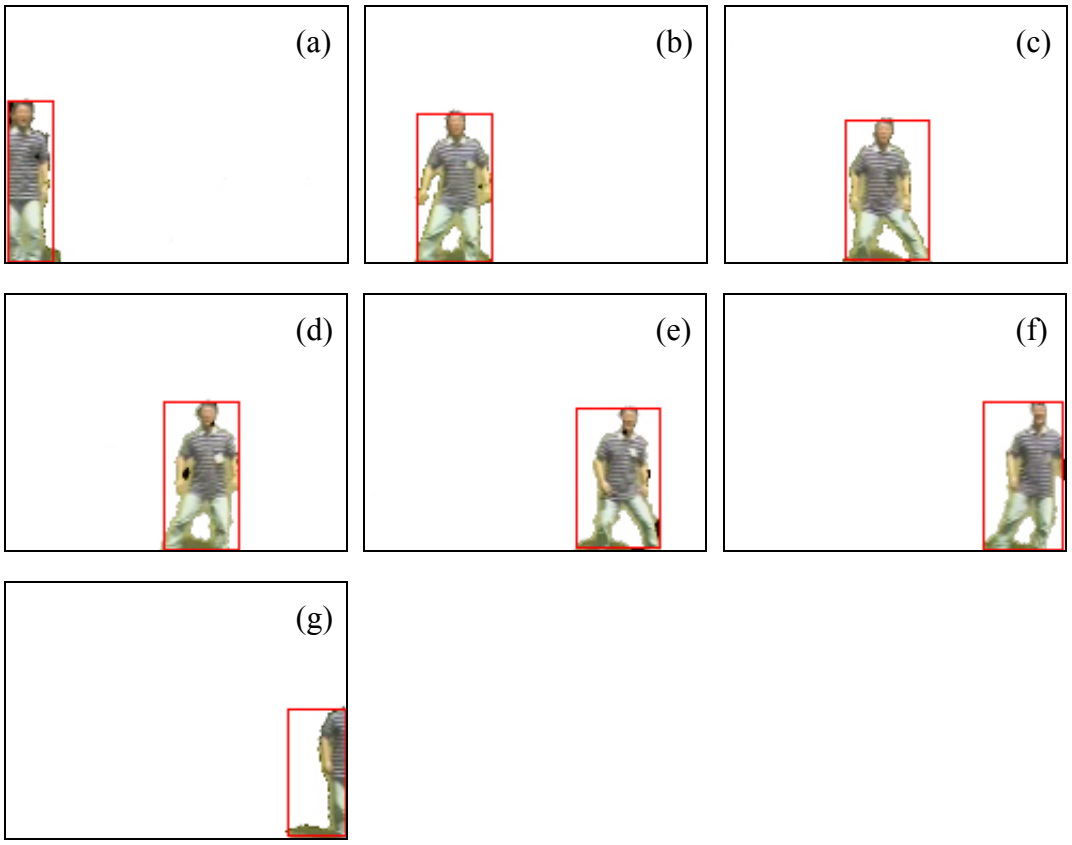


Fig. 34. Selected key objects for the ETRI_B sequence
(a) object appears; (b) change in motion; (c) change in motion; (d) change in motion;
(e) change in motion; (f) change in motion; (g) object is disappearing;

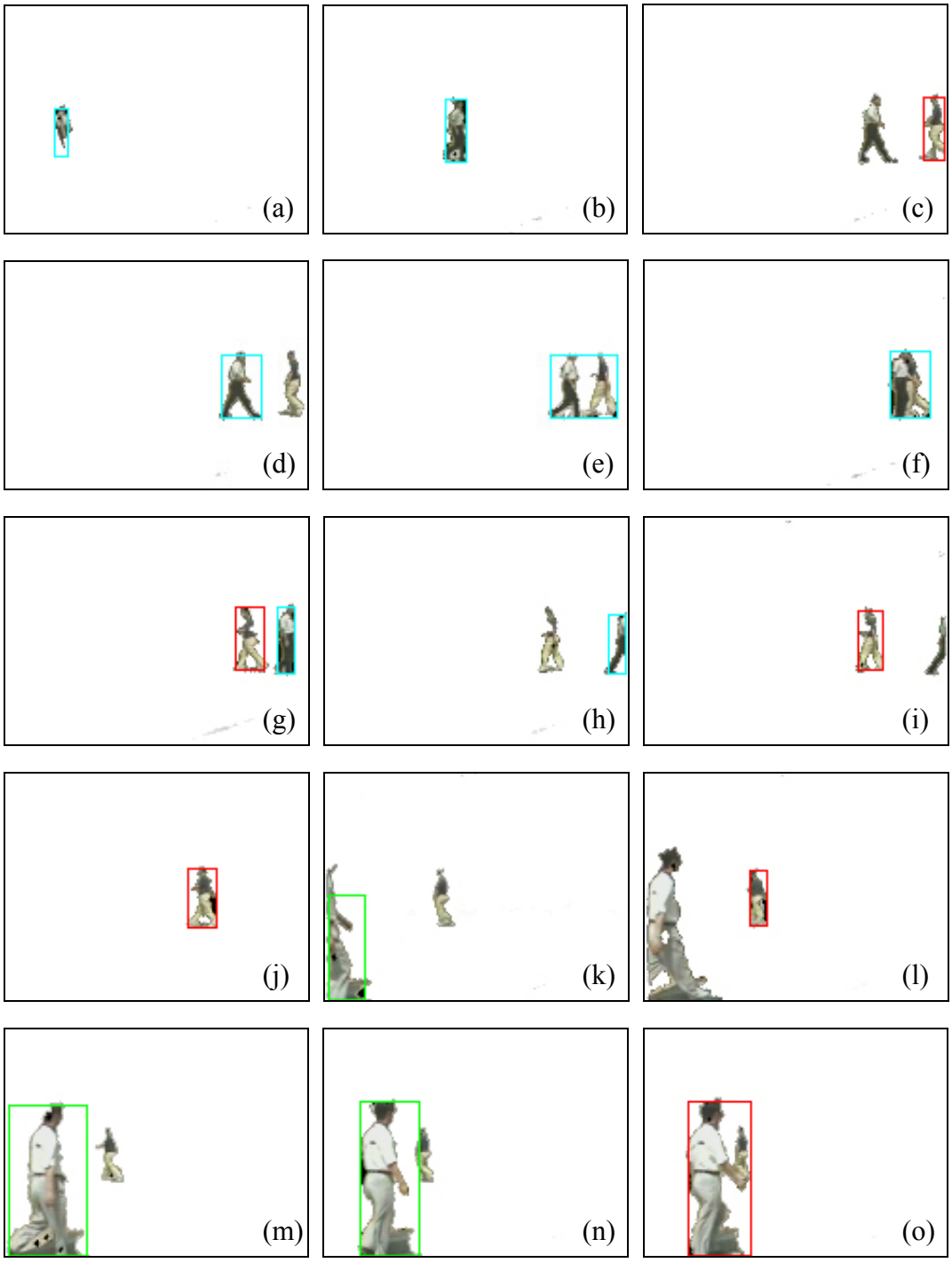


Fig. 35. Parts of the abstraction of the ETRI_C sequence
(a) object appears; (b)change in position; (c) object appears;
(d) change in position;(e)occlusion event; (f)change in motion;
(g) split event; (h)object is disappearing; (i)change in position;
(j) change in motion; (k)object appears; (l)change in position;
(m) change in motion; (n) change in motion; (o)occlusion event

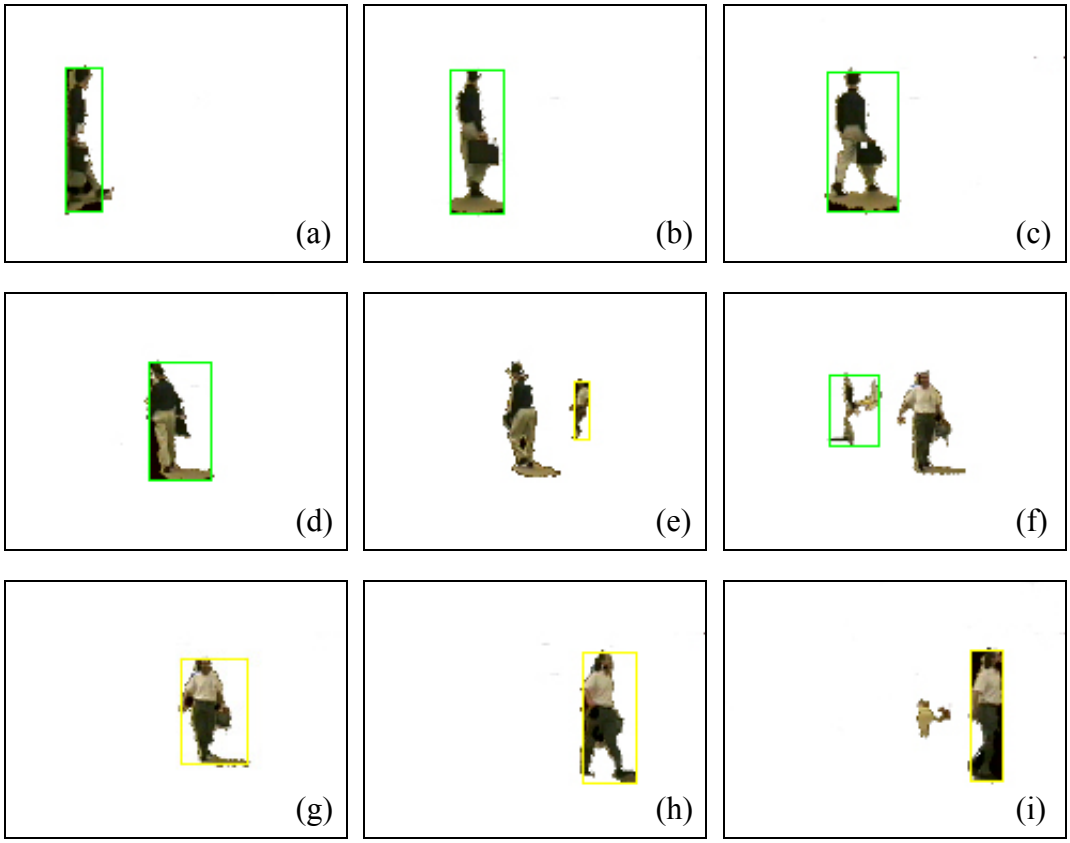


Fig. 36. Parts of the abstraction of the hall monitor sequence
(a)object appears; (b)change in motion; (c)change in motion;
(d)change in motion; (e)object appears; (f)object is disappearing;
(g)change in motion; (g)change in motion; (g)object is disappearing;

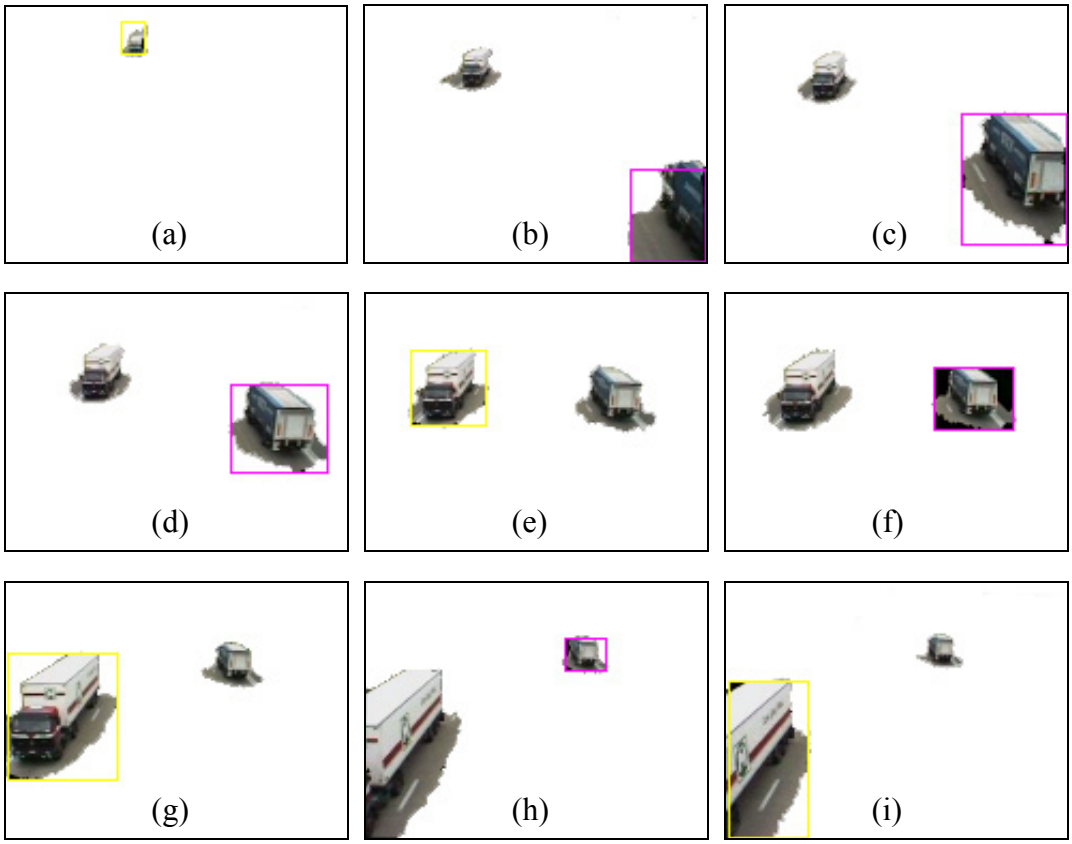


Fig. 37. Parts of the abstraction of the speedway sequence (a)object appears; (b)object appears; (c)change in motion; (d)change in motion; (e)change in motion; (f)change in motion; (g)change in motion; (g)change in motion; (g)change in motion;

Table 4 shows the statistics of the generate abstractions. The results show that the generated abstractions are very compaction and the object-level semantics and events are also represented in the abstractions. In the next section we will show how to integrate the abstraction algorithm to provide on-line alarming.

Table 4. Statistics of the abstraction

Object abstraction	Selected key VOPs.	Total VOPs
Object 1 (ETRI_A)	9	240
Object 7 (ETRI_A)	7	295
Object 3 (ETRI_B	10	140
Object 30 (ETRI_B)	7	59
Object 33 (ETRI_B)	6	167
Object 5 (ETRI_C)	7	176
Object 6 (ETRI_C)	15	171
Object 3 (speedway)	6	128
Object 1 (hall monitor)	5	235

4.5 Integrated System for Real-Time Video Object Tracking and On-Line Alarming

In this section, we will show the system we integrated for real-time tracking and on-line alarming using the algorithm we have implemented. Fig. 38 shows the system interface.

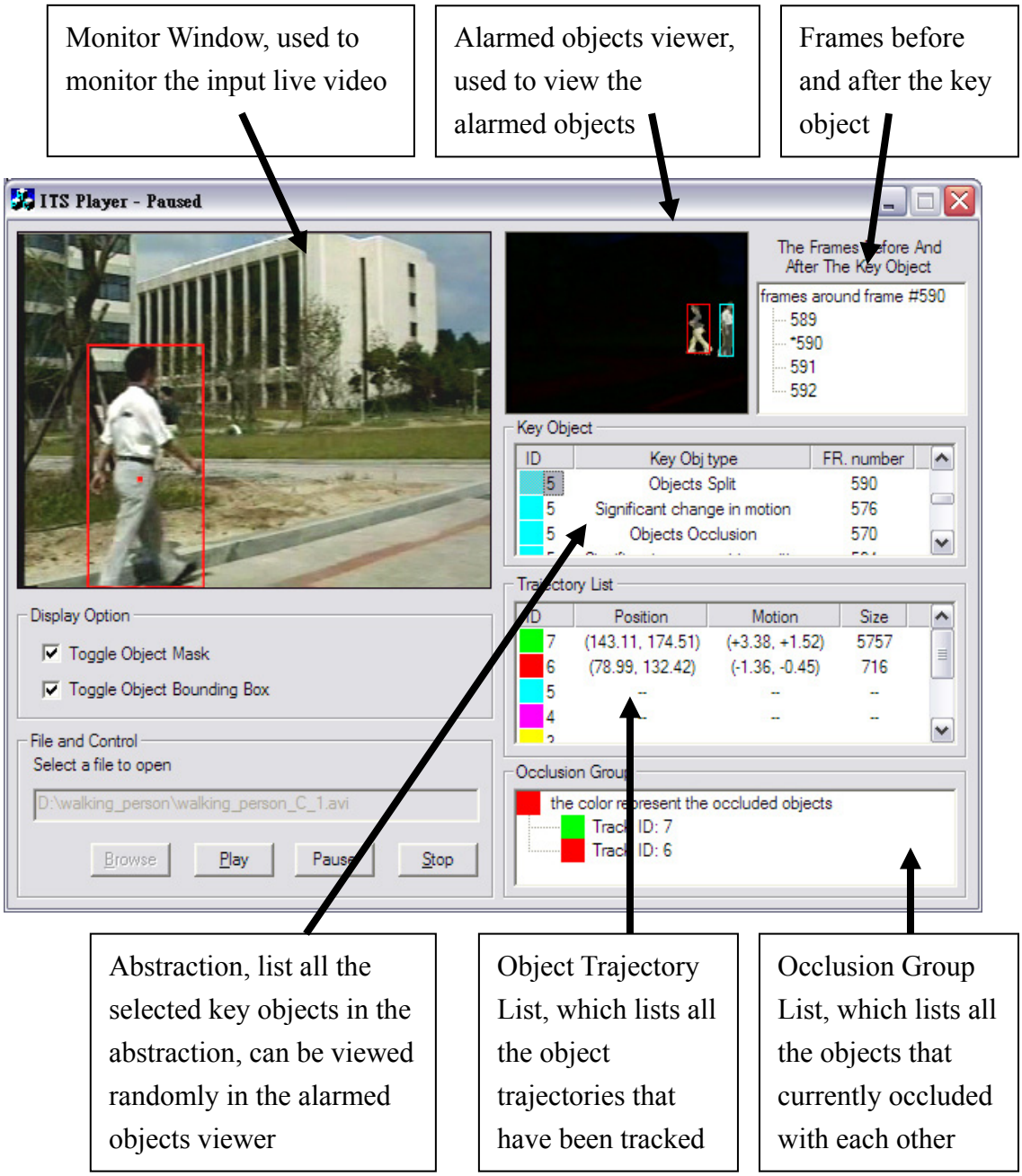


Fig. 38. Interface of the integrated system

The system tracks the video objects in the surveillance video in real-time and the abstractions for on-line events alarming are generated. While the live video is still being tracked, we can randomly click the key objects in the abstraction window to view what happened. Besides, all the tracked object trajectories and the occlusion groups are also listed to provide object information in detail.

We implement the system in Visual C++ with Microsoft Direct Show. Our testing platform is a computer with Pentium 1.6GHz CPU and 256MB RAM. The video sequences are the Speedway, ETRI_A, ETRI_B, ETRI_C and the hall monitor and all the sequences are in the format of 320 by 240. The performance can reach 16 frames per second.

Chapter 5

Conclusion and Future work

In this thesis, we presented a system for object-based video tracking and abstraction on surveillance videos. We adopted a simple but effective matching function that uses the motion vector difference as the matching criterion. We also designed a state-based object occlusion reasoning model to track and detect occlusion and split events robustly. Besides, we designed a novel video abstraction algorithm that generating abstractions by selecting key objects with important semantics and events. Based on this system structure, we implemented a real-time tracking system with on-line alarming using the generated abstractions. With the abstractions and on-line alarms, important events can be captured more efficiently and thus it is valuable to the monitoring applications and systems such as ITS because a lot of time and manpower can be saved.

To improve the performance and the robustness of the system, some enhancements can be done in the future:

- Removing the shadow regions to the object masks more precisely.
- Improving the segmentation algorithm so that it is more robust to lighting variation and complex scenes.
- To handle more complicated occlusion and split events such as multiple objects occlusions
- Extracting more semantics from the object in the video abstraction algorithm, for example to capture the actions of the objects.

In addition, since the object-based abstractions are very valuable and useful, the system can be further extended for the content retrieval and management. To achieve this, we can use the MPEG-7 descriptors to describe the contents with the detected events and generated

abstractions. And thus we can manage a database for surveillance and monitoring videos and important contents we are interested in can be retrieved efficiently. We believe that the extraction of content will be more and more important and one day such kind of systems will be widely adopted in the future.

Reference

[1] Yaakov Tsaig and Amir AverBuch, “Automatic Segmentation of Moving Objects in Video Sequence: A Region Labeling Approach,” IEEE Transactions on Circuits and Systems for Video Technology, Vol.12, NO. 7, pp.597 – 612, 2002

[2] J.C Choi, S.-W Lee, and S. –D. Kim, “Spatio-Temporal Video Segmentation Using a Joint Similarity Measure,” IEEE Transactions on Circuits and Systems for Video Technology”, Vol.7, NO. 2, pp. 279 – 286, 1997

[3] D. Wang, “Unsupervised Video Segmentation Based on Watersheds and Temporal Tracking,” IEEE Transactions on Circuits and Systems for Video Technology,” Vol.8, NO. 5, pp. 539 – 546, 1998

[4] Hieu T. Nguyen, Marcel Worring, and Anuj Dev, “Detection of Moving Objects in Video Using a Robust Similarity Measure,” IEEE Transactions on Image Processing, Vol.9, NO. 1, pp.137 – 141, 2000

[5] T. Aach, A. Kaup and R. Mester, “Statistical Model-Based Change Detection in Moving Video,” Signal Processing, Vol.31, NO. 2, pp.203-217, 1993

[6] A. Neri, S. Colonnese, G. Russo, and P. Talone, “Automatic moving object and background separation,” Signal Processing, Vol.66, pp.219-232, 1998

[7] D. D. Giusto, F. Massidda, and C. Perra, “A Fast Algorithm for Video Segmentation and Object Tracking,” The 14th International Conference on Digital Signal Processing, Vol.2, pp.697 – 700, Cagliari Univ., Italy, 2002

[8] Shao-Yi Chien, Shyh-Yih Ma, and Liang-Gee Chen, “Efficient Moving Object Segmentation Algorithm Using Background Registration Technique,” IEEE Transactions on Circuits and Systems for Video Technology”, Vol.12, NO. 7, pp. 577 – 586, 2002

[9] E.P. Ong, B.J. Tye, W.S. Lin, and M. Etoh, “An Efficient Video Object Segmentation Scheme,” Proceedings of IEEE International Conference on Acoustics, Speech and Signal

Processing, Vol.4, pp.IV-3361 – IV-3364, Singapore, 2002

[10] Jinhui Pan, Chia-Wen Lin, Chuang Gu, and Ming-Ting Sun, “A Robust Video Object Segmentation Scheme with Prestored Background Information,” IEEE International Symposium on Circuits and Systems, Vol.3, pp.803 – 806, Seattle, WA, USA, 2002

[11] Rita Cucchiara, Costantino Grana, Massimo Piccardi and Andrea Prati, “Detecting Moving Objects, Ghosts and Shadows in Video Streams,” IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.25, NO. 10, pp.1337 – 1342, 2003

[12] Franco Oberti, Simona Calcagno, Michela Zara, and Carlo S. Regazzoni, “Robust Tracking of Human and Vehicles in Cluttered Scenes With Occlusions,” Proceedings of International Conference on Image Processing, Vol.3, pp.629 – 632, Genova, Italy, 2002

[13] Changick Kim and Jeng-Neng Hwang, “Fast and Automatic Segmentation and Tracking for Content-Based Application,” IEEE Transactions on Circuits and Systems for Video Technology, Vol.12, NO. 2, pp.122 – pp.129

[14] Yi-Wen Chen, Duan-Yu Chen and Suh-Yin Lee, “Moving Object Tracking for Video Surveillance in Compressed Videos,” The 7th International Conference on Internet and Multimedia Applications and Systems, pp.695 - 698, 2003

[15] Young-Kee Jung, Kyu-Won Lee, and Yo-Sung Ho, “Content-Based Event Retrieval Using Semantic Scene Interpretation for Automated Traffic Surveillance,” IEEE Transactions on Intelligent Transportation Systems, Vol.2, NO. 3, 2001

[16] Yu Huang, Thomas S. Huang, and Heinrich Niemann, “Segmentation-Based Object Tracking Using Image Warping and Kalman Filtering,” Proceedings of International Conference on Image Processing, Vol.3, pp.601 – 604, Urbana, IL, USA, 2002

[17] Guangzhi Cao, Jingping Jiang and Jiaqian Chen, “An improved object tracking algorithm based on Image Correlation,” IEEE International Symposium on Industrial Electronics, Vol.1, pp.598 – 601, Hanzhou, China, 2003

[18] Bilge Gunsel and A. Murat Tekalp, “Content-Based Video Abstraction,” International

Conference on Image Processing, Vol.3, pp.128 – 132, NY, USA, 1998

[19] Jeho Nam and Ahmed H. Tewfik, "Video Abstract of Video," IEEE 3rd Workshop on Multimedia Signal Processing, pp.117 – 122, Minneapolis, MN, USA, 1999

[20] Wen-Gang Cheng and De Xu, "An Approach to generating two-level video abstraction," International Conference on Machine Learning and Cybernetics, Vol.5, pp.2896 – 2900, Beijing, China, 2003

[21] SangKeun Lee and Monson H. Hayes, "A Fast Clustering Algorithm For Video Abstraction," Proceedings of International Conference on Image Processing, Vol.2, pp.563 – 566, 2003

[22] Changick Kim and Jeng-Neng Hwang, "Object-Based Video Abstraction Using Clustering Analysis," Proceedings of International Conference on Image Processing, Vol.2, pp.657 – 660, Palo Alto, CA, USA, 2001

[23] Changick Kim and Jeng-Neng Hwang, "Object-Based Video Abstraction for Video Surveillance Systems," IEEE Transactions on Circuits and Systems for Video Technology, Vol.12, NO. 12, pp.1128 – 1138, 2002

[24] A.M Ferman, Bilge Gunsel and A. Murat Tekalp, "Object-Based Indexing of MPEG-4 Compressed Video," International Conference on Acoustics, Speech, and Signal Processing, Vol.4, pp.2601 – 2604, NY, USA, 1997

[25] Bernal Erol and Faouzi Kossentini, "Automatic Key Video Object Plane Selection Using the Shape Information in the MPEG-4 Compressed Domain," IEEE Transactions on Multimedia, Vol.2, NO.2, pp.129 – 138, 2000

[26] Quen-Zong Wu, Hsu-Yung Chang, and Kuo-Chin Fan, "Motion Detection Based on Two-Piece Linear Approximation for Cumulative Histograms of Ratio Image in Intelligent Transportation Systems," Proceedings of IEEE International Conference on Networking, Sensing & Control, Vol.1, pp.309 - 314

[27] Jong Bae Kim, Hye Sun Park, Min Ho Park and Hang Joon Kim, "Unsupervised Moving

Object Segmentation and Recognition Using Clustering and A Neural Network," International Joint Conference on Neural Networks, Vol.2, pp. 1240 – 1245, Taegu , South Korea, 2002

[28] M. Hu, "Visual Pattern Recognition by Moment Invariants," IRE Transactions on Information Theory, Vol.IT-8, NO.2, pp.179 – 182, 1962

[29] Rafael C. Gonzalez, Richard E. Woods, Digital Image Processing, 2nd edition , Addison-Wesley, January 15th 2002