國 立 交 通 大 學

資訊科學與工程研究所

博 士 論 文

以人類為基礎的視訊處理及其在監控上的應用

Human-based Video Processing and its Application to Surveillance

研 究 生：賴育駿

指導教授：廖弘源　教授

中 華 民 國 一百零一 年 一 月

以人類為基礎的視訊處理及其在監控上的應用
Human-based Video Processing and its Application to Surveillance

研 究 生：賴育駿	Student：Yu-Chun Lai
指導教授：廖弘源	Advisor：Hong-Yuan Mark Liao

國 立 交 通 大 學
資 訊 科 學 與 工 程 研 究 所
博 士 論 文

A Thesis
Submitted to Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in

Computer and Information Science

January 2012

Hsinchu, Taiwan, Republic of China

中華民國一百零一年一月

以人類為基礎的視訊處理及其在監控上的應用

學生：賴育駿　　　　　　　　　　　　　指導教授：廖弘源　博士
林正中　博士

國立交通大學資訊科學與工程研究所博士班

摘　要

近年來以人類為基礎的視訊處裡一直是相當熱門的研究題目，主要原因為人類通常是影片中的拍攝對象，例如，電影，監視器影片，以及運動影片。因此，如果可以在視訊影片中針對人類的部分加以處理，對於視訊內容的分析會相當的有幫助。常見的人類為基礎的視訊處理包含了人類偵測，切割，以及動作辨識等技術。並且可分為針對儲存影片的 off-line 處理以及針對即時環境的 on-line 處理。在本論文中，我們提出以人類為基礎的視訊處理技術並且將其應用在智慧型監控系統上。在第一項的研究中，我們針對已經錄製完畢的影片，提出了以背景資訊為基礎的場景分割方法。我們利用 Mosaic 的技術將屬於前景部分(通常是人物的部分)的資訊移除並且試著重建被遮蔽的背景。接著根據背景的資訊取出低階的視覺特徵來估測影片中兩 shot 之間的相似程度。並且參考電影製作的法則將 shot 群組化找出影片中不同場景之間的邊界位置。在找尋出場景邊界位置之後，可以簡化後續的視訊分析工作。在第二項的研究主題中，我們針對即時的監控系統提出我

們的主動式攝影機網路動作規劃技術。主動式攝影機可以拉近來觀測目標，可以提供較清楚的影像，因此非常符合智慧型監控系統上的需求。因此，我們提出一個線性生產規劃賽局(Linear Production Game)解法來控制攝影機網路中主動式 Pan Tilt Zoom 攝影機的參數。我們提出的非線性函式可以更加有效的攝影機去追蹤多個觀測目標，並且經由參數的拓展以及加上新的線性限制條件，可以轉換為一個線性生產規劃賽局(Linear Production Game)。由於線性生產規劃賽局可以在多項式時間內求得最佳解，因此我們提出的方法相當有效率以及精確。在第三項的研究主題中，我們針對人類動作辨識問題提出一個以局部特徵為基礎的辨識技術。我們根據局部特徵的表示法提出一個人類動作辨識的架構。兩種不同的局部特徵，包含動作的長期趨勢以及短期外型變化分別被抽取出來用來描述人類動作。最後經由 adaboost 的學習方法取出具有鑑別力的局部特徵組合來辨識人類動作。

Human-based Video Processing and its Application to Surveillance

Student：Yu-Chun Lai　　　　　　　Advisors：Dr. Hong-Yuan Mark Liao
Dr. Cheng-Chung Lin

Institute of Computer Science and Engineering
National Chiao Tung University

Abstract

In recent years, human-based video processing has attracted a great deal of attention in the field of computer vision. This is because human usually is the major subject in a video such as movie, surveillance video, and sport video. Therefore, a video processing technique based on human can provide rich information for video content analysis. Generally, common human-based video processing includes human detection, human segmentation, human motion recognition, and so on. Furthermore, according to the real-time requirements of an application, it can be categorized to the off-line processing for a video storage and the on-line processing for a real-time environment. In this dissertation, we put our emphasis on the human-based video processing and apply these techniques to an intelligent surveillance application. In the first topic, we propose a scene segmentation approach based on the analysis of background information for the off-line processing. The mosaic technique is utilized to remove the foreground parts (human) and reconstruct the occluded background.

According to the background information, several low-level visual features are integrated to compute the similarity measure between two shots; moreover, the rules of film-making are used to guide the shot grouping process. After the boundaries among different scenes are detected, the following video analysis processing can be simplified. In the second topic, we proposed an active camera network reconfiguration technique for an on-line surveillance system. Since an active camera (for example, a pan, tilt, zoom camera) be able to fixate a human subject to obtain a large view of people, it is suitable for intelligent surveillance system. Therefore, a camera network reconfiguration solution is proposed to adjust pan, tilt, and zoom parameters in a PTZ camera network for video surveillance application. The non-linear objective function we proposed better utilizes a network's cameras to track multiple targets. We also show that, by expanding the unknown parameters and imposing new constraints, the non-linear objective function can be converted into a linear production game (LPG) problem. Since an LPG yields an optimal solution that can be evaluated in polynomial time, the proposed method is efficient and accurate. In our third work, a human motion recognition framework based on local feature representation is proposed. A clay based feature to describe long-term movement trend and a motion history image (MHI) based feature to describe short-term shape variation, are extracted respectively. Then, the AdaBoost approach is applied to select

a best feature set for discriminating the human motions.

誌　　　　謝

漫長的求學階段終於畫上了句點，在這段期間中受到許多人的照顧以及幫助，謹在此向幫助我的師長、家人、以及朋友們獻上我感謝的心意。

首先要感謝是廖弘源老師，感謝老師提供了這麼好的環境可以讓我無後顧之憂的在研究上發展，並且以身作則教導我們什麼是做研究的態度跟思考方式，如果不是在碩士班的時候有機會參加廖老師實驗室的 meeting 並從中發現更多研究上的樂趣，我想我也許就不會走上學術這條路。同時也要感謝陳良華老師，不僅在課業以及研究上給予我很多的幫助，在學期間也給我許多鼓勵以及建議。感謝石勝文老師，在十分忙碌的生活中撥空指導我論文的內容以及寫作上的方法。感謝林正中老師在系上事務上給予的許多幫助以及建議。在此並感謝百忙中抽空指導我口試的莊仁輝教授、林志青教授、林嘉文教授與賴尚宏教授，對本論文的指導以及建議。

感謝我的父母，對我的任何選擇都如此的支持，我想我是非常幸運的人，可以追求自己想要做的事情，自由的選擇自己的路。雖然路途並非非常順暢，常常讓父母擔心，不過總算是結束學業上的部分，希望往後的路可以讓父母不再為我擔憂。感謝我的兄長，一起討論研究以及提供我許多就業上的資訊。

感謝在身旁的學長們，學億學長以及志文學長在我剛進實驗室，不知如何是好的時候給了我許多的建議跟協助，使我能夠慢慢的習慣研究的生活，並且有了可以討論研究以外的事情的對象。感謝敦裕學長適時的提點我要多花心思在重要的研究部分，使我避免在研究上繞的更遠。感謝祐銘學長在求學期間一路上的作伴，使我不會有一人孤軍奮戰的感覺，在學期間常常麻煩學長花時間跟我討論以及給我建議，真的是非常感謝。感謝士章學長、易聰學長以及家棟，跟你們一起討論總是可以激發我新的想法跟看法。

感謝在身旁的朋友以及學弟妹們，常一起行動的誌鴻以及興源，也一起體驗

了許多事情。常幫忙許多事務的堯麟。感謝 Amy 以及亦雲，從一開始進到實驗室時就一直給我許多的幫助，直到離開實驗室都還需要兩位的幫忙，真的是非常感謝。其它還有許多在中研院以及交通大學的老師與同學們，謝謝你們給予我的教導以及協助。

求學期間我很幸運的可以認識這些良師以及益友，這會是我人生之中珍貴的寶藏。除了感謝各位之外，也誠心的祝福各位能夠幸福快樂並達成自己的目標。

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In recent years, human-based video processing has become a popular research topic in the field of computer vision. This is because human usually is the major subject in a video such as a movie, surveillance video, and sport video. Therefore, a video processing technique based on human can provide rich information to video content analysis. Generally, common human-based video processing includes human detection, human segmentation, human motion recognition, and so on. Furthermore, according to the real-time requirements of an application, it can be categorized to the off-line processing and the on-line processing. These techniques can be widely applied for important applications, such as video surveillance [2, 3], video annotation [4, 5], and human computer interface [6, 7]. However, human-based video processing is difficult due to two reasons. First, because a conceptual human body movement is combined by complex articulated motions belonging to a huge number of segments, handling for complex motion with high degrees of freedom is necessary. Second, since processing human video in each frame is limited, a set of frames, especially spatial-temporal information, utilized to process human video is an alternative. In this dissertation, we focus on handling these challenging issues of the human-based video processing and its application to surveillance.

1.2 Overview of the Dissertation

In this dissertation, we put our emphasis on the human-based video processing in surveillance application. In a surveillance system, human-based video processing includes the off-line processing for a video storage and the on-line processing for a real-time surveillance environment. We proposed different human-based video processing techniques for the both categories. For the off-line processing, we proposed a scene segmentation technique to appropriately segment video into story units according the video content. We select a scene as a story unit based on the storyboard used in film-making. After the boundaries among different scenes are detected, the following video analysis processing can be simplified. For the on-line processing, we focus on two important issues in a traditional closed-circuit television (CCTV) surveillance system. The first issue is how to capture a large enough human view for providing good quality evidences. An active camera (for example, a pan, tilt, zoom (PTZ) camera) is an appropriate selection to deal with this task. However, a surveillance system usually is integrated by multiple cameras because a camera network can cover widely areas and reduce the number of blind spots. The issue therefore is expanded to a PTZ camera network reconfiguration problem. Therefore, we focus on the problem how to reconfigure the parameters of a PTZ camera network to capture people in cameras' field of views. The second issue of a traditional CCTV surveillance system is lack of real-time reaction because it only records all camera contents into a storage. To make the surveillance system available for detecting and alarming in real-time, human motion must be recognized. Therefore, we proposed our human motion recognition approach based on analyzing motion regions and their structure.

1.2.1 Scene Segmentation

Scene extraction is the first step toward semantic understanding of a video. It also provides improved browsing and retrieval facilities to users of video database. This work presents an effective approach to movie scene extraction based on the analysis of background images. Our approach exploits the fact that shots belonging to one particular scene often have similar backgrounds. Although part of the video frame is covered by foreground objects, the background scene can still be reconstructed by a mosaic technique. The proposed scene extraction algorithm consists of two main components: determination of the shot similarity measure and a shot grouping process. In our approach, several low-level visual features are integrated to compute the similarity measure between two shots. On the other hand, the rules of film-making are used to guide the shot grouping process. Experimental results show that our approach is promising and outperforms some existing techniques.

1.2.2 PTZ Camera Network Reconfiguration

In this work, we propose a non-linear objective function that better utilizes a network's cameras to track multiple targets. We also show that, by expanding the unknown parameters and imposing new constraints, the non-linear objective function can be converted into a linear production game (LPG) problem. Since an LPG yields an optimal solution that can be evaluated in polynomial time, the proposed method is efficient and accurate. The results of simulations and a real-world experiment demonstrate the proposed method's potential.

1.2.3 Human Motion Recognition

In this work, a local feature-based human motion analysis framework is proposed. Local features extracted from the motion regions are utilized to construct the relationship among different features, which is used to generate a spatial structural feature set, representing a human motion. In this work, we propose a clay based feature to describe long-term movement trend and a motion history image (MHI) based feature to describe short-term shape variation, respectively. In addition, the AdaBoost approach is applied to select a best feature set for discriminating the human motions. Our experiments demonstrate that the proposed approach can achieve high average recognition results with a noise tolerance capability.

1.3 Dissertation Organization

The remainder of this dissertation is organized as follows. In Chapter 2, a mosaic-based scene segmentation method is introduced. Then, A linear production game solution for capturing human motion is proposed in Chapter 3. In Chapter 4, the proposed framework for human motion recognition is described in detail. Finally, Chapter 5 contains some concluding remarks and future work.

Chapter 2

Scene Segmentation

In this chapter, we describe the proposed framework for segmenting scenes from a movie using background information. First, the scene segmentation problem and the motivation of this work are first introduced. We then propose our approach based on background information. Next, the segmented scene are shown in the experimental results section. Finally, the concluding remarks are described.

2.1 Introduction

The advances in low cost mass storage devices, higher transmission rates, and improved compression techniques have led to the widespread use and availability of digital video. Video data offer users of multimedia systems a wealth of information and also serve as a data source in many applications including digital libraries, publishing, entertainment, broadcasting and education. However, because of the large amount of data and unstructured format, efficient access to videos is not an easy task. To make the original video data in a database available for browsing and retrieval, it must be analyzed, indexed and reorganized. The suitably organized video data have the right structure for non-linear browsing and for content-based retrieval through large amount of data. To derive a structured representation of a video, each video sequence is usually decomposed into a hierarchical structure using shots and scenes as construction units. A shot is a sequence of frames that were continuously captured by the same camera. The definition of a scene is not as straightforward and usually refers to a common environment that is shared by a group of consecutive shots [8]. For example, we could see many consecutive shots (taken by different cameras) share the similar visual content because they are produced in the same environment such as a meeting room or a sports field. Generally, a video scene is basically a story unit that shows the same objects and allows one event to be fully presented. Shots in a video are analogous to words in a language in that they convey little semantic information in isolation. On the other hand, scenes reflect the dramatic and narrative structure of a video. One scenario is that humans remember different events after having watched a digital movie. Such an event can be a dialogue, an action scene, or a group of shots unified by location. Therefore, scene extraction is the first step toward greater semantic understanding of video content. The objective of video scene extraction is to cluster video shots into several groups, such that the shots within each group are related by some common aspects. Nowadays, there exist various types of videos, including movies, news casts, sitcoms (situation comedies), commercials,sports, and documentary videos. Some of them have ”story units” such as movies, while others (e.g., sports) do not. In this work, we concentrate on movies. Usually, there are two steps to extract video scene structures after shot boundary detection. The first step is to represent visual content of one shot and to define the similarity measure between two shots. The second is to group correlated consecutive shots into a scene. The compact representation of video shot content for shot similarity measure remains one of the most challenging issues. In our approach, the similarity measure is based on the background information obtained through a mosaic technique. By aligning all images of a video sequence onto a common reference frame, the mosaic technique is able to generate the static background of a video scene. Although the background image alone is not an effective video content representation for some tasks such as video retrieval, it is sufficient for the task of scene extraction. It is also worth noting that the way shots are grouped into a scene generally depends on the type of scene under analysis as well as on the video genre. The scenes of a TV-news report are different from the scenes of a basketball game, a documentary, or a movie. Hence, it is important to aggregate the shots by considering a model for the scene. In our approach, we exploit cinematic rules to devise a shot grouping algorithm.

2.2 Background and Motivation

Numerous methods for shot boundary detection have been proposed [9]. While shots are marked by physical boundaries, scenes are marked by semantic boundaries. Hence, scene boundary detection is far more difficult than shot boundary detection. Current techniques for video scene extraction can be broadly classified into three categories. The first groups shots that are visually similar and temporally close into a scene [10–19]. In these approaches, video shots are mostly represented by a set of selected key-frames. Low level features such as color, texture, motion, and shape are extracted directly from key-frames. Then, a classic clustering algorithm or simple peak detection is used to detect scene boundaries. However, the limitation of key-frame-based shot representation is that a frame taken from a shot often fails to represent the dynamic contents within the shot. When a sequence of shots is considered as a scene, it is often because the shots are correlated by the same environment rather than by visual similarity in terms of key-frames. In the second category, emphasis is put on the integration of audio and visual information. Various methods have been proposed to determine a shot boundary as a scene boundary if the visual and audio content change simultaneously [20, 21]. However, how to determine scene boundary efficiently still remains a difficult issue since the relationship between audio segments and visual shots is complicated. The third category exploits characteristics embedded in specific video domains, such as sports and news casts [22–25]. Since this approach is based on specific application models, it normally achieves high accuracy.

The main drawback is that an a priori model needs to be constructed beforehand for each application. The modeling process is time consuming and requires good domain knowledge and experience. In this chapter, we present a scheme for automatic video scene extraction based on visual information only. When a scene is composed of several shots, there will be at least one aspect in common between these shots. To measure the common aspects between two shots, we define the shot similarity using background information. Our approach is based on the following observations: (i) each frame of a video can be divided into foreground objects and background scenery; (ii) in most cases, while objects may move, appear and disappear, the background in a scene does not change significantly. These observations suggest that it would be more effective to focus on the background to detect a scene, which is a collection of shots unified by a common locale. However, in a single frame, most of the physical location information is invisible since it is concealed by the foreground objects. One solution to this problem is to use a mosaic technique. A mosaic is a panoramic image obtained by aligning all images of a video sequence onto a common reference frame [26, 27]. The resulting mosaic is an efficient and compact representation of a collection of frames [28]. In the case of a static mosaic, the moving objects blur out or disappear. Only the stationary objects and the background are displayed in the constructed mosaic. Therefore, in our approach, we use a mosaic technique to reconstruct the background scene of each video frame. Then, we compute the color and texture distribution of all the background images in a shot to determine the shot similarity. Since our method does not depend on the content of key-frames only, it is able to fully exploit the spatiotemporal information contained in video sequences.

2.3 Mosaic Construction

A shot is the basic unit for video indexing. To facilitate subsequent video analysis, in our system, the original video sequences are segmented into shots [29]. We have also proposed an algorithm to align frames from a video shot to build the static mosaic of the background [30]. Here, we briefly describe the generation of mosaics, and refer the reader to Ref. [30] for further details. We first need to derive the transformation between partially overlapping frames. Assuming background motion (due to camera motion) is the dominant motion in the video, the image motion of the majority of scene points can be approximated by the following transformation model:

$$u(x,y) = a_1x + a_2y + a_3, \tag{2.1}$$

$$v(x,y) = -a_2x + a_1y + a_4, \tag{2.2}$$

where $(u(x,y), v(x,y))$ is the motion vector at image position (x,y). This model is a special case of an affine model with six parameters. However, experimental results show that our model is better than the general affine model. This is because we impose more constraints on the models parameters to avoid some undesirable transformations implied in affine model such as skewing. In this step, we need to obtain the motion vectors (or displacements) between successive frames. Existing approaches are based on feature matching or optical flow computation. These techniques are computationally intensive. To reduce the processing time, we use the motion vectors encoded in the MPEG-1 [31] video stream directly. Fig. 2.1 shows a frame of a baseball game sequence and its corresponding motion vectors. Given the motion vector, the image motion parameters can be estimated by a robust regression technique called least-median-squares [32]. Let the motion vectors of each frame be denoted as $\{[u_1(x_1,y_1), v_1(x_1,y_1)], \ldots, [u_n(x_n,y_n), v_n(x_n,y_n)]\}$. The least-median-squares method can be described as

$$\min_{\hat{a}}\{median[(a_1x_i + a_2y_i + a_3 - u_i)^2 + (-a_2x_i + a_1y_i + a_4 - v_i)^2]\} \tag{2.3}$$

where $\hat{a} = (a_1, a_2, a_3, a_4)$ is the parameter vector. One distinctive property of the algorithm is that it can tolerate up to 50% of the outliers in the data set, i.e., half of the data set can be arbitrary without significantly effecting the regression result. Therefore, this technique can robustly estimate the motion of the majority of scene points (background) and will not be biased by the minority scene points (moving object). Once the transformations between successive frames have been determined, the transformations can be composed to obtain the alignment between any two frames of the video sequence, and in particular, between the current frame and the reference frame. In most cases, we choose the first frame of the sequence as a reference. After all the frames have been aligned to a reference frame, the next step is the selection of pixels to be put into the resulting mosaic. The gray value of each pixel of the mosaic is computed by applying an appropriate temporal operator to the aligned frames. The temporal average operator is effective in removing temporal noise, but the moving objects appear blurred, with ”ghost-like” traces in the resulting mosaic. The temporal median operator can remove the moving objects and produce a static mosaic of the background, but it is computationally expensive and is an off-line process. Therefore, we propose a novel scheme that is both effective in deleting moving objects and feasible for the on-line creation of panoramic images. Our approach is based on the observation that each overlapping pixel of the aligned frames will fall into one of the two categories: background or moving object. Since background motion is the dominant motion of video and we want to build the mosaic of background, we select the pixel that appears most frequently in the temporal domain. In Fig. 2.2, some sampled frames of ”Terminator II” sequence are shown. We observe that the camera has large movement in this sequence. Fig. 2.3 shows the static mosaic where only the background of the scene is visible. Using the background mosaic, the background scene image of each video frame is reconstructed. In the subsequent processing, we focus on these background image sequences.

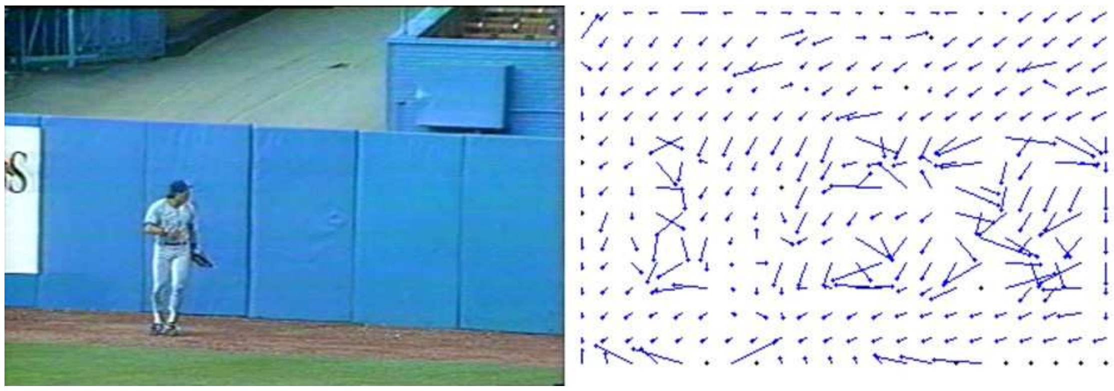


Figure 2.1: A frame of a baseball game sequence and its corresponding motion vectors.

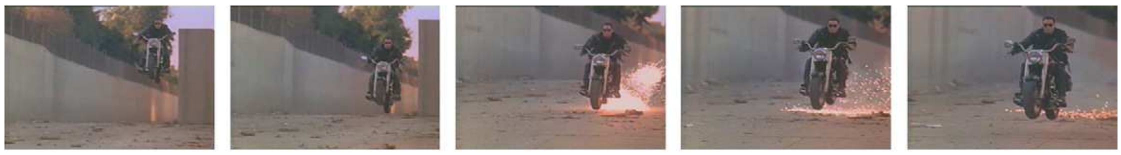


Figure 2.2: Sampled frames of a ”Terminator II” sequence.

2.4 Shot Similarity Measure

This section describes the color and texture features used in our work and how these features are taken into account in the formulation of shot similarity measure. A major requirement for shot similarity measure is to define a content representation that captures the common aspects or characteristics of the shot. One common method is to select one key-frame from the shot and use the image features of that key-frame as an abstract representation of the shot. For shots with fast changing content, one key-frame per shot is not adequate. Besides, the content description it provides varies significantly with the key-frame selection criterion. To avoid these problems, a more feasible approach is to consider the visual content of all the frames within a shot for shot representation. Color is one of the most widely used visual features in video content analysis. Most scene extraction algorithms compare color histograms between key-frames to determine the shot similarity measure. The histogram-based approach is relatively simple to implement and provides reasonable results. However, due to its statistical nature, the color histogram cannot capture the spatial layout information of each color. When the image collection is

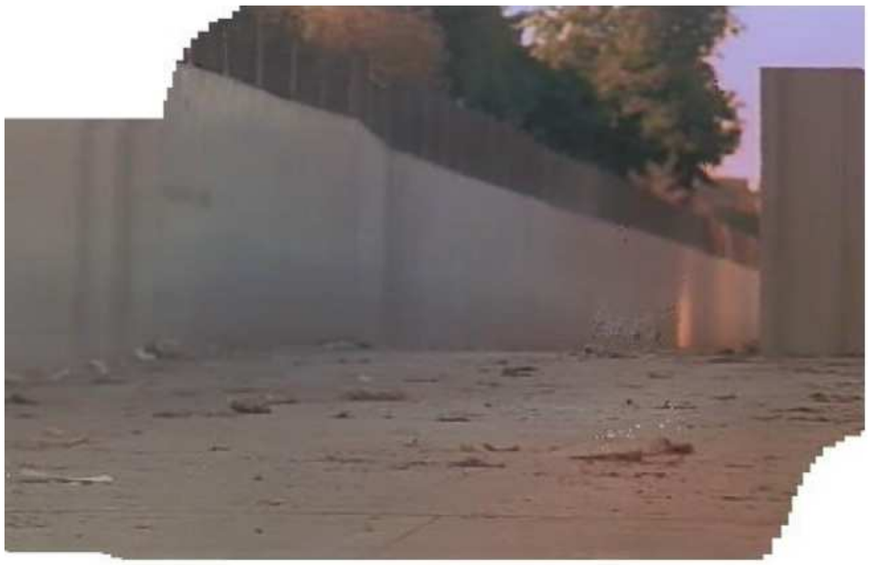


Figure 2.3: The static mosaic of the ”Terminator II” sequence.

large, two different content images are likely to have quite similar histograms. To remedy this deficiency, in our approach, the distribution state of each color in the spatial (image) domain is also taken into account. The color histogram of an image is constructed by counting the number of pixels of each color. The main issues regarding the construction of color histograms involve the choice of color space and quantization of the color space. The RGB color space is the most common color format for digital images, but it is not perceptually uniform. Uniform quantization of RGB space yields perceptually redundant bins and perceptual holes in the color space. Therefore, non-uniform quantization may be needed. Alternatively, HSV (hue, saturation, intensity) color space is chosen since it is nearly perceptually uniform. Thus, the similarity between two colors is determined by their proximity in the HSV color space. When a perceptually uniform color space is chosen, uniform quantization may be appropriate. Since the human visual system is more sensitive to hue than to saturation and intensity [33], H should be quantized finer than S and V. In our implementation, the hue is quantized into 20 bins. The saturation and intensity are each quantized into 10 bins. This quantization provides 2000 $(= 20 \times 10 \times 10)$ distinct colors (bins), and each bin with non-zero count corresponds to a color object. Since we are interested in the whole shot rather than single image frame, only one histogram is used to count the color distribution of all background images within a shot. Then, each bin of the resulting histogram is divided by the number of frames in a shot to obtain the average histogram. Next, several spatial features are calculated to characterize the distribution state of each color object in each image frame. Assuming a set of pixels $S = \{(x_1, y_1), ..., (x_n, y_n)\}$ belong to color object c_i, k is the image size, and m is the total number of 4-connected pixels in S. Then, we define

1. the density of distribution as

   $f_{i1} = \frac{n}{k}$,

2. the compactness of distribution as

   $f_{i2} = \frac{m}{n}$,

3. the scatter as

   $f_{i3} = \frac{1}{n\sqrt{k}} \sum_{j=1}^{n} \sqrt{(x_j - x_\mu)^2 + (y_j - y_\mu)^2}$,
   where $x_\mu = (1/n)\sum_{i=1}^{n} x_i$ and $y_\mu = (1/n)\sum_{i=1}^{n} y_i$.

   To define the fourth feature, the image is partitioned equally into p blocks of size 16×16. A block is active if it contains some subsets of S Let the number of active blocks in the image frame be q, we define

4. $f_{i4} = \frac{q}{p}$.

After the spatial features of all images are computed, we take average of these values, respectively. Let $\overline{f_{i1}}$, $\overline{f_{i2}}$, $\overline{f_{i3}}$, and $\overline{f_{i4}}$ be the average feature values of a color object c_i in a shot, for two color objects c_i and c_j, the difference in the spatial distribution within a shot is defined as

$$D_s(c_i, c_j) = \frac{1}{4}(|\overline{f_{i1}} - \overline{f_{j1}}| + |\overline{f_{i2}} - \overline{f_{j2}}| + |\overline{f_{i3}} - \overline{f_{j3}}| + |\overline{f_{i4}} - \overline{f_{j4}}|). \tag{2.4}$$

Texture refers to the visual patterns that have properties of homogeneity that do not result from the presence of only a single one color or intensity only. It contains important information about the structural arrangement of objects and their relationship to the surrounding environment. We define the coarseness of an images texture in term of the distribution density of the edges. The Canny edge detector is used to extract edges from an image. The edge location indicates sharp intensity variation. Psychophysical experiments have shown that the human visual system is sensitive to the high-frequency regions of an image such as edges. The detected edge image is partitioned into a set of 16 $\times$ 22 blocks. A block is textured, if the number of edge points in the block is greater than a threshold (=30, in our setting). Then, we can compute the ratio of the textured block of each image and its average value over a shot. The texture similarity between two shots is determined by the minimum of the two average values. In Fig. 2.4, two images with different level of texture coarseness are shown. Fig. 2.5 shows the detected edge image partitioned into a set of 16 $\times$ 22 blocks. Histogram intersection is a popular similarity measure used for color-based image matching [34]. It yields the number of pixels that have same color in two images. In our work, we extend this idea to shot similarity measure. Let A,B be the set of all color objects in shot S_1 and S_2, respectively, for a given $u \in A$, its similar color object in B is some $v \in B$ such that $\|u-v\| < \varepsilon$, where $\|u-v\|$ denotes the Euclidean distance between u and v in the HSV color space, and ε is a threshold (=3, in our setting). Then, (u,v) is called a similar color pair. Let $\Omega = \{(u,v)|(u,v) \in A \times B, (u,v)$ is a similar color pair $\}$, the shot similarity measure between S_1 (with the average histogram $\overline{H_1}$) and S_2 (with the average histogram $\overline{H_2}$) is defined as

$$ShotSim(S_1,S_2) = \frac{1}{k} \sum_{(u,v)\in\Omega} \{W(D_s(u,v))min(\overline{H_1}(u),\overline{H_2}(v))\} + w_t \times min(t_1,t_2), \quad (2.5)$$

where k is the image size; t_1 and t_2 are the average ratios of textured block for shot S_1 and S_2, respectively; w_t is the weight of texture feature; D_s is the difference in spatial features as defined in Eq. (4); and W is a weight function defined as

$$W(x) = \frac{1}{1+e^{a \times x+b}}$$

The weight function W is the general form of the sigmoid function which is frequently used in neural networks computation [35], where a and b are parameters. In our work, it is used to fuse the spatial distribution information with a histogram. The construction of this weight function is motivated by the psychophysical observation that the effect of spatial distribution on human perception is progressive [36]. Only when the difference in spatial features is greater than a threshold, humans perceive significant visual variation. The property of the sigmoid function fulfills this requirement. In our system, we set $a = 10$ and $b = -5$. As shown is Fig. 2.6, the functions value becomes significantly small for $x > 0.75$.

It is noted that a given color object in shot S_1 may have more than one similar color objects in shot S_2 as illustrated in Fig. 2.7. To avoid the overlapping contribution in calculating shot similarity, after each step of $min(\overline{H_1}(u), \overline{H_2}(v))$, $\overline{H_1}(u)$ and $\overline{H_2}(v)$ are all subtracted by $min(\overline{H_1}(u), \overline{H_2}(v))$

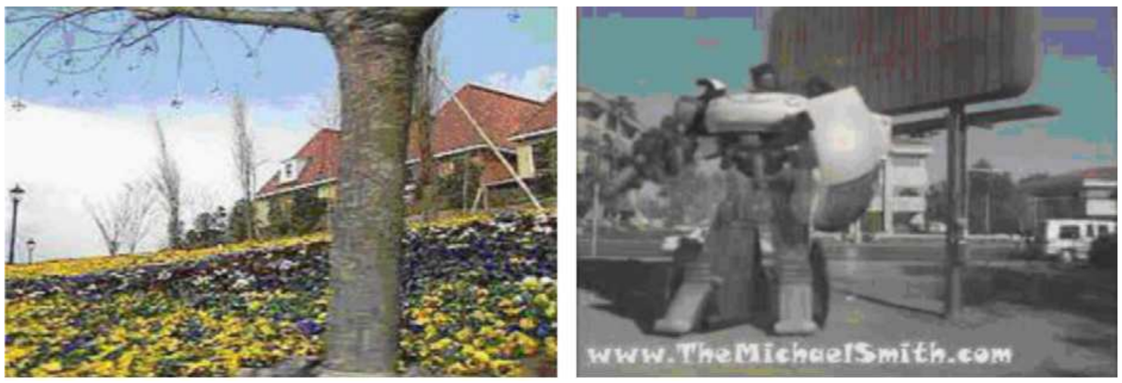


Figure 2.4: Two images with different texture coarseness.

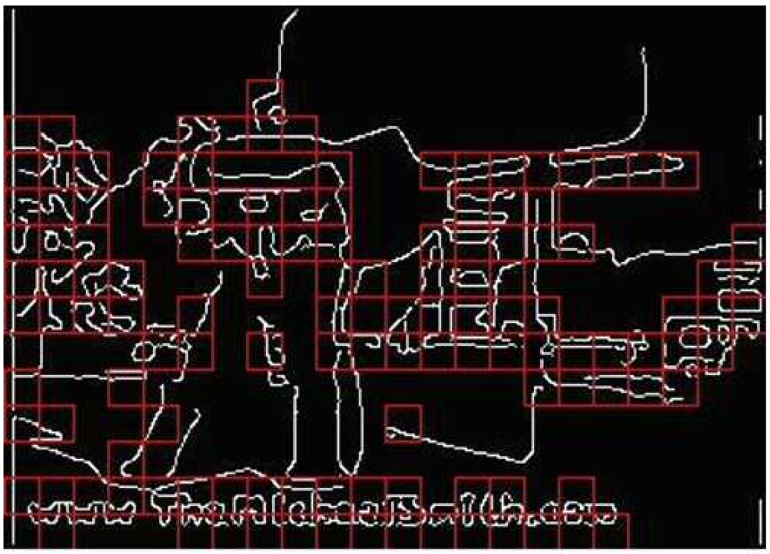


Figure 2.5: The detected edge image is partitioned into a set of 16 × 22 blocks.

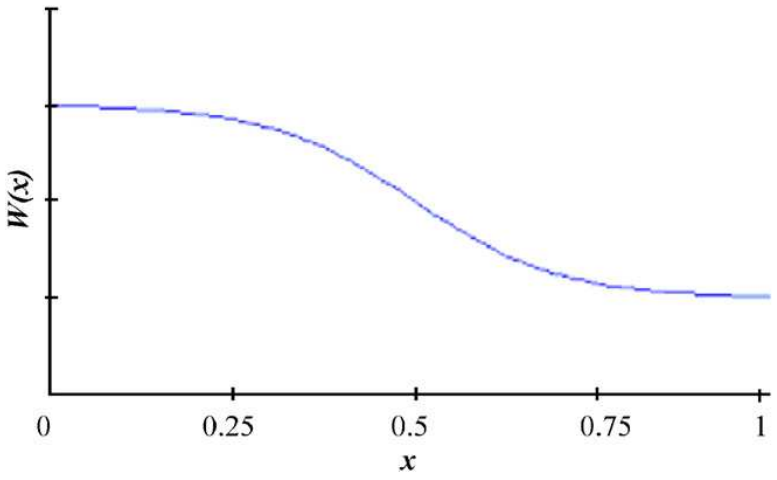


Figure 2.6: Sigmoid function with parameters *a* = 10 and *b* =-5.

2.5 Scene Extraction

This section describes the details of applying the cinematic model to group correlated consecutive shots into one scene. Movies directors, while filming scenes, also control the pace of a film in order to sustain the viewers interest. One important factor known to influence the pace of a movie is the Montage. Montage usually refers to a model that defines the usage of editing effects to assemble the shots into a smooth sequence in physical time and/or space and in the psychological association of ideas [37]. In order to convey an idea that has a strong resonance with viewers, Montage is widely used as the

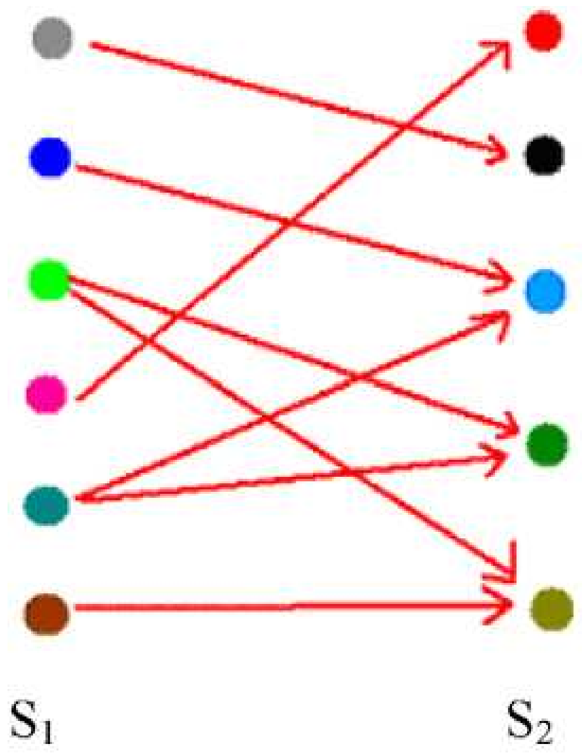


Figure 2.7: Finding similar color object pairs.

basis to model scenes. In most situations, Montage can be simplified as a set of cinematic rules. Commonly used rules include [38, 39]:

1. Parallel rule: It is used to compose scenes involving multiple themes, where shots from different themes are shown alternately. This rule is frequently used to model interactions between two parties such as conversations, hunting, and chasing.

2. Concentration rule: It starts with long distance shot, and progressively zooms into close-up shots of the main objects. It is used to introduce the main objects and their context.

3. Enlargement rule: It is the reverse of the concentration rule. It is used to introduce the context of the current main object before switching to other objects that possibly share a similar context. Thus, it typically signals the transition to a new scene.

4. Serial Content rule: It is used to model scenes that preserve the continuity of loca-

tion, time, space, and topic.

Together, these rules can be used to model most types of scenes. We use this knowledge to develop a two-pass algorithm for scene boundary detection suitable for feature movies. The first pass of the algorithm deals with the detection of potential scene boundaries. This is achieved by computing the shot similarity between two consecutive shots (see Fig. 2.8). Given a sequence of shots $S_1, ..., S_n$, if $ShotSim(S_{i+1}, S_i) < T_\mu$ then a potential scene boundary is detected. The threshold T_μ is empirically set to be the mean of all shot similarities, i.e.,

$$T_\mu = \frac{1}{n-1} \sum_{i=1}^{n-1} ShotSim(S_{i+1}, S_i). \tag{2.6}$$

Any two adjacent potential scene boundaries delineate a candidate scene. Thus, all shots are grouped into a set of candidate scenes (see Fig. 2.9). The above algorithm assumes that all shots in a scene take place in the same location and share many common backgrounds. This is true for most of the scenes composed using the serial content rule and parallel rule (such as a conversation in a studio). However, the algorithm tends to oversegment the more complex scenes composed using parallel or concentration/enlargement rules. In the outdoor chasing scene (also modeled by parallel rule), the escapee and pursuer shots are shown alternatively. But the background of both types of shots may be different. One important component of a scene defined by the concentration/enlargement rule is the close-up shot where more than half of the frame is occupied by the foreground object. Because of the limitations of the mosaic technique, the background information can not be recovered completely. Thus, the close-up shot will be identified as belonging to another scene. To handle such scenes, we need to merge the candidate scenes further. Let $G_1 = S_{11}, ..., S_{1m}$ and $G_2 = S_{21}, ..., S_{2n}$ be two candidate scenes consisting of m and n shots, respectively. The scene similarity between G_1 and G_2 is

$$SceneSim(G_1,G_2)=\frac{1}{m\times n}\sum_{i=1}^{m}\sum_{j=1}^{n}ShotSim(S_{1i},S_{2j}) \tag{2.7}$$

Two scenes $G1$ and $G2$ are visually similar, if $SceneSim(G_1,G_2) > T_\mu - \sigma/2$ where T_μ (defined in Eq. (6)) and σ are, respectively, the mean and variance of all similarity measures between every two adjacent shots. Because several simple scenes are merged into one complex scene, this threshold should be smaller than that of the first pass. Given a sequence of candidate scenes, the second pass of the scene extraction algorithm mainly consists of the following merging process:

Step 1: Set the expanding scene to be the first scene.

Step 2: Compare the expanding scene with two subsequent scenes B and C (see Fig. 2.10).

Step 3: If the expanding scene and scene C are visually similar, then

1. merge the expanding scene, scene B and scene C into one scene;
2. set the expanding scene to be the merged scene;
3. go to Step 2.

Step 4: If the expanding scene and scene B are visually similar, then

1. merge the expanding scene and scene B into one scene;
2. set the expanding scene to be the merged scene;
3. go to Step 2.

Step 5: Set the expanding scene to be scene B and go to Step 2.

This process is repeated until no more scenes can be merged. It is noted that B and C always refer to the two scenes immediately following the current expanding scene and they are always updated in Step 2.

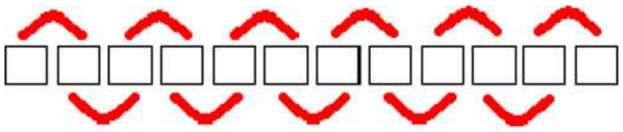


Figure 2.8: Computing the similarity between every two adjacent shots.

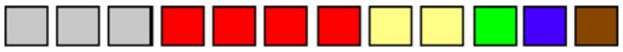


Figure 2.9: Shots are grouped into several candidate scenes.

2.6 Experimental Results

Six test videos in MPEG-1 format were used to evaluate our scene extraction algorithm: one home video and five full-length movies. The home video ”Lgerca_lisa_1” is an MPEG standard test video with ground truth from original video provider. The genres of movies include action, drama, comedy, thriller, and music. The testing with five different genres of movies would ensure that the overall performance of the algorithm is not biased toward a specific movie kind. To get the ground truth of other videos, two graduate students were invited to watch the movies and then asked to give their own scene boundaries. The intersection of their segmentation was used as the ground truth for the experiments. In movies, there is usually not a concrete or clear boundary between two adjacent scenes due to editing effects. Therefore, we follow Hanjalics evaluation criterion [12]: if the detected scene boundary is within four shots from the boundary detected manually, this boundary is counted as a correct boundary. Basic information about the test videos and the experimental results are shown in Table 2.1. As shown in Table 2.1, our algorithm correctly extracts all the six scenes of the first video (Lgerca_lisa_1) without any missed

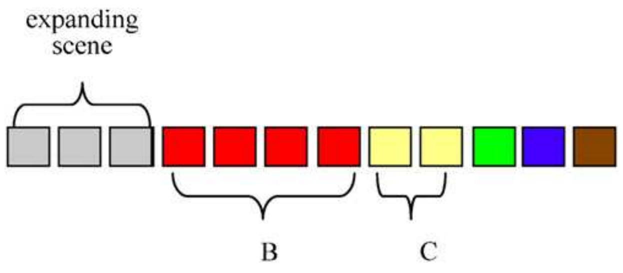


Figure 2.10: An expanding scene and the two subsequent scenes.

or false detection. Fig. 2.11 shows the shot grouping result of the first video, where the first frame of each shot is displayed. Fig. 2.12 shows one extracted scene (consisting of 4 shots) from another video ”LittleVoice”. This scene is composed by the parallel rule and has different backgrounds in the first and second shot. Our shot group algorithm is able to identify both shots as belonging to the same scene. However, our algorithm has some false and missed detections in the other test videos. The false detection is due to the significant change of lighting such as explosions and flashing lights. This could perhaps be improved by a more sophisticated visual similarity measure. The missed detection is mainly caused by inappropriate setting of the merging threshold. As the threshold depends on the variance (σ) of all similarity measures between every two adjacent shots, a too large σ results in under-segmentation of the video. This type of error occurs in some video scenes with very inconsistent pace. For performance comparison, we implement the well-known scene extraction algorithm proposed by Yeung et al. [10]. In their approach, a video sequence is first segmented into shots. Then, a time-constrained clustering algorithm is used to group visually similar and temporally adjacent shots into clusters. The visual similarity between two shots is measured by comparing color histograms of the respective key-frames. Finally, a scene transition graph is constructed based on the clusters, and cutting edges are identified to construct the scene structure. For fair comparison, the parameters of Yeung's algorithm are tuned to achieve the best performance. To measure the performance quantitatively, two metrics are used:
$recall = \frac{D}{D+MD}, precision = \frac{D}{D+FD}$
where D is the number of scene boundaries detected correctly, MD is the number of missed detection and FD is the number of false detection. Table 2.2 shows the performance comparison. As ”Lgerca_lisa_1” is an MPEG standard test video, some related works have also used it as test video. According to the reports of Lin [14] and Ngo [15], their approaches have three and two false detections, respectively.

Table 2.1: Accuracy measures for six test videos.

Video Title	Genre	Duration (in minutes)	No. of Scenes (ground truth)	Correct Detection	Missed Detection	False Detection
Lgerca_lisa_1	Home Video	15	6	6	0	0
Dungeons & Dragons	Action	107	66	54	12	6
Little Voice	Drama	96	141	110	31	24
Hot Chick	Comedy	104	104	71	33	52
Bugs	Thriller	82	76	58	18	26
Walk the Line	Music	136	118	70	48	35

Table 2.2: Performance comparison for scene extraction.

Video Title	Our Approach		Yeung's Approach	
	Recall	Precision	Recall	Precision
Lgerca_lisa_1	100%	100%	100%	85.7%
Dungeons & Dragons	81.8%	90.0%	74.2%	81.6%
Little Voice	78.0%	82.1%	72.3%	72.3%
Hot Chick	68.3%	57.7%	54.8%	50.9%
Bugs	76.3%	69.0%	59.2%	64.3%
Walk the Line	59.3%	66.7%	47.5%	56.0%

2.7 Concluding Remarks

In this chapter, we have proposed a mosaic-based algorithm for extracting scene structures from digital movies. Our approach is based on the idea that shots belonging to one particular scene often have similar backgrounds. Using a mosaic technique, the background of each video frame can be recovered. The color feature and texture feature of each background image are integrated to compute shot similarities. Based on the movie making model, our algorithm is able to group correlated shots into one scene. The computation is costly, but the spatiotemporal information of videos is fully exploited to achieve scene extraction. Experimental results show that the proposed approach works reasonably well in detecting most of the scene boundaries. Compared with some existing techniques [10, 14, 15], our approach is promising. Our approach can be applied directly to organize videos and can be utilized to provide browsing/retrieval facilities to the users.

As scene is a subject concept to reflect human perception, our future work will focus on investigating an adaptive technique to perform user-oriented scene extraction. The proposed shot similarity measure does not use motion feature to capture temporal variation in a video. Thus, another future research issue will be the integration of motion information into the proposed shot similarity measure for other tasks such as video retrieval.

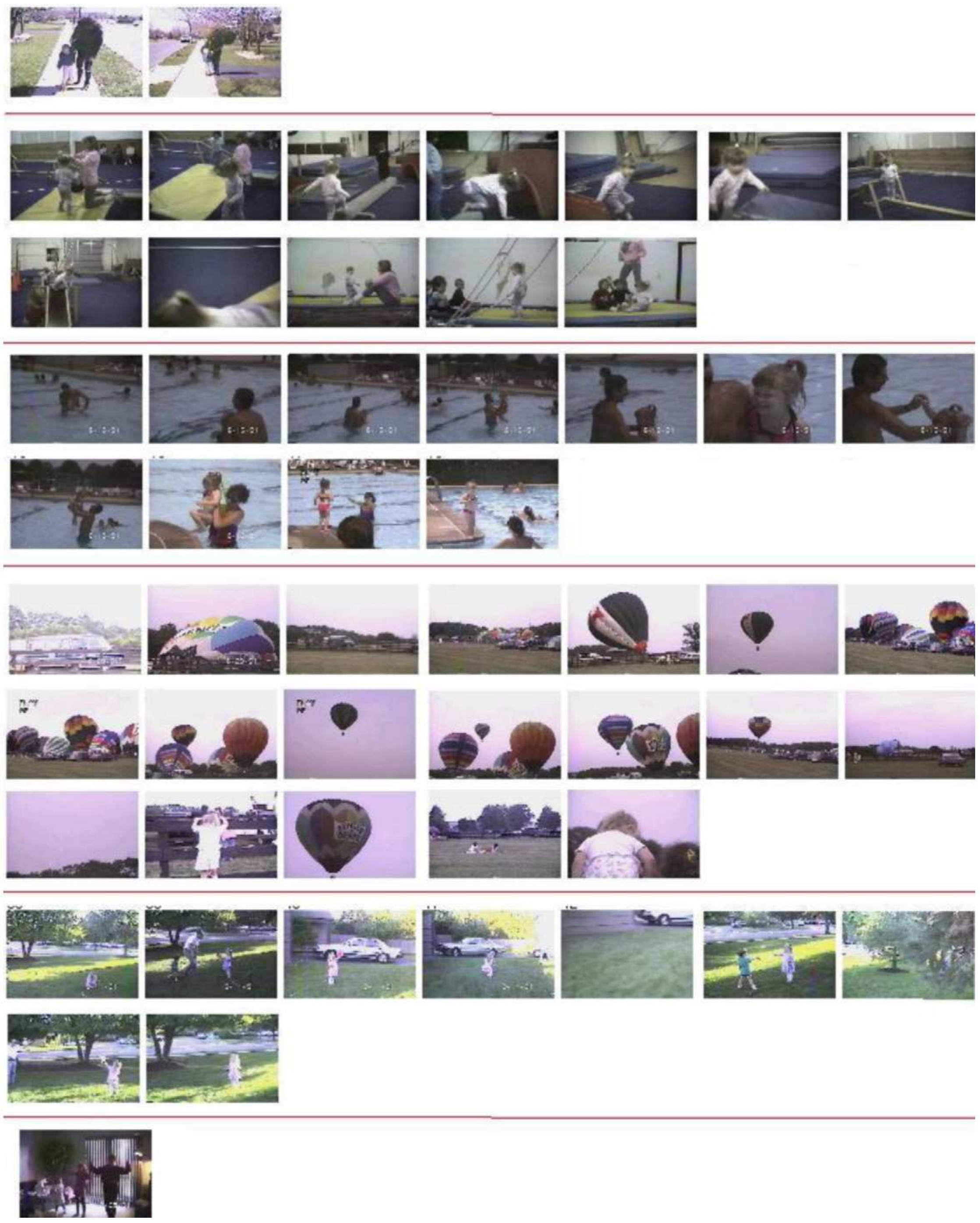


Figure 2.11: Shot grouping result of "Lgerca_lisa_1".

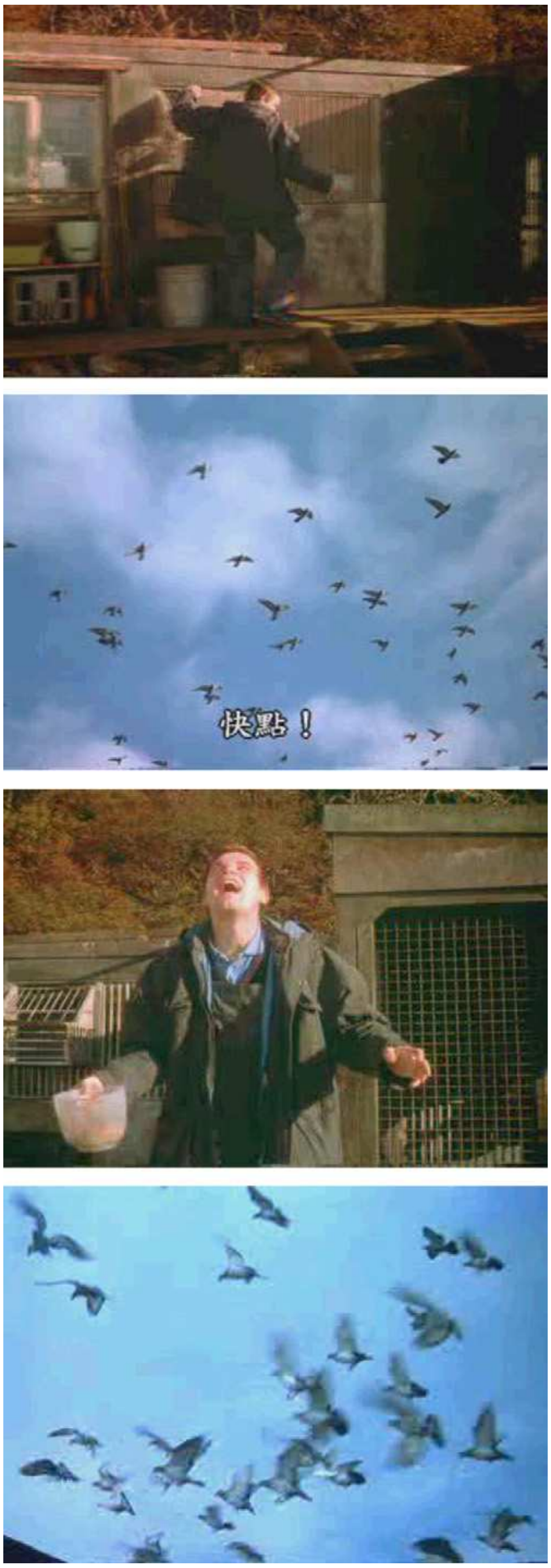


Figure 2.12: A scene extracted from ”Little Voice”.

Chapter 3

PTZ Camera Network Reconfiguration

In this chapter, we propose a linear production game solution for a pan, tilt, and zoom (PTZ) camera network to reconfigure cameras' parameters. First, we give an introduction about this research topic. The proposed approach is then presented. Next, simulations and a real world experiment are detailed. Finally, the conclusion is given.

3.1 Introduction

Intelligent video surveillance systems have been used for several years, and they are now widely deployed in important places, such as airports, all over the world. A single camera can provide useful information for event detection and target tracking [2, 3]; however, a surveillance system based on a camera network can reduce the number of blind spots and improve the system's reliability [40–46]. Camera networks are usually comprised of a heterogeneous collection of cameras, including panoramic cameras, fixed cameras, infrared cameras, and pan-tilt-zoom (PTZ) cameras. Among the different types of imaging devices, PTZ cameras are the most important components of an intelligent surveillance system because their field of view (FOV) can be changed in response to different task requirements. However, incorporating PTZ cameras into a surveillance system raises a challenging issue: How can the cameras be controlled and coordinated to accomplish a given task? Most surveillance tasks performed by PTZ cameras are related to three functions: tracking multiple targets, improving evidential quality and maximizing surveillance coverage.

Target tracking involves target detection as well as temporal and/or inter-view target correspondence matching. For example, Lim et al.'s approach [40] tracks the targets observed in FOVs and constructs a dynamic scene model containing the position, velocity, and view-dependent visibility of each target. The system tries to accomplish three objectives, namely, initial detection of moving objects, tracking of moving objects, and scheduling cameras to monitor activities. Cameras are assigned to tasks by solving a bipartite matching problem so that tasks are accomplished in order of priority. In the PTZ camera network developed by Ukita and Matsuyama [41], when the system detects a new target, nearby cameras that are idle are assigned to track the target. The system is simple and effective provided that the number of cameras is greater than the number of targets. Qureshi and Terzopoulos [42] introduced a multi-camera tracking system in which calibrated wide-FOV cameras are used to locate targets, and PTZ cameras are used to fixate on the located targets. The PTZ network operates according to heuristic rules designed to track targets cooperatively. Their method was further improved so that it can be applied to an uncalibrated multi-camera surveillance system connected with wireless communication network [47]. Since the wireless communication range is limited, Qureshi and Terzopoulos assumed that each camera can only communicate with its neighbors. Therefore, cameras within a communication range can share information of targets to be tracked. Adding/removing camera nodes can be accomplished very easily with their method. However, the flexibility is exchanged for security because, in general, a camera network sufficiently sharing information with a centralized server can outperform a camera network sharing information locally. In summary, a cooperative target tracking approach can utilize the camera resources efficiently and enable the cameras to support each other to recover the tracking when a tracking task fails.

While a solution to the target tracking problem can provide each target's trajectory, a surveillance system usually requires more information. For example, it is often necessary to capture the face of a human target or the license plate of a vehicle at a high resolution. These applications are related to improving evidential quality [44, 45]. In addition, PTZ cameras are frequently used to extend the coverage of the surveillance area by pan-tilt scanning. Piciarelli et al. [46] proposed an approach that reconfigures the pan-tilt-zoom parameters of all PTZ cameras based on the probability of observing an event in a specific location. Song et al. [43] applied game theory to maximize the surveillance coverage in their decentralized system, and adopted a sequential optimization strategy to achieve the Nash equilibrium [48]. Under this method, one PTZ camera is selected at random each time and its parameters are tuned, while those of the other cameras remain unchanged. After the Nash equilibrium is achieved, the cameras should cover the entire surveillance area at an acceptable resolution. When a human operator decides to track a specific target at a higher resolution, the target will be assigned to the most appropriate PTZ camera, which will then be excluded from the game. As a result, the remaining cameras have to adjust their parameters and try to maintain the maximum surveillance coverage. Based on the result of [43], they introduce a distributed consensus algorithm to solve target tracking and activity recognition problems [49] [50]. They modified the utility function in the game theoretic control framework [43] and then applied the method to control the cameras to cover the entire surveillance area. They also proposed to use a decentralized Kalman filter algorithm to track the position and velocity of each target. The game theoretic framework has two advantages: 1) it can be implemented easily; and 2) only a small amount of information needs to be exchanged. However, we remark that the Nash equilibrium is not necessarily an optimal solution. Moreover, tracking a specific target at a higher resolution is treated as an exceptional task that cannot be optimized by using the same game theory framework. Another potential problem of their method is that there is no mechanism to suppress investing too many resources (i.e., cameras) in tracking a single target.

Reconfiguring PTZ cameras to achieve any of the three objectives is intrinsically a combinatorial optimization problem. Computing the optimal solution is very time consuming and, therefore, existing methods usually reduce the problem into a bipartite matching problem, which assigns tasks to cameras. However, the task assignment formulation does not fully utilize the camera network. For example, when the number of tasks is greater than the number of cameras, some tasks will have to be abandoned despite that a camera may accomplish multi-tasks at the same time.

In this chapter, we propose an optimal and flexible solution to the PTZ network coordination problem. We show that the problem can be formulated as a linear production game (LPG). The LPG is about how a group of collaborative players utilizing their limited resources to create various products yielding the maximum payoff given that the price of each product is known. Players in the LPG of a PTZ network are the cameras. Each camera can control its FOV by selecting the PTZ parameters. Although the number of all PTZ combinations is very large, due to the limited speed of PTZ actuators, only a small set of new PTZ settings has to be considered. The new FOV corresponding to each new PTZ setting is the product of the game. Resources owned by each player (i.e., camera) are the targets which are observable to the camera. The observability of a target to a camera is determined by checking whether the camera can select a feasible PTZ settings to observe the target. The price of a product (i.e., the new FOV of a camera) is evaluated by examining the video quality of each target in the FOV. The goal of this cooperative game is to select a set of FOVs for the cameras to maximize the total payoff (video quality of the targets). The LPG is a special case of a linear programming problem. While a linear programming problem may be infeasible or unbounded [51], the LPG always has an optimal solution that can be evaluated in polynomial time. Therefore, many techniques, such as branch-and-bound and cutting-plane techniques [52], can be applied to solve the camera network reconfiguration problem.

Table 3.1: Symbol table used in Section 3.2

n	camera number
m	detected target number
$\mathbf{g}_t^k$	status vector of target k at time t
$\mathbf{b}_t^k$	3D bounding box of target k at time t
$\mathbf{v}_t^k$	velocity of target k at time t
U_t	the status of m targets at time t
$\hat{U}_{t+1}$	the predicted status of m targets at time $t+1$
ϕ_j^i	the j-th feasible FOV in i-th camera
Φ^i	the feasible FOVs in i-th camera
w_i	number of feasible FOV in i-th camera
$Q(.)$	the quality function of an FOV combination
$Q_k(.)$	the quality function by observing k-th target

3.2 Problem Formulation

Suppose a surveillance system contains n calibrated PTZ cameras, each of which is controlled by a network-connected processor. In addition, a fixed (non-PTZ) camera in our system is seen as a PTZ camera with only one available FOV. Furthermore, let m be the number of targets detected in the surveillance area. Each detected target is represented by a status vector denoted by $\mathbf{g}_t^k = \left[\mathbf{b}_t^k, \mathbf{v}_t^k\right]$, where $\mathbf{b}_t^k$ and $\mathbf{v}_t^k$ are, respectively, the 3-D bounding box and the velocity of target k estimated at time t. The target's status $U_t = \left\{\mathbf{g}_t^k \middle| k = 1, 2, ..., m\right\}$ and the static background constitute a dynamic scene model (targets history positions and a top-view scene model) that can be used to predict the status of all the targets m at time $t+1$, expressed as $\hat{U}_{t+1} = \left\{\hat{\mathbf{g}}_{t+1}^k \middle| k = 1, 2, ..., m\right\}$. The model is maintained by a central information processing node (a central server) that gathers information about the detected targets and the camera parameters from each camera node. It is assumed that the camera network has been calibrated and the homography (a point to point mapping matrix) between any two of the cameras is known so that the information about the detected target can be integrated. The central information processing node is also responsible for determining the optimal camera parameters.

3.2.1 Parameters to be Determined

Let ϕ^i denote the i-th camera's FOV, which is controlled by the pan-tilt-zoom parameters of the camera. We assume that the relationship between the FOV and the parameters is known. Therefore, the problem of determining the optimal camera parameters is transformed into a problem of selecting the optimal FOV for each camera. Because of the limitation of the lens motor speed, a camera can only change its parameters locally in a short time. Hence, given each camera's current parameters, a set of feasible FOVs can be constructed and expressed as follows:

$$\Phi^i = \left\{ \phi^i_j \middle| \, j = 1, 2, ..., w_i \right\}, \tag{3.1}$$

for i = 1, 2, ..., n, where w_i is the number of feasible FOVs of the i-th camera. The PTZ camera coordination problem is formulated as the following combinatorial optimization problem:

$$\left(\phi^{1*}, ..., \phi^{n*}\right) = \arg \max_{\phi^i \in \Phi^i, i=1,...,n} Q\left(\phi^1, ..., \phi^n\right), \tag{3.2}$$

where $Q(.) : \Phi^1 \times \cdots \times \Phi^n \longmapsto \mathbb{R}$ is a function mapping $\left(\phi^1, ..., \phi^n\right)$ to a real quality value.

In the next subsection, we explain how to assess the quality of a set of FOVs for different goals.

3.2.2 Quality Function of A Camera's FOV

Under the dynamic scene model, the locations of predicted targets can be computed for each camera's FOV. The predicted bounding box is defined as a region of interest (ROI). In a visual surveillance system, assessing the quality of an FOV usually involves the following two steps.

1. Determine whether the FOV includes some ROIs. An FOV without any ROIs

should be evaluated as having the lowest quality.

2. Evaluate the dimensions (width and height) of each ROI. The resolution of the ROI should be sufficient to accomplish the given task. Aldrige and Gilbert [1] suggested different resolution requirements for different tasks. If a resolution is lower than the suggested value, a low quality value should be assigned to it. Conversely, if the resolution is higher than the suggested value, the quality value should be upper bounded or reduced to induce camera zoom out for monitoring a larger area.

For most surveillance tasks, the quality of each camera's FOV can be evaluated individually and the total quality function, $Q\left(\phi^1,...,\phi^n\right)$, can be simplified as follows:

$$Q\left(\phi^1,...,\phi^n\right) = f\left(q_1,q_2,...,q_n\right), \tag{3.3}$$

where $q_i = Q\left(\phi^i\right)$ for i = 1, 2, ..., n, and $f(\cdot) : \mathbb{R}^n \longmapsto \mathbb{R}$ is a function that maps the n individual quality values to a total quality value. We discuss possible choices of $f(\cdot)$ later in this section. Furthermore, the quality function of each FOV, say ϕ^i, can be expressed as a function of the qualities of individual ROIs. Since the quality of each ROI can be evaluated independently, it is reasonable to compute the quality of an FOV as follows:

$$Q\left(\phi^i\right) = \sum_{k=1}^{m} Q_k\left(\phi^i\right), \tag{3.4}$$

where $Q_k\left(\phi^i\right) \triangleq Q\left(\hat{\mathbf{b}}_{t+1}^k;\phi^i,\hat{T}_{t+1} - \left\{\hat{\mathbf{g}}_{t+1}^k\right\}\right)$ is the non-negative quality of observing the k-th target with FOV ϕ^i. The value is zero when $\hat{\mathbf{b}}_{t+1}^k$ is not observable in FOV ϕ^i or it is completely occluded by other targets at $\hat{T}_{t+1}$. The simplest form of $f(\cdot)$ in Equation (3.3) is a linear summation function given by

$$Q_L\left(\phi^1,...,\phi^n\right) = \sum_{i=1}^{n} Q\left(\phi^i\right). \tag{3.5}$$

However, since the qualities of a target in all views count toward the total quality function, maximizing (3.5) makes all the cameras pursue high video quality targets and ignore low quality ones.

To resolve the problem, we adopt the following non-linear quality function in this work:

$$Q_{NL}\left(\phi^1, ..., \phi^n\right) = \sum_{k=1}^{m} \max_i Q_k\left(\phi^i\right), \tag{3.6}$$

where only the maximum ROI quality of each target counts toward the total quality. Thus, the quality of a solution that favors a specific target will be lower than that of a solution that assigns the cameras to monitor different targets.

3.3 The Proposed Approach

In this section, we show that the non-linear objective function (3.6) can be converted into a linear function by expanding the set of feasible solutions and imposing new constraints. Let T_j^i and $\left|T_j^i\right|$ denote the set of targets covered by FOV $\phi^i{}_j$, i.e., the j-th feasible FOV of the i-th camera (refer to equation (3.1)), and the number of targets in T_j^i respectively. The total number of subsets of T_j^i is $2^{\left|T_j^i\right|}$, and h is an index of the subset. For each subset $S_{j,h}^i \subset T_j^i$, $1 \le h \le 2^{\left|T_j^i\right|}$, we can construct a virtual FOV that ignores any target not in $S_{j,h}^i$, i.e., $Q_k\left(\phi^i{}_{j,h}\right) = 0$, for all $k \notin S_{j,h}^i$ (see Figure 3.1).

Notably, introducing virtual FOVs into the system increases the number of expanded feasible FOVs to $\sum_{i=1}^{n}\sum_{j=1}^{w_i} 2^{\left|T_j^i\right|} \le n\, w_{max}\, 2^{\left|T_{max}\right|}$, where w_{max} and $\left|T_{max}\right|$ are the maximum number of feasible FOVs of a camera and the maximum number of targets in an FOV, respectively. From the complexity analysis, it is obvious that the computation load is linearly proportional to the number cameras and is exponentially proportional to the number of targets in an FOV. The exponential growth of the variables may lead to the scalability problem. However, since a PTZ camera is mainly used to acquire high definition images of targets by choosing a proper zoom setting, the number of targets observed by a PTZ

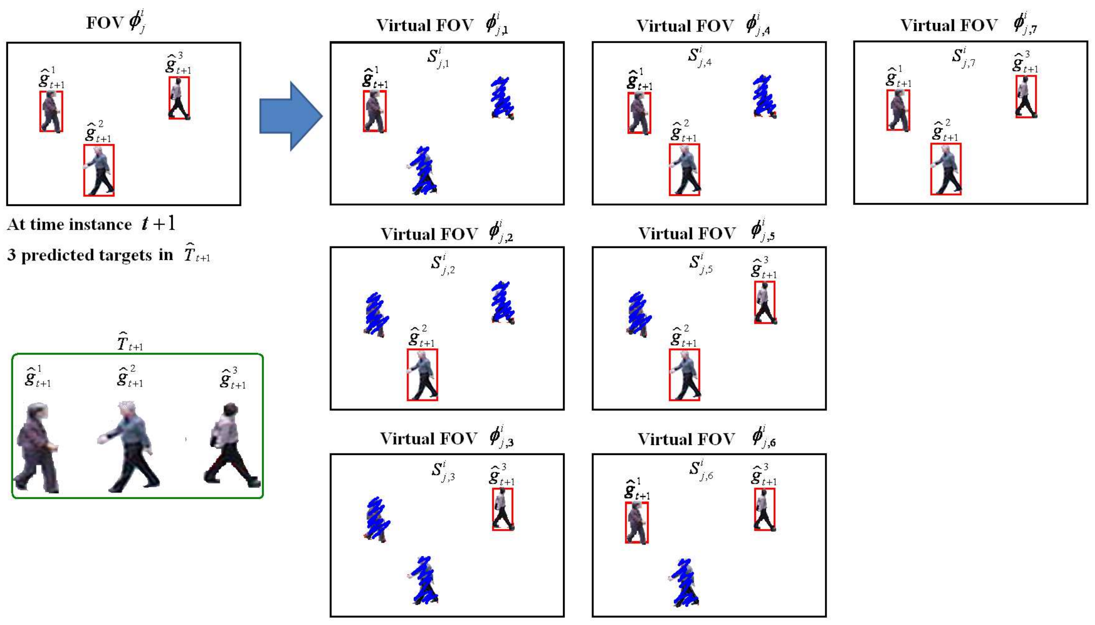

FOV ϕ^i_j
At time instance $t+1$
3 predicted targets in $\hat{T}_{t+1}$
$\hat{T}_{t+1}$
$\hat{g}^1_{t+1}$
$\hat{g}^2_{t+1}$
$\hat{g}^3_{t+1}$
Virtual FOV $\phi^i_{j,1}$
Virtual FOV $\phi^i_{j,2}$
Virtual FOV $\phi^i_{j,3}$
Virtual FOV $\phi^i_{j,4}$
Virtual FOV $\phi^i_{j,5}$
Virtual FOV $\phi^i_{j,6}$
Virtual FOV $\phi^i_{j,7}$
$S^i_{j,1}$
$S^i_{j,2}$
$S^i_{j,3}$
$S^i_{j,4}$
$S^i_{j,5}$
$S^i_{j,6}$
$S^i_{j,7}$

Figure 3.1: The decomposition of an FOV containing $\left|T^i_j\right|$ targets into $2^{\left|T^i_j\right|}-1$ virtual FOVs.

camera is very limited. To give an impression about the typical number of targets covered by the FOV of an camera in different surveillance applications, we tabulate in Table 3.3 the suggested target sizes with respect to four different applications described by Aldrige and Gilbert [1]. According to the data shown in Table 3.3, the maximum number of targets in an FOV of a camera is less than 15 for the recognition and identification applications. In practice, the maximum target number will be much smaller than this value and the solution can be computed very efficiently. The scalability problem emerges only when one uses too few PTZ cameras to observe too many targets. In that case, PTZ cameras are operating at the wide-angle (low resolution) mode, and the video content is less informative. To acquire useful surveillance videos, one should consider introducing more PTZ cameras into the network. Hence, although from the algorithmic point of view, the proposed method might suffer from scalability issues, in practice, the scalability issues can be ignored.

By replacing ϕ^i_j with the virtual FOVs, $\phi^i_{j,h}$, $h=1,\ldots,2^{\left|T^i_j\right|}$, the number of feasible FOVs to be assigned to the i-th camera becomes $\sum_{j=1}^{w_i} 2^{\left|T^i_j\right|}$. To select the optimal FOVs,

Table 3.2: Symbol table used in Section 3.3

T_j^i	targets covered by the FOV ϕ^i_j
$\left\|T_j^i\right\|$	number of targets covered by the FOV ϕ^i_j
$S^i_{j,h}$	subset of T_j^i
w_{max}	the maximum number of feasible FOVs of a camera
$\|T_{max}\|$	the maximum number of targets in an FOV
$\phi^i_{j,h}$	virtual FOV expanded from ϕ^i_j
$x^i_{j,h}$	binary variables to indicate $\phi^i_{j,h}$ is selected
o_{ijhk}	binary coefficient to indicate whether the k-th target is observable in $\phi^i_{j,h}$
q_k^*	the quality of the k-th target evaluated by one of the optimal FOVs

Table 3.3: The required target size of different surveillance applications suggests in [1].

Applications	Suggesting target height(%) in a CCTV FOV	Suggesting target size in a 640×480 FOV(average human width-height ratio 0.3442 [53])	Maximum suggesting target number in an FOV
Monitor and control	$> 5\%$	9×24	1422
Detection	$> 10\%$	17×48	376
Recognition	$> 50\%$	83×240	15
Identification	$> 120\%$	199×576	2

we define the binary variables $x^i_{j,h} \in \mathbb{B}$ ($\mathbb{B} \triangleq \{0,1\}$) to indicate whether the (j,h)-th virtual FOV of the i-th camera, i.e., $\phi^i_{j,h}$, is selected. Therefore, the optimization problem in FOV selection can be rewritten as follows:

$$\max_{\mathbf{x}} \sum_{i=1}^{n} \sum_{j=1}^{w_i} x^i_{j,h} \sum_{k=1}^{m} Q_k\left(\phi^i_{j,h}\right), \tag{3.7}$$

subject to

$$\sum_{j=1}^{w_i} \sum_{h=1}^{2^{\left|T^i_j\right|}} x^i_{j,h} \leq 1, \tag{3.8}$$

for $i = 1, 2, \ldots, n$, and

$$\sum_{i=1}^{n} \sum_{j=1}^{w_i} \sum_{h=1}^{2^{\left|T^i_j\right|}} x^i_{j,h}\, o_{ijhk} \leq 1, \tag{3.9}$$

for $k = 1, 2, \ldots, m$, where $\mathbf{x} = \left[x^1_{1,1}, x^1_{1,2}, \cdots, x^n_{w_n, 2^{\left|T^n_{w_n}\right|}}\right]$ and the binary coefficient $o_{ijhk} \in \mathbb{B}$ indicates whether the k-th target is observable in the virtual FOV $\phi^i_{j,h}$. The constraint specified in (3.8) ensures that each camera can only be assigned one FOV at a time, and (3.9) guarantees that the quality of a target can only be evaluated by a single FOV because the repeated target selection violates the constraint in equation (3.9) (the summation value is larger than 1.)

The relation between the solutions to (3.6) and (3.7) can be derived by changing the summation order of (3.7) as follows:

$$\sum_{k=1}^{m} \left[\max_{\mathbf{x}} \sum_{i=1}^{n} \sum_{j=1}^{w_i} x^i_{j,h}\, Q_k\left(\phi^i_{j,h}\right)\right] = \sum_{k=1}^{m} q^*_k, \tag{3.10}$$

where q^*_k is the quality of the k-th target evaluated by one of the optimal FOVs.

The objective function and the constraints given in Equations (3.10), (3.8) and (3.9) form a linear programming problem. Since a linear programming problem may be infeasible or unbounded [51], it is important to show that the above formulation yields an optimal solution that can be solved efficiently. In the following subsection, we show that

Table 3.4: Symbol table used in Section 3.3.1

N	n player set
v	a payoff function of the game
S	a coalition, subset of N
$b_k(.)$	total number of the k-th resources of a coalition
p	number of products
$\mathbf{A}$	a resource-product matrix
c	obtained price by selling a product
$\mathbf{C}$	the core of a game

our formulation can be regarded as an LPG [54]. The LPG is a special case of linear programming problems which has the following nice properties

1. The optimal solution of an LPG always exists, and

2. The solution can be computed in polynomial time by solving its dual problem [54, 55].

3.3.1 Linear Production Game (LPG) Problem

An LPG is a special case of a cooperative game that is usually denoted as (N, v), where N is a set of n players, namely $\{1, 2, ..., n\}$, and $v : 2^N \to \mathbb{R}$ is a payoff function that maps a coalition to a payoff value [56]. Specifically, the payoff of an empty set is zero. A coalition, say S, is a subset of all players, i.e., $S \subseteq N$. The LPG described in [54] involves maximizing the payoff of n cooperative players who own m different types of resources. The total number of the resources type k in a coalition S are denoted as $b_k(S)$, $k = 1, 2, ..., m$, and those resources can produce p different products. A resource-product matrix $\mathbf{A}$ is used to specify the relationships between the amounts of the different types of resources required to produce each product. The (i, j)-th entry of the resource-product matrix, denoted by a_{ij}, describes the amount of the i-th resource required to produce the j-th product, which is non-negative. Then, each product can be sold at a different price given by $\mathbf{c} = (c_1, c_2, ..., c_p)$. In this game, the maximum payoff can be obtained by solving the following linear programming problem

$$v(S) = \max_{\mathbf{x}} \mathbf{c}^\top \mathbf{x}, \tag{3.11}$$

subject to $\mathbf{Ax} \le b_k(S)$ and $x_i \ge 0$ for all i.

The above optimization problem indicates that all players in the coalition S should combine their resources to produce a set of products that maximize the revenue $v(S)$.

Let $\mathbf{y} = [y_1 \cdots y_n]^\top$ denote the payoff vector, where $y_i = c_i x_i$, $i = 1, 2, ..., n$. In cooperative game theory, an imputation describing a payoff vector is defined as follows:

$$\left\{ \mathbf{y} | \mathbf{y} = [y_1, y_2, ..., y_n]^\top, \sum_{i \in N} y_i = v(N), y_i \ge v(\{i\}) \, \forall i \in N \right\} \tag{3.12}$$

Equation (3.12) indicates that each player benefits by cooperating in a game because the resulting payoff, i.e., y_i, of the i-th player is not less than the payoff derived by playing alone, i.e., $v(\{i\})$. $v(\{i\})$ means the maximum payoff value obtained by the i-th players resource only. Moreover, in the cooperative game $v(N)$, the sum of the payoffs to all the players is equal to the total payoff. The solution concept of a cooperative game is called the *core*. It consists of one or more imputations that satisfy the following condition: all the payoffs of a subset in the imputation must be maximized. In the payoff vector, none of the players has a motive to leave the coalition because doing so would reduce his/her payoff. The core is defined as follows:

$$\mathbf{C} = \left\{ \mathbf{y} | \mathbf{y} = [y_1, y_2, ..., y_n]^\top, \sum_{i \in N} y_i = v(N), \sum_{i \in S} y_i \ge v(S) \, \forall S \subseteq N \right\} \tag{3.13}$$

Notably, the coefficients in the LPG equations are all positive. As the coefficients in the objective function (3.7) and in the constraints (3.8) and (3.9) are also all non-negative, the optimization problem can be easily mapped to an LPG problem. The role mapping between the camera network reconfiguration problem and the LPG is listed as follows:

1. The players are the cameras.
2. The resources of a player are the targets covered by a camera at the next time instance.
3. The products are the selected FOVs for the cameras.
4. The price of each product (i.e., an FOV) is the quality of each selected FOV.

This formulation is similar to a bounty hunter game in which the cameras act as hunters who use FOVs to "hunt" targets. Different hunters will receive different rewards (i.e., video quality) when obtaining a target. This is a cooperative game because all the hunters tend to let the hunter who may receive the greatest reward to catch a target. The existence of a core of the game ensures that all players will obtain the maximum payoff.

In this work, we adopt the branch-and-cut algorithm [57] to compute an optimal integer solution. Since an optimal solution guarantees that (3.10) is an upper bound of (3.6), the LPG solution is equivalent to the optimal solution of the non-linear objective function (3.6).

3.4 Simulations and A Real World Experiment

To compare the performances of different approaches, we conducted three computer simulations and a real experiment. Throughout the four experiments, the following simple quality function is adopted.

$$Q_k\left(\phi^i_{j,h}\right) = \begin{cases} 1+\lambda\, z^i_{j,h}, & \text{if target } k \text{ is covered by } \phi^i_{j,h} \\ 0, & \text{otherwise,} \end{cases} \tag{3.14}$$

where $\lambda = 0.01$ in the experiments and $z^i_{j,h} \in [0,1]$ is the zoom level of the FOV $\phi^i_{j,h}$. When the lens is fully zoomed out, $z^i_{j,h} = 0$. Conversely, $z^i_{j,h} = 1$ indicates that the lens is fully zoomed in. Therefore, $\lambda z^{i}_{j,h} \leq 0.01$. The quality function defined in (3.14) indicates that our goal is to control the PTZ cameras to capture as many as possible the targets appearing at the area under surveillance while favoring high zoom level camera settings. The goal can be applied to real-world applications. For example, the most common application is to record all targets' trajectories in the scene. Then, when an event occurred, the recorded trajectories can be utilized to trace a target (for example, a criminal) and to reconstruct his escaping route. For simplicity, we assume that the image resolution of any static camera included in the surveillance network is too low that the image quality function always returns zero. Hence, the optimal solution will be a control strategy that uses the PTZ cameras to capture the highest number of targets.

In the first two experiments, we simulated a surveillance system containing three virtual PTZ cameras, a virtual static camera covering the entire area under surveillance, and several virtual pedestrians with different paths and movement speeds. In total, we produced five pedestrian sets each containing 15 moving objects. Figure 3.2 shows the simulation environment in which the total coverage of each PTZ is depicted as a fan-shaped region and a moving target is represented as a trajectory with time ordering.

We compared the performance of the proposed non-linear objective function LPG (NOF-LPG) method with that of the following four methods: the linear sum method, i.e., the linear objective function LPG (LOF-LPG), which maximizes Equation (3.5); Song et al.'s method (SONG) [43]; Lim et al.'s method (LIM) [40]; and the exhaustive search (ES) method. To find an optimal solution, the ES algorithm tests all possible combinations $\prod_{i=1}^{n} w_i$ of the feasible FOVs (refer to Equation 3.1)) of all the cameras. The small-scale tests enable us computing an ES solution as a reference to compare different approaches. The simulated system was implemented in C++ on a PC with an Intel Core 2 DUO E8400 3.00Ghz CPU.

The first simulation compares the efficiency and accuracy of the five algorithms on the five pedestrian sets. The average computation times using NOF-LPG, LOF-LPG,

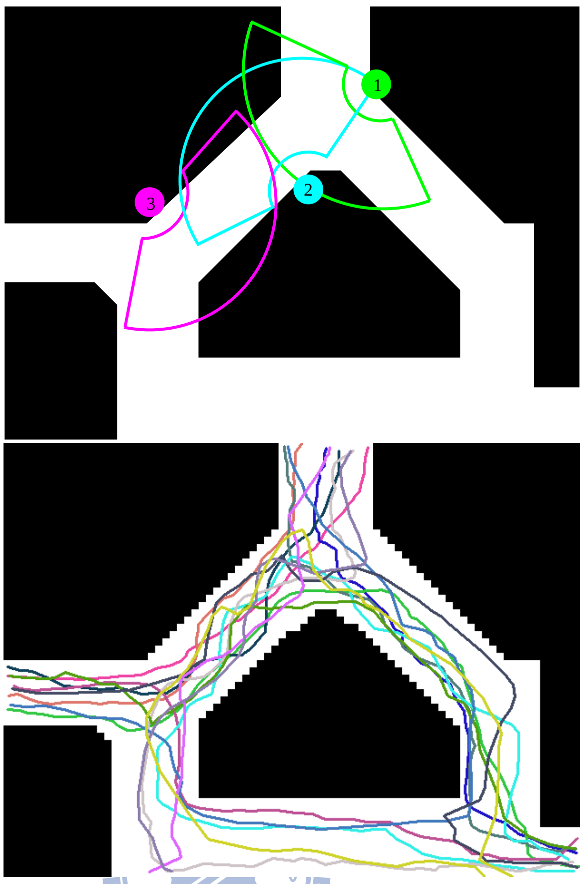


Figure 3.2: The simulated surveillance environment with three PTZ cameras and fifteen moving targets in Pedestrian set 1

SONG, LIM, and ES are shown in Table 3.5. The computational load of NOF-LPG is exponentially proportional to the maximum number of targets in an FOV, i.e., $\left|T_j^i\right|$ (refer to Section 3.3) but the number of targets in an FOV is normally small. Therefore, the efficiency of NOF-LPG is acceptable. In the simulation, the maximum number of targets in an FOV is five, so the computation time of the proposed method is satisfactory.

Figure 3.3(a) shows the true number of moving objects in pedestrian set 1 and the number observed by ES. Note that the cameras may not be able to observe all the moving objects simultaneously because of the limitations of their FOVs. Therefore, the numbers of observed targets derived by the ES algorithm are used as a benchmark. In the simulation, the positions of all pedestrians are provided by the global-view static camera. Figure 3.3(b) shows the performances of the four compared methods using pedestrian set

Table 3.5: The average computation time required for one iteration by the NOF-LPG, LOG-LPG, SONG, LIM, and ES methods

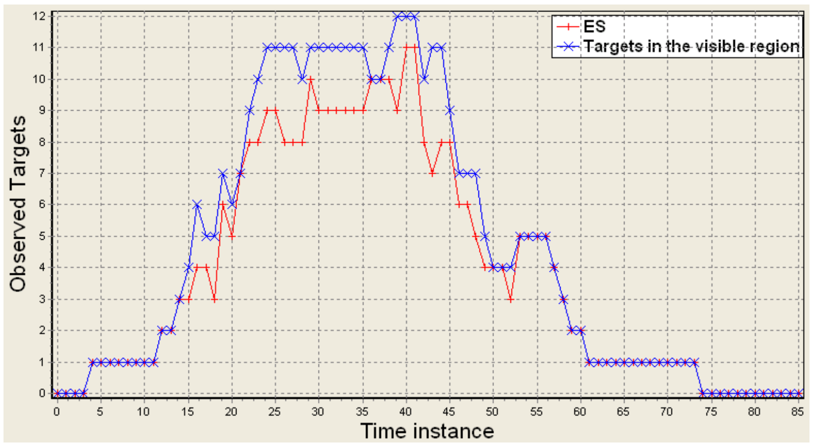
	Average computation time in the first simulation	Average computation time in third simulation
NOF-LPG	334.14 ms	1115.72 ms
LOF-LPG	247.76 ms	1112.89 ms
LIM	236.33 ms	1073.44 ms
SONG	245.01 ms	1086.99 ms
ES	2330.37 ms	$O(45^{30})$

Table 3.6: The percentages of targets observed by the compared methods

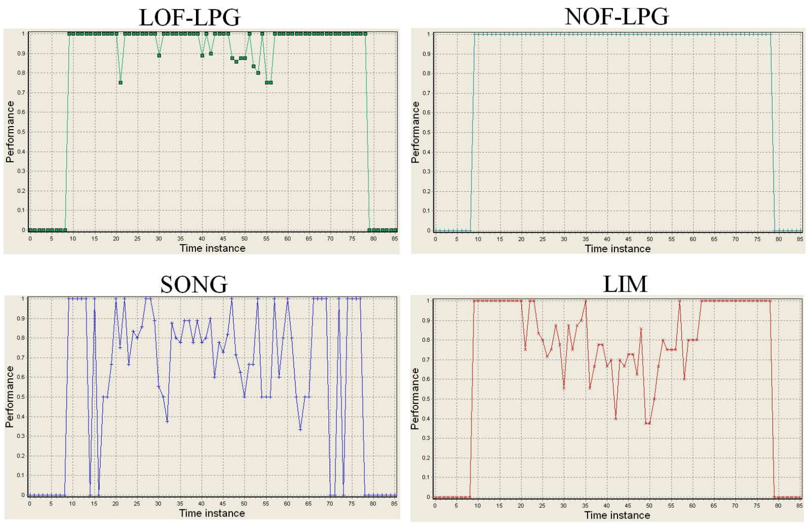
	Pedestrian set 1	Pedestrian set 2	Pedestrian set 3	Pedestrian set 4	Pedestrian set 5
NOF-LPG	100.0%	100.0%	100.0%	100.0%	100.0%
LOF-LPG	96.4%	93.7%	93.9%	93.6%	93.2%
LIM	75.6%	71.7%	84.6%	77.9%	77.0%
SONG	73.4%	76.0%	66.3%	72.4%	68.4%

1, where all data are normalized by the corresponding reference values (the result of ES); and Table 3.6 details their performances on the five pedestrian sets. The NOF-LPG and LOF-LPG methods outperform the other two methods. The results show that the number of targets observed using NOF-LPG is equal to that of the ES algorithm. Furthermore, the LOF-LPG method outperforms the SONG and LIM methods because it utilizes a centralized optimization technique. The proposed NOF-LPG method outperforms the other three methods because it adopts a better objective function.

As mentioned in Section 3.2.2 that the linear summation function (3.5) encourages all cameras to pursue high video quality targets without a mechanism to suppress too many cameras focusing on the same group of targets. Therefore, the LOF-LPG method may lose track of some targets. Figure 3.4 shows an example of this effect found in the first simulation at time instance 21. This figure shows that target 4 is ignored by the LOF-LPG algorithm because the sum of the quality values of targets 2 and 6 is greater than that of target 4. Conversely, the NOF-LPG appropriately controls cameras' FOVs to obtained more targets.



(a)



(b)

Figure 3.3: (a) The true number of moving objects in Pedestrian set 1 and the number observed by the exhaustive search (ES) method; (b) the normalized numbers of targets observed by the compared methods (Pedestrian set 1).

In the second simulation, we assess the effects of two types of noise that are inevitable in a real environment. The first type of noise is caused by target detection errors, which yield an incorrect number of targets; and the second type is target location errors caused by target tracking and/or prediction errors. The miss detection rate is set at 10% in this simulation, and the target position is perturbed by zero mean white Gaussian noise with a standard deviation of 0.5 meters. In addition, a 20% gross location error is generated by a random walk whose step size and orientation angle are uniformly distributed within $[2,5]$ meters and $[0,2\pi]$ respectively. The number of moving targets and the method

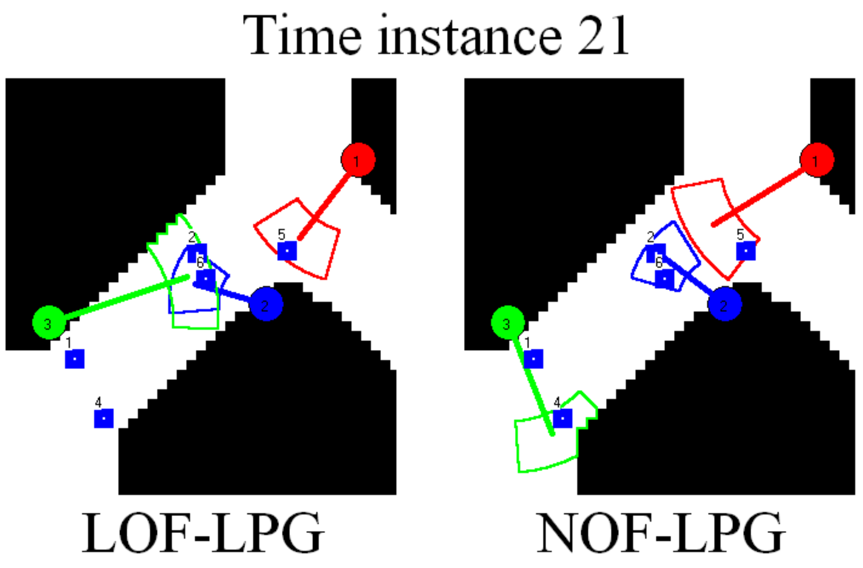


Figure 3.4: The snapshot of NOF-LPG and LOF-LPG at time instance 21.

Table 3.7: The percentages of targets observed by the compared methods in noisy pedestrian sets

	Pedestrian set 1	Pedestrian set 2	Pedestrian set 3	Pedestrian set 4	Pedestrian set 5
NOF-LPG	92.5%	88.8%	91.0%	87.0%	89.9%
LOF-LPG	91.0%	87.1%	85.8%	86.1%	85.4%
LIM	72.9%	64.0%	75.1%	69.0%	71.5%
SONG	68.5%	66.5%	67.1%	61.3%	65.1%

used to compute the reference values are the same as those in the first simulation. The results of the second simulation, shown in Table 3.7, demonstrate that incorrect target information does degrade the decision quality. Even so, NOF-LPG still outperforms the other methods.

In the third experiment, we simulated a large scale surveillance system with 30 virtual PTZ cameras and 45 virtual pedestrians. Figure 3.5 shows the simulation environment. In the simulation, we tested the performance of different approaches under two different conditions: 1) the positions of the pedestrians are provided by a global-view static camera and 2) the positions of the pedestrians are estimated by the PTZ cameras. In the latter case, the positions of those pedestrians which are located outside the FOVs of all the cameras will be not available. Therefore, the control strategy is made with insufficient information and a degradation of tracking accuracy should be expected. Since the exhaustive search of this large-scale simulation is impractical due to the high computational complexity $O(45^{30})$, the number of tracked targets compute by NOF-LPG with a global-view static camera is used to replace the ES result as a benchmark. Table 3.8 shows the results of computer simulation. In this large-scale surveillance simulation, NOF-LPG achieved the best performance. Furthermore, when a global-view static camera is not available, 10% drop of the tracking performance is observed for every method tested in this experiment. Moreover, since the SONG method adjusts one camera at a time, each camera is adjusted every 30 iterations in this simulation. Hence, its performance is further degraded. The average computation time of the third simulation is shown in Table 3.5. The computation time of each method is approximately of the same order. In this simulation, the maximum targets appeared in any of the camera FOVs is only five. Therefore, the computation time of the proposed NOF-LPG method is slightly longer than that of the other methods.

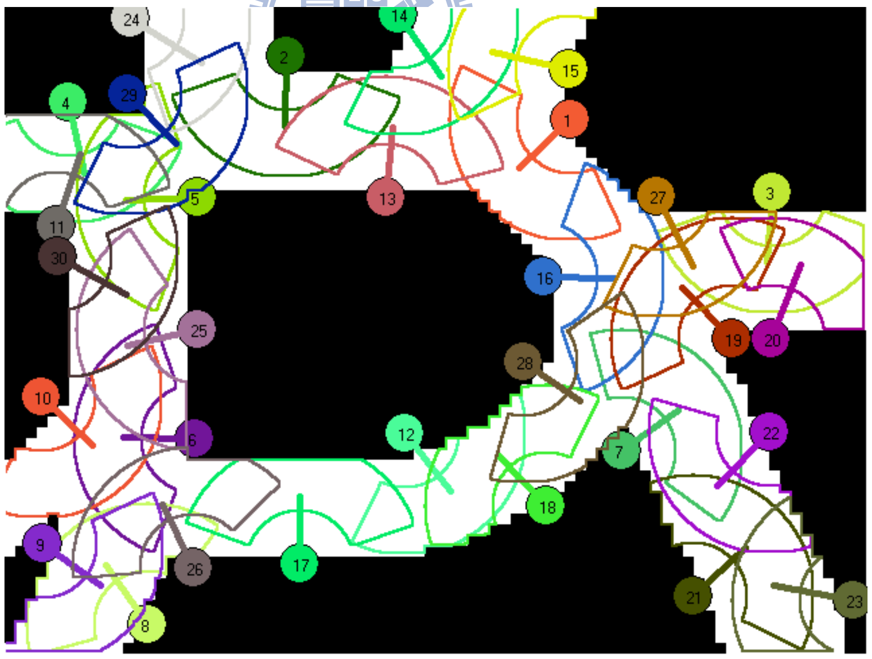


Figure 3.5: The scene involved with 30 virtual PTZ cameras in the third simulation.

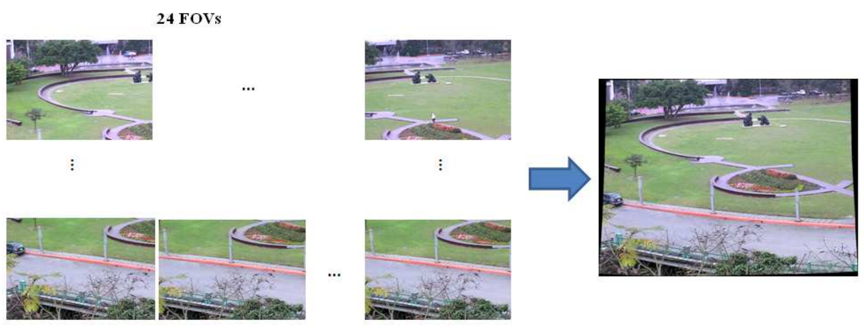
We also conducted an experiment using real-world data to evaluate the performance of the four methods. A camera network comprised of two PTZ cameras (AXIS AX-215) and

Table 3.8: The percentages of targets observed by the compared methods in a large scale scene.

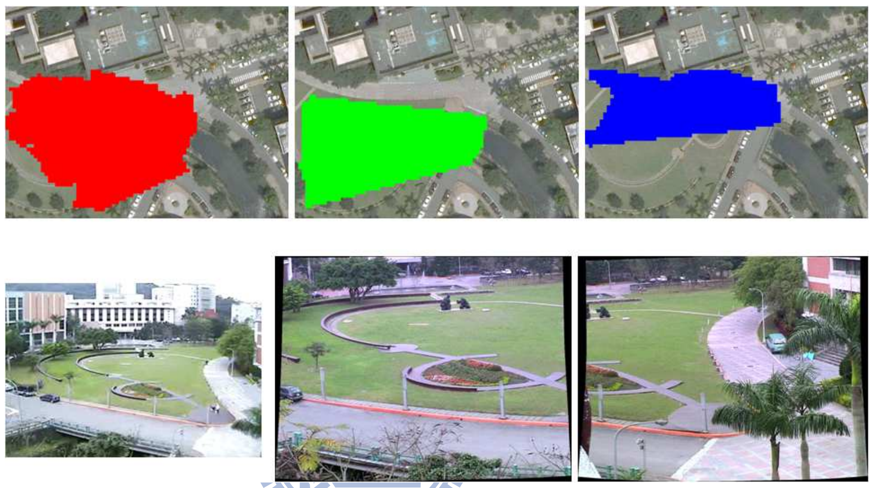
	Performance with a global fixed camera (normalized by the result of NOF-LPG with a global fixed camera)	Performance without any fixed camera (normalized by the result of NOF-LPG with a global fixed camera)
NOF-LPG	100.0%	90.0%
LOF-LPG	98.1%	87.5%
LIM	96.8%	73.0%
SONG	45.5%	35.6%

one fixed camera (D-link DCS-3220G) was deployed in an outdoor environment. There are quite a few automatic methods to estimate the image homographies, such as [58] and [59]. However, since estimating a homography only requires four pairs of correspondence points, it is not difficult to manually construct the correspondence points. Therefore, in the experiment, all cameras were calibrated by selecting landmarks manually and the homography map was computed by using functions in OpenCV [60]. The video resolution of each FOV was 352×240. At the system setup stage, 24 predetermined FOVs were assigned to each PTZ camera. To calibrate the homographies of the predetermined FOVs, the images corresponding to the 24 FOVs were acquired and used to build a panoramic image of the PTZ camera with Autostitch [61]. Therefore, the homography between any of the 24 FOVs and the corresponding panoramic image is known. At present, the 24 FOVs only contain two zoom levels because, if the FOV is too narrow, the image features would be insufficient (feature points < 4) to estimate the image homography reliably. The panoramic images of the two PTZ cameras and the images of the fixed camera are related by registering them to a top-view aerial image. Figure 3.6(a) shows the set of 24 acquired images and the registered panoramic image; and Figure 3.6(b) shows the visual coverage regions of the three cameras overlaid on a top-view image. Therefore, the homography between any two cameras is known.

The video from the fixed camera is used for target detection and prediction; and Yoo and Park's difference-based approach [62] is utilized to detect moving targets. Although the approach in [62] may fail to detect semi-stationary targets, it is efficient and robust



(a)



(b)

Figure 3.6: (a) A panoramic image derived by integrating 24 images; (b) the coverage regions of the three cameras overlaid on a top view image and the panoramic image of the three cameras.

to lighting variations; thus, it is very suitable for an outdoor real time system. A simple constant-velocity motion model is used to predict the locations of targets. Figure 3.7 shows a snapshot of the targets detected in the video stream of the fixed camera.

Since it is impossible to control the PTZ cameras by using the four compared methods simultaneously, we tested them one by one. For comparison, each algorithm was tested for 20 minutes (about 5400 frames) on a cloudy and windy day. At each iteration, the system required about 0.8 seconds to estimate the parameters of the constant-velocity model of each moving target in order to predict its next position. Then, new parameters were determined and sent to the PTZ cameras. Usually, the cameras took 0.6 seconds to move

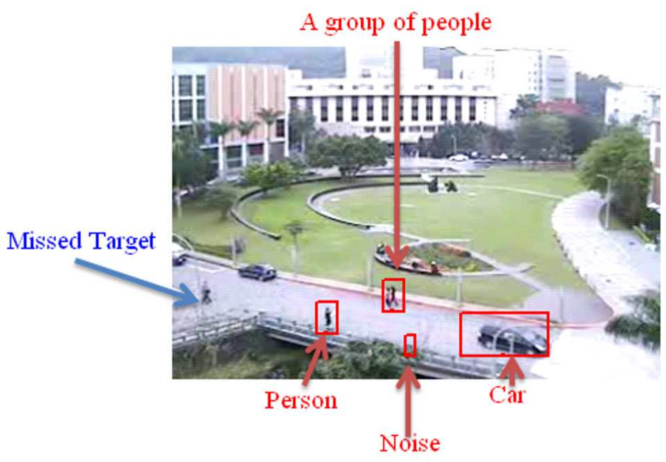


Figure 3.7: Detected targets (a single person, a group of people, a car, and noise)

to the assigned FOV. Finally, the targets observed by the FOVs were counted to evaluate the performance. Figure 3.8 shows the NOF-LPG results at three time instances. In video frame 1470, several groups of people are detected and the two PTZ cameras are instructed to observe different groups of targets for maximizing the observed target number. Video frames 3358 and 3444 exemplify a target hand-over between cameras, where the group observed by camera 2 in frame 3358 is passed to camera 3 in frame 3444. As in the simulations, the exhaustive search method is applied to obtain reference data. However, since it is impractical to test all combinations of the cameras' FOVs when determining the optimal solution in a real environment, the exhaustive search is based on the video recorded by the fixed camera. Each feasible FOV of the PTZ cameras is mapped to a region in the image of the fixed camera. Then, all combinations of the mapped regions are utilized to search for the maximum number of observed targets as reference data. Figure 3.9 shows the corresponding reference data of the four video sequences used to test the NOF-LPG, LOF-LPG, LIM and SONG methods; and Table 3.9 details the performances of the four algorithms in the real-world experiment.

Although the performance of each algorithm is affected by shadows, wind, calibration errors, target detection errors, and prediction errors, the results demonstrate that NOF-LPG and LOF-LPG still outperform LIM and SONG.

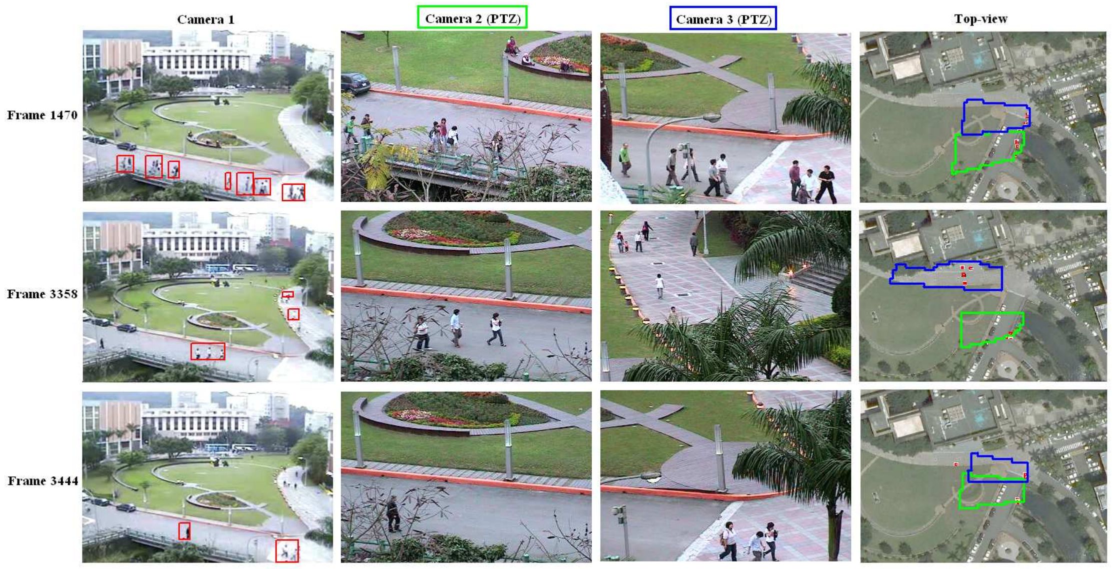


Figure 3.8: The tracking results of frames 1470, 3358, and 3444 using NOF-LPG.

Table 3.9: The percentages of targets observed by the compared methods in a real environment

	Performance (normalized by the result of exhaustive search in the fixed camera)
NOF-LPG	71.0%
LOF-LPG	67.3%
LIM	60.2%
SONG	62.3%

3.5 Concluding Remarks

In this chapter, we have considered the process used to assess the quality of a set of FOVs and proposed a non-linear objective function to reduce the number of unattended targets in a surveillance region. When the quality of each FOV is evaluated individually, it is easy to derive the objective functions of different PTZ network problems. The proposed approach provides an optimal solution for the PTZ network coordination problem. We have also shown that the non-linear optimization problem can be transformed into a linear production game problem that is guaranteed to yield an optimal solution. The optimal solution of LPG can be computed in polynomial time so our approach is efficient. The

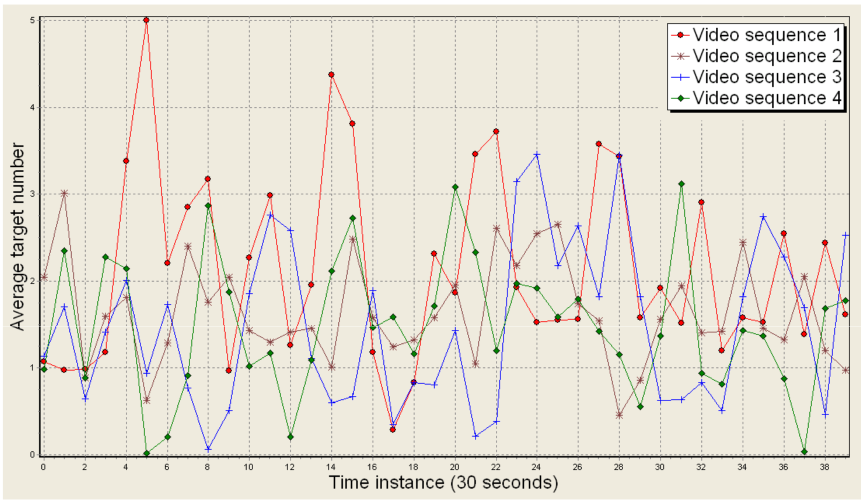


Figure 3.9: The average number of targets identified by the ES algorithm in the video sequences used to test the NOF-LPG (video sequence 1), LOF-LPG (video sequence 2), LIM (video sequence 3) and SONG (video sequence 4) methods.

branch-and-cut method is adopted to solve the PTZ parameters selection problem. Computer simulations and a real-world experiment were performed to evaluate the proposed method in a multi-target surveillance environment. The results show that the method achieved the highest tracking rate among the compared methods.

Chapter 4

Human Motion Recognition

In this chapter, a local feature-based human motion analysis framework is described. First, we introduce the human motion recognition problem and review the related works. The proposed approach about feature extraction and training process is then detailed. Next, the experiment results are shown. Finally, we present our conclusion.

4.1 Introduction

In the past ten years, human motion recognition has attracted a great deal of attention in video surveillance field [63–68] due to its broad applications. Human motion analysis based on video is a vivid research topic, and it is known to be a difficult task due to two reasons: Firstly, human body movement involves complex articulated motions belonging to a large set of segments, thus the developed recognition algorithm must have the capability to handle a complex motion with high degrees of freedom. Secondly, it is not practical to analyze human behavior in frame-by-frame base; an alternative approach would be utilizing the spatial-temporal information, which greatly increases the amount of data to be processed. The existing works dealing with the above addressed two issues can be divided into two categories. The first category is called the statistical-based methods. The basic idea associated with this category is analyzing the low-level features to discriminate different human motions. Haritaoglu et al. [63] proposed the W4 system, which computes the vertical and horizontal projections of a silhouette to determine the global posture of a human subject. Bobick and Davis [64] constructed two temporal templates called motion energy images (MEI) and motion history images (MHI) by stacking a set of consecutive frames. Moment-based features are then extracted from MEI and MHI and used to perform template matching. Ke et al. [69] aligned video frames to generate a spatial-temporal volume. Classifiers of local features are trained by observing the variations of vertical and horizontal optical flows in the volume. Subsequently, a feature selection process is applied to find the best features for recognizing a motion. However, it is hard to obtain a complete posture by statistical-based methods, because image segmentation is still an ill-posed problem. The second category is called the trajectory-based methods. Methods belonging to this category utilize the trajectories of body parts to describe human motions. Trajectories have higher tolerance for environmental noise than low-level features, thus they have a better discriminating capability. For example, Min and Kasturi [65] extracted limbs' trajectories by tracking the maximum magnitude points of the optical flows and used them as landmark points to train the descriptors. Hidden Markov Model (HMM) is adopted and trained to recognize the human motion trajectory. A view invariant representation based on trajectories of human body parts was proposed by Yilmaz and Shah [70]. Their system recognizes and retrieves human actions from multiple videos captured from different views or from a single moving camera. Although the trajectory-based methods do not reply on assumptions pertinent to the completeness of posture detection, the robustness of landmark detection is still another challenging issue. Moreover, the challenging research issues in field of object recognition and categorization also include that the developed method is expected to be able to tolerate situations, such as when objects belong to the same category but differ in appearance (for example, chairs with different numbers of legs, as shown on the bottom left of 4.1) and when an object is viewed from different viewing directions. These issues might be overcome by local feature-based representation methods. The essential idea of such methods is to determine reliable object parts and use them to represent and recognize the specified object. During the partial features matching process, the object's structure is better to be taken into consideration to enhance the robustness, which also provides clues for identifying outlier features. For example, Leibe et al. [71] learned a codebook of local appearance and then proposed the Implicit Shape Model (ISM) to indicate the appearance location of an object. Opelt et al. [72] trained the fragment features by Adaboost to obtain a feature set that provides the best recognition result. The concept of those approaches is illustrated in Fig. 4.1.

The above-mentioned challenging issues also exist in research on human motion recognition. First, the same category of objects with different appearances is analogue to a motion composed of different movement styles, such as waving at different speed or arm positions. Second, a motion may be captured from different viewing angles, which is similar to the case of an object being viewed from different viewing directions. Therefore, we propose to adopt the idea of the local feature-based representation methods and apply it in motion recognition tasks.

Extracting a set of suitable local features is an important step in the local feature-based representation method. However, it is not suitable to directly applying the method in [71] or [72] (designed for static object recognition) to human motion recognition, because the spatio-temporal relationship provides rich information for describing a human motion which was not taken into account in [71] and [72]. Therefore, a motion-fitted feature extraction approach taking into account the spatio-temporal relationship is proposed in this chapter.

Utilizing whole body posture information [63, 64, 68] is a common approach for human motion recognition. In Fig. 4.2, the source images, whole body postures, static regions, and moving regions are shown from the top to the bottom row, respectively. The moving regions have higher discriminating power than the static regions but constitute only small parts of the whole body postures. In contrast, the static regions occupy the dominant parts of the whole body postures. Hence, it would be easily biased if whole body postures are used for human motion recognition. We proposed to apply frame difference technique to extract the moving regions, and use only those regions having higher discrimination capability for the recognition tasks. In this chapter, a human motion recognition framework adopting local feature-based representation is proposed. This approach manages to handle the missing information problem, a key issue for both the statistical- and trajectory-based methods, which often arises due to the imperfect image processing results and the noisy environments. The experimental results show that the proposed approach yields high average recognition accuracy all above 86%. Even when the extracted information is incomplete and noisy, the method still provides a reliable recognition results.

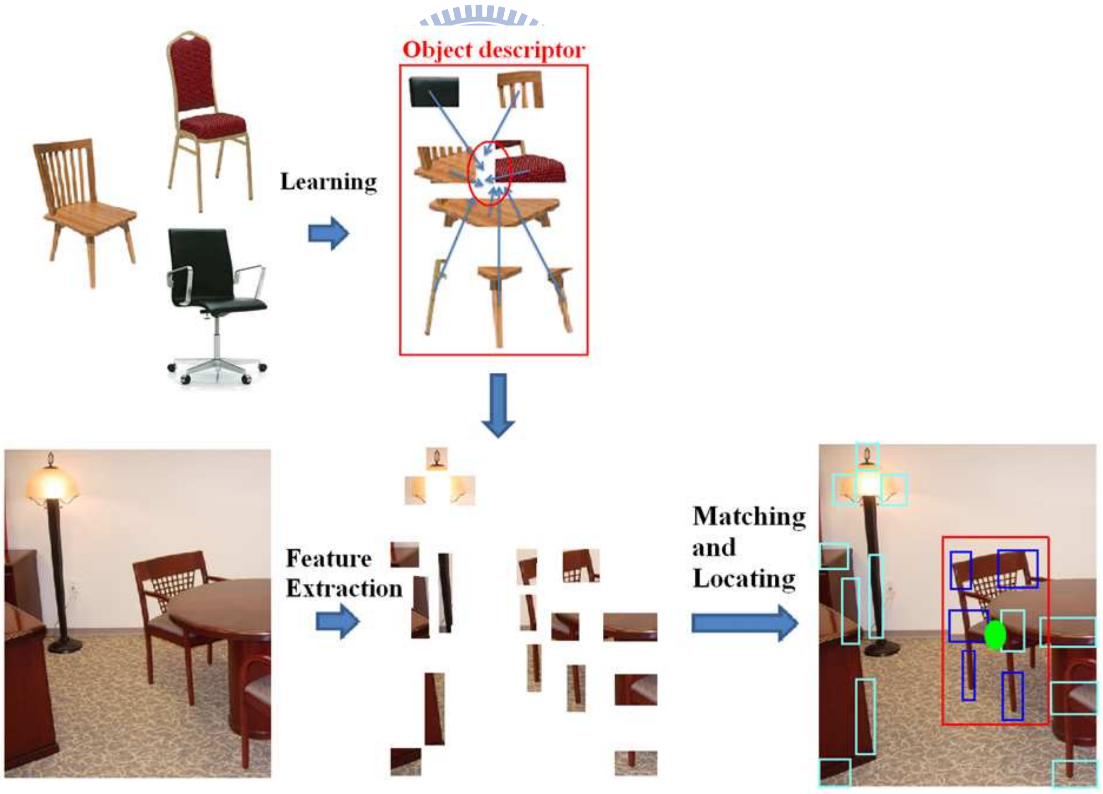


Figure 4.1: A concept of local feature-based representation and object detection.

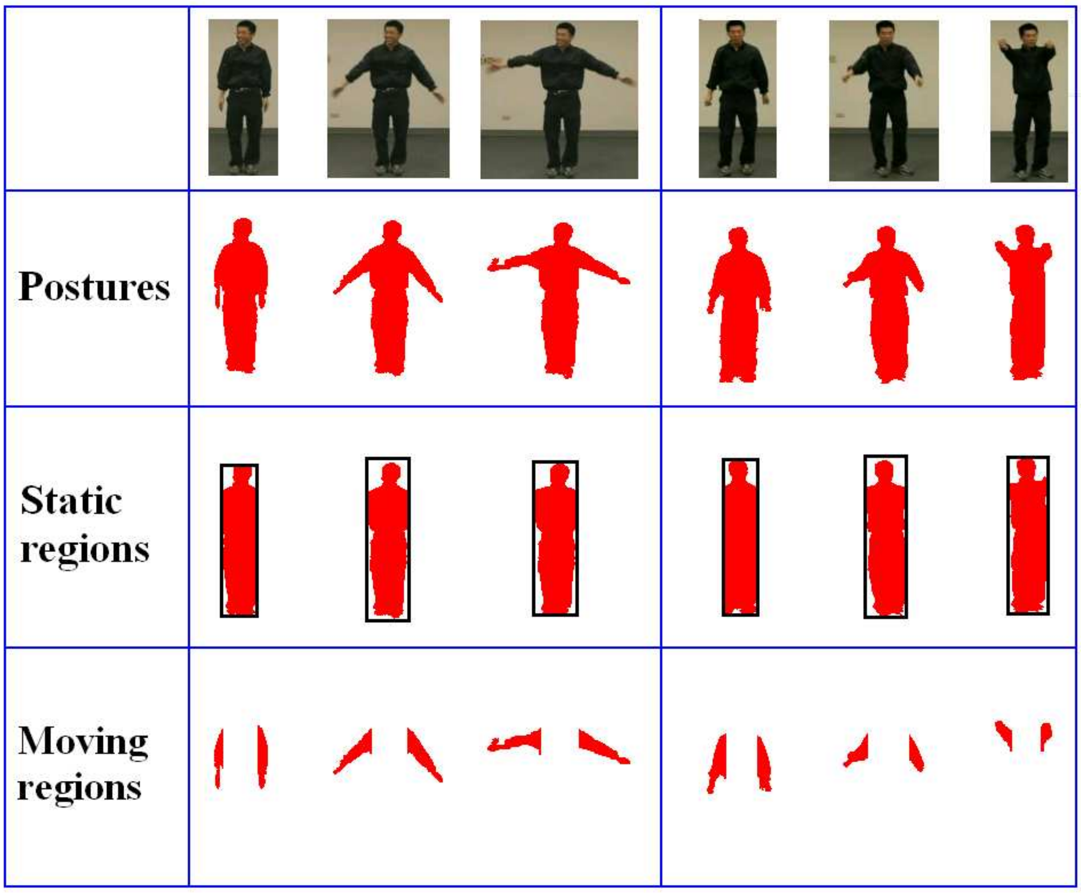


Figure 4.2: An example includes two different human motions, their posture, static regions, and moving regions.

4.2 Local Feature-based Human Motion Recognition Framework

To overcome the drawbacks and limitations of the existing methods mentioned in the last section, a local feature-based human motion recognition framework is proposed. The flow chart of the framework is depicted in Fig. 4.3. In the learning phase, the system first determines whether the input human motion is a whole-body motion. If it is, then an alignment process is performed and followed by the limb motion extraction. Otherwise, the limb motion extraction can be directly applied to the source images. Next, the motion features are extracted from the resulting limb motions and are categorized into two types according to the motion durations, namely the long-term and the short-term features. Different mathematical models are used for representing the long-term and the short-term features. The relations among those local limb features are constructed and used for training the human motion recognition. Finally, in the learning phase, a human motion detector is built based on the analysis results of the local limb features. In the recognition phase, the same feature extraction process is performed to extract the local limb features from the input testing video data. The extracted features are then compared with the stored ones in the human motion detector. If a match is identified, the system returns the type and the position of the motions.

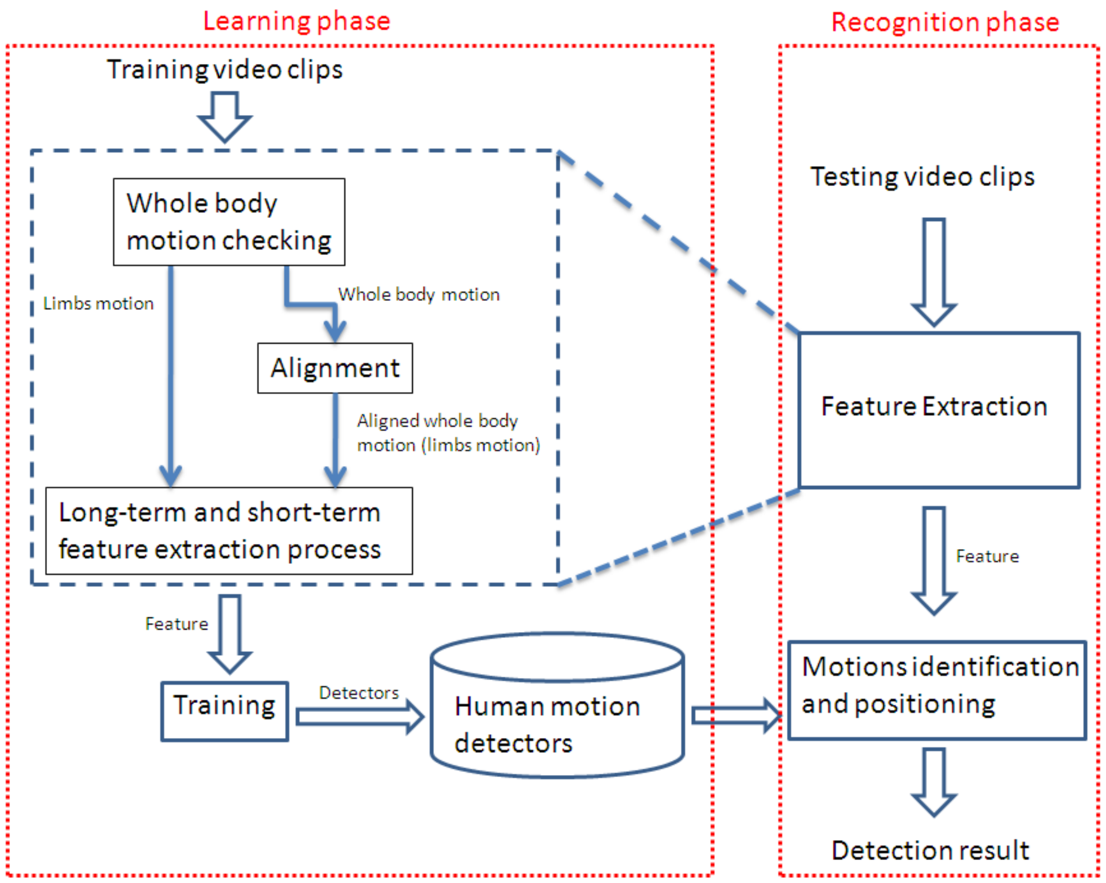


Figure 4.3: An overview of the proposed human motion recognition framework.

4.3 Local Feature-based Representation

In order to extract only the local motion features, it is essential to analyze the position of the moving human object within each image. If the motion was performed at the same location, the variance of the positions over a period of time is small, it is considered as a limb motion sequence. In a contrast to that, the motion is considered as a whole-body motion sequence, such as walking or running, and an alignment process is necessary before the local motion feature extraction takes place. Clay representation [73] is used for long-term feature to represent trajectories, and a method for accumulating the moving edge patches is introduced for the case of short-term feature. All these methods will be described in detail in this section.

4.3.1 Whole Body Motion Alignment

When dealing with the whole-body motion sequences, the body motion alignment is an important step towards the accuracy of the whole recognition system. The aim is to extract only the local limb motions as precisely as possible, and use them as the features to train the motion detector. First, the moving edge maps of a torso across different frames are traced to measure the whole body displacement vector. The moving edge map can be obtained by frame difference technique and Canny edge detector. Next, the Chamfer-matching [74] method is applied to estimate the nearest distance from the pixels in the current moving edge map to the next time instant. The Chamfer-matching extracts the displacement with lowest distance within a search area and then assigned as a displacement vector to align whole body motion between two frames. The search area of the whole body displacement can be bounded in a small region, because the time interval between adjacent frames is short. As a result, the estimated vector is applied to align the edge maps.

4.3.2 Clay Representation of Trajectories (Long-term Feature)

In [73], we proposed a grouping-based trajectory extraction approach that utilizes a clay model to represent the trend of a limb's trajectory, which is robust against noisy environment. In this chapter, we adopt the clay representation of trajectories in [73] as the long-term feature by considering its spatio-temporal characteristic. The details about the trajectory extraction and clay representation are stated as follows.

Based on the assumption that all points belong to the same moving articulated part have homogeneous movement trends during a period of time, as shown in Fig. 4.4(a), the trajectory of a moving articulated part can be determined. We should notice that no landmark points should be selected, providing better noise tolerance capability than a single landmark point tracking approach, which would possibly rely on a noisy point tracking result.

First, the edges of adjacent frames are extracted by frame differencing the two frames, adopting the Canny edge detector. Then, a uniform sampling is utilized for generating the moving edges. The corresponding point pairs are obtained by Shape Context [75], forming the motion flows with the continuative linkage to form rough trajectories, as shown in Fig. 4.4(b). The short trajectories with too few intersecting points are considered as noises, a conventional nearest-neighborhood (NN) clustering is adopted to assign all trajectories to different clusters. Figure 4.4(c) shows a clustering result. The detail about the NN clustering for the trajectories can be obtained in [73]. Next, a voting processing based on the neighboring region within a certain radius r is achieved according to the points belong to the same cluster at each timestamp, as shown in Fig. 4.4(d), yellow points belong to one cluster, and purple points belong to another cluster. In addition, the voting process concept is shown in Fig. 4.4(e). High overlapping regions are assigned as the candidate for the moving articulated parts as shown in Fig. 4.4(f) (the yellow region and the purple region). Then, the trajectories of articulated parts are extracted by linking the centroids of the candidate regions across the time axis. Finally, the Gaussian-filtered trajectories can be generated, as shown in Fig. 4.4(g).

After trajectory extraction, a suitable trajectory representation is required for recognition, being able to work efficiently and handle translation, scaling, and noisy trajectories. The clay representation we proposed in [73] fulfils above-mentioned criteria well. Based on the clay representation, a trajectory can be thought as the force that pushes the limb. When the force is applied to a clay-like material, the deformed clay material is utilized to represent the trajectories. To build our clay model, we adopt the approach proposed in [76], which builds a real-time 3D virtual clay model and a move tool to simulate the clay deformation. Our model is modified from the original 3D model to fit our 2D extracted trajectories. In addition, our model is manipulated to allow performing an inner force. In our proposed clay model for representing trajectories, at first, the location of the clay is assigned by the bounding box of the trajectories. Next, the sequential motion vectors of a trajectory are applied to the clay, according to its time ordering, as shown in Fig. 4.5(a), from left to right and top to bottom. As a result, the bounding box (the blue rectangle in the left part of Fig. 4.5(b)) of non-zero density regions is extracted and normalized to a fixed size feature array (the right part of Fig. 4.5(b)) as a clay pattern for further matching.

After the clay pattern constructed, a normalized vector $\overrightarrow{p_c^i p_r^j}$, directing from $p_c^i = (x_c^i, y_c^i)$, the centroid of i-th clay pattern, to the j-th reference point p_r^j which is obtained by the mean centroid of all clay pattern in the j-th video. The vector is attached to be the clay feature for the following training process. Although the proposed clay representation of trajectories can reveal the movement trend of a moving limb, it still lacks the shape information. On the other hand, since the trajectory only consider about the long-term trend of a moving limb, the short-term property and the isolated movement are still unconsidered. To achieve the above-mentioned requirements, we propose a so-called accumulated moving edge patches as the short-term feature to compensate for the shortage of only adopting the clay representation of trajectories as the long-term feature. The details about accumulated moving edge patches are described in the following subsection.

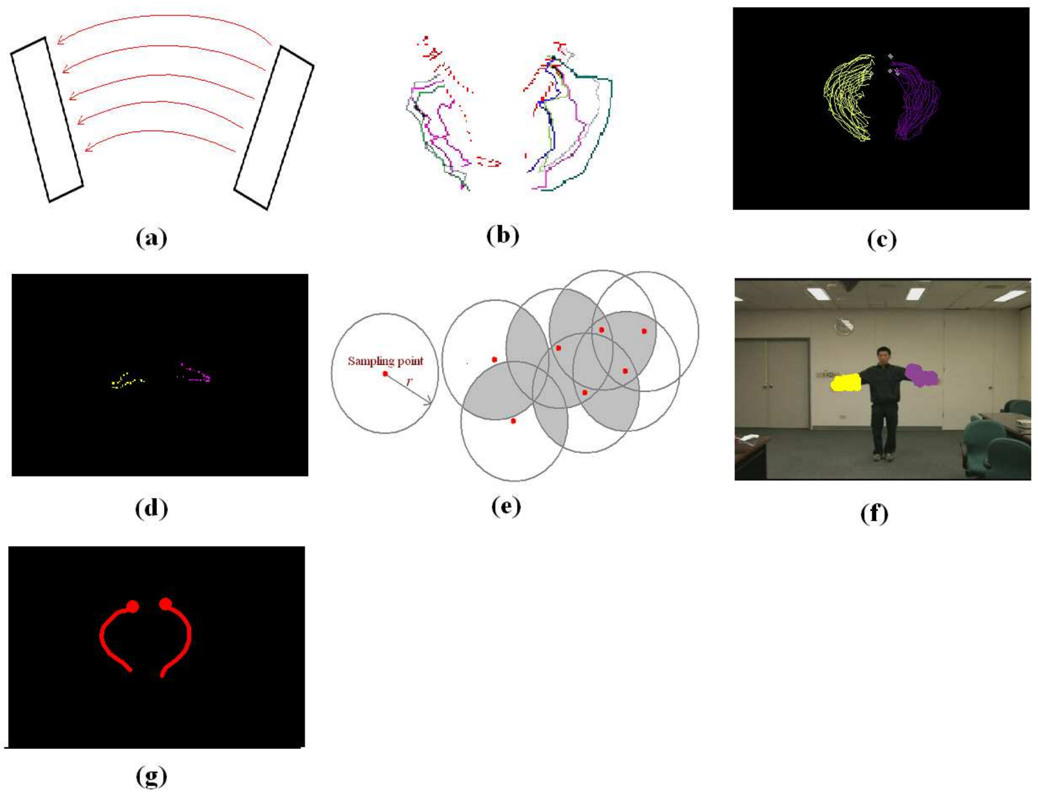
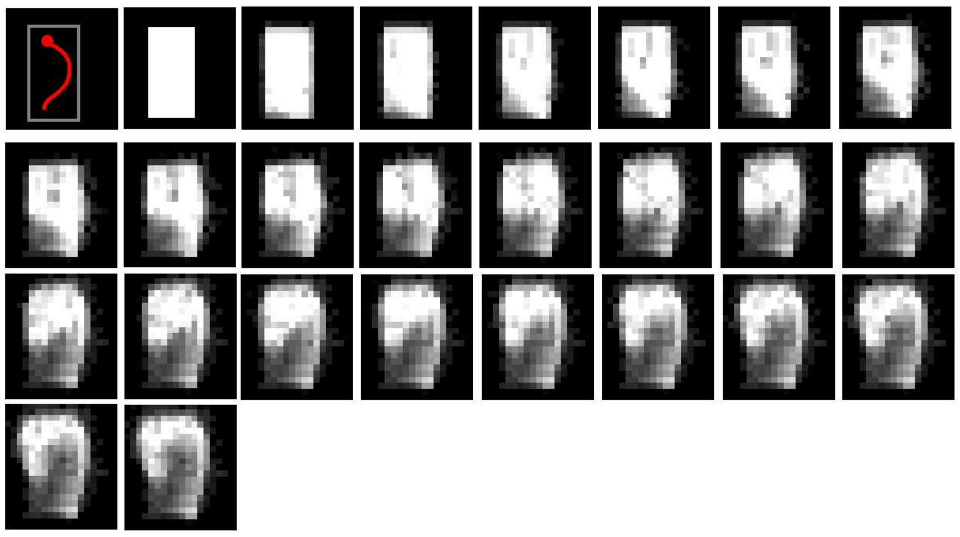


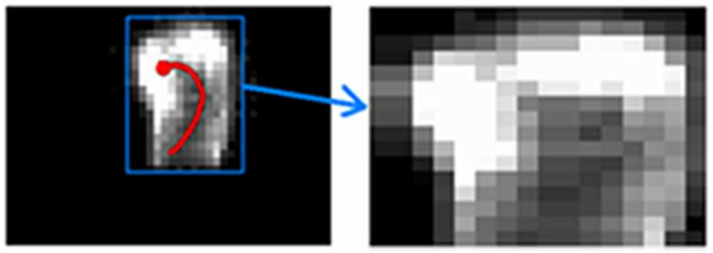
Figure 4.4: (a) Trajectories generated by the sampling points of the same moving articulated part; (b) generating trajectories in a real case; (c) trajectories after clustering; (d) sampling points clustered at a timestamp; (e) voting neighboring regions based on sampling points; (f) generated candidate regions at a timestamp; (g) extracted trajectories.

4.3.3 Accumulated Moving Edge Patches (Short-term Feature)

First, a moving edge map at timestamp t, $E(x,y,t)$, is extracted with the shapes' information of the moving limbs (the same process in the Section 4.3.2). Next, these edge maps are accumulated based on the MHI [64] to determine the causality of moving edges along the temporal axis. According to the concept of MHI, the MHI value decreases with the magnitude proportional to the duration from the current timestamp to the beginning timestamp. The MHI is calculated as follows:



(a)



(b)

Figure 4.5: (a) A piece of clay deformed by a sequential motion vectors according to the time orders; (b) a 2D clay pattern produced from a trajectory (red line in the left part).

$$H_\tau(x,y,t) = \begin{cases} \tau & ,ME\,(x,y,t) > 0 \\ \max\,(0, H_\tau\,(x,y,t-1) - 1), & otherwise \end{cases} \tag{4.1}$$

$$ME\,(x,y,t) = E\,(x,y,t) * G(x,y) \tag{4.2}$$

where $E(x,y,t)$ is the moving edge map at timestamp t, $G(x,y)$ is a 2D Gaussian filter, and $ME(x,y,t)$ is a blurred version (Gaussian filtered $E(x,y,t)$) of the moving edge map $E(x,y,t)$. The Gaussian filter process to the moving edge map $E(x,y,t)$ in each frame can suppress the noisy edge detection result due to the variation of lighting condition, shadow, and texture on the foreground subject. Furthermore, based on the Gaussian filtered moving edge map, the accumulated MHI result in a period of time is more reliable.

In MHI, a time axis sliding window size should be determined as the signal observing time duration. In our proposed method, an adaptive time axis sliding window size, s, is automatically determined according to the video content property. The number of moving edge pixels in each frame are counted frame by frame and accumulated across the time-axis. If the accumulated number is greater than an adaptive threshold θ_w, the current pattern will be saved as an MHI. The adaptive threshold θ_w is set by multiple of the average moving pixel number $\theta_w = mN_{avg}, m > 1$, N_{avg} is the average moving pixel number. Moreover, to solve the human body size scaling problem and increase available features, adaptive thresholds$(\theta_w^1, \theta_w^2, \theta_w^3, \ldots)$ are set automatically based on the different average moving pixel number. After generating the MHIs, each MHI is broken into pieces of a local feature. An example of the proposed accumulated moving edge patches generation is shown in Fig. 4.6. First, a biggest yellow circle in the left-most figure in Fig. 4.6 is set to bound a MHI pattern, meanwhile, the circle is uniformly divided along the radius (the rest yellow circles in the left-most part in Fig. 4.6) and the angle direction (the green line segments). Each intersecting point (yellow circles and green lines) is set as a center point (blue rectangles in the second figure in Fig. 4.6 from the left) of a feature patch. At each center point, the neighboring square rectangle regions (the square rectangles with different colors in the third part from the left in Fig. 4.6) are copied to be a square feature patch, and the length l_{patch} is determined by a fixed ratio of a bounding circle's radius b_r as: $l_{patch} = fb_r, f = [0,1]$. Here, several different patch size ratios are set for handling scaling and noise problems. Finally, each extracted patch is normalized to a fixed size patch. A reference point p_r^j is set automatically by the centroid of all patterns in the j-th time axis sliding window. Finally, the normalized vectors $\overrightarrow{p_c^i p_r^j}$, directing from $p_c^i = (x_c^i, y_c^i)$, the centroid point of the i-th patch, to a reference point p_r^j, as shown in the right-most part in Fig. 4.6, $\overrightarrow{p_c^i p_r^j}$, is attached to be the accumulated moving edge patch for the following training process.

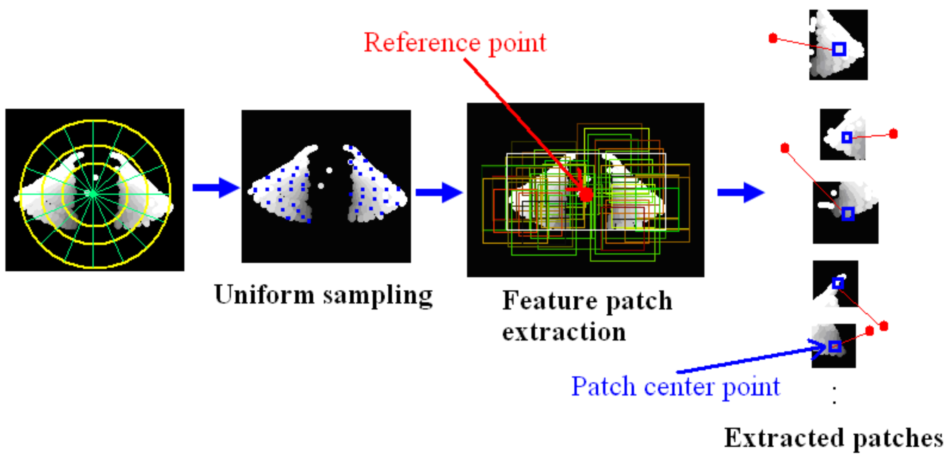


Figure 4.6: The sampling and the accumulated moving edge patch extraction process [right-most: red dots: reference points; patch center point-¿patch 'centroid' point].

4.4 Learning and Recognition

In the previous section, a large number of potential features are extracted to ensure that long-term and short-term features can be kept for further training and recognition. However, because the features may contain redundant information, we apply Viola and Jones' [77] Adaboost learning approach to select key features for representing a limb's motion. In addition, the Classification and Regression Trees (CART) scheme [78] is adopted to build the weak learner for learning the limb's motions. By using the CART scheme, the high discriminating feature bins are selected for identifying a motion. After all weak learners have been constructed by CART, the AdaBoost algorithm [77] apply weak the learners to the training data set, as a result, the best weak learners can be generated. The best weak learner is combined into a strong learner in each iteration. In our framework, two features (clay representation of trajectories and accumulated moving edge patches) are trained separately. When the above two features are trained, the features are fused by the adaptive weights describing later. For a testing video clip in Fig. 4.3, the two features in the proposed method are extracted separately. The extracted features will be compared with the trained features sets. For a patch satisfying the rules in the strong learner, the 2D Gaussian voting [79] approach is utilized for voting. As shown in Fig. 4.7(a), the concept of Gaussian voting is illustrated. The rectangle patches are the matched patches by checking the strong learner. The 2D Gaussian voting is applied to the location of patches' reference points (the red arrows and blue circles) based on the attached vectors $\overrightarrow{p_c^i p_r^j}$ of each matched patches. Therefore, the response map is generated after all matched patches voted, as shown in Fig. 4.7(b). The response map is calculated as:

$$R(x,y) = \sum_{i=1}^{n_m} \frac{1}{2\pi\sigma^2} \cdot \exp^{\frac{-Dist(L_i,x,y)}{2\sigma^2}} \cdot \alpha(L_i) \tag{4.3}$$

$$Dist(L_i,x,y) = \sqrt{(x-x'_{L_i})^2 + (y-y'_{L_i})^2} \tag{4.4}$$

where $Dist(L_i,x,y)$ is the Euclidean distance from (x,y) to (x_{L_i},y_{L_i}), (x_{L_i},y_{L_i}) is the location pointed by the reference vector $\overrightarrow{p_c^i p_r^j}$ in the matched weak learner L_i, and $\alpha(L_i)$ is the weighting value of the associated weak learner L_i to represent the importance of the matched patch:

$$\alpha(L_i) = \begin{cases} \alpha'(L_i) \times w^{accu}, \ IF\ L_i \in f_a \\ \alpha'(L_i) \times w^{traj}, \ IF\ L_i \in f_t \end{cases} \tag{4.5}$$

Where f_a and f_t represent the accumulated pattern and the trajectory feature set, respectively. The weight w_{accu} is the ratio between total alpha value of the moving edge patches (short-term feature) and all alpha values, i.e., $w_{accu} = \sum_{i=1}^{n_a} \alpha_{accu}^i / (\sum_{i=1}^{n_a} \alpha_{accu}^i + \sum_{j=1}^{n_t} \alpha_{traj}^j)$, where n_a and n_t represent the patch number belonging to the set f_a and f_t, respectively. Similarly, w_{traj} represents the ratio between total alpha value of the trajectory patches (long-term feature) and all alpha values, i.e., $w_{traj} = \sum_{j=1}^{n_t} \alpha_{traj}^j / (\sum_{i=1}^{n_a} \alpha_{accu}^i + \sum_{j=1}^{n_t} \alpha_{traj}^j)$. Therefore, the proposed method can automatically select the weight of w_{accu} and w_{traj}, according to the video content property. After constructing the response map, a rectangle window is slid in the response map to find the best match according to the searching the region with the maximum voting value, as shown in Fig. 4.7(b). The position with the maximum value identifies location belongs to a motion. The locating for the position with a maximum value is achieved by the calculation:

$$(x,y) = \arg\max_{x,y} W(x,y) \tag{4.6}$$

$$W(x,y) = \sum_{i=x-(\varepsilon/2)}^{x+(\varepsilon/2)} \sum_{j=y-(\varepsilon/2)}^{y+(\varepsilon/2)} R(i,j) \tag{4.7}$$

where ε is search window size, and $R(x,y)$ s the voting value at the location (i,j) in the response map. An example of a response map and the motion locating process is shown in Figure 4.7(b). The center of the yellow square rectangle in Figure 4.7(b) shows the detected position with the maximum value in the response map.

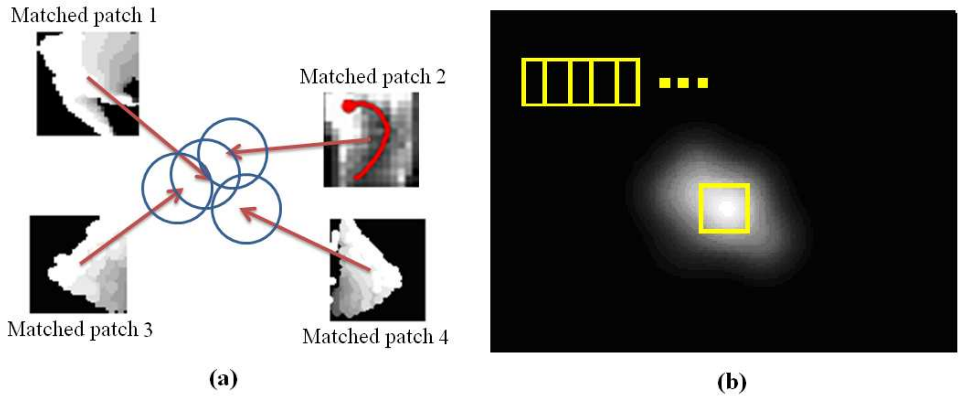


Figure 4.7: (a) The concept of 2D Gaussian voting process [17]; (b) a sliding window used to locate a motion's position.

4.5 Experimental Results

In our experiments, a large number of test video clips were used to evaluate the performance, including 10 different actions performed by 5 actors. The video clips were encoded in the standard MPEG-1 format with the resolution of 352×240. The actors in the test videos occupied from 11% to 15% of a frame's area. The actions are shown in Fig. 4.8, which includes eight limb motions (labeled as actions 1-8) and two whole body motions (walking and running, labeled as actions 9 and 10). The limbs motions were repeated five times, and the whole body motions were repeated four times in each video clips. Totally, the dataset contained 240 video clips.

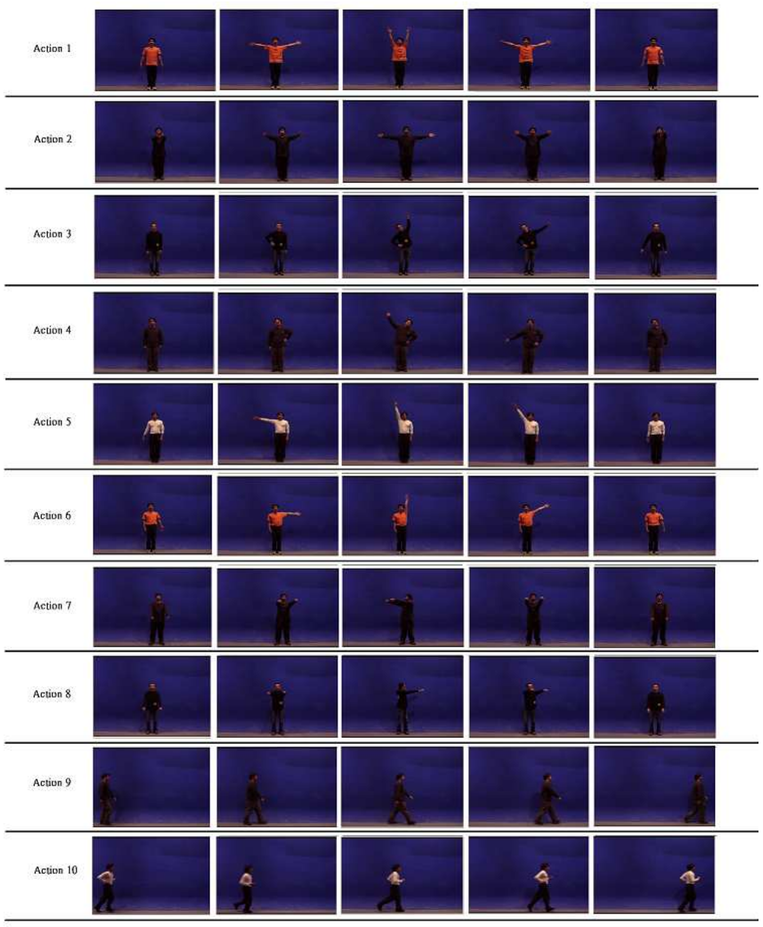


Figure 4.8: The ten actions in our test database.

4.5.1 Leave-One-Out Cross-Validation Strategy

In the first experiment, the leave-one-out cross-validation strategy was applied to test our approach. In each iteration, one actor was chosen for test and the other actors were used to train all motion detectors. In our experiments, we adopted the adaboost and the CART structure weak learner implementation provide by GML AdaBoost Matlab Toolbox [80]. Furthermore, each feature patch (long-term features and short-term features) was normalized to 16x16 bins histogram. The recognition results are listed in Table 4.1. In this experiment, the average recognition rate, 95.8%, can be calculated from Table 4.1.

Table 4.1: The confusion matrix illustrates the performance of our approach.

	Action 1	Action 2	Action 3	Action 4	Action 5	Action 6	Action 7	Action 8	Action 9	Action 10
Action 1	1	0	0	0	0	0	0	0	0	0
Action 2	0	1	0	0	0	0	0	0	0	0
Action 3	0	0	0.96	0	0.04	0	0	0	0	0
Action 4	0	0	0	1	0	0	0	0	0	0
Action 5	0.04	0	0.08	0	0.88	0	0	0	0	0
Action 6	0.04	0	0	0	0	0.96	0	0	0	0
Action 7	0	0	0	0	0	0	1	0	0	0
Action 8	0	0	0.04	0	0	0	0.08	0.88	0	0
Action 9	0	0	0	0	0	0	0	0	0.9	0.1
Action 10	0	0	0	0	0	0	0	0	0	1

Table 4.2: The recognition rates achieved when different percentages of the extracted features were dropped.

Dropped percentage	10%	30%	50%	70%	90%
Best	95.8%	95.4%	93.7%	91.2%	88.3%
Worst	94.5%	92.5%	92.9%	88.3%	82.5%
Average	95.1%	94.3%	93.3%	89.7%	86.0%

4.5.2 Noisy Environment Testing: Random Discarding Features

Recall that the advantage of our local feature-based approach was robust on incomplete data. To verify this aspect, the extracted features were discarded randomly to simulate a noisy matching process. Therefore, in this experiment, the recognition process repeated five times to ensure the consistency; meanwhile, the best, worst, and average results are recorded. The results of the noisy matching process are shown in Table 4.2. In this experiment, our approach still achieved the average recognition rate about 86% when 90% of the features were dropped. It shows that our approach is able to use only a few features to recognize a motion by preserving the key local variations and structures. Even under severe feature dropping situations, the key information can be kept to provide high discriminate capability.

4.5.3 Noisy Environment Testing: Long-term Stationary Discarding

To test how our method can handle the long-term stationary noise, a discarded pat-tern situation is utilized to simulate that some parts of a frame is occluded and the human behavior cannot be captured by a camera. All moving edges occurred in these regions were dropped to simulate long-term noise. Similarly, the discarded patterns were generated repeatedly five times, and the best, worst, and average cases are recorded. Fig. 4.9(a) and (b) show a discarded pattern (red areas) and the remaining moving edges (white areas), respectively. In Fig. 4.9(a), 30% features are dropped, and In Fig. 4.9(b), the trajectory and the moving edge patches are seriously damaged caused by spatial discontinuity situations. However, the recognition results, shown in Table 4.3, indicate that our approach can still work well even under the severe noisy situation. When 10% of the spatial information was dropped, the average recognition rate was very close to the original data without dropping anything. When 30% and 50% of the spatial information were dropped, the average recognition rates still achieved 84.99% and 74.56%, respectively.

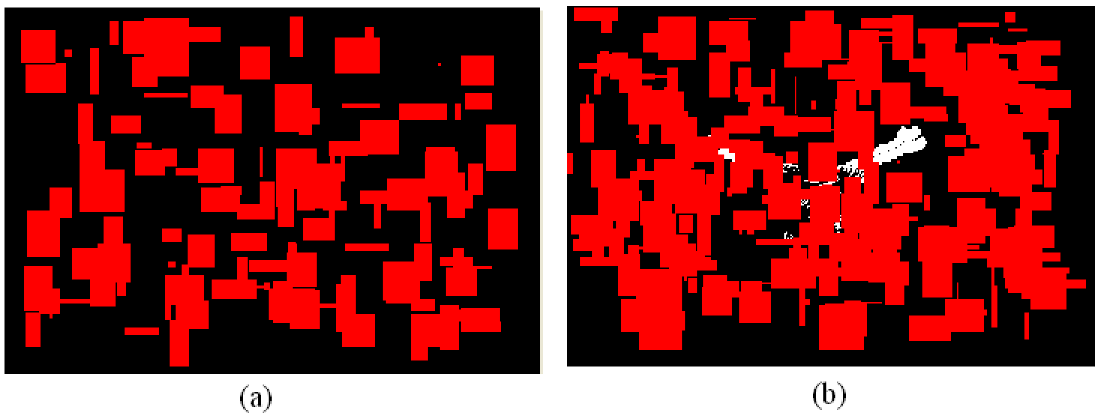


Figure 4.9: (a) A randomly discarded pattern (30% of the features dropped); (b) an example of the remaining information (50% dropped) .

Table 4.3: The recognition rates when different percentages of spatial information are dropped.

	Drop 10% spatial information	Drop 30% spatial information	Drop 50% spatial information
Best	95.8%	89.1%	79.5
Worst	93.3%	81.2%	72.5%
Average	94.8%	84.9%	74.5

4.5.4 Weizmann Dataset Testing

Finally, the Weizmann dataset [81] was used for comparison. The dataset included ten actions performed by 9 actors. Five of the actions were local limb motions and the rest were whole body motions. Table 4.4 shows the leave-one-out cross-validation (the same strategy as our first experiment) recognition results. In the experiment, the total recognition rate of the ten actions is 87.4% because of obscure local variations caused by the low resolution dataset (180×144). For example, the ”skip (single leg jump)” motion was often confused with ”jump” or ”walk” motions because its local variation was not significant enough to differentiate it in low resolutions. On the other hand, Table 4.5 lists the reported results of other state-of-the-art approaches. Table 4.5 shows that our approach is better than Niebles and Li [66] and Scovanner et al. [67]. In the last row, it shows that Weinland and Boyer’s approach [68] yields higher recognition rates than our method, which mainly because their method relies on two assumptions: First, the authors assume that the training silhouette set is reliable. Second, the authors assume that the center of a human subject can be located easily. In both of the assumptions, a reliable background model or a manual selection for the foreground human subject is required. Therefore, although their approach has higher recognition rates, our approach is more practical to be implemented in the real-world environment, because our proposed approach is not based on the abovementioned assumptions.

Table 4.4: The resultant confusion matrix using the Weizmann dataset.

	bend	jack	jump	pjum	run	side	skip	walk	wave1	wave2
bend	0.88	0	0	0	0	0	0	0	0	0.11
jack	0	1	0	0	0	0	0	0	0	0
jump	0	0	0.88	0	0	0	0	0.11	0	0
pjum	0	0	0	1	0	0	0	0	0	0
run	0	0	0	0	0.88	0	0	0.11	0	0
side	0	0	0	0	0	1	0	0	0	0
skip	0	0	0.44	0	0	0.11	0.22	0.22	0	0
walk	0	0	0	0	0	0	0	1	0	0
wave1	0	0	0	0	0	0	0	0	0.88	0.11
wave2	0	0	0	0	0	0	0	0	0	1

Table 4.5: The recognition results derived by non-background modeling approaches

Approach	Recognition rate on Weizmann dataset
Niebles and Li [66]	72.8%
Scovanner et al. [67]	82.6%
Our approach	87.4%
Weinland and Boyer [68]	93.6%

4.5.5 Limitations and Discussion

However, our approach has the following limitations: First, the motion detection result greatly depends on the accuracies of the body motion alignment. If the alignment is inaccurate due to the missing edges in the torso part, then small portion of the main body would be included in the limb's motion and cause false training. Second, in order to classify human motion by local motion features, the system would work well only if videos with reasonably high resolutions are provided for training and testing. In addition, based on our observation from the adaboost learning results, the clay representation of trajectories (long-term feature) usually provides higher weighting (higher alpha values) than accumulated moving edge patches (short-term feature), because the former provides richer information. But, in general the number of long-term features is usually less than the short-term features. Due to this fact, the developed system can automatically adjust the weightings according to the number of each type of features.

4.6 Concluding Remarks

In this chapter, a local feature-based human motion analysis framework was proposed. In our approach, a human motion is described by a spatial structural feature set. The features are directly extracted from the moving regions, and thus the background modeling process usually included in a conventional system is not required. Two types of local feature-based representations for human motions were proposed, they are: 1) a clay representation of a trajectory which describes the long-term movement trend of the motion; 2) moving edge patches derived by multi-scale MHIs which stores short-term variation and shape information. All the extracted features are selected by an adaboost approach to achieve the best discrimination capability form the training data. Finally, the trained detector is used to recognize an unknown human motion. The experiments showed that our approach is robust even on lack-of-information and noisy situations. Therefore, our approach provides a suitable alternative way to recognize a human motion in real-world applications.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this dissertation, we have presented human-based video processing and its application to surveillance. First, an off-line processing to segment the scenes in a video was discussed in Chapter 2. Second, we proposed an PTZ camera network reconfiguration approach for a real-time environment in Chapter 3. Finally, a local feature-based human motion analysis framework was described in Chapter 4.

In Chapter 2, we have proposed a mosaic-based algorithm for extracting scene structures from digital movies. Our approach is based on the idea that shots belonging to one particular scene often have similar backgrounds. Using a mosaic technique, the background of each video frame can be recovered. The color feature and texture feature of each background image are integrated to compute shot similarities. Based on the movie making model, our algorithm is able to group correlated shots into one scene. The computation is costly, but the spatiotemporal information of videos is fully exploited to achieve scene extraction. Experimental results show that the proposed approach works reasonably well in detecting most of the scene boundaries. Compared with some existing techniques [10, 14, 15], our approach is promising. Our approach can be applied directly to organize videos and can be utilized to provide browsing/retrieval facilities to the users. As scene is a subject concept to reflect human perception, our future work will focus on investigating an adaptive technique to perform user-oriented scene extraction. The proposed shot similarity measure does not use motion feature to capture temporal variation in a video. Thus, another future research issue will be the integration of motion information into the proposed shot similarity measure for other tasks such as video retrieval.

In Chapter 3, we have considered the process used to assess the quality of a set of FOVs and proposed a non-linear objective function to reduce the number of unattended targets in a surveillance region. When the quality of each FOV is evaluated individually, it is easy to derive the objective functions of different PTZ network problems. The proposed approach provides an optimal solution for the PTZ network coordination problem. We have also shown that the non-linear optimization problem can be transformed into a linear production game problem that is guaranteed to yield an optimal solution. The optimal solution of LPG can be computed in polynomial time so our approach is efficient. The branch-and-cut method is adopted to solve the PTZ parameters selection problem. Computer simulations and a real-world experiment were performed to evaluate the proposed method in a multi-target surveillance environment. The results show that the method achieved the highest tracking rate among the compared methods.

In Chapter 4, we proposed a local feature-based human motion analysis framework. In our approach, a human motion is described by a spatial structural feature set. The features are directly extracted from the moving regions, so the background modeling process in a conventional system is not required. A local feature-based representation for a motion is proposed, based on two features: 1) a clay representation of a trajectory describes long-term movement trend; 2) moving edge patches derived by multi-scale MHIs provides short-term variation and shape information. All the extracted features are selected by an adaboost approach to achieve the best discrimination capability form the training data. Finally, the trained detector is used to recognize an unknown human motion. The experiments showed that our approach is robust even on lack-of-information and noisy situations. Therefore, our approach provides a suitable alternative way to recognize a human motion in real-world applications.

5.2 Future Work

In our motion recognition approach, the trained human motions are segmented as atomic motions. However, the videos captured from a surveillance system and the videos obtained from scene segmentation usually contain more than one atomic motions. Therefore, the atomic motion segmentation problem should be handled in the future. Moreover, our local feature-based representation is not a view invariant representation; therefore, the atomic motions with different views should be handled to improve our system usability in the future.

Bibliography

[1] J. Aldrige and C. Gilbert, *Performance Testing CCTV Perimeter Surveillance Systems*, Number 14–95. White Crescent Press, Home Office Police Scientific Development Branch, Woodcock Hill, Sandridge, St Albans, Herfordshire, UK, 1996.

[2] I. Haritaoglu, D. Harwood, and L.S. Davis, "W4: real-time surveillance of people and their activities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809–830, 2000.

[3] O. Javed and M. Shah, "Tracking and object classification for automated surveillance," in *Proceedings of the European Conference on Computer Vision*, 2002, vol. 2423, pp. 343–357.

[4] H. Miyamori and S.I. Iisaku, "Video annotation for content-based retrieval using human behavior analysis and domain knowledge," in *IEEE International Conference on Automatic Face and Gesture Recognition*, 2000, pp. 320–325.

[5] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[6] R. Sharma, I. Pavlovic, and T.S. Huang, "Toward multimodal humanvcomputer interface," *Proceedings of the IEEE*, vol. 86, no. 5, 1998.

[7] G.R. Bradski, “Computer vision face tracking for use in a perceptual user interface,” *Intel Technology Journal*, vol. 1, no. Q2, 1998.

[8] F. Beaver, *Dictionary of Film Terms*, Twayne Publishing, New York, 1994.

[9] I. Koprinska and S. Carrato, “Temporal video segmentation: a survey,” *Signal Processing: Image Communication*, vol. 16, pp. 477–500, 2001.

[10] M. Yeung, B.L. Yeo, and B. Liu, “Segmentation of video by clustering and graph analysis,” *Computer Vision and Image Understanding*, vol. 71, no. 1, pp. 97–109, 1998.

[11] J.R. Kender and B.L. Yeo, “Video scene segmentation via continuous video coherence,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1998, pp. 367–373.

[12] A. Hanjalic, R.L. Lagendijk, and J. Biemond, “Automated high-level movie segmentation for advanced video-retrieval systems,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 4, pp. 580–588, 1999.

[13] Y. Rui, T.S. Huang, and S. Mehrotra, “Constructing table-of-content for video,” *Multimedia System*, vol. 7, no. 5, pp. 359–368, 1999.

[14] T. Lin, H.J. Zhang, and Q.Y. Shi, “Video content representation for shot retrieval and scene extraction,” *International Journal of Image and Graphics*, vol. 1, no. 3, pp. 507–526, 2001.

[15] C.W. Ngo, T.C. Pong, H.J. Zhang, and R.T. Chin, “Motion-based video representation for scene change detection,” *International Journal of Computer Vision*, vol. 50, no. 2, pp. 127–142, 2002.

[16] B.T. Truong, S. Venkatesh, and C. Dorai, "Scene extraction in motion picture," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 1, pp. 5–15, 2003.

[17] W. Tavanapong and J. Zhou, "Shot clustering techniques for story browsing," *IEEE Transactions on Multimedia*, vol. 6, no. 4, pp. 517–527, 2004.

[18] Z. Rasheed and M. Shah, "Detection and representation of scenes in videos," *IEEE Transactions on Multimedia*, vol. 7, no. 6, pp. 1097–1105, 2005.

[19] Y. Zhai and M. Shah, "A general framework for temporal video scene segmentation," in *Proceedings of the International Conference on Computer Vision*, 2005, pp. 1111–1116.

[20] J. Juang, Z. Liu, and Y. Wang, "Integration of audio and visual information for content-based video segmentation," in *Proceedings of the International Conference of Image Processing*, 1998, pp. 526–530.

[21] H. Sundaram and S.F. Chang, "Video scene segmentation using video and audio features," 2000, pp. 1145–1148.

[22] A. Merlino, D. Morey, and M.T. Maybury, "Broadcast news navigation using story segmentation," in *ACM International Conference on Multimedia*, 1997, pp. 381–391.

[23] Y. Ariki, M. Kumano, and K. Tsukada, "Highlight scene extraction in real time from baseball live video," in *ACM InternationalWorkshop on Multimedia Information Retrieval*, 2003, pp. 209–214.

[24] L. Xie, P. Xu, S.F. Chang, A. Dirakaran, and H. Sun, "Structure analysis of soccer video with domain knowledge and hidden markov models," *Pattern Recognition Letters*, vol. 25, no. 7, pp. 767–775, 2004.

[25] Y. Zhai, A. Yilmaz, and M. Shah, "Story segmentation in news using visual and text cues," in *International Conference on Image and Video Retrieval*, 2005, pp. 92–102.

[26] M. Irani, P. Anandan, J. Bergen, R. Kumar, and S. Hsu, "Efficient representations of video sequences and their applications," *Signal Processing: Image Communication*, vol. 8, no. 4, pp. 327–351, 1996.

[27] M. Irani and P. Anandan, "Video indexing based on mosaic representations," *Proceedings of the IEEE*, vol. 86, no. 5, pp. 905–921, 1998.

[28] A. Aner and J. Kender, "Video summaries and cross-referencing through mosaic-based representation," *Computer Vision and Image Understanding*, vol. 95, no. 2, pp. 201–237, 2004.

[29] L.H. Chen, C.W. Su, H.Y. Liao, and C.C. Shih, "On the preview of digital movies," *Journal of Visual Communication and Image Representation*, vol. 14, no. 3, pp. 357–367, 2003.

[30] L.H. Chen, Y.C. Lai, C.W. Su, and H.Y. Liao, "Extraction of video object with complex motion," *Pattern Recognition Letters*, vol. 25, no. 11, pp. 1285–1291, 2004.

[31] D.L. Gall, "Mpeg: a video compression standard for multimedia applications," *Communications of the ACM*, vol. 34, no. 4, pp. 46–58, 1991.

[32] P.J. Rousseeuw and A.M. Leroy, *Robust Regression & Outliers Detection*, Wiley, New York, 1987.

[33] X. Wan and C.C. Jay Kuo, "A new approach to image retrieval with hierarchical color clustering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 5, pp. 628–643, 1998.

[34] M.J. Swain and D.H. Ballard, “Color indexing,” *International Journal of Computer Vision*, vol. 7, no. 11, pp. 11–32, 1991.

[35] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice- Hall, New Jersey, USA, 1999.

[36] B.B. Boycott, *Color Vision*, Cambridge University Press, Cambridge, UK, 2001.

[37] L. Giannetti, *Understanding Movie*, Prentice-Hall, Englewood Chiffs, NJ, 1987.

[38] G. Davenport, T.A. Smoth, and N. Princever, “Cinematic primitives for multimedia,” *IEEE Computer Graphics and Applications*, vol. 11, no. 4, pp. 67–74, 1991.

[39] T.S. Chua and L.Q. Ruan, “A video retrieval and sequencing system,” *ACM Transactions on Information Systems*, vol. 13, no. 4, pp. 373–407, 1995.

[40] S.N. Lim, L.S. Davis, and A. Elgammal, “A scalable image-based multi-camera visual surveillance system,” in *Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance*, 2003, pp. 205–212.

[41] N. Ukita and T. Matsuyama, “Real-time cooperative multi-target tracking by communicating active vision agents,” *Computer Vision and Image Understanding*, vol. 97, no. 2, pp. 137–179, 2005.

[42] F.Z. Qureshi and D. Terzopoulos, “Surveillance in virtual reality: System design and multi-camera control,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[43] B. Song, C. Soto, A.K. Roy-Chowdhury, and J.A. Farrell, “Decentralized camera network control using game theory,” in *Proceedings of the ACM/IEEE International Conference on Distributed Smart Cameras*, 2008, pp. 1–8.

[44] A.D. Bagdanov, A.D. Bimbo, and W. Nunziati, "Improving evidential quality of surveillance imagery through active face tracking," in *Proceedings of the International Conference on Pattern Recognition*, 2006, vol. 3, pp. 1200–1203.

[45] N. Bellotto, E. Sommerlade, B. Benfold, C. Bibby, I. Reid, D. Roth, L. V. Gool, C. Fernandez, and J. Gonzalez, "A distributed camera system for multi-resolution surveillance," in *ACM/IEEE International Conference on Distributed Smart Cameras*, 2009, pp. 1–8.

[46] C. Piciarelli, C. Micheloni, and G.L. Foresti, "PTZ camera network reconfiguration," in *ACM/IEEE International Conference on Distributed Smart Cameras*, 2009, pp. 1–7.

[47] F.Z. Qureshi and D. Terzopoulos, "Smart camera networks in virtual reality," in *Proceedings of the IEEE*, 2008, vol. 96, pp. 1640–1656.

[48] E. Rasmusen, *Games and Information - An Introduction to Game Theory*, Wiley-blackwell, 1991.

[49] C. Soto, B. Song, and A.K. Roy-Chowdhury, "Distributed multi-target tracking in a self-configuring camera network," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1486–1493.

[50] B. Song, A. Kamal, C. Soto, J. Farrell C. Ding, and A. Roy-Chowdhury, "Tracking and activity recognition through consensus in distribution camera network," in *IEEE Transactions on Image Processing*, 2010, vol. 19, pp. 2564–2579.

[51] R.J. Vanderbei, *Linear Programming: Foundations and Extensions*, International Series in Operations Research & Management Science Springer-Verlag, 2008.

[52] S.P. Bradley, A.C. Hax, and T.L. Magnanti, *Applied Mathematical Programming*, Addison-Wesley, 1977.

[53] R. Lackore, *Firefighter Anthropometric Data White Paper*, FAMA Technical Committee Chassis Subcommittee, 2007.

[54] G. Owen, “On the core of linear production games,” in *Mathematical Programming*, 1975, vol. 9, pp. 358–370.

[55] J. Derks and J. Kuipers, “On the core of routing games,” *International Journal of Game Theory*, vol. 26, no. 2, pp. 193–205, 1997.

[56] J.M. Bilbao, *Cooperative Games on Combinatorial Structures*, Kluwer Academic Publishers, 2000.

[57] J. Mitchell, *Branch-and-cut algorithms for combinatorial optimization problems in Handbook of Applied Optimization*, Oxford University Press, 2002.

[58] S. Calderara, A. Prati, and R. Cucchiara, “Hecol: Homography and epipolar-based consistent labeling for outdoor park surveillance,” *Computer Vision and Image Understanding*, vol. 111, no. 1, pp. 21–42, 2008.

[59] S.N. Sinha and M. Pollefeys, “Towards calibrating a pan-tilt-zoom camera network,” in *The fifth Workshop on Omnidirectional Vision, Camera Networks and Non-classical cameras*, 2004.

[60] OpenCV library, http://opencv.willowgarage.com/wiki/.

[61] M. Brown and D. Lowe, “Automatic panoramic image stitching using invariant features,” *International Journal of Computer Vision*, vol. 74, no. 1, pp. 59–73, 2007.

[62] Y. Yoo and T.S. Park, “A moving object detection algorithm for smart cameras,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2008, pp. 1–8.

[63] I. Haritaoglu, D. Harwood, and L. Davis, "Ghost: A human body part labeling system using silhouettes," in *Proceedings of the International Conference on Pattern Recognition*, 1998, pp. 77–82.

[64] A.F. Bobick and J.W. Davis, "The recognition of human movement using temporal templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 3, pp. 257–267, 2001.

[65] J. Min and R. Kasturi, "Extraction and temporal segmentation of multiple motion trajectories in human motion," in *Computer Vision and Pattern Recognition Workshop*, 2004, p. 118.

[66] J.C. Niebles and F.F. Li, "A hierarchical model of shape and appearance for human action classification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[67] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *15th international conference on Multimedia*, 2007, pp. 357–360.

[68] D. Weinland and E. Boyer, "Action recognition using exemplar-based embedding," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–7.

[69] Y. Ke, R. Sukthankar, and M. Hebert, "Efficient visual event detection using volumetric features," in *Proceedings of the International Conference on Computer Vision*, 2005, pp. 166–173.

[70] A. Yilma and M. Shah, "Recognizing human actions in videos acquired by uncalibrated moving cameras," in *Proceedings of the International Conference on Computer Vision*, 2005, pp. 150–157.

[71] B. Leibe, A. Leonardis, and B. Schiele, "Combined object categorization and segmentation with an implicit shape model," in *ECCV'04 Workshop on Statistical Learning in Computer Vision*, 2004, pp. 17–32.

[72] A. Opelt, A. Pinz, and A. Zisserman, "Incremental learning of object detectors using a visual shape alphabet," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006, pp. 3–10.

[73] Y.C. Lai and H.Y. Mark Liao, "Human motion recognition using clay representation of trajectories," in *Intelligent Information Hiding and Multimedia Signal Processing*, 2006, pp. 335–338.

[74] H.G. Barrow, J.M. Tanenbaum, R.C. Botles, and H.C. Wolf, "Parametric correspondence and chamfer matching: Two new techniques for image matching," in *Proceedings of 5th International Joint Conference on Artificial Intelligence*, 1977, pp. 659–663.

[75] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[76] G. Dewaele and M.P. Cani, "Interactive global and local deformations for virtual clay," in *the 11th Pacific Conference on Computer Graphics and Applications*, 2003.

[77] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001, pp. 511–518.

[78] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, *Classification and Regression Trees*, Wadsworth International Group, Belmont, California, 1984.

[79] M.S. Drew, Z.N. Li, and Z. Tauber, "Illumination color covariant locale-based visual object retrieval," *Pattern Recognition*, vol. 35, no. 8, pp. 1687–1704, 2002.

[80] Graphics and Media Lab Projects, http://research.graphicon.ru/machine-learning/gml-adaboost-matlab-toolbox.html.

[81] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2247–2253, 2007.