

**Human Detection and Pose  
Estimation for Motion Picture  
Logging and Visualisation**

Ziran Wu

PhD

University of York  
Department of Electronics

September 2012

# Abstract

---

This thesis contributes to the research area of Computer-Vision-based human motion analysis, investigates techniques associated in this area, and proposes a human motion analysis system which parses images or videos (image sequences) to estimate human poses. A human motion analysis system that combines a novel colour-to-greyscale converter, an optimised Histogram of Orientated Gradients (HOG) human body detector, and an improved Generalised Distance Transform and Orientation Maps (GDT&OM) pose estimator, is built to execute key-frame extraction.

The novel colour-to-greyscale conversion method that converts RGB images to chroma-edge-enhanced greyscale images by employing density-based colour clustering and spring-system-based multidimensional scaling, is proved to be superior compared with other methods such as Color2Grey and Ren's method. The weakness of the novel method is that it is still parameter dependent and does not perform well for some images.

We make improvement on Histogram of Orientated Gradients by employing a modified training scheme and using pre-processed data, and the performance is improved by achieving similar true detection rate but much lower false detection rate, compared with the original HOG scheme.

We discuss the GDT&OM method and develop the original GDT&OM human detector to a human pose estimator using results of human detection. Meanwhile we also investigate a pose estimation method based on locations and orientations of human body parts under the assumption of body parts can be accurately located.

Then we integrate all methods to build a key-frame extraction system which is more

intelligent than conventional approaches as it is designed to select frames representing content of videos. We finally apply our methods to build a video logging system, which automatically records actions of gymnastic videos according to the actions displayed. Both systems perform well for a small set of motion categories. However they are object-dependent systems that need users to manually select target objects, and the performance is limited by the human body detector and pose estimator.

# List of Contents

---

Abstract .....	i
List of Contents .....	iii
List of Figures .....	vii
List of Tables .....	xi
Acknowledgements .....	xii
Author's declaration .....	xiii
Chapter 1 Introduction .....	1
1.1 Background .....	1
1.2 Overview of Human Motion Analysis .....	3
1.3 Discussion on 2D and 3D.....	4
1.4 Terms and Definitions of the Thesis.....	6
1.5 Motivations and Presuppositions of the Thesis.....	7
1.6 Overview of the System Proposed in the Thesis.....	8
1.7 Structure of the Thesis.....	10
Chapter 2 Prior Work in Human Motion Analysis .....	12
2.1 Human detection .....	15
2.1.1 Human detection method for moving human bodies	15
2.1.2 Signal-processing-based human detection	17
2.1.3 Machine-learning-based human detection	20
2.2 Human tracking .....	23

2.2.1 Model-based tracking	23
2.2.2 Region-based tracking	24
2.2.3 Feature-based tracking	25
2.3 Human structure analysis .....	26
2.4 Human action recognition .....	29
2.4.1 Pose estimation	29
2.4.2 State-space based approaches	30
2.4.3 Mixed representation approaches	34
2.5 Colour-to-greyscale mapping .....	38
2.6 Human motion analysis systems .....	40
2.7 Prior work and the research of the thesis .....	41
Chapter 3 Pre-processing .....	43
3.1 The weakness of previous methods.....	44
3.2 Colour clustering .....	45
3.2.1 The method based on complete-linkage	48
3.2.2 The method based on $k$ -means clustering	49
3.2.3 The method employing spatial-density estimation	51
3.2.4 The method based on smoothing and thresholding in 3D histogram scatter	54
3.2.5 The method based on smoothing and watershed algorithm in 3D histogram scatter	57
3.2.6 The result comparison of three clustering methods	60
3.3 Multidimensional scaling of cluster prototypes .....	62
3.4 Mapping of all colours to greyscale .....	66
3.5 Comparison against other methods .....	68
3.5.1 Qualitative performance	68
3.5.2 Quantitative performance	73
3.6 Conclusions .....	79
Chapter 4 Human Detection.....	80
4.1 Histogram of Orientated Gradients .....	81

---

4.1.1	Extracting HOG features	81
4.1.2	HOG training phase	84
4.1.3	Post-processing by multi-scale object localisation	85
4.1.4	Experimental results	87
4.2	Modifications on HOG training .....	89
4.2.1	Extra training data	89
4.2.2	Experiments and results	91
4.3	A Comparison: Haar-like features and AdaBoost .....	93
4.3.1	Haar-like features	94
4.3.2	AdaBoost classifier	94
4.3.3	Cascade detector	95
4.3.4	Experiments and results	95
4.4	Using Pre-processed images.....	97
4.4.1	HOG detection using greyscale images for both training and testing	98
4.4.2	HOG detection using greyscale images to refine colour detections	100
4.5	Conclusions .....	104
Chapter 5 Pose Estimation .....		105
5.1	Theory .....	107
5.1.1	Generalized Distance Transform	107
5.1.2	Orientation Maps	108
5.1.3	Template Matching	108
5.1.4	Templates	109
5.2	Detection Experiments for Frame Sequences .....	111
5.2.1	Background Cancellation	111
5.2.2	Template Modification	114
5.3	Template Matching for Pose Estimation .....	117
5.4	Estimating Poses by Locations of Body Parts.....	120
5.4.1	Preparing experimental data	121
5.4.2	Model training and testing	121
5.5	Conclusions and Discussion.....	123

Chapter 6 Practical Applications .....	125
6.1 Key-frame Extraction System .....	125
6.1.1 HOG human detection .....	127
6.1.2 GDT&OM template matching .....	129
6.1.3 Perceived motion energy .....	131
6.1.4 Experimental results .....	133
6.2 Video Logging.....	137
6.2.1 HOG human detection .....	138
6.2.2 Template matching .....	140
6.2.3 Producing video logs .....	142
6.3 Conclusions .....	149
 Chapter 7 Conclusions and Further Work .....	 151
7.1 Conclusions .....	151
7.2 Further Work .....	153
7.2.1 Stage one: improvement in HOG human detection .....	153
7.2.2 Stage two: improvement in the robustness .....	154
7.2.3 Stage three: development in high-level behaviour recognition .....	155
 Appendix I.....	 157
Shell script for HOG training .....	157
Shell script for HOG testing .....	159
 Appendix II .....	 161
Cascade configuration .....	161
 Bibliography.....	 162

# List of Figures

---

Figure 1.1 System flowchart .....	9
Figure 1.2 Structure of the main body of the thesis .....	10
Figure 2.1 Overview of human motion analysis .....	14
Figure 2.2: An example of contour templates .....	17
Figure 2.3: Hierarchical part-template matching .....	18
Figure 2.4: Gavrilu's method (partial view).....	19
Figure 2.5 Human structure models .....	27
Figure 2.6 Procedure of Ramanan's iterative parsing method .....	28
Figure 2.7 Represent poses by CHORs.....	30
Figure 2.8: A Hidden Markov Model.....	32
Figure 2.9: The nodes of a Bayesian Network.....	33
Figure 2.10 Examples of mixed features.....	35
Figure 2.11 An colour-to-greyscale example .....	38
Figure 3.1 The problem of colour bars.....	45
Figure 3.2 Clustering and remapping colours for an image.....	48
Figure 3.3 Colour clustering using complete-linkage .....	49



---

Figure 3.4 Colour clustering using $k$ -means and $k$ -means++ clustering.....	51
Figure 3.5 A 2D view of SDE .....	52
Figure 3.6 An SDE result .....	54
Figure 3.7 The process of colour clustering using smoothing and filtering.....	57
Figure 3.8 Comparison of the thresholding mechanism and the watershed algorithm .....	58
Figure 3.9 The result of using the watershed algorithm.....	60
Figure 3.10 Re-mapped by means of colour clusters.....	61
Figure 3.11 An example of spring graphs .....	64
Figure 3.12 Proportion of runs finding minimum energy solution .....	66
Figure 3.13 Flowchart of the entire procedure.....	68
Figure 3.14 Comparison between our method and others.....	71
Figure 3.15 The comparison between our method and Ren's method.....	77
Figure 4.1 The flowchart of HOG training and detection.....	81
Figure 4.2 The procedure of HOG extraction .....	82
Figure 4.3 Three variants of HOG descriptors.....	83
Figure 4.4 The flowchart of HOG training .....	85
Figure 4.5 Merging HOG detections by multi-scale object localisation.....	86
Figure 4.6 Results of the original HOG detector .....	88
Figure 4.7 Examples of the original HOG detections with overlapping detection windows on body parts .....	90

---

Figure 4.8 Three mechanisms of cropping negative images from human images.....	91
Figure 4.9 Results of the HOG detector using extra training data.....	93
Figure 4.10 Types of Haar-like features.....	94
Figure 4.11 The structure of the cascade detector.....	95
Figure 4.12 Results of the HOG detector using greyscale images.....	99
Figure 4.13 Flowchart of the “iteratively detecting” scheme .....	101
Figure 4.14 Results of the HOG detector using greyscale images and “iteratively detecting” scheme .....	103
Figure 5.1 The diagram of GDT&OM.....	106
Figure 5.2 Images of producing a GDT image .....	108
Figure 5.3 Examples of templates.....	110
Figure 5.4 Orientations of templates pixels .....	111
Figure 5.5 The flowchart of template matching.....	112
Figure 5.6 GDT&OM result of each step.....	113
Figure 5.7 Modified templates .....	114
Figure 5.8 Template set for GDT&OM template matching: .....	118
Figure 5.9 An example of marking body part locations.....	121
Figure 6.1 The diagram of the key-frame extraction system .....	127
Figure 6.2 Examples of true and false detections .....	129
Figure 6.3 Template set for key frame selection .....	130
Figure 6.4 Triangle mechanism.....	133

Figure 6.5 Comparison of key-frames extracted by PME and template matching .. 134

Figure 6.6 Some “bad” key-frames ..... 136

Figure 6.7 Examples of gymnastic images ..... 138

Figure 6.8 Detection results ..... 140

Figure 6.9 Templates used for logging gymnastic videos ..... 142

Figure 6.10 A visualised rings video log ..... 145

Figure 6.11 A visualised balance beam video log ..... 146

# List of Tables

---

Table 3.1 The comparison of results .....	77
Table 3.2 Evaluation results by using the thinned ground truth.....	78
Table 4.1 Detection results of the original HOG approach .....	89
Table 4.2 Results of the HOG detector using extra training data.....	92
Table 4.3 Results of using Haar-like features and AdaBoost.....	97
Table 5.1 Matching results of using modified and non-modified templates.....	116
Table 5.2 The result of action recognition with different $n$ .....	118
Table 5.3 True numbers of classifications for each action type ( $n=5$ ) .....	119
Table 5.4 A comparison with other methods .....	120
Table 5.5 The result of body-part based pose estimation.....	122
Table 6.1 The HOG result for the Weizmann dataset.....	128
Table 6.2 Comparison of “good” key-frames achieved. ....	135
Table 6.3 Performance of the video logging system .....	147
Table 6.4 Incorrect classification results .....	148

# Acknowledgements

---

First and foremost, I would like to show my deepest gratitude to my parents and grandparents, who provided me good education when I was young, offered me financial support and cared my daily life all the time during my PhD course.

Secondly I would like to give my sincere appreciation to my supervisor, Prof. John Robinson, and thesis advisor, Dr. John Dawson, both of whom have provided me with valuable guidance and advice in every stage of my research and thesis. Without their enlightening instruction, impressive kindness and patience, I could not have completed my thesis. Their keen and vigorous academic observation also enlightens me in my future study and work.

I shall extend my thanks to Dr. Matthew Day, Mr. Edward Munday and Dr. Andrew Pomfret in Room T224, for all their kindness and help. I would also like to thank all other people, especially the PhD students in the Department of Electronics, who have helped me to develop my academic knowledge and competence. My sincere appreciation also goes to then teachers and classmates of my BEng course in Zhejiang University and MSc course in the University of York. Last but not least, I'd like to thank all my friends, for their support and encouragement.

# **Author's declaration**

---

All of the material contained in this thesis has been presented before.

# Chapter 1

## Introduction

---

### 1.1 Background

Human motion analysis in Computer Vision studies and develops methods and applications in which images are processed to produce information based on the apparent human motion in the images and videos. Although the concept of human motion analysis in Computer Vision is presented by Aggarwal et al. in 1999 [1], associated research began in 1970s [2] [3]. This interest is motivated by a wide spectrum of applications, such as athletic performance analysis, surveillance, human machine interfaces, content-based image/video storage and retrieval, and video conferencing.

In recent years, research on human motion analysis has attracted more and more attention and develops fast. A famous conference, IEEE International Conference on Automatic Face and Gesture Recognition (FG), was human motion analysis as an important component and its range of topics tracks the development of the field. In FG96 (the 2nd, in Killington, Vermont, USA), papers on body tracking, gesture recognition and behaviour analysis are proposed. At that time, the body tracking method presented had strict limitations: target person and lighting should be static, and only one person in the scene [4]. Meanwhile human behaviour and gesture recognition was based only on locations and movement of hands [5] [6]. These methods are relatively simple and impractical. In FG04 (the 6th, in Seoul, Korea)

techniques were significantly proved. More complicated models and advanced devices were used for pose and gesture recognition [7] [8] [9]. The latest conference was held in 2011 (the 9th in Santa Barbara, USA), in which a new and more challenging topic was interested: human communicative behaviour analysis [10] [11]. The 10th, FG2013 in Shanghai, China, will include regular topics such as gesture, motion analysis as well as a novel one – psychological and behavioural analysis. According to the development of FG, it can be summarised that research on human motion has developed through the following stages:

- 1) Conventional analysis on a single body part and simple behaviours: since at that time the processing speed of computers was not sufficient, researchers used simple models and features under ideal assumptions. Therefore complex behaviours could not be studied.
- 2) Complex body models and behaviours: with developed techniques, using more complicated models to study advanced body motion became feasible.
- 3) Higher level of human behaviour (group interaction) and interdisciplinary research (psychology): now computers are required to understand not only content a person performs, but the thoughts reflected by his behaviours, and interaction between people as well. Methods from different areas are combined. For example, to understand the content of a group meeting, motion and face analysis, sound and voice analysis, as well as psychology methods can be integrated together as a system.

Techniques for human motion analysis developed quickly due to many successful analysis methods and rapid development of computer hardware. Developments in statistics, image processing and machine learning have led to advanced methods that can achieve higher accuracy and faster processing speed. Meanwhile developments in computer hardware makes complicated computation affordable, so fast or even real-time process for human motion analysis can be achieved. Generally speaking, human motion analysis systems, such as Microsoft Kinect, now are leaving laboratories and becoming practical in daily life.



Fundamentally, human motion analysis systems can be classified into two categories based on types of data used: 2D approaches and 3D approaches. 2D approaches process images without depth information of objects, while 3D approaches have depth information available and use it. Generally speaking, 2D data requires less storage space but raises processing difficulty. Considering that most images and videos in daily life are 2D, it is an important issue to develop 2D approaches. Hence, this thesis focuses on investigating human motion analysis methods and systems using 2D image and videos.

## **1.2 Overview of Human Motion Analysis**

Human motion is a concept that includes several components of object information in images and videos: low level components such as locations and movement of entire human bodies and body parts, and high level components such as human actions, which are reflected by low level components. Fundamentally, human motion analysis consists of a set of topics: human detection, human tracking, human pose estimation and human action recognition. Although in practice different systems employ different processing strategies, usually it is a consecutive procedure of these four topics: human bodies need to be detected first, then tracking and body structure analysis can be executed, and finally actions recognised.

Human detection in Computer Vision indicates a category of methods that estimate locations of human bodies in images or video frames. Objects to be located can be full or part of (usually upper half) human bodies. The most common way to do this is to use a window to scan of each image and find regions similar to a trained model. Methods of human detection can be divided into two categories: Signal-processing-based methods, which classify objects by matching specific spatial information, and machine-learning-based methods, which statistically learn models and classify objects. Human tracking methods are applied to estimate movement of human bodies in frame sequences. Movement of human bodies can be achieved by analysing

differences between frames, or finding out connections between located human bodies in consecutive frames. Considering that in frame sequences there are possibly multiple persons and stationary bodies, using body locations can achieve more accurate results, at the expense of more complex computation.

Human pose estimation classifies human bodies to pose categories. Generally, it can be considered as a problem of multi-class classification that categorises samples to several signal-processing-based or machine-learning-based models.

Human action recognition classifies input images or videos by human actions represented in them. For videos, the relationship of human body information (i.e. locations, shapes, movement etc.) in different frames is analysed to determine action categories. This type of methods can analyse complex actions. For images standing alone, human poses are usually used to estimate actions. Research on human motion analysis is based on a lot of prior art. The associated research areas of human motion analysis mainly include image processing and machine learning. Image processing methods convert input image data to appropriate features, signals or information. For example, gradient and edge information is important in human motion analysis, and these are extracted and analysed by image processing methods. Machine learning methods are used to build models according to training data, and classify or estimate testing data by applying these models. Algorithms such as Support Vector Machine (SVM) [12], boosting [13] and Random Forest [14] are common and popular methods applied in this area.

### **1.3 Discussion on 2D and 3D**

It is emphasised that we use 2D images and videos as input, since 2D approaches are essentially different from 3D ones. Research on 3D approaches is also a major topic in motion analysis. There are a number of examples of 3D motion analysis.

An important issue of 3D motion analysis is the way of obtaining 3D data. Usually

3D data is captured by some specific devices. The paper [7] suggests a method that uses multiple cameras to take images of a person wearing a multiple-coloured suit, to reconstruct the 3D view of the person and estimate his posture. A research group in Carnegie Mellon University builds a 3D captured human motion dataset (CMU multi-Modal Activity Database, CMU-MMAC) [15] that uses a motion capture device set, which includes 12 infrared cameras, wired and wireless internal measurement units, and wearable devices such as BodyMedia and eWatches. This dataset contains videos of human bodies which perform a number of actions along with their 3D representation of each body parts.

A recent successful example is Microsoft Kinect for Xbox360 game consoles, which provides reliable real-time performance being built in an affordable, light-weighted hardware device. Kinect uses a 3D sensor consisting of an infra-red laser projector along with a CMOS sensor to capture depth images according to intensity of reflected infra-red laser. Images usually indicate depth of human bodies, since this device is designed to capture motions of people playing games. The key technique in the Kinect system is joint position regression [16]. After getting the depth of the human bodies, the system attempts to obtain their structure information. It decomposes the bodies to more than ten joints by a voting mechanism. A regression forest which consists of a set of decision trees [17], takes the responsibility of the voting.

The three approaches above exemplify the difference between 2D and 3D approaches is the way of capturing data. Kinect uses a pair of infrared devices: a projector and a sensor, to obtain depth images. Alternative methods such as using multiple fixed cameras and capturing suits, are also hardware constrained. In other words, using a single RGB camera, which is the most common situation in daily life, cannot produce 3D data. Meanwhile, Kinect's infra-red projectors have distance and environment constrains, e.g. surveillance, group sports games, etc. Hence taking depth pictures is inconvenient in some applications. In addition, there is a huge amount of 2D data which has already produced and need to be processed. Therefore, although the technology implemented on Kinect is very successful, the application is constrained. However, although 3D technologies are developing fast, 2D images and

videos are still in the dominant position at present. That is the reason why human motion analysis methods for regular 2D images and videos are so important and irreplaceable. Of course, 2D and 3D approaches are tightly associated, since many theories and algorithms can be applied by both 2D and 3D methods.

## **1.4 Terms and Definitions of the Thesis**

In this thesis, we discuss concepts of motion analysis and pattern recognition, so a number of terms need to be defined and explained to avoid confusion. The following keywords are used in the thesis:

**Action and Motion:** We define the term “action” as basic, simple behaviours of humans, such as walking, running, jumping, etc., so action recognition has the same meaning of basic behaviour understanding. For complex behaviours we do not use the word “action”. Generally, the term “motion” means the movement of an object. However in this thesis, human motion stands for not only the concept of movement of human bodies, but human actions or behaviours. Therefore human motion analysis studies both movements and actions in images and videos.

**Detector and classifier:** in the thesis, a detector locates target objects, and a classifier classifies objects into two or more categories. In our work, human detectors are essentially object classifiers that categorise image regions into human and non-human categories. The terms detect/classify and detection/classification can be distinguished in the same way as above.

**Stationary image and frame sequence:** in the thesis, a stationary image means an image that stands alone and does not have any relative information from other images. A frame sequence is a pipeline of images that are extracted from a video, and each frame can be treated as an independent image. So here the terms “frame sequence” and “video” have the same meaning.

Shade and shadow: shade and shadow are two important concepts in human motion analysis. Shade is defined as dark regions on objects due to lighting condition, while shadow is dark regions having obvious contours caused by objects sheltering light. Both shade and shadow usually make motion analysis more difficult by changing gradient information, e.g. weakening edges between objects and adding luminance edges.

## **1.5 Motivations and Presuppositions of the Thesis**

The motivations of the thesis are to advance the area of human motion analysis, to investigate methods that improve performance of human detection and human pose estimation, to design a system that consists of a pipeline of motion analysis methods, and to present and discuss experimental results of the system.

The presuppositions which are assumed in our work throughout the thesis are as follows:

- 1) Images and videos are captured by conventional 2D RGB or greyscale cameras.
- 2) The resolution of data images must be larger than the minimum detection window, e.g.  $64 \times 128$  for HOG detectors. If a human body is much smaller than the minimum window size, it will be missed in the detection phase. There is no maximum resolution for data images, although high resolution images will require long processing time.
- 3) Both image and video data used is uncompressed. Compressed data need to be decompressed to be applied to the system. Hence compressed images are converted to raw data and compressed videos are converted to frame sequences. Lossless compression has no impact on the system. However, lossy compression affects gradient and edge information of images, and causes difference in results.
- 4) We have not considered the noise issue at present. The system requires

relatively low-noise data. The performance of the system in noisy situations is to be tested in the further plan, and improvement for noisy data will be made.

- 5) Images and videos used are all digital samples with a quantisation of  $2^8$ , which means each pixel has a value range of  $[0, 255]$ .
- 6) Video cameras are fixed, which means that frame differences caused by movement of cameras can be ignored.
- 7) We focus on processing videos containing one entire human body. Although multiple bodies can be processed in our system, we only discuss the one-body situation in the thesis. Meanwhile, a part of a body is treated as a non-object.
- 8) Action types in an action set which are used to categorise human bodies are significantly different between each other, to avoid interference.

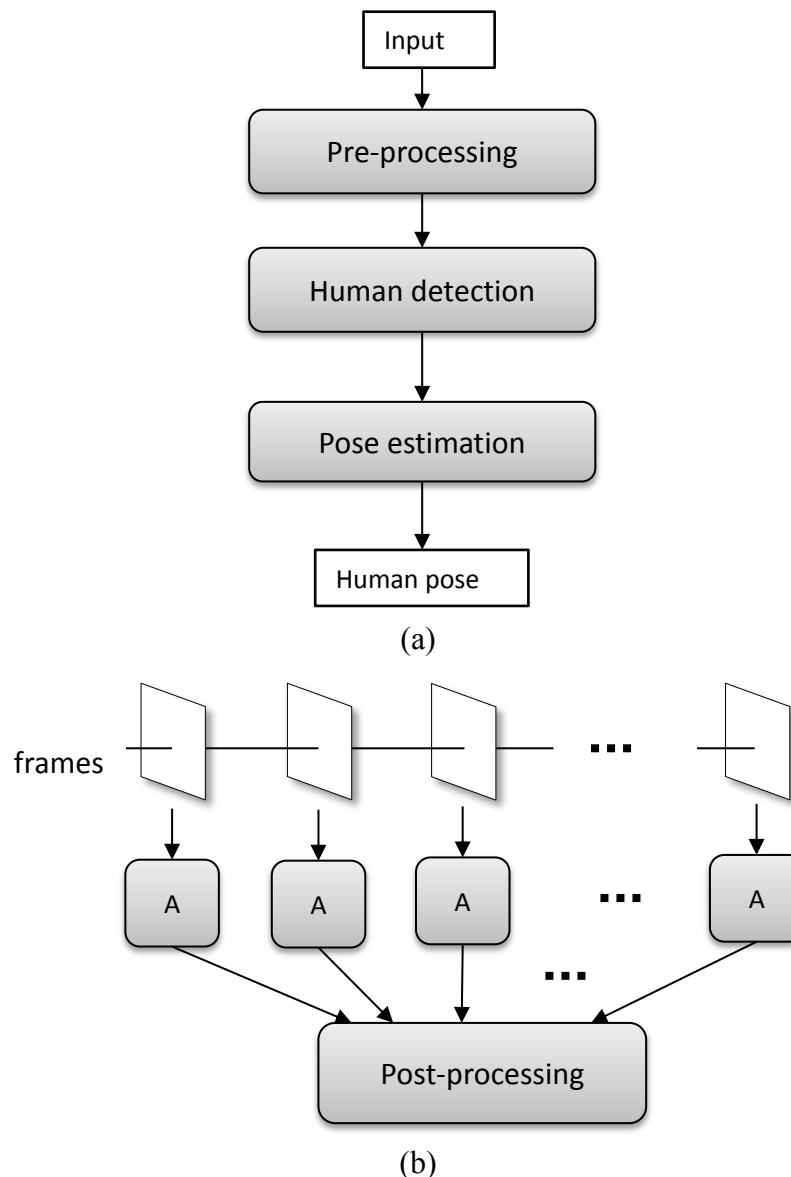
This thesis makes four main contributions:

- 1) A novel colour-to-greyscale conversion method using density-based colour clustering and spring systems is proposed.
- 2) The HOG human detection method [18] is improved by applying pre-processed data and new training strategy.
- 3) The GDT&OM template matching method [19] is improved and modified for pose estimation.
- 4) A content-based key-frame extraction [20] [21] system is proposed.
- 5) A video logging system [22] that automatically recording human actions is presented.

## 1.6 Overview of the System Proposed in the Thesis

This thesis presents a 2D human motion analysis system which is used to detect human bodies and estimate human poses in still images, as shown in Figure 1.1.

The system can be applied to frame sequences when each frame is processed separately. Besides, we have proposed a pre-processing method to emphasis colour edges for colour images and enhance the detection performance. Figure 1.1a indicates the structure of the system for a single image in our work. If the input is a video, we design to process and estimate the pose in each frame separately, and then post-process the estimated results to achieve a final result, which is shown in Figure 1.1b.



**Figure 1.1 System flowchart**

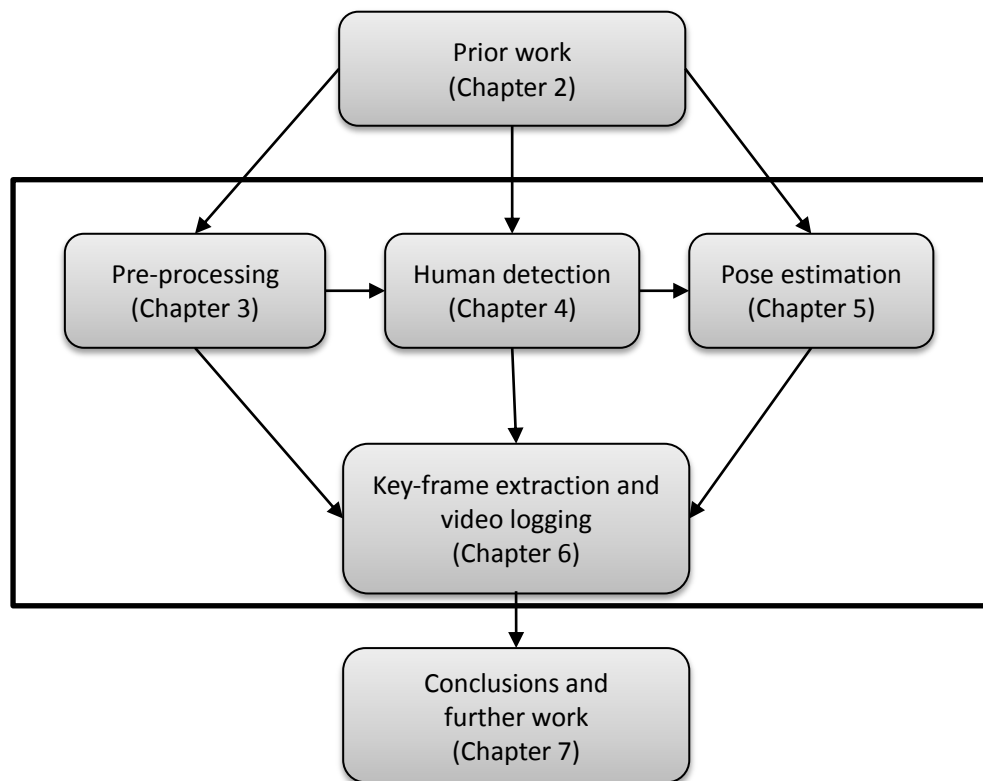
**(a) Flowchart of the system in the thesis, (b) Diagram for processing frame sequences, A represents the procedure in (a).**

## 1.7 Structure of the Thesis

The structure of this thesis follows the structure of our human motion analysis system, as shown in Figure 1.2.

Chapter 2 carries out a literature review on the work associated to the area of human motion analysis. It explains the definitions and concepts in this area, and divides the area into four topics: human detection, human tracking, human pose estimation and human action recognition, investigates and compares representative theories and methods proposed by other researchers in each topic, and finally introduces two practical human motion analysis systems.

Chapter 3 presents a novel colour-to-greyscale conversion method which is designed to produce greyscale images from RGB images with enhanced colour edges. The procedure and the experimental results of this novel method are illustrated in this chapter. We also make both quantitative and qualitative comparisons between our method and other methods.



**Figure 1.2 Structure of the main body of the thesis**



Chapter 4 illustrates an improved HOG human detector. The basic theory of HOG, the improved training scheme, the usage of pre-processed data and the experimental results are presented in this chapter. Another method which integrates Haar-like features and AdaBoost classifiers [13] [23] is also introduced as a comparison.

Chapter 5 mainly introduces a template-based human detection method named GDT&OM, and how we modify and apply it to estimate human poses. This chapter also introduces a pose classification method based on locations of body parts.

Chapter 6 describes the integration and implementation details of the entire human motion analysis system, and illustrates two practical applications of the system – key-frame extraction and video logging. We apply the system to a set of short videos to extract key-frames. A conventional method of key-frame extraction which uses PME (Perceived Motion Energy) [20] is implemented as a comparison. Then, finally another system that creates logs which record human actions for a set of extended footages is investigated, which is the final goal of the research. A set of gymnastics videos is used in the experiment. The system recognises and distinguishes significant actions in these videos and makes the log by recording these actions along the timeline.

Chapter 7 makes conclusions for the thesis and investigates the superiority and weakness of the system proposed. Then the further work which is expected to improve the performance of the system is also described.

# Chapter 2

## Prior Work

### in Human Motion Analysis

---

Human motion analysis is a branch of motion analysis, which is a large and important area in computer vision and image processing. It concentrates on recognising or describing the motions of human bodies in images or videos. The earliest research in this area can be traced back to the 1970s [2] [3] [24]. Johansson described a method called moving light displays (MLD) in 1975 [2] to analyse the movement of the joints in human bodies. In this paper the author introduced a method that simply represented a human body by stick figures which were lines linked by joints, whose motion were tracked and estimated in order to recognize the action of the human body. This was one of the earliest papers on human motion analysis. Nowadays, with the development of technologies of algorithms and hardware, human motion analysis is starting to step out of the laboratories and come into people's daily life.

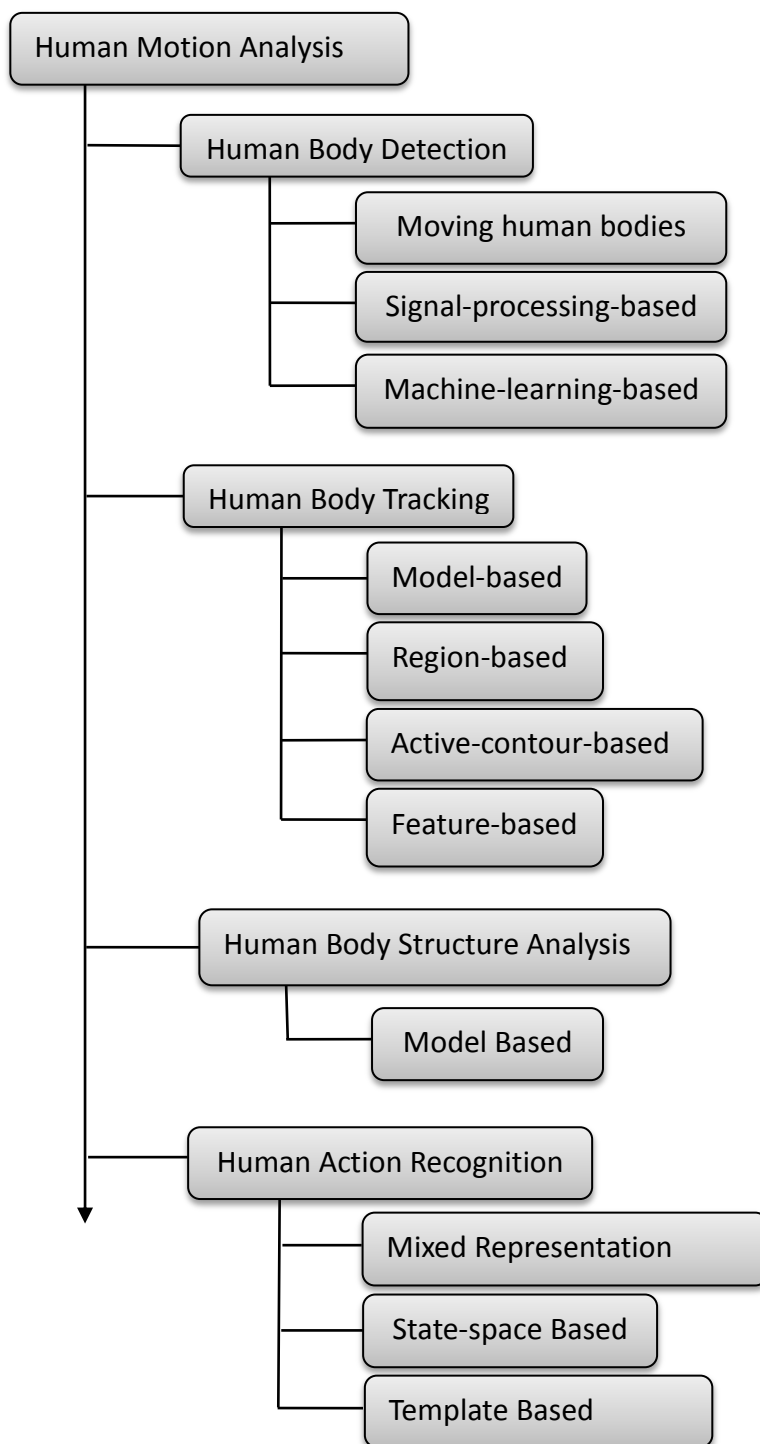
As mentioned in Chapter 1, human motion analysis in computer vision generally has four topics: human detection, human tracking, human action recognition (or human activity recognition) and body structure analysis. The survey [1] only stated 3 of these, without including human detection. However in most research projects, detection is the basic step for other processes. Papers introducing whole motion analysis systems usually put the detection phase as the starting phase [25] [26]. Other papers which do not discussing human motion detection also take the

detection result as their input. So in this chapter human detection is treated as an important part of human motion analysis and introduced in detail. Some other surveys [27] [28] do not include body structure analysis, which nowadays is popular and applied in many practical applications. This chapter will introduce that as well.

Human detection detects the whole or part of human bodies in the input images. Human tracking considers whether in frame sequences the detected human bodies are for the same person or for different persons. In other words it estimates the motion trace of people in the input frame sequences. Human motion recognition focuses on recognising the behaviour of the detected person. Body structure analysis considers the locations, movement and relationships of human body parts. Each topic in human motion analysis can also be divided to several subtopics. Figure 2.1 shows the whole structure of this area.

Generally speaking, two types of data are used in human motion analysis: stationary images and frame sequences (or called videos). Stationary images do not have information in the time domain, so tracking methods cannot be applied to these images. And of course there are no time-domain-based approaches to action recognition for these images either. Tracking methods are usually applied to frame sequences, and relationship between frames can be analysed and used to do action recognition.

In the research, we found that most state of art, human detection methods do not achieve very good performance, in other words, they do not achieve very high detection rate against a very low error rate. So we did some research and experimentation in the area of pre-processing, in order to improve the detection performance. Therefore in this chapter, the related work for pre-processing will also be introduced.



**Figure 2.1 Overview of human motion analysis**

## 2.1 Human detection

Human Detection is a kind of image region classification. Essentially it classifies the areas containing human bodies and those without human bodies. According to the requirements of the system, the detection target can be either full bodies or part of bodies. For example, a surveillance picture may have multiple kinds of objects, such as pedestrians, buildings, vehicles, animals, blank background, etc. The goal of the human detector is finding the pedestrians' bodies and rejecting other objects. Without accurately distinguishing the human bodies from other objects, the further phases such as tracking and recognition cannot be correctly executed, because the detection phase provides the input to the further phases.

Although human detection is a branch of object classification, three factors make it more complicated compared with some other kinds of object classification: body styles, clothing conditions and poses. The appearances of the target objects (human bodies) vary hugely due to these three factors. It is easy to imagine how much difference there is between a fat, little, sitting boy wearing a jacket and a tall, thin, dancing lady wearing a dress.

There are three categories of approach to detect human bodies: moving human body approaches, signal-processing-based approaches and machine-learning-based approaches. There are also multiple types of detection targets, such as full bodies, part of bodies, human riding bicycles, or even human groups.

### 2.1.1 Human detection method for moving human bodies

This kind of method can only be applied to frame sequences, as stationary images only have static objects. The differences between frames, which are caused by motion, are used as information about moving objects.

Moving object blobs are typical features which represents the motion regions. Lipton et al. [29] used the dispersion and regions of the blobs as input to classify the moving objects into human bodies, vehicles and clutter, and optimised the result by temporal consistency constraints. Collin et al. [30] used spatial-time-mixed information, such

as blob dispersion, blob area, apparent aspect ratio of the blob bounding box, and camera zoom, to classify the blobs in each frame to find possible target regions by a viewpoint-specific three-layer neural network classifier. Then they kept the results in histograms which reflected the probabilities of which class the objects belonged to, and selected the most likely class label as the output. In that paper, they classified the objects into humans, vehicles, human groups and clutter.

Kuno et al. [31] used the silhouettes of the input images to build a human detection system. They extracted regions of brightness variation, selected regions caused by moving objects, merged regions in to silhouette patterns according to distances. Then they treated the silhouettes as combinations of some body segmentations so that the silhouette pattern features can be therefore extracted. Finally they classified the silhouettes by making judgments of silhouette patterns by using shape parameters such as the mean and the standard deviation of silhouette projection histogram and the aspect ratio of the circumscribing rectangle of moving regions.

Other work such as Cutler and Davis [32] considered the similarity of periodic human motion, and presented an approach that detects periodic human motion by estimating similarity with motion prototypes. This technique implements detection, tracking and recognition when executing time-frequency analysis on the periodic motion. A similar approach presented by Lipton [33] uses optical flow. Under the assumption that non-rigid moving objects such as human bodies more frequently generate residual flow than rigid objects do, the algorithm can distinguish human motion apart from other moving objects such as vehicles. These approaches are based on the assumption that human motion can be considered as periodic, and build motion models to match the target.

The weakness of this type of approach is obvious. It can only detect moving objects, so these approaches cannot be applied to still images. Even for videos, if the target is static, or a part of the target is static, these approaches do not work. Hence, the applications of these approaches are limited. In Section 5.2, we use blobs and silhouettes produced by frame difference to estimate motion regions and cancel complicated backgrounds. We combine blobs and silhouettes with edge images so

that moving edges will be extracted, and then a template matching is applied to locate targets.

### 2.1.2 Signal-processing-based human detection

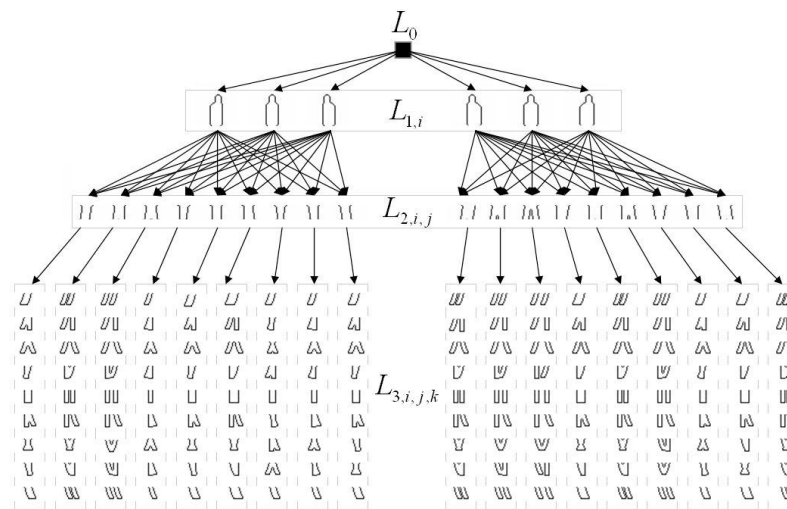
Signal-processing-based human detection is defined as a kind of method that matches signal information at specific locations. Usually these methods use template matching techniques in order to find regions that are similar to the pre-defined templates in the input images. The most common templates are the contours of human bodies, such as the examples in Figure 2.2. The edges are treated as specific locations on which information of pixels such as intensity and orientations are used for matching. The simplest way to match the templates is extracting the edges of the input images and comparing the edges with the contours. However, since sharps and poses of human bodies vary hugely, simple pixel-to-pixel matching will cause significant errors. A number of advanced techniques designed to increase matching accuracies have been proposed.



**Figure 2.2: An example of contour templates**

Some advanced techniques, such as matching by Hausdorff distances [34] [35], have also been applied in this area. The paper [35] which was presented by Yu and Leung, described a curve segment Hausdorff distance (CHD). This method extends the conventional point-to-point Hausdorff distance to a curve-to-curve distance, which improves the matching accuracy and the computational efficiency. However, at present, the edge-to-edge matching is not a widely applied solution for direction. Instead, researchers are often considering matching images and templates after transforms, which are image processing methods that keep and enhance positive information, such as edges between different objects and orientations, and meanwhile remove or weaken negative information, such as shade and overlaps.

Lin et al. [36] described a hierarchical part-template matching method, which segments the whole human contour to three parts: the upper body  $L_1$ , the middle body  $L_2$  and the lower body  $L_3$ . Each part has a number of different shapes representing different body poses, as the diagram shown in Figure 2.3. According to Figure 2.3, in  $L_1$  6 upper body parts are examined, in  $L_2$  each upper body candidate in  $L_1$  connects to 9 middle body parts, and in  $L_3$  each middle body candidate in  $L_2$  also connects to 9 lower body parts. Meanwhile according to the nature of human body shapes and poses, different higher level templates connect to different lower body template sets. There are 486 combinations that compose the contour templates of  $L_1$ ,  $L_2$  and  $L_3$  into an entire human outline under a hierarchical combining rule. In the matching phase, the edges of the images match the templates hierarchically from  $L_1$  to  $L_3$ , and optimise the result by maximising the joint likelihood. The combination with calibration and background subtraction is also discussed in the paper.

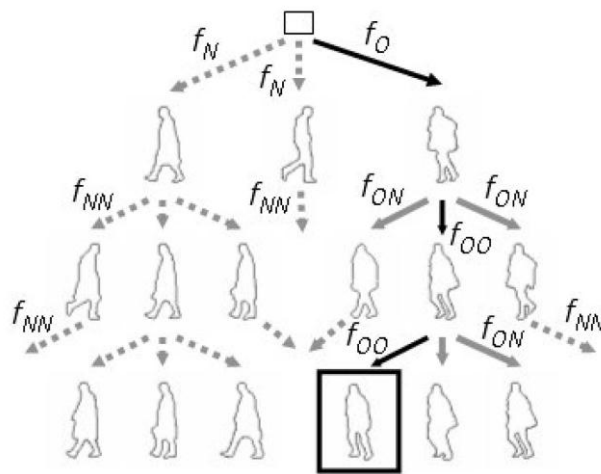


**Figure 2.3: Hierarchical part-template matching (from the paper [36])**

Gavrila described a probabilistic approach to hierarchical template matching [37]. This method constructs a hierarchical tree clustering similar exemplars (templates) and uses distance transforms (DTs) [38] to measure the similarity between images and exemplars. Then it executes matching hierarchically from root (the node on the top) of the tree to the leaves (the nodes on the bottom). This paper contributes to building a Bayesian model to estimate the posteriori probability of the object class.



Figure 2.4 indicates the procedure of this method. In Figure 2.4, an input contour is classified to the leaf node marked by a rectangle in solid black (optimal path). The symbol  $OO$  means both the parent and the current nodes are on the optimal path,  $ON$  means the parent node is on the optimal path but the current is not, and  $NN$  means neither the parent nor the current node is on the optimal path. Multiple training samples are used to pass the procedure above and their distance measurements and path information on each node are collected. Then a Bayesian model [39] is applied to estimate the probability on each node to boost the processing speed.



**Figure 2.4: Gavrila's method (partial view), from the paper [37]**

Thanh et al. [19] presented a weighted template matching method that employed a generalized distance transform and an orientation map (GDT&OM). From the image to be processed, this approach builds a GDT image which indicates the distance to the nearest edge of each pixel, and calculates the orientations both on the image and the templates. The information of the GDT image and the OM are both applied to measure the similarity between the detection windows and the templates. With these advanced features, the accuracy of the detection is significantly improved. This method is implemented and developed in the research. The reason why we employ this method is that it gives a good matching accuracy due to the employment of orientation maps. With this advantage relatively high accuracy of posture estimation will be obtained via this template matching method. The detail of the implementation will be discussed in further detail in Chapter 5.

Seemann et al. presented an advanced 4D Implicit Shape Model (4D-ISM) learning from articulations and viewpoints [40] [41]. In the recognition phase the algorithm employs an ISM voting space for each codebook entry. The spatial occurrence distribution and the associated shape will be stored. The highlight of this approach is that articulation clusters (which is similar to template sets) will be generated automatically in the training phase.

To make a conclusion for the template-based approaches, the following advantages and disadvantages are achieved.

Advantages: 1) They do not need a big set of image data to train the system. 2) The amount of computation is relatively small.

Disadvantages: It is very difficult to find appropriate templates. The templates are usually made manually, or extracted from a few existing images, which are not robust ways. If we want to detect humans with different poses, clothes or body styles, more templates will be needed. When the number of templates increases, the interference between templates grows quickly. So at present, template matching approaches are applied to limited range of poses, clothes and body styles.

### **2.1.3 Machine-learning-based human detection**

We define machine-learning-based human detection as a kind of method classifies unknown object samples by experience of known samples. These methods statistically summarises information from all training images and employ methods of machine learning and pattern recognition. Unlike signal-processing-based methods, these methods do not concern signals at specific locations, but treat information in the detection widow as an entirety, build statistical models which use the features extracted from the training data, and employ the models to make classifiers (detectors). Features are computer vision methods designed to represent important information of images. For example, Harris corners are used to indicate interest points of images; Haar-like features and histogram of orientated gradients are descriptors that reflect gradient and edge information; Principal components represent statistical features of the whole image. The classification methods can be

boosting/AdaBoost classifier, Bayes classifier, support vector machine (SVM), neural network, nearest neighbourhood etc.

The concept of Haar-like features is developed from Haar wavelets theory. It extracts specific locations in the detection windows and represents them by adjacent rectangular regions. Differences which are used as features in classification, are calculated between each pair of adjacent rectangular regions by comparing their pixel intensities. This difference is then used to categorize subsections of an image. Boosting/AdaBoost classifiers are a kind of mechanism that combines weak classifiers into a strong classifier. These classifiers are often constructed by cascade in the presented methods [13] [23]. The basic idea is: each weak classifier has the duty of classifying a part of the features and passing the classification result to the higher level. This mechanism can significantly accelerate the efficiency of the classification. So boosting/AdaBoost is often applied to build real-time systems. The paper [13] published by Viola and Jones joins together a base line of Haar-like features with AdaBoost approach (We call this category of methods Haar-AdaBoost for short in this thesis). Many researchers developed their approaches based on this. A typical Haar-like feature and boosting approach is described by Kruppa and Schiele [42] based on Lienhart's face detector [43] [44] and this is a standard approach in OpenCV. The paper [42] described the Haar-like feature and boosting approach for face detection. The detector introduced could also be applied to bodies with a different set of configuration and training data. Kruppa and Schiele provided the configuration files for the human detector to detect upper, lower and full bodies.

Histogram of Orientated Gradients (HOG) is a popular topic in the area of human detection in recent years. It collects and stores the histograms of the gradients and orientations as the features. The original approach of HOG, presented by Dalal and Triggs [18] combined HOG with SVM, which was a slow classifier. Recently more and more researchers combine HOG with an AdaBoost classifier [45], which makes the computation much more efficient. In paper [45] the HOGs of small image regions are extracted to build weak classifiers with SVMs, and the strong classifier, which executes categorisation for large regions, is composed by the weak classifiers.

The system proposed by the thesis applies the HOG method as the human detector. In Chapter 4, research, experiments and discussions on raising the performance of the HOG approach will be introduced. Meanwhile a comparison between HOG and Haar-like feature/AdaBoost will be presented in Chapter 4.

A more recent concept in the area of human detection is employing random forests. A random forest is a training and classification mechanism that is becoming popular recently in the areas of machine learning and pattern recognition [14]. A random forest consists of a number of random trees. Each tree is constructed by the values of a random vector sampled independently. All trees in the forest must have the same distribution model. The classifier uses a voting mechanism to make decisions. Gall et al. [26] introduced a novel approach employing Hough forests. Hough forests are random forests consisting of random trees and were adapted to perform a generalized Hough transform in an efficient way. This algorithm was based on the codebook detector and mapped the image features to a Hough space. It learnt the models by mapping from the extracted features named “cuboids” to their corresponding votes. The term “cuboid” stands for a local image patch (two dimensional) or a video spatial-temporal neighbourhood (three dimensional) in a generalised scene depending on the task. A Hough forest is applied to learn the model of mapping from the appearance of cuboids to their corresponding probabilistic Hough votes which are leaf nodes. In other words, a leaf node is trained to store the probability of a patch belonging to an object class and the decisions of these nodes are summarised to produce the final classification result. The construction of the random trees followed the standard way of a random forest framework [14]. The Hough forests did not only apply to detection, but also tracking and recognition.

Comparing with signal-processing based approaches, machine-learning-based approaches usually need more training data and more computational resource. However, more training data means more data variations are involved. For example, various human body shapes and poses can be concerned by using this kind of method. Machine-learning-based approaches essentially combine great amount of information to build reliable, robust detectors.

## 2.2 Human tracking

For videos, tracking is an indispensable especially when there are multiple targets detected. As for further processing, we need to connect the detected information for the same person who acts in different frames. Tracking is also designed to identify and track human body parts in videos. Tracking is an issue that is related to the features of position, speed, shape, texture and colour of an object. Generally there are four types of tracking methods:

### 2.2.1 Model-based tracking

Human bodies can be represented in three ways: stick figures, 2D contours and 3D volumes. Each way supports some method for tracking. Stick figure approaches approximate human bodies as the bones and joints, and analyse the relationship of the bones and joints to track the motion. Neglecting the texture, 2D contour approaches analyse the projection of human bodies on the image. 3D volume models represent human bodies as spheres, cylinders and centurms, which can describe much more detail of the bodies, and of course, are much more complicated.

An approach based on stick figures, or 2D skeletons, was proposed by Karaulova et al. in paper [46]. The idea is to use Hierarchical Principal Component Analysis (HPCA) to describe the movement (called kinematics in the paper) and Hidden Markov Models (HMMs) to describe the action (called dynamics in the paper). The hierarchy has a top level storing the whole body information and lower levels storing information of the body parts. 2D and 3D data can both be used as training data being treated as stickmen and decomposed into parts.

Jin et al. recently proposed a paper that tracked human using colour-histogram and HOG [47]. As HOG is an edge based feature, contour information was used in this approach. The colour information was collected by histograms along with HOG features. As the distribution of human tracking is non-linear and non-Gaussian, the authors employed Particle Filters [48] to build the tracking system. They collect the particles, which are the differently-weighted samples of the distribution, near the

centre of the object, calculated the differences between every two neighbour states (the centres of the human), updated the weights and particles, and finally estimated the new centre of the objects. The highlight of this method is a combination of colour and edge information, which differ from other methods that only use greyscale information.

For 3D cases, the most common model is also a skeleton model, which contains 3D position and rotation information for human body parts and joints. Tong and Bian presented an SLDM approach [49]. SLDM stands for shared latent dynamical model and is developed from Gaussian processes models. This model executes mappings from a shared latent space to both state space and observation space as well as a dynamical mapping in latent space. This paper starts from the 3D human skeleton model which has 49 degrees of reconstruction freedom, runs an SLDM to estimate the dynamics, compute observations, and reconstructed the motion trace.

Zhang et al. also presented a skeleton body model for both 2D and 3D upper body tracking [50]. The model decomposes a 2D human upper body into 6 parts, each of which is represented by a reference point and a set of state variables. The model also provides the relationship between the body parts so that size and location of each part are constrained by others. Then the model can be also extended to a 3D version. The upper body model is applied to a dynamic Bayesian network [51] to achieve tracking. However samples of upper bodies with hidden parts, e.g. arms hidden behind torsos, which is considered as a major difficulty in human tracking, are not presented in this paper.

### **2.2.2 Region-based tracking**

Region-based tracking usually builds Gaussian models for human bodies and background, tracks the small regions, such as heads, arms, legs, and then combines them together to achieve the target. The problem is the influence of shadows when processing the moving objects. One solution is based on the colour information. The edges of different regions can be found by colour difference. The other problem which is more difficult to solve, is that there are overlaps between people. Some advanced detecting algorithms based on multiple cameras are able to do this, but it is out of this topic.

The paper proposed by McKenna et al. [52] describes a method that combines colour and gradient information in motion segmentation, which aims to solve the shadow problem. Regions, people and groups are considered as three levels in tracking. A human body consists of a number of regions, which can be merged and split with bounding boxes. And a human group is composed by one or more human bodies. So a tracking process starts from detecting and tracking moving regions, and composes upper levels step by step by using region information.

### **2.2.3 Feature-based tracking**

Feature-based tracking includes feature extraction and feature matching. It treats the objects to be tracked as sets of features instead of whole shapes. So that even if part of the target object disappears, the algorithm can still track the rest. However the type of the features is limited. It is too difficult and expensive to track multiple complicated features. That is to say, low-level features as points and grids are easier to extract and track, but lines or blobs, which are relatively high-level features, are more difficult to extract and track. There is a trade-off between complexity and efficiency [28].

According to Polana and Nelson, they extracted a person with a rectangular rim, and selected the centroid as the feature point [53]. In their paper, when two persons are shading each other, they will be distinguished by the speed of the centroid. Except for the centroid, other features such as colour, texture can also be involved in feature-based tracking.

Jang and Choi presented a method using points and lines with Kalman Filtering [54]. They extracted high-level features such as shapes, textures, colours and edges. Kalman filter estimates motion parameters which help the system to track movement of non-rigid objects by feature energy minimisation.

In conclusion, comparing these four types of tracking methods have their own characteristics. Model-based tracking is traditional and the most straightforward approach. But shading and unusual poses cause problems for selecting models. Region-based tracking is more flexible than model-based tracking and widely

applied. However it more depends on moving regions and the overlapping problem is very difficult to work out through methods using a single camera. Feature-based tracking takes part of the object as tracking target, which means that even part of a tracked object is hidden, some features of the object remain visible. But in practice high precision tracking needs high-level features, while high-level feature matching raises difficulty and expense [28].

Tung and Matsuyama presented a practical approach that tracks human motion by local features [55]. The paper proposes a system that consists of two models: detection and tracking, and switches between the two models. The detection model is used to locate important human body parts such as faces/heads and hands, and the tracking model is used to keep updating the locations of the body parts based on an adaptive colour-based particle filter with an optical flow estimator. Detection results are used as references to correct tracking estimations.

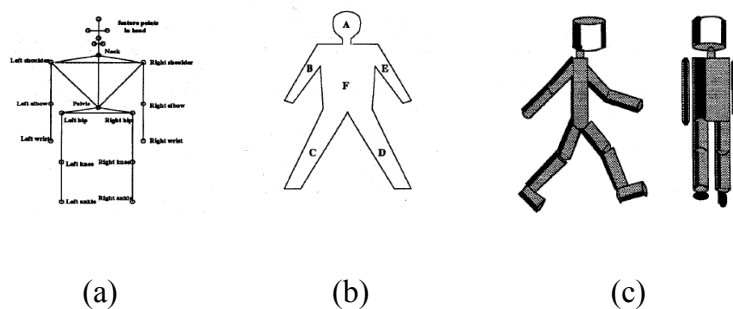
Since the system proposed in the thesis is initially designed to analyse human motions in stationary images, tracking methods are not applied. However, in the applications, videos are also used and the system performs detection in each frame. When the system process videos with multiple people, a tracking method is required to identify different people through frames. This part of work will be carried out in the future.

## **2.3 Human structure analysis**

Human structure analysis works to find locations, orientations and sizes of the human body parts. It has a tight relationship with posture estimation as given the information of the body parts, it will be not difficult to identify the poses. Actually there are several papers discussing the methods that estimate the poses from the body parts information [25] [56] [57]. The discussion of this research is put in Section 2.6.1.



To represent the parts of human bodies, there are mainly three ways: the skeleton models, the contour models with marks on every part and the volumetric models. The skeleton or stickman models are the most common representation and nowadays most approaches for 2D images and videos are based on it. The advantage of the stickman models is that they are constructed by joints which allow the body parts to move and rotate. This is flexible and compatible for the nature of human motion. The contour models are usually fixed outlines of human bodies. So they can only estimate approximate information for the body parts. Especially when the number of the body parts is large, using contours models is not an appropriate solution. While the volumetric models are more concerned with the size and depth of the body parts, which is more compatible for 3D analysis. An example of 3D structure analysis will be introduced in section 2.7. In this section we only discuss 2D cases.

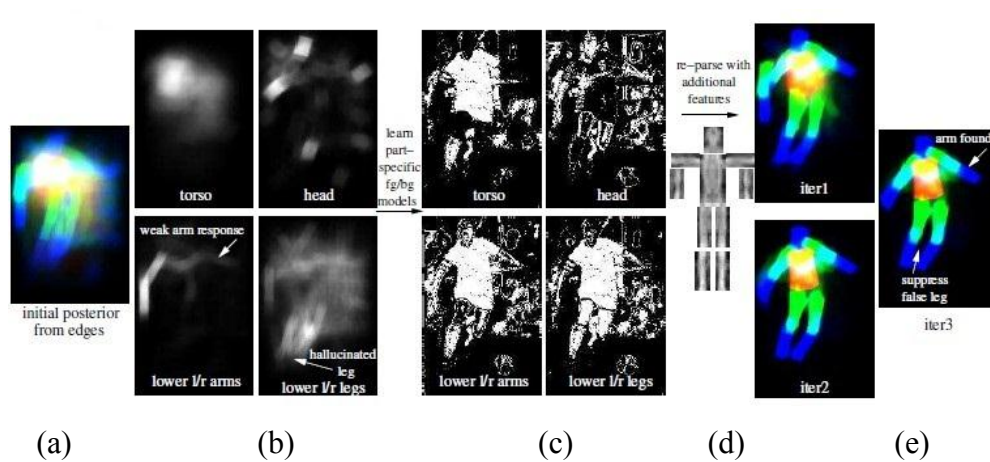


**Figure 2.5 Human structure models, from the paper [1]**

**(a) a 2D stickman model, (b) a contour model, (c) a 3D volume model.**

A successful approach of body structure analysis called iterative parsing is proposed Ramanan [56] [58]. This technique has been implemented in many other researchers' work [25] [57] as it achieves relatively high accuracy of body part segmentation. Figure 2.6 indicates the procedure in Ramanan's paper. At the beginning, as the colour picture in the very left, Figure 2.6(a), iterative parsing begins with estimating the localisations of the body parts by matching with a deformable pose model based on edges. The matching results are usually poor at this stage. This is the initial parse of the procedure, which only gives a rough estimation of the parts. Then the initial parse is applied to build a foreground/background colour histogram model for all body parts which are regions on the image at this stage, as Figure 2.6(b) showed. In Figure 2.6(c), the region model is applied to the input image as masks so that the

matching region will be reduced. Figure 2.6(c) (d) (e) illuminate that the deformable model is used to parse the image and reduce the matching region iteratively, so that the program can achieve accurate locations and orientations finally. Figure 2.6(d) is the deformable model and Figure 2.6(e) is the final result in which different colours stand for different body parts. Normally setting the number of iterations to 3 meets most demands of accuracy according to Ramanan’s experiments.



**Figure 2.6 Procedure of Ramanan’s iterative parsing method**

**(a)the result of the initial parse, (b)foreground/background model of different body parts,(c)masks on the input image,(d)the deformable model,(e)results of parsing by the deformable model in 3 iterations. (From the paper [58]).**

There are some problems with iterative parsing. First of all, it cannot achieve good performance for small scale figures and some common poses. Secondly, as the sizes of the parts in the deformable model are fixed, overlapping between body parts will occur in the output result. Thirdly, the algorithm does not know how to deal with the invisible body parts in some images, and this is the major problem that we do not apply this in our work. Besides template matching, we also introduce an approach that uses locations of body parts to build classification models. However the performance of the classifier strongly depends on the accuracy of part location, and Ramanan’s method is still not accurate enough, which will be discussed in detail in Chapter 5. But in all, iterative parsing is an impressive approach in the research topic of body structure analysis, and has the potential of improvement.

## 2.4 Human action recognition

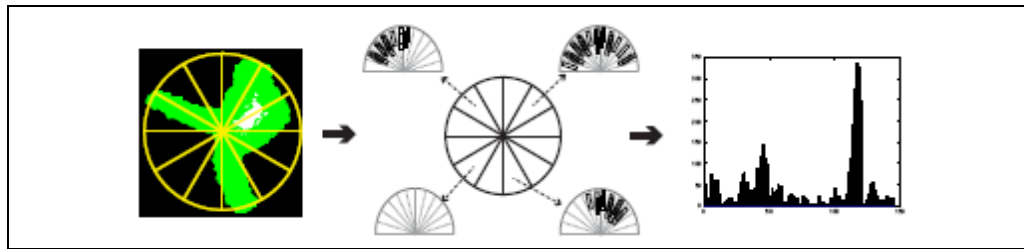
Human action recognition is highly valuable in practical applications. The methods vary according to the data types. For frame sequences, the information in the time-domain is the key. The approaches are usually state-space based. For still images, action recognition can be considered as pose estimation.

### 2.4.1 Pose estimation

Pose estimation in still images is usually related to human detection. Pose estimation can be considered as a multi-kernel classification problem, and human detection can be considered as a single-kernel classification problem. Therefore the methods used for human detection, either signal-processing-based or machine-learning-based, can be applied to pose estimation. In another point of view, it also has a tight connection with human structure analysis. If the structure of a body, which means the positions of the parts of the body, has been found, it is very easy to identify the pose.

Wang et al. [59] located the parts of the bodies by comparing images to be classified with standard images. In the paper, a deformable template matching algorithm which extracts a set of sample points from the edge and found the assignment between the sample points, is employed. After a spectral clustering mechanism, still images will be classified into different action clusters.

Ikizler [57] presented another estimation method. They used Ramanan's method [56] to extract the body structures and Histogram of Oriented Rectangles (HORs) to representing the pose. They used the extracted rectangular regions which are histogrammed over 15° orientations to build HORs. They designed spatial circular grids and circular HORs for still images, see Figure 2.7. They applied Linear Discriminant Analysis to reduce the feature dimensions and trained one-vs-all SVM classifiers for each action class. They built the SVM classifiers by using RBF (Radial basis function) kernels [60].



**Figure 2.7 Represent poses by CHORs, from the paper [57]**

Pose estimation is used in another type of application – motion capture. Tangkuampien and Suter represented [61] a marker-less motion capture system that employs multiple cameras and ground-truth poses. The learning mechanism, which is called Kernel Subspace Mapping (KSM), uses two Kernel PCAs (KPCA) [62] to learn pose and image manifolds separately, and then connect the two manifolds by using Local Linear Embedding [63]. The significance of the system is that it accurately coordinates human motion and reduces the dimensional redundancy by KSM.

Poppe [64] proposed a method that extracted features of orientation histograms of human silhouette gradients (HOSG). The method is similar to HOG but only uses outlines of human silhouettes instead of all gradients. Then a common spatial patterns (CSP) is applied which differs from conventional classifiers by focusing on differences between classes rather than on modelling classes individually. This method will be a comparison with our system in Chapter 5.

In conclusion, the two approaches mentioned above both extracted the body structures first, represented the structures by some kind of features, and finally classify the extracted body to a predefined class. This part of prior work illustrates that simple human actions can be recognised by estimating poses, which is also performed by the system of this thesis.

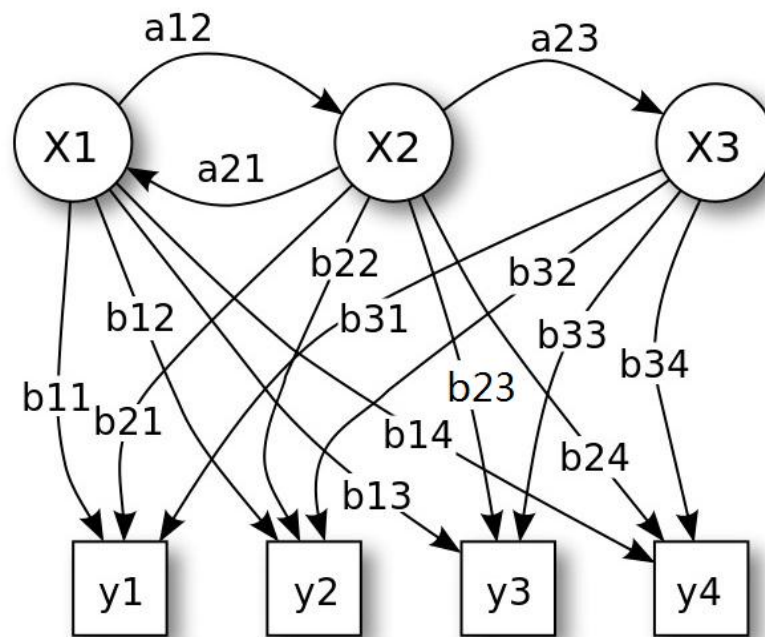
#### **2.4.2 State-space based approaches**

State-space-based approaches define every static pose as a state, and treat the motion as a course of those states. Then if the system can find the relationship between the states by a kind of probability model, the action will be recognized. For a frame-sequence, state-space based action recognition has two steps: 1) identify the state of

the human body for every frame. 2) classify the state sequence to an action class. For the first step, the pose estimation mentioned in the previous section can also be used to identify states. However, the accuracy of the pose estimation is a big problem to carry out. Therefore researchers usually employ simpler ways, for example, locating the positions of the hands, measuring the distance between two feet etc., to represent the states. For the second step, some statistical models are applied.

The most popular model is HMMs [65] (Hidden Markov Models). Hidden Markov Model is a statistical method assuming the system to be a Markov process with unobserved states, but its output dependent on the states is visible. Each state has a probability distribution over the possible output tokens. Therefore the sequence of tokens generated by an HMM gives some information about the sequence of states. The word “hidden” indicates the state sequence through which the model passes, not the parameters of the model. The usage of HMMs in motion recognition includes two phases: training and matching. Training is the process that defines the parameters of an HMM and optimizes the states, in order to generate the patterns that are used to match the features of defined motions. Figure 2.8 indicates the structure of HMMs. In Figure 2.8,  $X_i$  ( $i=1,2,3$ ) are hidden states, and there are converting probabilities between them, denoted as  $a_{ij}$  ( $i=1,2,3; j=1,2,3$ ). The symbols  $y_n$  ( $n=1,2,3,4$ ) represent observations, and the relationship between  $X_i$  and  $y_n$  can be concluded as a set of probabilities  $b_{in}$ . That is to say, in every time  $t$ , the varying hidden state will give an observation according to the distribution of  $b_{in}$ . Given 3 of these 4 sets of variables, the remaining can be estimated set according to the HMM theory.

A recent work using HMM is Li’s paper [66] which discusses an improved HMM. Firstly the method represents each action by several action graphs (each graph is the silhouette of a human body and can be treated as an action state). Unlike conventional HMM, the states are shared by all action types, so a pool of action states is built, so the action graphs can be trained effectively and efficiently according to the author. Finally the paper proposes a silhouette matching algorithm combining both shape and motion features to identify motion states.



**Figure 2.8: A Hidden Markov Model, from the website [67]**

The paper proposed by Yamato et al. [68] illuminates a tennis-motion recognition method which employs a compact observation codebook formed by clustered grid-based silhouette mesh features, uses Baum-Welch algorithm to train HMMs and estimate the probabilities by Viterbi algorithm [65]. Feng and Perona mapped key poses to states in a static HMM [69]. They simplified the observation model by reducing the flexibility to achieve more efficient training. There is a lot of further research on HMMs, such as Coupled Hidden Markov Models (CHMMs) [70] and Variable Length Markov Models (VLMMs) [71], applied for action recognition. The CHMMs coupled two or more HMMs which might or might not have relationships. The two state sequences both influence the output. In that paper the author states that this multiple-process model is faster and more robust than single-process models in vision processing. VLMMs is a kind of random processes with variable memory length, in contrast to the conventional  $n$ -th order HMMs. It was more flexible when processing the action recognition cases.

Another type of methods, called Bayesian Networks (BNs) [72], is a kind of probabilistic graphical models the same as Markov Models. As Figure 2.9 shows, each node ( $X_2, X_4, X_5$ ) in the graph represents a random variable, while the arrows

between the nodes represent probabilistic dependencies among the corresponding random variables. The tables indicate those probabilistic relationships. These conditional dependencies in the graph are often estimated by using known statistical and computational methods. These probabilities can be trained by machine-learning. The training process is a set of iterative schemes that look for a maximum likelihood for the parameters which are treated as localized message passing operations.

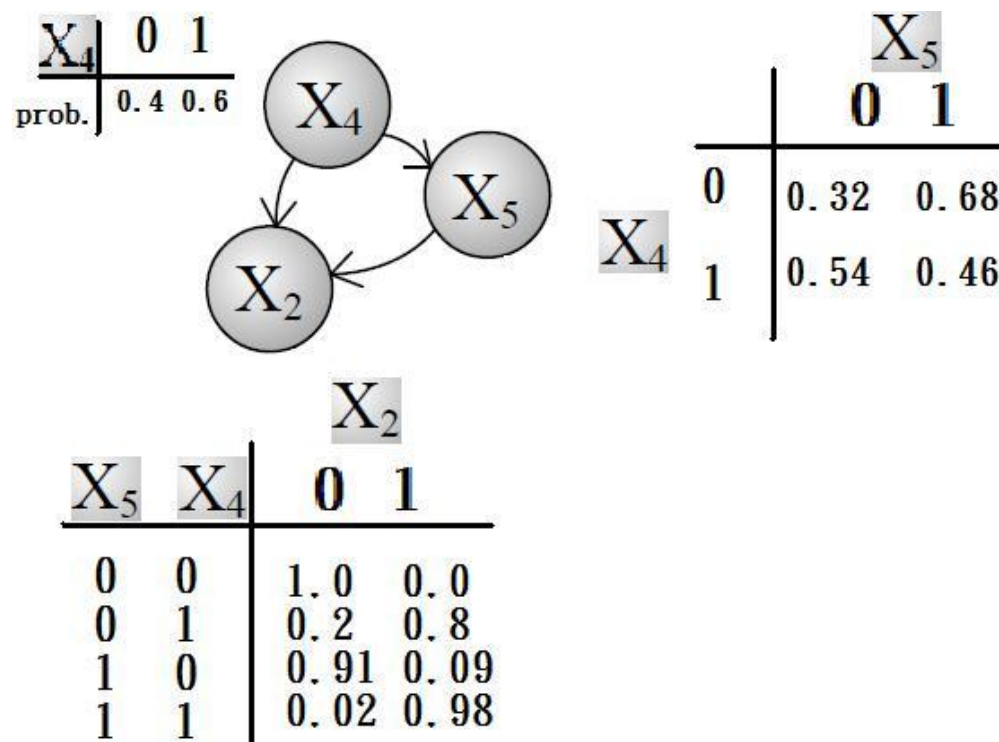


Figure 2.9: The nodes of a Bayesian Network, from the paper [39]

Remagnino et al. [73] developed a model for the classification and annotation of multi-agent actions, using BNs on two levels. The lower level is a set of BNs called behaviour agents. They contain input nodes associated with characteristics such as speed, acceleration and heading, which are the input information to hidden nodes. The upper level which is also BNs consist of behaviour agents and nodes corresponding to motion direction. Each behaviour agent connects to the upper level, a situation agent, via a behaviour node. A situation agent collects information from its nodes and determines the motion type in an interaction node. When the Euclidean separation between two behaviour agents fell below a specified threshold, the

situation agents, would be activated.

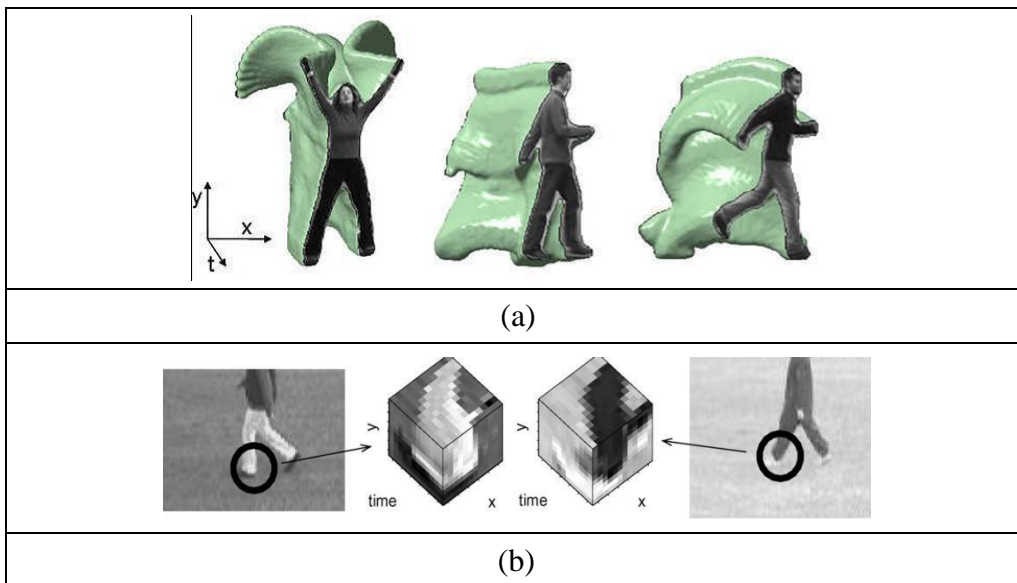
The state-based approaches are elegant methods. However, many factors constrain these techniques in practice. First of all, it is constrained by the nature of human action. As known to all, human action is diversified and complicated. Multiple persons act in multiple styles for the same action. And there are thousands of action types. To solve the problems above need robust, efficient methods and huge data sets. Therefore building a practical recognition system is an enormous, formidable and expensive project. Secondly, it is constrained by the accuracy of state identification. Just as mentioned above, pose estimation and body structure analysis are the main methods to identify the motion states. However, the accuracy of the approaches in these two topics still needs improvement at present. Accurate state identification cannot be guaranteed in the state of the art. Hence most state-based approaches work under the assumption that the states have been given or use relatively simple state sets. Because of the two reasons above, state-based approaches now can only process some simple motion cases and limited number of motion classes. There is still a long way to go in practice.

Since at present the system is designed to analyse simple human actions which can be identified by human poses, state-based methods are not used. For complex human behaviours that consist of several actions, state-based method can be applied as a high level recognition mechanism, which will be carried out in the further work.

### **2.4.3 Mixed representation approaches**

The mixed representation approaches form another big category of the recognition methods. The idea of this category is combining and representing the motion information in both the spatial domain and the time domain. Thus state estimation can be bypassed compared with the state-based approaches. According to the way of extracting representation features, this type can be divided into global and local approaches.





**Figure 2.10** Examples of mixed features

**(a)space-time blobs, (b)space-time cuboids features. (From the paper [27])**

Gorelick et al. described an action recognition method in the paper [74] based on space-time (or called spatial-temporal in some papers) shape (Figure 2.10(a)) generated from a human action. It is a global representation method which composes information in the spatial and time domain to global features. A space-time shape, which includes all of the human poses in the motion period, was a visual concatenation of 2D silhouettes.

The local features such as space-time saliency and space-time orientations, as well as the global features of the shapes are extracted. These features are used to do action classification and clustering. Local approaches extract space-time information in some particular positions. Laptev et al. defined local motion events in the paper [75] to represent the position and the shape of associated space-time neighbourhoods. These features are extracted by an extended Harris corner detector that is able to detect corners in the spatial-temporal space. They compared the performance of a nearest neighbourhood classifier and an SVM. They also compared the local and global space-time events in the experiment. The paper [76] represented by Jhuang et al. describes a biological inspired system based on a hierarchical feed-forward architecture. The system extracts three types local space-time features: gradient-based, optical-flow-based and orientation-based. Then a neurobiological model is

applied to match with the spatial-temporal templates hierarchically with three stages. Finally the system applies a linear multi-class SVM classifier to achieve classification. Another recent approach using local features proposed by Oshin et al. [77] that uses Ferns [78] to train and classify local spatial-temporal features (interest points). Fern is an improvement of the Naive Bayesian classification framework [79], a simple probabilistic classifier based on applying Bayes' theorem with strong independence assumptions. The Fern classification is simple, efficient and robust. These features are extracted by both Laptev's method and Dollar's method [80].

The three papers above argue that a human motion can be described as a set of local space-time features which exploit the appearances of human body parts and their movements (Figure 2.10(b)), present methods that measure the similarity between the 3D local features to train a set of motion models. Laptev's paper also introduces a mechanism for local velocity adaptation and evaluates the influence by camera motions, which is an important contribution by the researchers.

Mined hierarchical compound features are a type of novel idea presented by Gilbert [81]. Their methods extract an over-complete set of simple 2D corners in both space and time. A hierarchical process is employed to encode the neighbourhood of these corners along with the searching area increasing. Meanwhile they used data mining to learn the most distinctive and descriptive features at each stage of the hierarchy. The features would become more and more significant, strong, or in the authors' words, "complex, discriminative and sparse" as the hierarchical levels go up because frequently appearing features would be enhanced. According to these learned features, a label of the actions can be assigned by the maximum response for each frame and finally a decision of action class for the sequence can be decided. This approach was claimed to be very fast and able to achieve real-time processing.

As mentioned above, the paper [26] described a solution not only for human detection, but tracking and action recognition as well. This approach used space-time feature cuboids which were a kind of mixed representation. To execute the tracking stage, the process was very similar to detection, just assembling the detections into tracks by using the confidence as observation for a particle filter. There was some

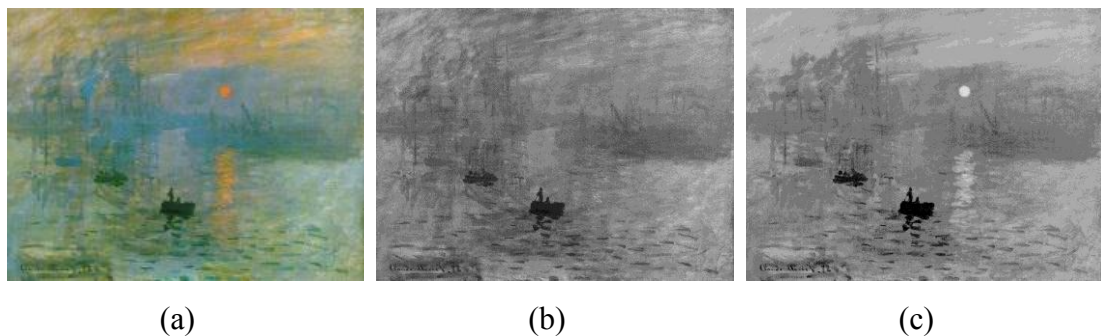
additional information such as colour and texture added to achieve better performance. The training process for recognition was using sets of motion sequences (tracks) for different motion classes to build Hough forest models. In the classification phase, the same voting mechanism as the detection phase is applied. The recognition precision was encouraging. The precision for all of the testing datasets was over 90% and better than most other approaches being compared.

Mixed representation approaches are a group of relatively new methods for action recognition. The benefit of using these methods is that the system can avoid classifying the motion states and sometimes even detection and tracking. They unified the problems on space and time domains into the same problem, which can make the process concise and faster. However, as there are differences of characters between the space-domain and the time-domain, finding a set of appropriate features to represent the characters of the space-domain and the time-domain, as well as their relationships, is extremely important and somehow tough. We argue that this kind of approach possibly represents the future direction in human motion analysis.

Additionally, the system presented by the thesis is a global-based approach, so a human detection method is employed. However a local-based method is a variation for human motion analysis and does not need to do global detection. Instead human bodies and motions are represented as a combination of local features, and action recognition can be treated as local feature matching. Hence local methods stand for a different way of describing motion videos, which provides another direction of developing our system. Besides, for a human motion video, some local features are not significantly influenced by changing the viewpoint, therefore using local features is a way that solves the problem of moving cameras, which is a key limitation of the system presented in this thesis. Hence the research on local features will be an important issue in the future.

## 2.5 Colour-to-greyscale mapping

This section is not directly related to human motion analysis. However in the thesis, colour to greyscale mapping is the pre-processing for human detection. Considering that most detection approaches at present essentially relate to the gradients or edges of the images, and the data used are usually colour image or videos. Most implementations of detections only take the luminance of a colour image as the input, which wastes the colour information. For example, considering two neighbour regions with far different colours but very close luminance values in an image, the edge between the two regions is very strong in the colour image but is very weak in its luminance image. The waste of the colour information has a negative effect on the detection. In order to deal with this edge problem, colour to greyscale mapping algorithms, which map every pixel in the colour image to a greyscale value and enhance the edges between regions of different colours, are applied. Figure 2.11 shows an example of this process. It is noticeable that in the luminance image, the sun and the reflection on the river have disappeared. However in the colour-to-greyscale mapped image, there is a clear view of the sun and the reflection.



**Figure 2.11 An colour-to-greyscale example**

**(a) the original colour image, (b) the luminance image, (c)the greyscale image generated by Color2gray**

At present most colour mapping methods can be can be divided into two major categories: global methods and local methods. The global methods do not consider the spatial information in the image. They only focus on how to map the values from a 3D RGB space to a 1D space. So the given colour will always be mapped to the

same greyscale value. The mapping of colour to luminance is an example of global method. Local methods, in contrast to the global ones, map the colour of a pixel to greyscale values depending upon the spatial surround of the pixel.

Some global methods such as Grundland's method [82], Rasche's method [83] [84] and Alsam's method [85] employ advanced colour-to-greyscale mapping functions. The method of Grundland et al. converts the values from the RGB space to the YIQ space, in which the Y component represents the luminance. Then the dimensionality is reduced from the 3D YIQ space to the 2D YC space. Finally 2D values are converted to the 1D greyscale values. Rasche et al. worked out the mapping problem by minimising an objective function which contained the information of the colours in the picture. Alsam et al. adjusted the conventional luminance mapping function to an image dependent function which defines different weights for each colour channel.

There are more local based methods that achieve encouraging performance, such as Socolinsky's method [86], Bala's method [87], Color2gray method [88] and Drew's method [89]. Among these methods, Color2gray is a particularly successful example and usually considered as a comparison target. This method worked in the L\*a\*b space. For each pair of neighbour pixels, it computes the difference by combining their chrominance and luminance differences. Then it executes a squares optimisation to assign a greyscale image in which each pair of neighbouring pixel has a difference approaching their corresponding target difference scalar, which is decided by the chrominance and luminance differences of this pair in the source image. However, Color2gray has a critical weakness that it was highly computationally expensive as it is order  $N^2$  for an N-pixel image, because to achieve a good optimisation, each pixel should treat all other pixels in the image as its neighbour, according to [88], and when the value of a pixel is updated, the distances between all other pixels need to be computed. In other words, it will take an extremely long time when processing a large image by this method. What's more, to yield the most visually pleasing solutions, manual adjustment of many parameters is required.

Colour-to-greyscale conversion at present is well developed. Generally there are quite a few approaches that can meet the requirement in practice. The goal in this topic now is to develop methods that are more robust to deal with problems of some extreme pictures and less computationally expensive. In Chapter 6 a novel global approach that reaches a similar result to Color2Gray but with less resource consuming is introduced.

## **2.6 Human motion analysis systems**

There are a number of completed systems that operate for human motion analysis. Here “completed” means that the systems are able to take images or videos from cameras as input, and output the motion analysis result such as action recognition or body structure analysis. The approach of Hough forests discussed above is a good example. Besides, here Ferrari’s pose estimation and people retrieval system is introduced as a typical and complete example.

Ferrari et al. presented a system that combines detector, pose estimator and shot retrieval [25] [90]. The system is designed to find the frames which were similar to a proposed pose in a large number of movie shots. To achieve that, the system detects the upper bodies of people in the input images, extracts the structures of the bodies by identifying six body parts. Retrieval is executed according to the estimated pose.

The whole operation starts with an HOG detector developed by Dalal and Triggs [18], employing an SVM as the classifier. Then a foreground highlighting algorithm, which is based on the prior knowledge of human body structure and the Grabcut algorithm [91], is applied to tell apart the foreground (the human bodies) from the background in the images. To estimate the poses Ramanan’s iterative parsing algorithm [56] is employed to segment the detected bodies into heads, torsos, upper and lower arms, and store the location, orientation and size of each part. Three pose descriptors are introduced by the paper: 1) part positions; 2) part orientations, relative locations, and relative orientations; and 3) part soft-segmentations. These

descriptors work to extract the features from the information of detected body parts. For the final retrieval, the system has two operating modes, query mode and classifier mode.

Comparing with the HOG based pose retrieval, the Ferrari's system performed better in most situations. However, the accuracy (called precision in the paper, the rate of correctly locating body's position and identifying poses) is objectively low – no solutions could achieve the accuracy over 50%. But at least, this system gave a completed solution from original images to analysis result and proved some meaningful suggestions on how to link methods for each phase together.

## **2.7 Prior work and the research of the thesis**

In this chapter the big picture of human motion analysis area has been reviewed. The chapter introduced each topic in this research area, including the definition, the topic tasks, the basic concepts, the relationship to others, the recent contributions and the problems to be solved. What's more the topic of colour-to-greyscale, which is an additional part of the research of the thesis, is introduced as well. This chapter also gives an example of applications of human motion analysis.

The technical level in the state of art is the yardstick that evaluates our own work, and the advantages and disadvantages of existing methods can be summarised and compared so that the target of the proposed system in the thesis can be determined. Some existing techniques can be used, integrated and developed to build the system. Normally a novel method is based on several existing theories or methods along with integration and optimisation. Moreover, multiple approaches in the state of art give us different views of treating and solving problems.

Human motion analysis is a large, complicated research area. To build an accurate, reliable, robust and efficient system of human motion analysis is a difficult task.

There are still a number of problems to be worked out. The following points can be learnt from prior work:

- 1) Colour information are usually not used, which need to be improved because colour is an important way of identify objects in vision. Therefore the thesis discusses how colour information can be used to enhance edge representation and proposes a colour-to-greyscale method as a pre-processing procedure in Chapter 3.
- 2) There are a number of object detection methods that can be employed to locate human bodies. HOG at present is considered as the best approach for body detection. In Chapter4 we implement and improved the HOG method and compare it with the Haar-AdaBoost approach.
- 3) Building a general system to analyse all types of human motions is very difficult. At present most approaches are designed to solve a certain range of motion problems. The system represented by the thesis is designed to analyse motion of a full human body with a fixed camera.
- 4) A video can be treated in different points of view. To recognise the human action in a video, in the static view, its key-frame can be used for recognition; in the time state view, the linkage of poses can be used for recognition; while in the global view, shape variance can be integrated into an entire feature for recognition. Chapter 6 illustrates two applications of the system. The first one, key-frame extraction, is designed to extract the most representative frame of a video clip. The second one, video logging, is designed to recognise important actions in a video clip to build a state chain.
- 5) Local features of motion images and videos have the advantage of solving the problem of moving camera, and can be a topic in the further work.



# Chapter 3

## Pre-processing

---

As mentioned in Chapter 3, an HOG detector [18] collects histograms of gradients extracted from the training and testing images and uses them as the features. Haar-like [13] features are also shaped-like and responsive to, local edges. Hence extracting gradients differently will influence detection results. Considering that the images we are using are RGB colour images, while the original HOG approach only employs the gradients of the channel which has the largest difference, this mechanism loses gradient information in other unused channels. We assume that high-valued gradients which reflect strong edges of the images will increase the accuracy of human detection. Under this assumption, we have developed a series of methods to enhance the gradients by using the difference information of the three channels.

In digital images, edges are caused by different materials and shades/shadows. In human detection, we usually need to keep the information from material edges and ignore the shade/shadow edges. There is an argument that edges between different objects which consist of different materials are usually indicated by different colours. This may not always be correct in certain scenarios. Of course, sometimes different materials have the same colour, which makes the edges very weak. It is a natural problem in human vision, and is out of consideration of this thesis. On the other hand, due to the lighting conditions of the pictures, shades and shadows will also lead to edges. Considering the lighting condition is not predictable, the edges caused by shades and shadows will confuse human body detectors. This is to say, human

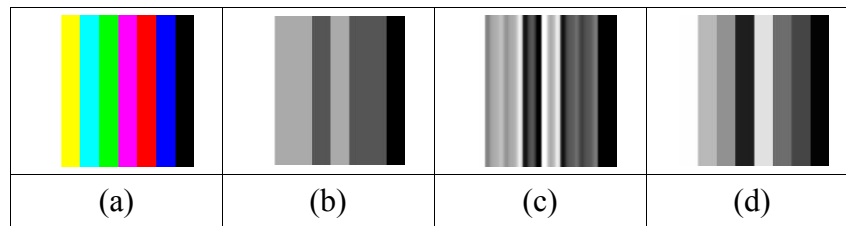
body detectors cannot distinguish the edges caused by shade/shadow from those caused by different materials, which in the assumption of the thesis will cause incorrect detections.

In colour theory, colour differences between materials indicate different values in hue space. Differences between highlighted areas and shade/shadow areas on the same material mean the same hue value but different luminance values in hue space. In the case of human detection, clothes and human skin often appear in several regions which have the same colour type but different luminance values. In order to weaken the shade/shadow edges and to enhance the edges between clothes/skin and backgrounds, pixels which have close hue values are grouped and assigned with the same colour value, which can solve the problem described in previous paragraph. This chapter proposed a novel colour-to-greyscale conversion method that is based on the density distribution in the RGB space, and can be applied not only in the proposed system, but also other image processing methods using one-channel input.

### **3.1 The weakness of previous methods**

Current colour-to-greyscale converting methods can be categorised into global and local methods. Global methods of prior art usually convert the RGB values to greyscale values in fixed converting functions, without considering the structures and colour distributions of the input images. In such methods the mapping performance may be poor sometimes, especially in complex images.

Local methods are developed to execute conversion via analysing the structure and colour distribution of every image in its own right. The advantage of local methods is local differences, especially those on edges, will be discovered and edges will be enhanced significantly. However, these methods often cause low-frequency distortion, as shown in Figure 3.1.



**Figure 3.1** The problem of colour bars

**(a) Original, (b) Luminance, (c) Spatial enhancement of luminance at chrominance edges, (d) The mechanism in this thesis.**

Figure 3.1(b) indicates that chroma edges will be lost in a conventional colour-to-luminance conversion. Figure 3.1(c) is the result of Bala and Eshchbach's method [87] for colour bars, which brings colour boundaries back but causes a distortion problem. The low-frequency areas which are in the same colour in the original picture, have luminance variation in the output image. The reason is that the algorithm is more sensitive on the high-frequency areas, which leads to variations near the edges.

Our method is global and makes analysis on the colour density distribution for every input image. It maps RGB values to greyscale values according to the density distributions of input images in RGB spaces. We execute the mapping process as follows: firstly, we group the colours of an image into a number of clusters in a RGB space. Secondly, we map the 3 dimensional centroids of the clusters to a one dimensional greyscale space. Finally, we calculate the greyscale value of each pixel by measuring its distances to each centroid in the RGB space.

## 3.2 Colour clustering

We choose the RGB space as the image processing space. There are a number of colour models for digital images, for example, three component models such as RGB, HSV and Lab [92], and four component models such as CMYK [93] [94]. RGB is the most commonly used colour model. Compared with Lab and HSV, which use the brightness (or sometimes called lightness) as a component, the three

dimensions of the RGB model are all colour components and the same weight can be used for each dimension to measure distances between different colours. Compared with CMYK, it is easier to process and display data based on the RGB model. We use sRGB [95] which is a standard RGB colour space used on monitors, printers and the Internet.

Mapping RGB values to greyscale values via a colour clustering process is a novel idea in the proposed method. For an input image, we build a 3D histogram, in which the value of each point in the 3D space stands for the number of pixels which have the RGB values corresponding to the points' coordinates. For example, if there are 3 pixels having the colour value (128,128,128), the point in the 3D histogram will be evaluated assigned to 3. Therefore the histogram reflects the density distribution of the colours of the input image, as the example shown in Figure 3.2. We aimed to develop a mechanism that merges the non-zero points around the high density regions to colour clusters.

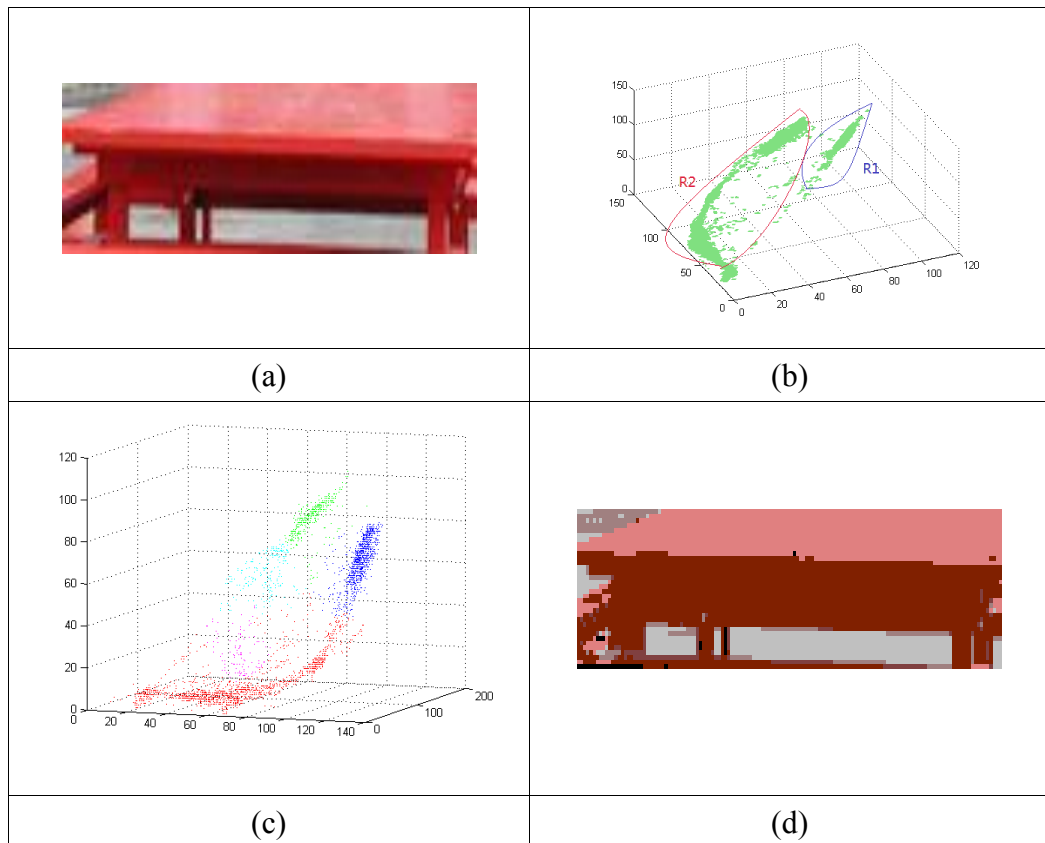
Figure 3.2 gives a typical example of how the pixels distribute in an RGB colour space. The high density region located along the diagonal corresponds to the grey pixels of the original image, and is denoted as  $R_1$ . The other high density region aside corresponds to the red pixels, and is denoted as  $R_2$ . The goal of the thesis is to merge the points near  $R_1$  and those near  $R_2$  into two different clusters. In practice, two clusters are not enough for the further process. So for in this particular case, we group the points into 5 clusters: dark grey, light grey, dark red, light red, and the others between the high density regions. The selection of the number of clusters is partly dependent on the image. We choose 5 clusters here due to the simple example in Figure 3.2. However, the final colour-to-greyscale result is not sensitive to the choice of the cluster number, which will be discussed in Section 3.3. We have implemented five different methods to group the points to clusters.

1. The first method is based on complete-linkage [96], a classical method that clusters points according to the distances between them, so the points will be grouped to a number of 3D regions.
2. The second method employs a widely applied algorithm --  $k$ -means clustering

and one of its improvements, *k*-means++.

3. The second method uses complete-linkage as a preliminary stage to group points to small clusters, and then merges the small clusters to large ones by employing spatial-density estimation.
4. The third method is based on getting separate clusters by smoothing and thresholding the whole RGB space, which can be completed by the following steps:
  - 1) Smooth the histogram with a 3D Gaussian filter to get a 3D colour density image;
  - 2) Segment the whole smoothed 3D histogram to a number of connected regions by employ a threshold.
  - 3) Select the *n* largest regions as the clusters and grow these by (3D) morphological opening until the clusters fill the entire 3D space and all colours are assigned to one of the clusters.
5. The fourth method has the same procedure as the method 2 except the second step. It employs a watershed algorithm to segment the histogram.

After grouping all the points to a number of clusters, we repaint the image in which each pixel is assigned by the average colour of its corresponding cluster. The average colour of a cluster corresponds to the centroid of the colour cluster in the RGB space. The result is shown in Figure 3.2(d), in which there are only 5 colours left.



**Figure 3.2 Clustering and remapping colours for an image**

**(a) Original image, (b) 3D histogram of colors in original, shown as a scatter diagram, (c) Classification of points in the scatter diagram following density-based colour clustering, (d) Original image recolored according to clusters found by the algorithm.**

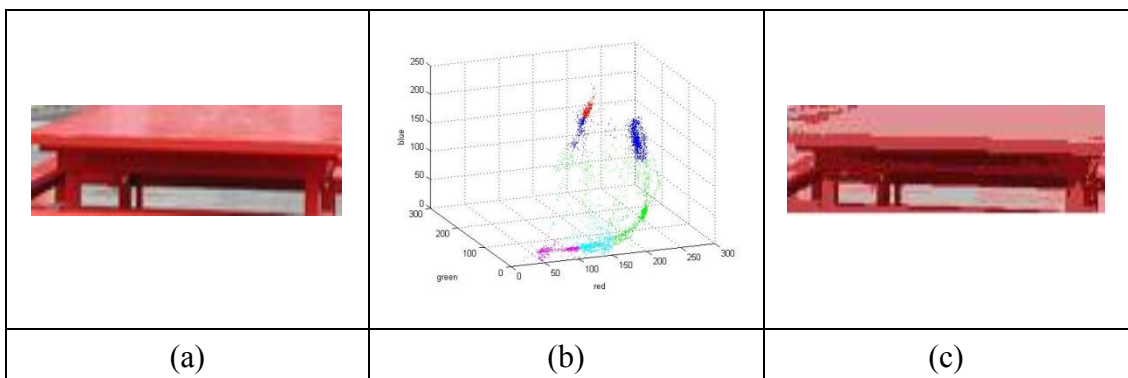
### 3.2.1 The method based on complete-linkage

This approach uses a classic clustering method called complete-linkage [96]. For a 3D space with  $N$  non-zero points (a non-zero point means this point corresponds to one or more pixels in the image), this method merges the points to clusters in the following steps:

- 1) Compute all of the distances between the non-zero points and make a distance table;
- 2) Merge the pair of points which have the smallest distance to a single point. For example point  $p_1$  and  $p_2$  is the closest pair in the table. Merge them and denote the new point as  $p_{12}$ ;

- 3) Considering a point  $p_i$  different from  $p_1$  and  $p_2$ , denote the distances to  $p_1$  and  $p_2$  as  $d_{1,i}$  and  $d_{2,i}$ , and set  $d_{12,i} = \max(d_{1,i}, d_{2,i})$  as the distance between  $p_i$  and  $p_{12}$ . Apply this to all of the points and updating the distance table;
- 4) Carry out Step 2 and 3 iteratively until only one point left.

After the complete-linkage process, a dendrogram that shows the grouping relationship of all the non-zero points is built up. Now the algorithm is designed to select a level of the dendrogram to divide the points into a number of clusters. The result of the clustering process is shown in Figure 3.3, in which 5 clusters are indicated.



**Figure 3.3 Colour clustering using complete-linkage**

Figure 3.3 illustrates that the side region that corresponds to the red pixels is divided to four clusters, which leads to red regions in different luminance in the output image. What makes things worse, there are some points corresponding to grey pixels that are grouped in to red clusters, which cause some grey pixels appearing in red in the output image.

### 3.2.2 The method based on $k$ -means clustering

K-means clustering [97] is a method that is commonly employed to cluster points in multiple dimensional spaces and measures clusters by distances and density.  $K$ -means clustering can be described as a course that minimises the within-cluster sum of squares:

$$\arg \min_{\mathbf{S}} \sum_{i=1}^k \sum_{x_j \in S_i} \|x_j - \mu_i\|^2 \quad (3.1)$$

where  $x_i$  are observations (data points),  $k$  is the number of sets  $\mathbf{S}=\{S_1, S_2, S_3, \dots, S_k\}$ ,  $\mu_i$  is the mean of points in  $S_i$ . The standard  $k$ -means algorithm is an iterative process between two steps [98]:

- 1) Assignment step: Assign each observation to the cluster whose mean is closest to it

$$S_i^{(t)} = \left\{ x_p : \|x_p - m_i^{(t)}\| \leq \|x_p - m_j^{(t)}\| \forall 1 \leq j \leq k \right\},$$

where  $t$  represents each iteration,  $m_i$  ( $i=1, 2, 3, \dots, k$ ) are the sets, and each  $x_p$  is assigned to exactly one  $S^{(t)}$ .

- 2) Update step: Compute the new means to be the centroids of the observations in the new clusters.

$$m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

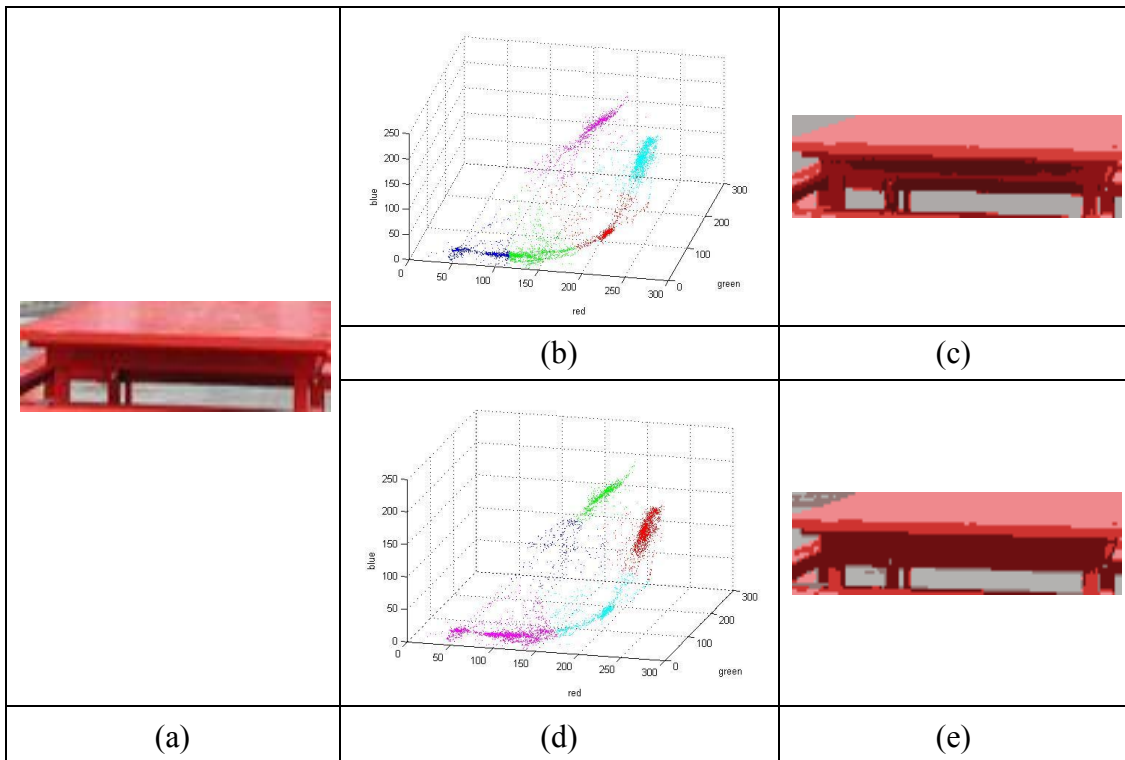
The algorithm keeps running until the assignments stop altering. Then the final assignments are used as the final clustering result.

In the conventional  $k$ -means algorithm, the starting points of each cluster (set) are randomly selected. Here we use an improved method named “ $k$ -means++” that optimally selects starting clusters [99]:

- 1) Select one centre uniformly and randomly from among the observations.
- 2) For each observation  $x$ , compute the distance between  $x$  and the nearest centre that was chosen in Step 1 (denoted as  $D(x)$ ).
- 3) Select one new observation randomly as a new centre, using a weighted probability distribution where a point  $x$  is chosen with probability proportional to  $D(x)^2$ .
- 4) Repeat Steps 2 and 3 until  $k$  centres have been chosen.
- 5) Execute the standard  $k$ -means clustering method with the chosen starting centres.



We use MATLAB's built-in function to implement the standard  $k$ -means algorithm and also apply a machine learning and data mining tool named WEKA to implement the  $k$ -means++ algorithm. The Euclidean distance is used to measure distances between points.



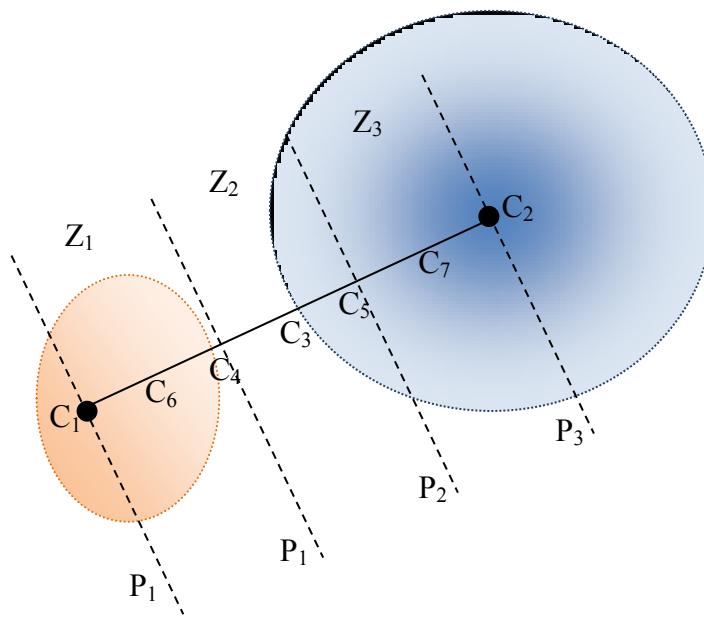
**Figure 3.4** Colour clustering using  $k$ -means and  $k$ -means++ clustering

Figure 3.4 illustrates that both  $k$ -means and  $k$ -means++ methods achieve clusters by measuring distances rather than by estimating density. For instance, the points representing red pixels are segmented into 4 clusters by  $k$ -means and 3 clusters by  $k$ -means++, although these points form a region with continuous density. However, the aim of this section is to cluster points representing the same colour type to one set, so the method is required to do clustering according to density information. Hence we have developed several methods instead of using existing algorithms.

### 3.2.3 The method employing spatial-density estimation

According to Figure 3.3, it is shown that complete-linkage can successfully group the points into small regions. But in the consideration of the entire space, it does not consider the distribution density of the points. An alternative way is to divide the

points to a relatively large number of small clusters first, and then to use a method called spatial-density estimation (SDE), which is a new method developed by us, to merge the points to a smaller number of large clusters. We still employ the complete-linkage method to build a dendrogram, and then group the points according to it. When there are a relatively small number of clusters left, such as 50 clusters, we pass the grouping result to SDE. To explain SDE more clearly, we present a simple case with 2 small clusters in a 2D view as shown in Figure 3.5.



**Figure 3.5 A 2D view of SDE**

In Figure 3.5, there are two groups of points: the blue group and the orange group. The colour depth of each group means density of points, while deeper colour represents higher density.  $C_1$  and  $C_2$  are the centroids of Group 1 and Group 2.  $C_4$  is the one-third point of  $C_1C_2$  and  $C_5$  is the two-thirds point of  $C_1C_2$ .  $C_3$ ,  $C_6$  and  $C_7$  are the middle points of  $C_4C_5$ ,  $C_1C_4$  and  $C_5C_2$  respectively.  $P_1$ ,  $P_2$ ,  $P_3$  and  $P_4$  are the planes perpendicular to  $C_1C_2$  at  $C_1$ ,  $C_2$ ,  $C_5$  and  $C_2$  respectively.  $Z_1$  is the space between  $P_1$  and  $P_2$ .  $Z_2$  is the space between  $P_2$  and  $P_3$ .  $Z_3$  is the space between  $P_3$  and  $P_4$ . To judge whether these two groups are in a same cluster or not, we attempt to estimate the densities of the non-zero points in  $Z_1$ ,  $Z_2$  and  $Z_3$  and compare them. We execute as follows:

- 1) Count the number of non-zero points in  $Z_1, Z_2$  and  $Z_3$ , and denote them as  $n_1, n_2$  and  $n_3$ ;
- 2) Calculate the average distances of the non-zero points in  $Z_1, Z_2$  and  $Z_3$  from  $C_6, C_3$  and  $C_7$  respectively and denote them as  $d_1, d_2$  and  $d_3$ ;
- 3) Compute the “densities” by

$$dens_i = \frac{n_i}{d_i^p} \quad (3.2)$$

where  $i=1, 2, 3, p$  is a positive power parameter. The reason why we use the average distance instead of space size is because it is very difficult to estimate the size of the space where the non-zero points distribute.

- 4) Compare  $dens_1, dens_2$  and  $dens_3$  with a weighting function to make the decision.

$$dens_2 \sim f(dens_1, dens_3) \quad (3.3)$$

where  $\sim$  means comparing. A simple example is: when the relationship between  $dens_2, dens_1$  and  $dens_3$  satisfies

$$\begin{cases} f(x_1, x_2) = \frac{1}{2}x_1 + \frac{1}{2}x_2 \\ dens_2 \geq f(dens_1, dens_3) \end{cases} \quad (3.4)$$

then the two clusters will be merged together.

We execute SDE for every two tiny groups so that all of the tiny groups will be merged into several large clusters. It can be noticed that the rule of comparing densities is the key issue to decide the number of clusters at last.

To estimate whether a point is between two planes or not, we use the equation which represents planes:

$$Ax + By + Cz + D = 0 \quad (3.5)$$

In the case in this thesis, the four planes mentioned above are all parallel, which

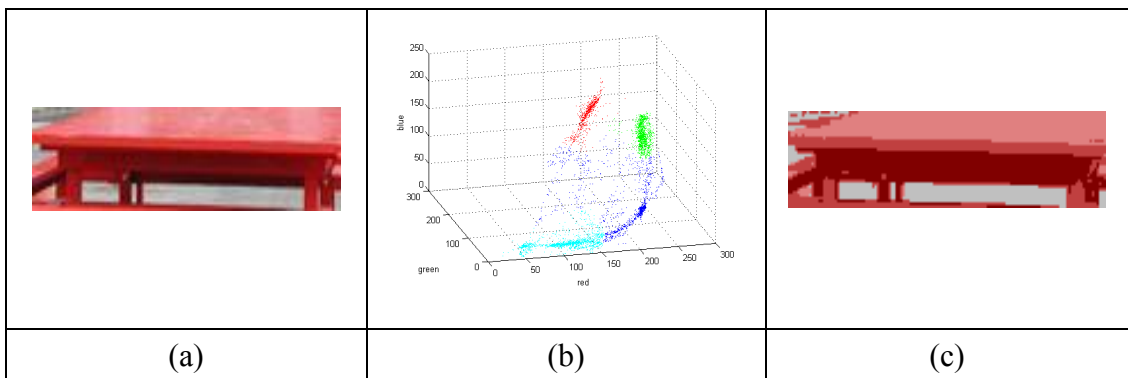
means any two of them (denoted as  $P_i, P_j$ ) can be represented as:

$$\begin{cases} Ax + By + Cz + D_i = 0 \\ Ax + By + Cz + D_j = 0 \end{cases} \quad (3.6)$$

Considering a point  $p_0$  with the coordinates  $(x_0, y_0, z_0)$ , let

$$\begin{cases} t_i = Ax_0 + By_0 + Cz_0 + D_i \\ t_j = Ax_0 + By_0 + Cz_0 + D_j \end{cases} \quad (3.7)$$

If  $p_0$  is at the position between  $P_i$  and  $P_j$ ,  $t_i$  and  $t_j$  will have contrary signs. In other words,  $t_i * t_j < 0$ . An example of SDE clustering method is shown in Figure 3.6.



**Figure 3.6 An SDE result**

According to Figure 3.6(b), SDE generates three clusters in the side region fewer than the method without SDE, which is what we want. Meanwhile, there is less interference between red and grey clusters, which leads to clear colour edges in the output.

### 3.2.4 The method based on smoothing and thresholding in 3D histogram scatter

The weakness of the two complete-linkage-based methods above, is that we have to build an  $N$  by  $N$  distance table for an  $N$ -pixel image to compare all of the distances every time when we merge the closest pair of points, and update the distance table. For a large image, this algorithm is much too expensive in computation and memory,

and sometimes the result cannot be achieved due to the limitation of memory. Therefore, we have developed other methods.

Different from the approach introduced in Section 3.1, this section introduces a method that uses smoothed 3D histogram scatters in RGB spaces. After making the 3D histogram scatter for a colour image, this approach has the following steps to group the points:

- 1) Smooth the scatter with a box filter to get non-zero regions;
- 2) Threshold the smoothed scatter to get a new scatter with several high-value connected regions;
- 3) Select the  $n$  largest connected regions as the cluster prototypes;
- 4) Group all the non-zero points to these prototypes;

In step (1), a Gaussian filter is applied. In practice, we use an  $N$  by  $N$  by  $N$  cube ( $N=3,5,7,9\dots$ ) as a box filter. After smoothing, the non-zero points will be linked to form non-zero regions. Most times at this stage, the connected regions are large while the number of them is small.

In the thresholding step, denoting a point in the scatter as  $p_{i,j,k}$  and the threshold as  $t$ , we process the scatter by:

$$p_{i,j,k} = \begin{cases} 1, & \text{if } p_{i,j,k} > t \\ 0, & \text{if } p_{i,j,k} \leq t \end{cases} \quad (3.8)$$

So the regions that have high densities will be left while the regions with low densities will be cleared to 0. As a result, there will be a larger number of smaller connected regions appearing in the scatter. Meanwhile, the largest connected regions in the scatter indicate the main colour clusters. The selection of the threshold is so important that it has a great influence on the clustering result. Considering that different sizes of images can cause different densities in the scatter, the selection of the threshold is usually designed to be adaptive according to image sizes. As the number of pixels in images increases, the values in corresponding histograms will increase. For images with the same content but different resolutions, there is a

proportional relationship between the values in the histogram and the number of pixels. So we set a linear relationship between thresholds and image sizes.

$$threshold = k * n + a \quad (3.9)$$

where  $k$  is the coefficient,  $n$  is the number of pixels and  $a$  is the offset.

In practice, there is a problem that as all of the non-zero points are set to one, we are not able to distinguish different connected regions as clusters. In order to represent every region by labelling them with different numbers, the painter's algorithm [100] is used here, which processes as follows:

- 1) Set the cluster number to 1, scan the scatter point by point;
- 2) If the value of the current pixel equals to 1, cluster number increases 1;
- 3) If the value of the current pixel equals to 1, set it to the scatter number, and the number of points in the current cluster increases 1;
- 4) Check the neighbour (upper, under, left, right, front, back) points of the current point, and return to Step 3;
- 5) Finish when the last point reached.

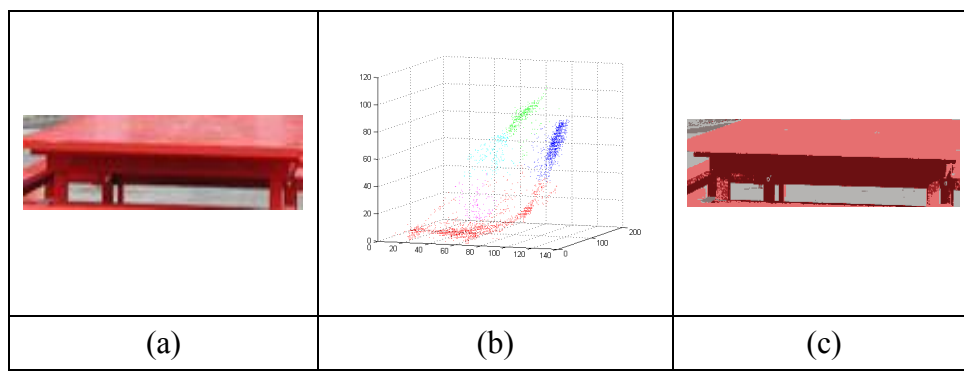
Now the number of the clusters and the number of points in every cluster has been obtained. If  $N$  clusters are required, the top  $N$  largest regions will be selected. Then it is necessary to classify all of the points corresponding to the colour image pixels into their nearest clusters. In order to work this out quickly, we implement an algorithm called "region expanding". The region expanding algorithm keeps expanding each selected cluster (connected region) at the same time in the 3D space until the connected regions reach the boundary and contact each other, so that the 3D space will be split into several sub-regions corresponding to the selected clusters and each point in the space can be classified to one of the cluster. The process is as follows:

- 1) Define a finishing marker  $f$ , set  $f$  to 1 at the beginning, and then scan the scatter point by point from the (0,0,0) point;
- 2) If the value of the current point equals to 0, set  $f$  to 0;
- 3) Check the neighbour (upper, under, left, right, front, back) points of the

current point, if any of them does not equals to 0, set the current point equalling to that neighbour;

- 4) When the last point is reached, if  $f$  equals to 0, return to Step 1 and repeat the process, else finish the process.

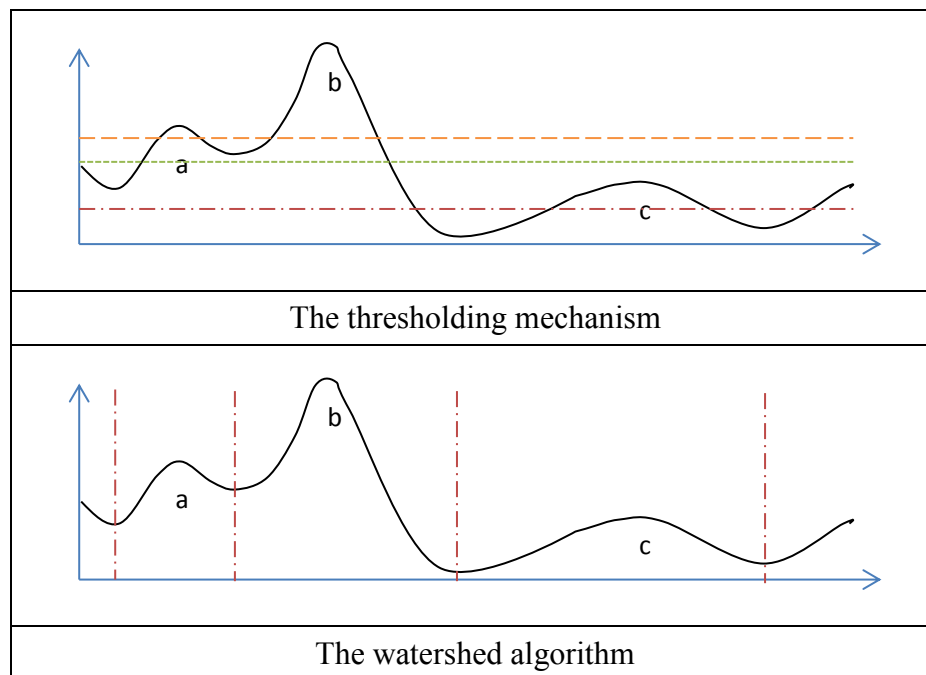
All of the non-zero points will be included in the selected clusters when the region expanding algorithm finishes. The work left is just simply representing the pixels corresponding to the points of each cluster with the cluster's average colour. The result of whole process in 2.2 is shown in Figure 3.7.



**Figure 3.7 The process of colour clustering using smoothing and filtering**

### 3.2.5 The method based on smoothing and watershed algorithm in 3D histogram scatter

This method just uses the watershed algorithm [101] instead of the simple thresholding mechanism to find the connected regions. To explain the difference between these two methods, a comparison is made in Figure 3.8.



**Figure 3.8 Comparison of the thresholding mechanism and the watershed algorithm**

Figure 3.8(a) indicates the thresholding mechanism with three different thresholds, which are represented by the horizontal dotted lines. The regions over the threshold will be considered as separate clusters. The problem of this method can be easily noticed. When we select a high-value threshold, such as the orange dashed line, region *c* which has many points but a low peak value, will be ignored. And in the further steps the points in region *c* will possibly be grouped to region *b* (cluster *b*). However we design to cluster region *c* as a separate region/cluster, so we have to decrease the value of the threshold. When the threshold goes down to a low value (the red dash-dotted line), region *c* becomes a separated cluster. However region *a* and *b* are grouped into the same cluster, which is also not what we want. This kind of paradox happens quite often when applying the thresholding mechanism. In addition, the threshold varies for different images. That is to say, we also need to develop a reliable method to find an appropriate threshold for every image even if there is no such paradox described above.

In contrast, the watershed algorithm can solve the problem mentioned above well. The vertical red dash-dotted lines in Figure 3.8(b) represent the watershed locations,



which are the boundaries between clusters. It is noticeable that the points in region a, b and c are all grouped separately. Meanwhile, the watershed algorithm does not require the selection of thresholds, which is an important but difficult issue in the thresholding process.

Now consider a 3D histogram scatter. To achieve the boundaries as Figure 3.8(b) in the case of this thesis, we invert the smooth 3D scatter first. So that the peaks will become troughs and the troughs will become peaks. After we process the inverted scatter via the watershed algorithm [101], watersheds will be found. These watersheds are the boundaries of the high density regions after the processed scatter has been inverted back. We employed the watershed algorithm illuminated by Meyer, which is the most common method. According to the paper [101], the watershed line of function  $f$  is defined as:

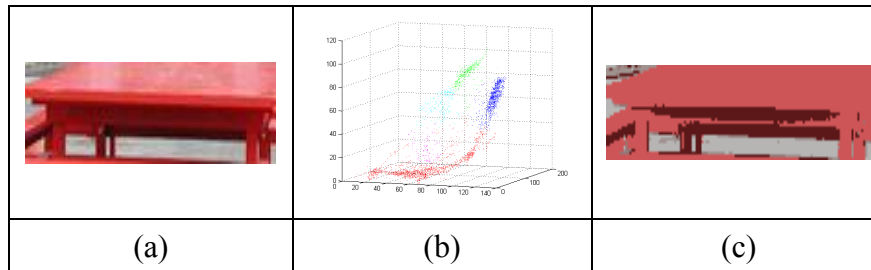
$$\text{Wsh}(f) = \text{supp}(f) \cap \left[ \bigcup_i (\text{CB}(m_i)) \right]^c \quad (3.10)$$

The theory of watershed algorithm is based on the knowledge of topography.  $\text{CB}(m_i)$  is defined as the catchment basin, and  $\text{supp}(f)$  is the support. In practice, we implement the procedure as follows:

- 1) A set of markers, pixels where the flooding shall start, are chosen. Each is given a different label.
- 2) The neighbouring pixels of each marked area are inserted into a priority queue with a priority level corresponding to the grey level of the pixel.
- 3) The pixel with the highest priority level is extracted from the priority queue. If the neighbours of the extracted pixel that have already been labelled all have the same label, then the pixel is labelled with their label. All non-marked neighbours that are not yet in the priority queue are put into the priority queue.
- 4) Redo step 3 until the priority queue is empty.

After we get the connected regions by using watershed algorithm, we cluster all the points corresponding to the pixels via the region expanding mechanism described in

Section 3.2.2. Figure 3.9 is the result produced by the method based on watershed algorithm.






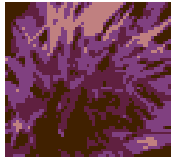
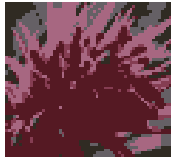
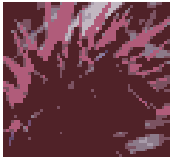






**Figure 3.9 The result of using the watershed algorithm.**

It can be noticed that the side region is only divided to two clusters, and there is almost no interference between red and grey clusters. Comparing with Figure 3.7, more regions are clustered to light red instead of dark red. It is difficult to say which is “correct” or “better”. However, since the method using watershed algorithm has the superiority of threshold-free, it is employed in the final approach.

### 3.2.6 The result comparison of three clustering methods

The advantages and disadvantages of these three clustering methods have already been described above. Now a comparison of the clustering results for a number of images can be made. As the complete-linkage-based method is limited by computer’s memory, we use small-sized input images. The results are shown in Figure 3.10.

	Original image	SDE-based	Threshold-based	Watershed-based
0				
1				
2				

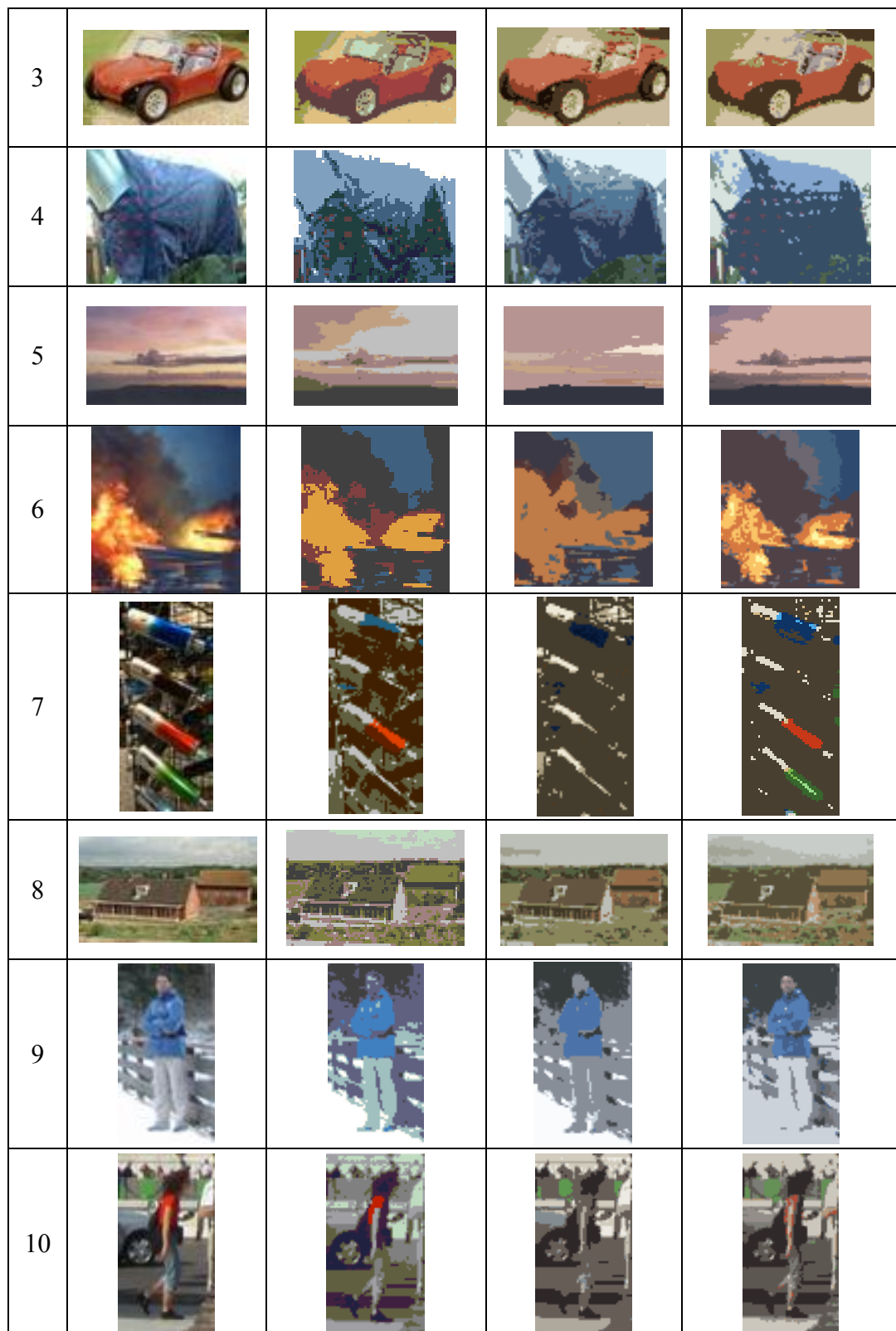


Figure 3.10 Re-mapped by means of colour clusters

In Figure 3.10, sample images can be divided into two types. Samples such as

Picture 0, 1, 2, 5, 6, 9 have two types of colours but have significant luminance variation. Others have more colour types. Picture 4 which has small purple dots in a blue region, and Picture 7 which has five different types of colours, are difficult to classify all of the colours correctly. Figure 3.10 indicates that watershed algorithm classifies colours better than the other two. For Picture 7, watershed algorithm finds the green colour but the other two cannot. For Picture 10, watershed also classifies green and red better than the others. However, there is an incorrect classification for Picture 3, in which a red region on the car is classified to green by watershed algorithm. The SDE-based method performs not fairly for some images. However this method needs to build a complete-linkage dendrogram, which cases a big computational cost for a large image. To design a practical approach, we choose the watershed algorithm.

### 3.3 Multidimensional scaling of cluster prototypes

After achieving the clusters of the points in the RGB space, we then calculate the centroid of each cluster by:

$$C = \frac{1}{n} \sum_{i=1}^n p_i \quad (3.11)$$

where  $C$  stands for the location of the centroid of a cluster,  $p_i$  is the coordinates of one of the points in the cluster, and  $n$  denotes the number of the points in the cluster. We define a cluster prototype which combines the centroid location and the number of the pixels (note it is not the numbers of the points because a point may be mapped by several pixels).

These prototypes are then represented as vertices in a graph with the edges representing the 3D distances between cluster prototypes weighted by the numbers of points in each cluster. The graph may be visualized as a 3D graph that represents the cluster centroids perfectly: irrespective of the numbers of points in each cluster,

the graph's minimum energy state is that of the original geometric distribution of points. We now use multidimensional scaling to determine the lowest energy 2D or 1D graph that represents the cluster distribution. Lowest-energy here corresponds to minimization of Kruskal's stress [102] [103], which is defined as

$$\sigma(X) = \sum_{i < j \leq n} w_{ij} (d_{ij}(X) - \delta_{ij})^2 \quad (3.12)$$

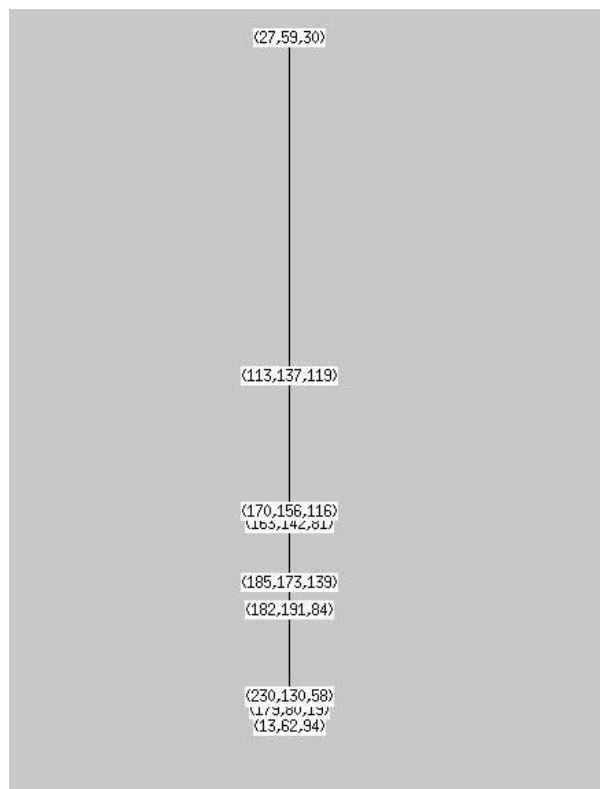
In the formula above  $w_{ij} \geq 0$  indicates a weight between a pair of nodes  $i$  and  $j$ , and in the case of this section the weight of a pair of nodes is the product of the point numbers of the two clusters.  $d_{ij}(X)$  is the Euclidean Distance between these two nodes.  $\delta_{ij}$  is the ideal distance between the nodes in the  $m$ -dimensional data space, and here is the distance between two centroids in the RGB space. The entire multidimensional algorithm, has the following steps [103]:

- 1) For a spring system with  $N$  nodes, compute the distances (or the inversed similarities) between each two nodes, thus  $N(N-1)$  distances totally. Order these distances from the largest to the smallest.
- 2) Use a  $\delta_{ij}$  to satisfy Equation (3.12) and to minimize the stress.
- 3) Move the nodes according to  $\delta_{ij}$  and build a new system configuration. So that distances and stress are updated, and  $\delta_{ij}$  is re-determined.
- 4) Go back to step (2) and repeat the process until reach a minimum stress and the system will stay at a dynamic stable state.

In practice we treat every connection between two nodes as a "spring" which has "forces" to the nodes. The "forces" come from the stresses between every two nodes. The nodes are moved by the "forces" and the entire system will be finally led to a dynamic stable state due to the energy minimising law. A 2D spring graph is useful for visualising the effects of cluster size and placement, but the 1D version yields a mapping of the prototypes to 1D which is then applied to convert to grayscale. In the case of the thesis we use the locations of the centroids as the positions of the nodes, and the numbers of the points in the clusters as the weights. The distance of a pair of clusters is defined as the Euclidean Distance between their centroids in the RGB space. The weight of their distance is the product of the weights of the two clusters.

So the system can compute the average stress of a set of prototypes.

We build up a 1D spring graph (the visualisation of a spring system) with the nodes initialised at random positions, as the solution of multidimensional scaling and average stress/energy minimising [103]. A typical 1D spring graph is shown in Figure 3.11. In Figure 3.11, a node is displayed by three numbers which are the coordinates of the centroid of a cluster prototype. For example, the node (27,59,30) corresponds to the cluster prototype of which the point (27,59,30) is the centroid. The vertical line stands for the greyscale from 0 in the top to 255 in the bottom, and the position of each node indicates its mapping value to the greyscale. When the spring system reaches a stable statue, which means the energy of the system is minimum, the positions of the nodes will be relatively static. Then the greyscale mapping will be determined according to the positions of the nodes.



**Figure 3.11** An example of spring graphs

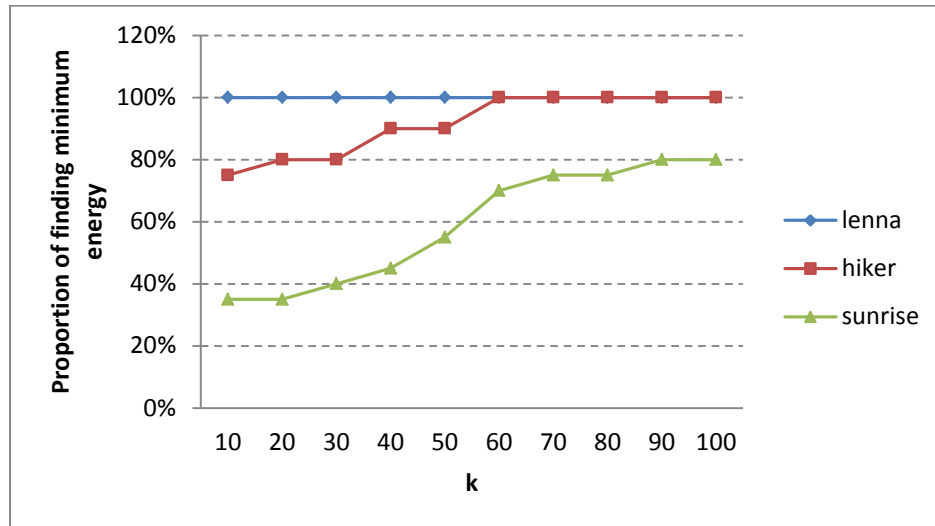
Here it is appropriate to discuss the reason why the final colour-to-greyscale result is not sensitive to the selection of cluster number  $n$ . According to Figure 3.11, a spring

system can drive the nodes whose centroids are near each other in the RGB space to close positions in the greyscale space. When the number of clusters is more than the number of colour types, the points of the same colour type will probably be clustered into two or more sets which have near centroids. However, the spring system will map their corresponding nodes to near positions. Since the procedure of mapping each pixel to greyscale is an inverse distance weighting function (see Section 3.4), the influence on the final result is small. Of course users can manually select the number of clusters for each image to achieve a best result for different situations and applications. However selecting an appropriate number of clusters can satisfy most images. The number can neither be too small, otherwise points with significant colours will be merged; nor be too large, otherwise the final result will be similar to the luminance image. Hence we use the number of clusters  $N=10$  for all experiments.

However, the spring graph method achieves only local minimisation. As we initialise the 1D spring graph with random node positions, sometimes a spring graph will reach a stable statue that only achieves a local minima instead of the global one. To work out this problem and get the global minima, we employ an iterative hill-climbing process to determine the minimum-energy configuration. We start this process by randomly initialise  $n$  spring graph instances, with one of them being the foreground instance and others being background instances. All of the instances are running in parallel and independently. When one of the background instances achieves a stress smaller than that of the foreground instance, we replace the foreground instance by the background one. Meanwhile a new background instance is started. An existing background instance will be killed and a new one will be started if the existing one cannot improve the stress after a duration  $t$ . This process is repeated and the entire process terminates when  $k$  instances in a row do not improve on the currently found minimum energy. In practice we set  $n=2$  and  $t=2000$  ( $t$  is an iteration number in the program).

We have run the algorithm on many different images for a number of times, assessing the ability of the multiple-start mechanism to find a global optimum. Figure 3.12 below shows the frequency of finding the minimum energy associated with a particular 3D-to-1D conversion given  $k$  for a typical image (Lenna, the blue

line), a moderately difficult image (Hiker, the red line) and for the most difficult image (Sunrise, the green line).



**Figure 3.12 Proportion of runs finding minimum energy solution**

The diagram shows that with  $k = 100$ , the probability of finding the minimum is at least 80%. This means that if we run the program 10 times for the most difficult image (Sunrise), it will be a probability of 0.9999999 ( $1 - 0.2^{10}$ ) to achieve a minimum stress.

### 3.4 Mapping of all colours to greyscale

Having got the optimum distribution of the prototypes by energy minimisation, the algorithm can now map the colour values to greyscale. For each pixel, we measure the Euclidean Distance to every prototype in the RGB space, and then use Shepard's method [104] to computer its greyscale value. Shepard's method, which is an inverse distance weighting algorithm, converts colour values to greyscale as:



$$u(\mathbf{x}) = \frac{\sum_{i=0}^N w_i(\mathbf{x})u_i}{\sum_{j=0}^N w_j(\mathbf{x})} \quad (3.13)$$

Where

$$w_i(\mathbf{x}) = \frac{1}{d(\mathbf{x}, \mathbf{x}_i)^p} \quad (3.14)$$

is an inverse distance weighting function. The function  $d(\mathbf{x}, \mathbf{x}_i)^p$  is the distance between the pixel and the prototype we have computed, and  $p$  is the power parameter which is a positive real number.

In practice, the method described above will bring a serious problem. In some cases, false edges will appear due to some pixels which should have high values but have been set very low values after the mapping process. We argue that the problem is pixels that are outside the hull of nodes so their values are being extrapolated rather than interpolated. Adding nodes  $(0,0,0)$  and  $(255,255,255)$  insures that all values interpolated. So we have got  $N+2$  cluster prototypes in practice.

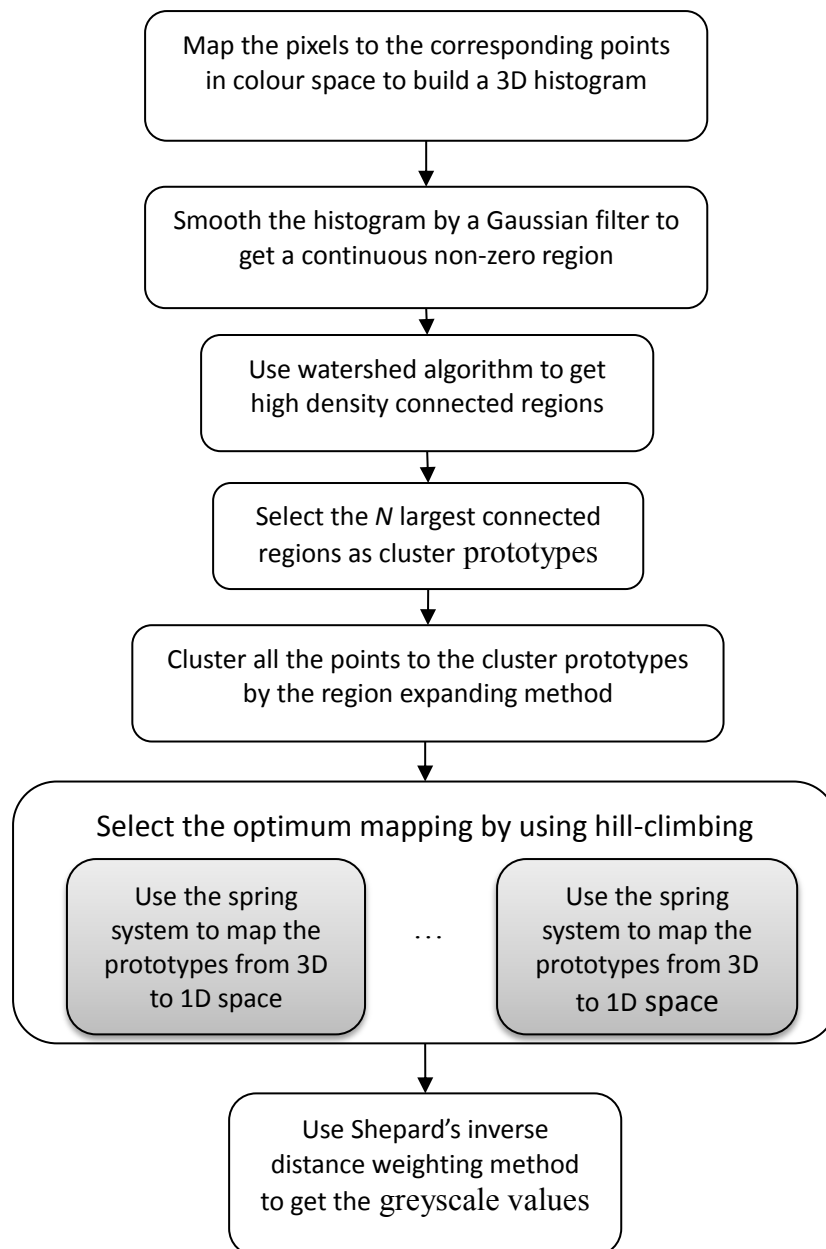


Figure 3.13 Flowchart of the entire procedure

## 3.5 Comparison against other methods

### 3.5.1 Qualitative performance

We have tested the proposed method by applying it to a number of images. The final

system is based on watershed method, and we call it Density-based colour to greyscale method. The whole colour-to-greyscale process is indicated by Figure 3.13.

A simplified approach called Colour-quantisation colour to greyscale method is also implemented for comparison. This method executes multidimensional mapping in the same way as Density-based method, but it just clusters the points in an RGB space to a number of fixed prototypes instead of watershed-based clustering. For example, it divides an RGB space, which is a cube with the scale from 0 to 255 in each dimension, to  $5 \times 5 \times 5$  regions, and each region is considered as a prototype. The centre of each region is defined as the location of the corresponding prototype. The numbers of non-zero points inside the regions are the point numbers of the prototypes.

We use luminance images and Color2gray [88] result as comparisons. We also add the result produced by Colour-quantisation (Colour-quant), i.e. the method just described. We extract the Canny edge of the result images to get a clear view. The proposed method that is superior to Color2gray int that it can process large sized images. So for the proposed method, we get greyscale images in large sizes at first, then scale it to the same size as the images used in Color2gray, and finally extract the edges. The comparison is shown in Figure 3.14. In Figure 3.14, the pictures “Lenna”, “Baboon” and are classic testing images, while “Sunrise” is a typical image that can reflect the performance of a method: simply taking the luminance for colour-to-greyscale conversion will case losses of chroma edges in this picture. The picture “Cells” contains lots of edges, “Hiker” has a very complicated background and “Walkers” has close colours on the door, the wall and the lady’s hair.

Here is the parameter configuration for each method:

Luminance: no parameters needed.

Color2gray: this method is parameter-dependent. We set its parameters  $\theta=45^\circ$ ,  $\alpha=10.0$ ,  $\mu=\text{full neighbour}$  (see the definition of its parameters in [88]).

Colour-quant: we set the division of RGB cubes to 5x5x5 regions.

- 1) Density-based: we set the number of selected colour clusters  $n=10$ , the number of hill-climbing candidates (background instances)  $hc=1$ , and in the phase of inverse distance weighting, we set the power parameter  $p=2$ .
- 2) According to the result shown in Figure 3.14, a summary can be made as follows:
- 3) For the image “Baboon”, the density-based method achieves the result with clear edges between hairs in different colours. Other methods obviously did worse than Density-based method.
- 4) Our method produces qualitative results on a par with the state of the art without human intervention. In particular, we see on the “Sunrise” image, that the equal-luminance sun/sky difference is preserved at least as well as Color2gray, and significantly better than the other two. Arguably the rendering of the colour blindness test is closer to that of normal colour vision than any of the alternatives.
- 5) Because of the single choice of the number of clusters  $N$ , images with fewer objects are over segmented by our method. This is evident in the “Hiker” image where the green grass has been divided into two clusters resulting in a busy edge image. On the other hand we note that the outlines of the hiker’s body are best defined in the density-based edge image, so this would be suitable as input to a subject stage of person detection and pose estimation.
- 6) The picture “Walkers” illustrates a case where doing without spatial information harms the density-based method: the woman’s hair is rendered close in grey value to the door behind her and so the edge is attenuated.

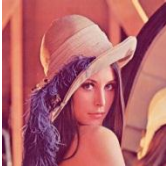
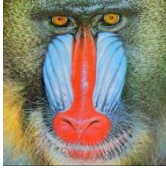


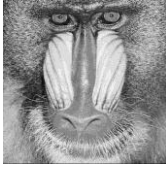



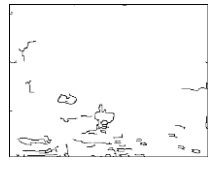

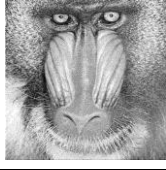


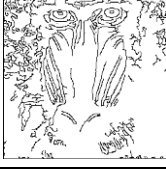


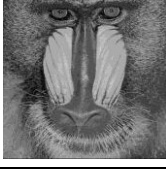



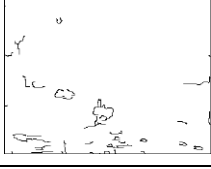

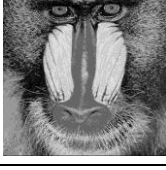



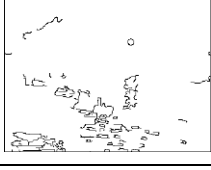
	Lenna	Baboon	Sunrise
Input			
Original grayscale			
Original edge			
Color2gray grayscale			
Color2gray edge			
Colour-quant grayscale			
Colour- quant edge			
Density-based grayscale			
Density-based edge			

Figure 3.14 Comparison between our method and others

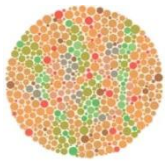


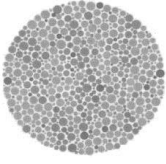


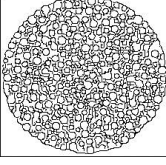
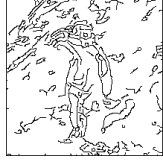

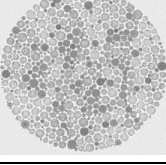


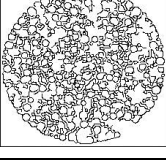
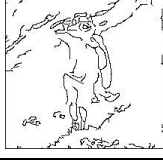
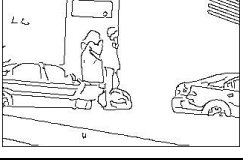
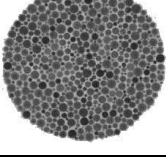


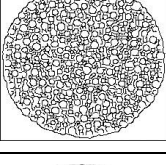
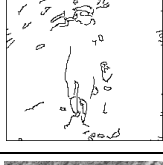

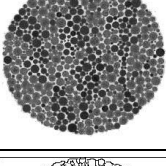


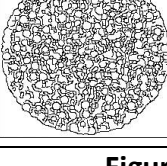
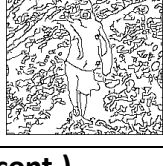
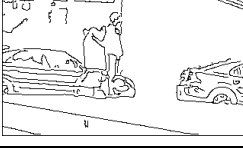
	Cells	Hiker	Walkers
Input			
Original grayscale			
Original edge			
Color2gray grayscale			
Color2gray edge			
Colour-quant grayscale			
Colour- quant edge			
Density-based grayscale			
Density-based edge			

Figure 3.14 (cont.)

### 3.5.2 Quantitative performance

We compare our method with Ren's method [105], which combines edges of luminance image (G-image) and a transformed image (R-image) which has enhanced colour information. In the beginning, the method computes differences between colour channels and produces a D-image:

$$D(i, j) = \omega_1 |r(i, j) - g(i, j)| + \omega_2 |r(i, j) - b(i, j)| + \omega_3 |g(i, j) - b(i, j)| \quad (3.15)$$

Where  $r, g, b$  represent the three channels of an input image, and  $(i, j)$  is the location (coordinate) of a pixel.  $\omega_1, \omega_2$  and  $\omega_3$  represent the weights for the differences, which are set to an equal value in Ren's experiment. An R-image is a smoothed colour image created by combining the corresponding G-image and D-image:

$$R(i, j) = k \frac{\omega_d G(i, j) + \omega_g D(i, j)}{\omega_d + \omega_g} \quad (3.16)$$

where  $k$  is a coefficient that scales intensity values in R-images within  $[0, 255]$ , and  $\omega_g, \omega_d$  are the weights defined as:

$$\begin{aligned} \omega_d &= 1.5 \times \text{Range}(D) + \sigma(D) \\ \omega_g &= 1.5 \times \text{Range}(G) + \sigma(G) \end{aligned} \quad (3.17)$$

where the function  $\text{Range}()$  returns the intensity range of an image, and  $\sigma()$  obtains standard deviations.

After obtaining transform images, Ren employed a Canny operator implemented by OpenCV to extract edges. There are two threshold parameters required:  $T_h$  which has a larger value and is used to find initial segments of strong edges, and  $T_l$  which is smaller and is used for edge linking. Ren proposed a mechanism to automatically determine  $T_h$  and  $T_l$ :

$$\begin{aligned}
 T_h &= \mu + \max\left(\frac{\mu}{2}, \frac{\mu}{\sigma}\right) \\
 T_l &= \frac{|\mu - \sigma|}{2}
 \end{aligned}
 \tag{3.18}$$

where  $\mu$  and  $\sigma$  are the mean and standard deviation respectively of an input greyscale image. Ren extracted edges from the G-image and R-image, and output their combination as the result:

$$E_f = E_G \cup E_R \tag{3.19}$$

To analyse the performance of his algorithm, Ren used a set of ground truth images which were obtained from the testing images by the following process:

- 1) An RGB test image was converted to the  $YC_bC_r$  space.
- 2) Edge images were determined from both the RGB and  $YC_bC_r$  spaces. To extract edges for a 3-channel image, Ren applied Canny operators on each channel separately with thresholds determined automatically, and then output the OR combination of the three edge images as the final result.
- 3) The RGB edges and  $YC_bC_r$  edges were OR-ed together to achieve a union image of edges.
- 4) Finally the edge image obtained in Stage 3 was modified manually to remove noise generated in the processes above.

To evaluate the performance of edge detection, Ren compared the output edge images with their corresponding ground truth pixel by pixel and accumulated the numbers of true and false edge pixels. Finally the pair of totals of true and false edge pixels was used to evaluate performance.

Ren's ground truth images are no longer available (Ren [106]), so we have replicated them following the above procedure. We then follow the same procedure for performance evaluation. We partly disagree with the subjective decisions made in step 4) in Ren's paper, (e.g. missing edges of the reflecting regions in 'pepper'), but we have tried to reproduce their images faithfully to allow fair comparison. The



ground truth images are shown in Figure 3.14.

In the colour-to-greyscale phase for our method, we set the cluster number to 10, and execute the spring process 10 times to achieve a global minima for each image.

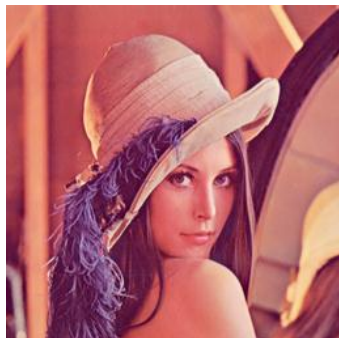


We have also implemented a simple but commonly used colour edge extracting method: on each pixel, the channel having the maximum gradient is used to extract edges. Color2gray is also inspected in this section. The final results are unions combined by the edges produced by the two methods above and luminance edges. The high and low thresholds for the Canny operator are automatically generated by Ren's method (Equation 3.18).

To evaluate the comparison results, we use Dice's coefficient [107] which is a similarity measure over sets  $X$  and  $Y$ :

$$s = \frac{2|X \cap Y|}{|X| + |Y|} \quad (3.20)$$

where  $s$  stands for the similarity.

In the experiment we find that our method achieves smaller numbers of both true and false edge pixels than those by Ren's method: slightly fewer true edge pixels against significant fewer false ones, see Table 3.1.

	Lenna	Pepper	House
A			



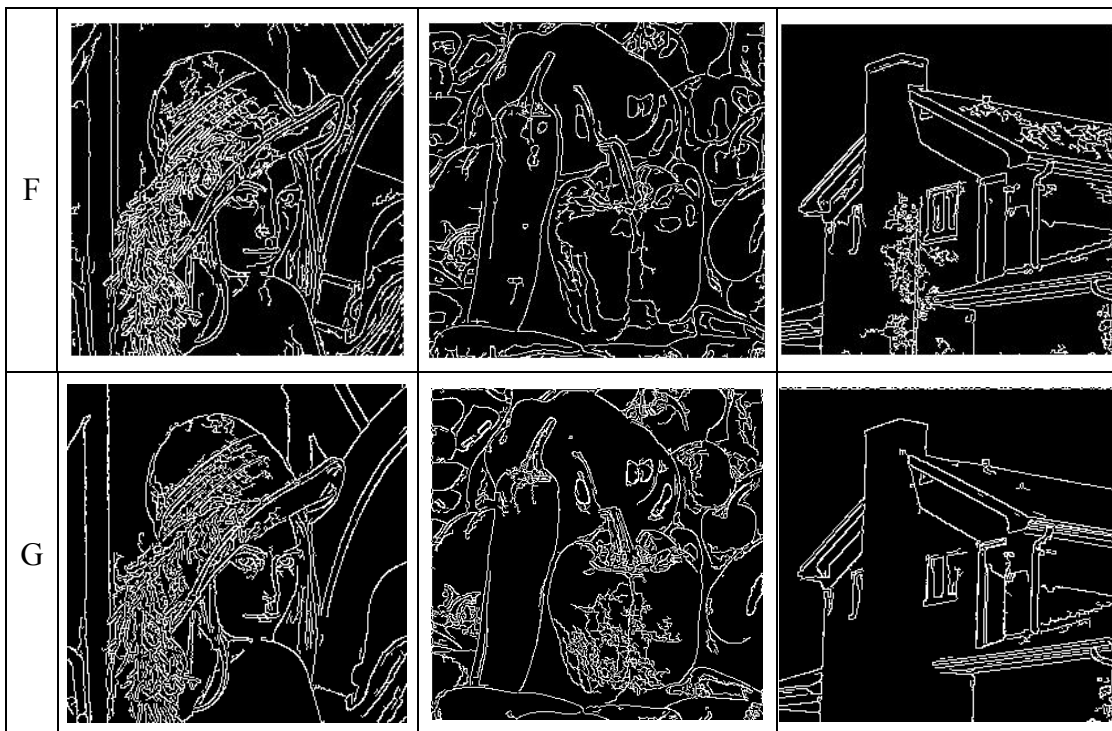


Figure 3.15 The comparison between our method and Ren's method

A: source images. B: ground truth images. C: Ren's edge images. D: our edge images with automatic thresholds. E: "thinned" edges. F: edges using the maximum gradients on each pixel. G: Color2gray edge images.

	lenna			pepper			house		
	true	false	<i>s</i>	true	false	<i>s</i>	true	false	<i>s</i>
Ren's	8141	2847	0.7650	7971	6852	0.6217	3895	2224	0.7198
Ours	8027	3000	0.7529	7906	5723	0.6468	3879	1532	0.7658
Color2gray	7680	2289	0.7579	7022	5905	0.5915	3530	1346	0.7357

Table 3.1 The comparison of results

First of all we reject the method that produces Figure 3.15F, since that there are heavy noise and disconnected edges appearing in the three images of Figure 3.15F.

Figure 3.15C shows that that Ren's method produces edge images with more noise than ours. Meanwhile, Ren's method misses some significant edges, such as the edges in the right corner of 'lenna' and the edges of the reflection regions in 'pepper'. Our method does better on these regions. Color2gray produces "clean" edge images for "lenna" and "house" at the cost of losing some edges, but performs the worst for "pepper" – making lots of noise and significantly losing some edges.

Considering the similarities with the ground truth images, according to Table 3.1, our method achieves the highest value for “pepper” and “house”. Here is a fact that our method can achieve similar numbers of true edge pixels with significant fewer false edge pixels for all of these three images but Ren’s method does find slightly more ‘true’ edges than us. We argue this is due to thick edges in the ground truth images. The ‘ground truth’ are produced by extracting edges in both RGB and  $YCbCr$  spaces, combining all edges with an OR operator. For a significant edge in an input image, corresponding edges in different channels are not always at exactly the same location, which leads to a thick edge in the union output. So we produce a new set of ground truth images whose edges are thinned from the ground truth images used above, see Figure 3.15E. The thinning is done by removing pixels from thick edges to leave the best single-pixel wide edges. This is a subjective judgement as was the original ground-truth creation process, but we argue that one-pixel-thick edges are a more appropriate representation of ground truth than the thick edges in Ren’s paper. Against these baseline images the performance is indicated by Table 3.2:

	lenna			pepper			house		
	true	false	$s$	true	false	$s$	true	false	$s$
Ren’s	7579	3409	0.7597	6984	7839	0.5997	3696	2407	0.7227
Ours	7760	3267	0.7763	7855	5744	0.7109	3854	1557	0.8083
Color2gray	7415	2554	0.7832	6676	6251	0.6240	3429	1447	0.7619

**Table 3.2 Evaluation results by using the thinned ground truth**

In Table 3.2, we see that the true detections fall comparing with Table 3.1. However the decrease is less for our method. These results show that Ren’s method obtains thicker edges but fewer edge details. Meanwhile, Ren’s method produces much more noise as shown by its large false detection number. Color2gray achieves the highest similarity with the ground truth for “lenna” this time, but worse than our method for both “pepper” and “house”. Meanwhile, Color2gray extracts the fewest true edges pixels among the three methods.

### 3.6 Conclusions

In this chapter we have introduced a colour edge extraction method that uses density-based clustering and colour-to-greyscale conversion. This new global method avoids the problem of low-frequency regions and generates greyscale images that accurately conserve the edge/region structure of the colour originals. Compared with Color2gray, it performs comparably to prior art without operator intervention in a relatively low computational expense. Compared with Ren's method, it produces more edge details with lower noise.

Our method ignores some edges between two regions which have similar colours but differ by luminance. However in some situations luminance edges are required, as the case of the last image in Figure 3.14. Although we do not intend to introduce spatial filtering for the reasons discussed in Section 1, spatial activity can be designed drive the selection of parameter values, in particular  $n$  the number of clusters.

Although this method is initially designed for improve the performance of the gradient-based human detection method, it also provides a solution of minimising loss of colour edge information when converting RGB images to greyscale. For any image processing method which depends on edges and requires one-channel input, if the used data is RGB images, then our conversion method can be applied to avoid losing colour edges.

# Chapter 4

## Human Detection

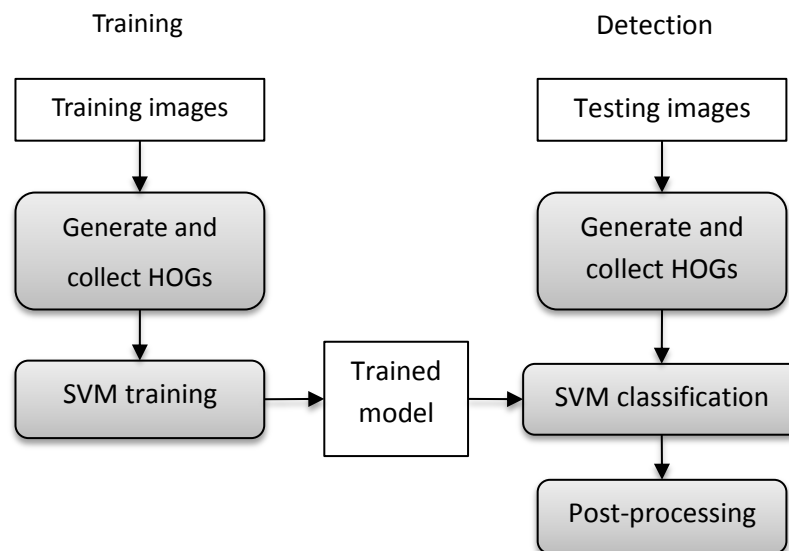
---

Human detection is the beginning phase of human motion analysis. A complete system of human motion analysis needs an accurate human detector to locate human bodies as the input of the following processing phases [25] [90]. Histogram of orientated gradients (HOG for short) for human detection, which was firstly proposed by Dalal [18] [108], is a popular method in the area of human detection in computer vision. Dalal designed an HOG training and detection scheme that achieves a relatively high true detection rate, compared with using other methods, such as Viola and Jones' method that employs Haar-like features and adaBoost classifier, which will be introduced in Section 4.3. Due to the good performance of HOG, we applied an HOG detector in the proposed system of the thesis. However, the original HOG scheme is not able to reduce its false detection rate to a practical level (see Section 4.1.4). Consequently, we made some modifications on the original scheme, including using extra training images and employing our colour-to-greyscale conversion method, to achieve a lower false detection rate and meanwhile to keep the true detection rate.

Section 4.1 introduces the theory and the original processing scheme of HOG. Section 4.2 illustrates how we add extra training data and shows the experimental results compared with the original one. Section 4.3 introduces procedures of employing our colour-to-greyscale conversion as pre-processing. Section 4.4 compares the improved HOG approach with the approach using Haar-like features and AdaBoost.

## 4.1 Histogram of Orientated Gradients

Like other machine-learning-based detection methods, in the training phase, the HOG method builds detectors by collecting image features and employing a classifier to learn statistical models. In the detection phase, it classifies testing windows into human or non-human according to trained models. Histograms of orientated gradients (HOGs) are extracted from input images as features and a linear support vector machine (SVM) [12] is selected as the classifier. The entire HOG training and detection structure is shown in Figure 4.1:

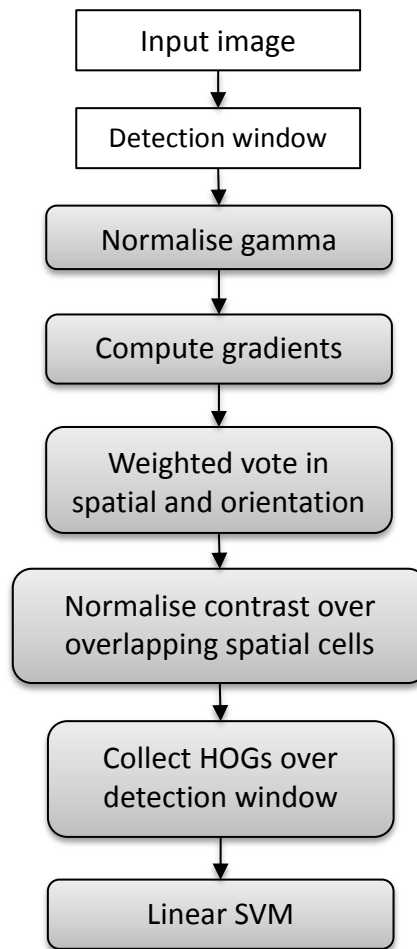


**Figure 4.1**The flowchart of HOG training and detection

According to Figure 4.1, both training and testing procedures begin with extracting and encoding HOG features (or descriptors), which will be introduced in 4.1.1. The mechanism of training models will be introduced in 4.1.2. The post-processing phase is a process called multi-scale object localisation, which will be introduced in 4.1.3.

### 4.1.1 Extracting HOG features

HOG features are based on edge intensities and orientations of images. A typical HOG extraction procedure is indicated by Figure 4.2:

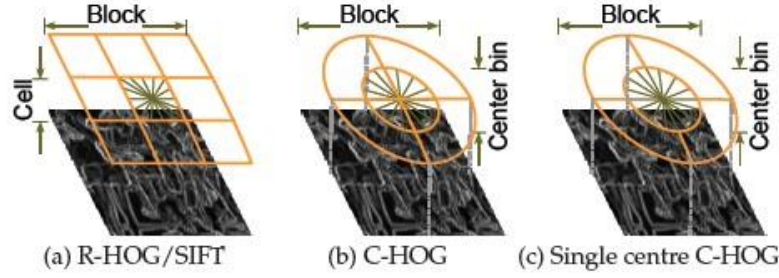


**Figure 4.2** The procedure of HOG extraction, form the paper [18]

For an input image, in the beginning, we scan it by a multi-scale rectangular mask and extract every scanned window from the image. Then for each window we scale it to a fixed size which is 64 by 128 in Dalal's approach and define them as detection windows. Considering that gradients vary due to local contrast and luminance variation, each window is processed by a gamma normalisation which executes square root in each colour channel (Dalal stated it was the best normalisation method in his experiment) at a low FPPW (false positive per window, restricting to greyscale reduce performance by 1% at  $10^{-4}$  FPPW). At the stage of computing gradients, Dalal argues in his PhD thesis that using the 1-D centred mask  $[-1,0,1]$  works best for human detection. Meanwhile orientations of gradients are spaced within  $0^{\circ}\sim 180^{\circ}$  and evenly distributed into  $\beta$  bins which are used to form histograms.



There are a number of HOG types, such as Rectangular HOG (R-HOG), Circular HOG (C-HOG) and single centre Circular HOG, as shown in Figure 4.3. Considering the nature of human detection (using a  $64 \times 128$  rectangular window), R-HOG is employed in the approach.



**Figure 4.3 Three variants of HOG descriptors, from the paper [108]**

There are two concepts called “cell” and “block” in the process of producing HOGs. A cell is a  $\eta \times \eta$  image region and a block consists of  $\zeta \times \zeta$  cells. For example, Figure 4.3(a) shows a block with  $\eta=3$  and  $\zeta=3$ . For each cell, a 1-D histogram which has  $\beta$  bins corresponding to orientation bins is formed by accumulating gradient orientations. The gradient energy of each pixel in the cell is used to vote into its corresponding bin of the orientation histogram. Dalal also states that using a Gaussian spatial window over a cell before accumulating voted orientations can improve detection performance.

So there are  $\zeta \times \zeta$  orientation histograms in each block. In order to solve the problem of gradient variation inside a block, Dalal performs four schemes of independent normalisations in each block. We use the scheme “*L2-Hys*” which is [109]:

$$\begin{cases} v \leftarrow v / \sqrt{\|v\|_2^2 + \varepsilon^2} \\ v = 0.2, \text{ if } v > 0.2 \end{cases} \quad (4.1)$$

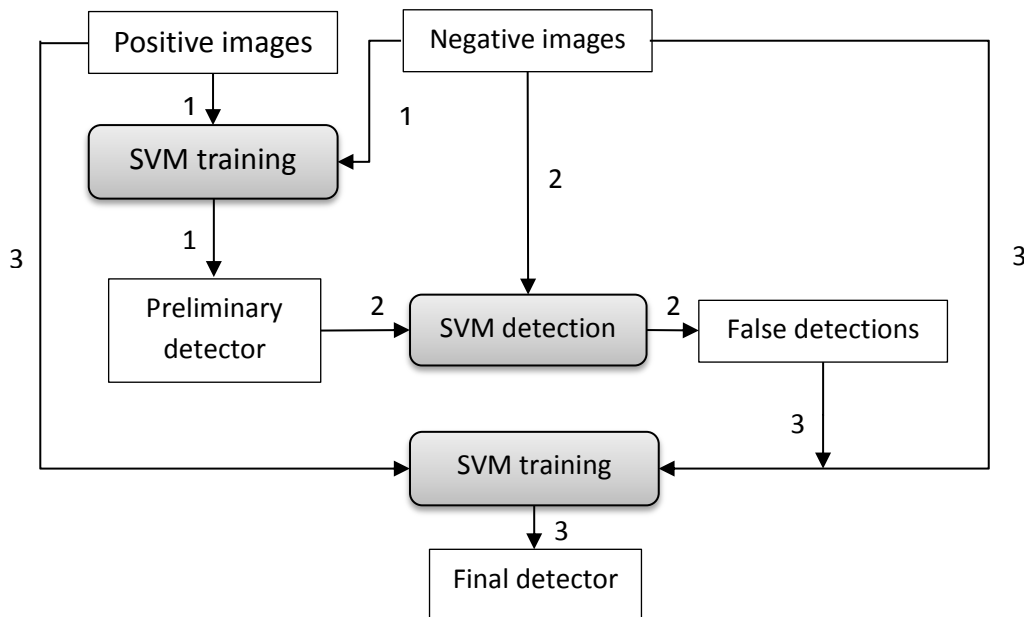
where  $v$  is the original descriptor,  $\|v\|_k$  is its  $k$ -time normalised one, and  $\varepsilon$  is a small constant to avoid division by zero. Meanwhile, there is a clipping mechanism limiting  $v$  under 0.2 during renormalisation. Finally we collect the normalised block descriptors which are defined as a HOG descriptor and put them to a vector.

### 4.1.2 HOG training phase

Building a human detector essentially is training a binary classifier that distinguishes positive (human) and negative (non-human) detection windows. Here a linear SVM is employed as the training and detection algorithm, which learns models by using HOG features from both positive and negative windows. Dalal also tried a Gaussian SVM which improved performance very little at the expense of much more computation time. So we also use the linear SVM as a comparison. The training dataset used includes positive images each of which contains a single detection window, and negative images each of which is large and contains many detection windows. The training phase contains 3 stages:

- 1) Dump HOG descriptor data for both positive and negative training images separately, and then train a preliminary detector by using the dumped data. For each negative image randomly extract a small number of detection windows (e.g. 10) as training samples.
- 2) Apply the preliminary detector to scan the negative images carefully to find false detections (called “hard” samples). Add these hard samples into the negative data.
- 3) Retrain the detector with the positive data and the updated negative data.

Figure 4.4 indicates the scheme of HOG training. Since there is huge number of negative samples if we extract all of the detection windows from the negative images, randomly extracting 10 samples from each negative image makes the training process much faster. To ensure the performance of the detector, hard samples are produced. Hard samples essentially are detection windows close to positive samples, and can help to reduce false detections after being added to negative samples. An example of the HOG training script is detached in Appendix and reflects the training procedure above.



**Figure 4.4** The flowchart of HOG training

### 4.1.3 Post-processing by multi-scale object localisation

Most detection methods usually face a problem after detectors scan an image and acquire detections: sometimes there are several detection windows overlapping and effectively representing the same object. Hence a fusion method that distinguishes different groups of detected windows and merges overlapping windows is required. Considering that an image region which has strong positive features (in the case of this section, it means a region looks like a human body strongly), usually obtains many detection windows (more than 5 for example), the fusion method is also designed to reject false detected regions which only have very few detection windows (1 or 2 for example).

Dalal's HOG approach determines a final detection from a group of overlapping detections according to their SVM scores, the number of detections and window scale ratios. A group that has a high peak SVM score, large number of detections and similar window scale ratios will be determined as a true positive. Conversely, the method will perform a rejection. Based on the theory proposed by D. Comaniciu et al

[110], this method, which is named non-maximum suppression, takes the following steps for a group of overlapping detections:

- 1) Represent each detection window (considered as a point) in 3-D position and scale space. The space consists of  $(x,y)$  coordinates in horizontal and logged scale values in vertical.
- 2) Estimate the uncertainty matrix for each point.
- 3) Merge the points to a mode by iteratively computing the mean shift vector for each point.
- 4) Finally obtain the bounding box from the mode that contains the final location and scale rate.

Figure 4.5 indicates the result of the merging method. It shows that for the man on the right, the case is simple and the result is good. For the woman on the left, the case is much more complicated due to a large number of multi-scale overlapping detections. The merging algorithm measures these overlapping windows by their sizes, locations and density to achieve a final result. A part of the woman's head is left out of the final detection window due to the large number of the preliminary detection windows which leave a part of the head out. The outcome is still not too bad, considering that those very small and very large detection windows are rejected. In addition, for the bicycle near the lower-left corner, only one detection window is located in this region and rejected by the fusion method.



**Figure 4.5 Merging HOG detections by multi-scale object localisation**

**Left: the detection result before merging windows; Right: the final detections**

#### 4.1.4 Experimental results

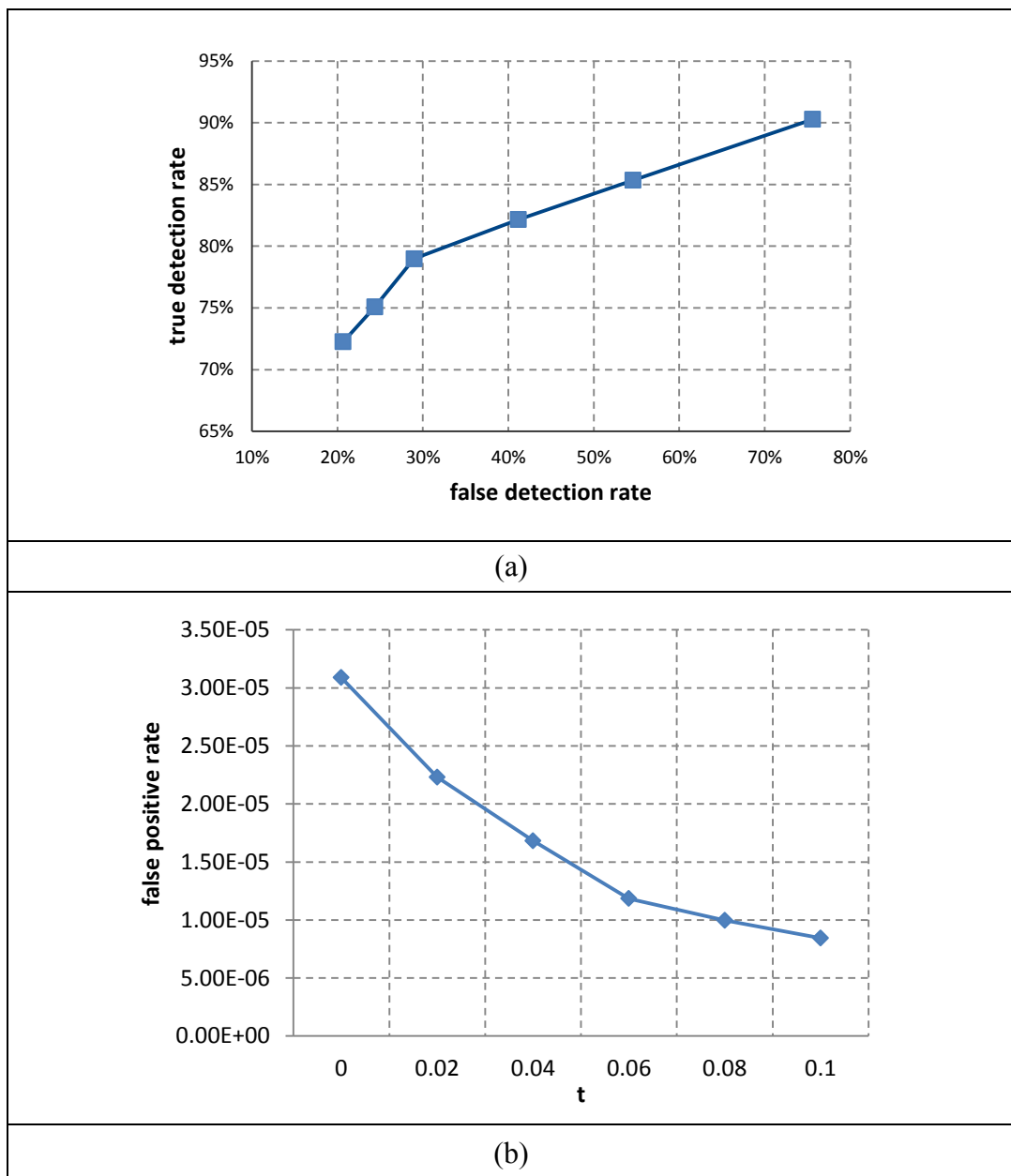
We use the same INRIA dataset as Dalal did. This dataset has 1208 positive training samples, each of which contains a single human body (can be a man, woman or child) walking or riding bicycles. We mirror these positive samples by their vertical middle line so that we get 2416 positive samples. The dataset also has 1218 negative training images. For each image we extract 10 detection windows randomly so that 12180 negative samples are extracted for the first training stage. There are also 288 positive testing images which contains 566 human bodies totally. Instead of testing sub-images that are cropped from the data images and scaled to  $64 \times 128$ , as Dalal did, we apply the trained detector to the 288 positive testing images to achieve more practical results.

We set the HOG parameters  $\eta=8$ ,  $\zeta=2$ ,  $\beta=9$ , and perform the *L2-Hys* scheme for block normalisation. We scale the detection window by 1.05 from the initial size ( $64 \times 128$ ) and set the scanning stride to 8 pixels in both height and width directions. A threshold  $t$  is needed for multi-scale object localisation, and we raise  $t$  from 0.00 to 0.10 with a step length of 0.02. The result is indicated in Table 4.1 and Figure 4.6. In Figure 4.6(a), the vertical axis represents the rates of true detection numbers to the total number of human bodies, which is:

$$\text{true detection rate} = \frac{\text{number of positive detections}}{\text{number of human bodies}}$$

In order to represent numbers of false detection comparably to true detections, we define the false detection rate as the rate of false detection numbers to the total number of human bodies on the horizontal axis, which is:

$$\text{false detection rate} = \frac{\text{number of false detections}}{\text{number of human bodies}}$$



**Figure 4.6 Results of the original HOG detector**

**(a) true detection rates against false detection rates, (b) the chart of false positive rates**

When  $t=0.00$  the true detection rate reaches over 90% but there are also 428 false detections, which means every true detection goes with 0.84 false detection. It is noticeable that as the threshold rises up, the false positive rate falls down significantly, from  $10^{-5}$  level to  $10^{-6}$  level according to Figure 4.6(b), and the true detection rate drop down at the same time. Therefore there is a trade-off between the true detection rate and the false positive rate. However the negative rate is still too

high even we use the highest threshold with the positive rate falling down to a low percentage. When  $t=0.10$ , the positive rate falls down to 72% while there are still 117 false detections, which means that more than one false detection occur when we get four true detections.

t	Positive	Negative
0	511	428
0.02	483	309
0.04	465	233
0.06	447	164
0.08	425	138
0.10	409	117

**Table 4.1** Detection results of the original HOG approach

## 4.2 Modifications on HOG training

### 4.2.1 Extra training data

The performance of the original HOG approach does not satisfy practical applications, since the false detection rate is relatively high. By studying the detection results, we notice that many false detections occur at human bodies parts and boundaries of foregrounds (human bodies) and backgrounds, see Figure 4.7. We suppose that this phenomenon is caused by the defect of the training mechanism which only uses pure positive and negative samples and does not treat body parts and foreground/background boundaries as negative samples. Hence we add some extra negative images which are extracted from human body images into the training dataset.



**Figure 4.7 Examples of the original HOG detections with overlapping detection windows on body parts**

We produce extra negative images by cropping windows from human images, each of which is either a single-person or a human-group picture. We have tried three different schemes of cropping body parts, termed “2x1ol”, “2x2” and “2x2ol”. For a human image, “2x1ol” means we use a cropping window with  $\frac{1}{2}$  of the image height and the full image width. The term “ol” means there is an overlapping window in the middle of the image, so 3 sub-images will be produced. Similarly, “2x2” means cropping sub-images in  $\frac{1}{2}$  height and width without overlapping windows, and 4 sub-images will be produced. “2x2ol” crops sub-images in the same size of “2x2” and has overlapping windows along both height and width. Therefore 9 sub-images will be produced. Figure 4.8 shows how the cropping method works.





Figure 4.8 Three mechanisms of cropping negative images from human images

## 4.2.2 Experiments and results

We selected 260 human images and cropped them by the three methods above. Finally we get 780 sub-images by “2x1ol”, 1036 ones by “2x2” and 2331 ones by “2x2ol” (we discard some sub-images whose sizes are smaller than  $64 \times 128$ ). After obtaining the extra negative images, we add to the initial dataset and train detectors as what we did in Section 4.1. The detection results compared with the original HOG training are illustrated in Table 4.2 and Figure 4.9. To make a fair comparison, we also add 780 negative images representing backgrounds and other objects for the

original HOG training. So in Table 4.2 and Figure 4.9, the “original+” detector is trained by 1998 negative images (not body parts images).

Table 4.1 and 4.2 indicate that the “original+” detector performs similarly to the “original” one, which means that adding negative images representing backgrounds or other objects cannot improve detection performance. However, according to Table 4.2 and Figure 4.9(a) a significant reduction of false detection numbers occurs when extra negative training images are used, and meanwhile the true detection rate only decreases slightly. So the conclusion can be made that using extra negative images which are body parts can improve the performance but using extra background negative data cannot. It can also be noticed that “2x1ol” performs the best, which pulls the false detection rate down by more than 25% but only loses 7 true detections when  $t=0$ . Along with  $t$  rising, it keeps a true detection rate close to the original HOG detector and a low false detection rate as well. “2x2” and “2x2ol” perform similarly, both better than the original one and worse than “2x1ol”. Meanwhile the chart of false positive rates shows the same information: false positive rate goes down all time as  $t$  rises, and “2x1ol” has the lowest false positive rate.

t	Original+		2x1ol		2x2		2x2ol	
	pos	neg	pos	neg	pos	neg	pos	neg
0	506	435	504	271	502	348	503	337
0.02	484	300	476	171	481	235	478	220
0.04	459	225	448	114	457	162	456	140
0.06	444	169	423	82	435	120	421	114
0.08	429	129	406	59	408	105	413	88
0.1	409	120	396	46	397	91	393	73

**Table 4.2 Results of the HOG detector using extra training data**

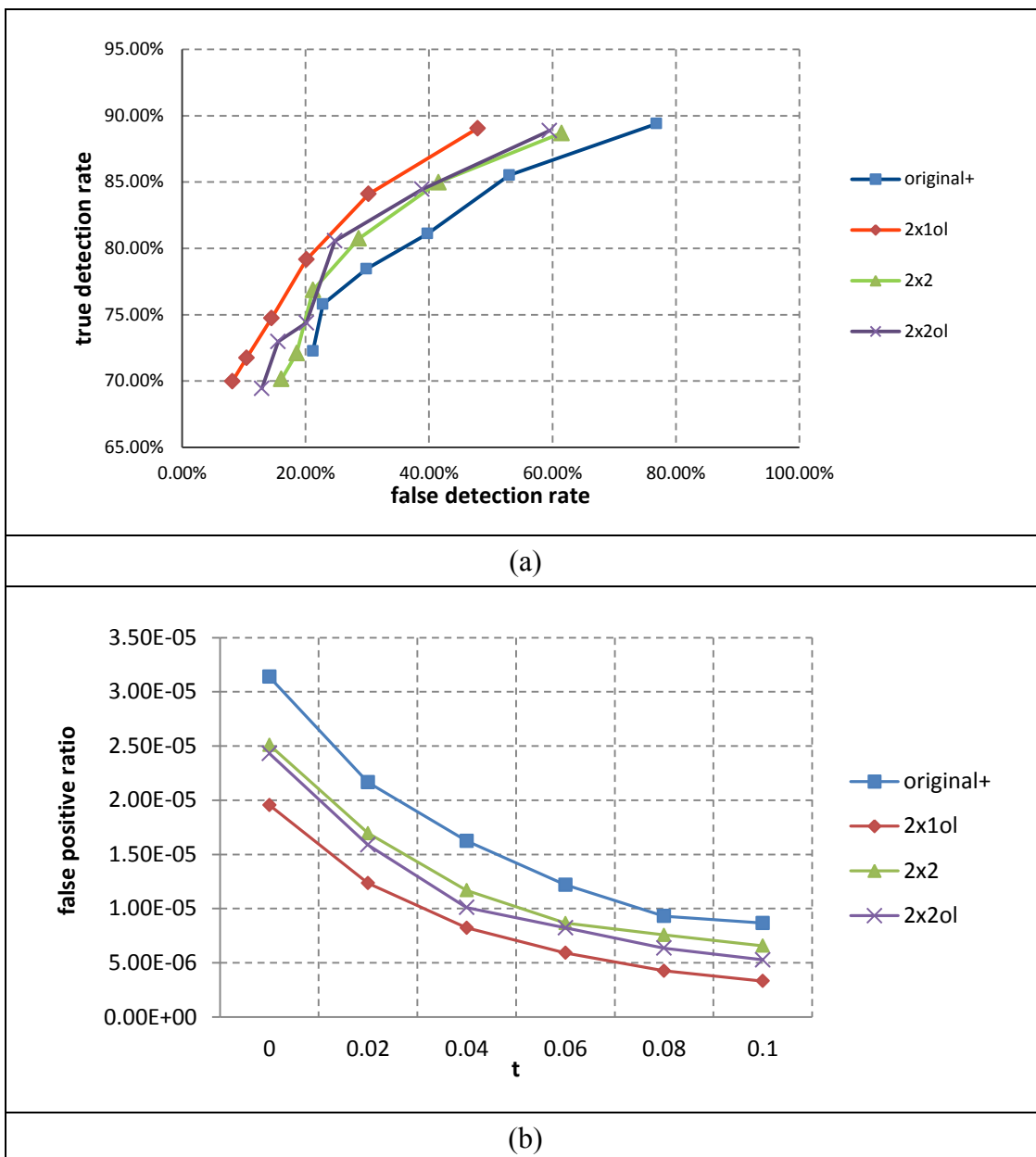


Figure 4.9 Results of the HOG detector using extra training data  
 (a) true detection rates against false detection rates, (b) false positive rates.

### 4.3 A Comparison: Haar-like features and AdaBoost

The approach using Haar-like features and AdaBoost classifiers (we call it the Haar-AdaBoost method below) is also implemented as a comparison. The Haar-AdaBoost

method which was firstly presented by Viola and Jones [13], was initially designed to achieve accurate, efficient human face detection, and then it has been widely extended to other types of object detection in computer vision. We implemented this approach for our human body detection based on Day's work [23].

### 4.3.1 Haar-like features

Haar-like features, presented by Viola and Jones, are differences between sums of rectangular image regions. In Figure 4.10, twelve types of features are indicated. In each feature type, the sum of its white region is subtracted by the sum of its black region. The diagonal features are less commonly used than the horizontal/vertical ones.

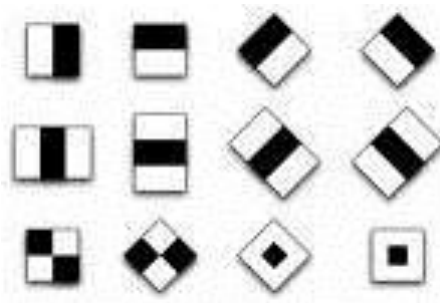


Figure 4.10 Types of Haar-like features

### 4.3.2 AdaBoost classifier

AdaBoost is a machine learning algorithm that builds a strong composite classifier by combining several weak elemental classifiers [13]. Each weak classifier processes quickly to roughly classify input objects and the entire classifier combines results generated by all of the weak classifiers to achieve a fast and accurate classification. The structure of a typical AdaBoost classifier can be illustrated by the following equation:

$$H(x) = \text{sign}\left(\sum_{t=1}^T \alpha_t h_t(x)\right) \quad (4.2)$$

where  $x$  is the input,  $H(x)$  is the classifier result (or final hypothesis),  $T$  is the number

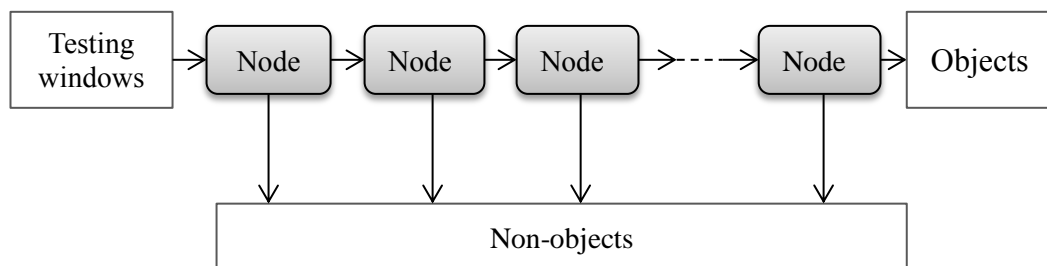
of weak classifiers,  $h_t$  represents the a week classifier weighted by its coefficient  $\alpha_t$ .

For each weak elemental classification  $h_t(x)$ , the only requirement is simply to reach an error  $\epsilon_t < 0.5$ . The error rate of an elemental classifier determines the value of its coefficient Viola and Jones state that the coefficient  $\alpha_t$  is given by the following equation [23]:

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \quad (4.3)$$

### 4.3.3 Cascade detector

Cascading is a method to apply AdaBoost classifiers sequentially by Viola and Jones. The cascade detector essentially is a degenerate decision tree which has a number of decision nodes each of which consists of a number of elemental classifiers to reject a proportion of non-objects and. Figure 4.11 illustrates the structure of the cascade detector [23].



**Figure 4.11 The structure of the cascade detector**

In order to train the detector more efficiently, a mechanism called bootstrapping is involved: when the algorithm has finished training a node, it substitutes the examples which would be rejected by the partial detector by other examples which would be accepted.

### 4.3.4 Experiments and results

Day extended Viola and Jones' face detection method by optimising the schemes of

the cascade, investigating the selection of elemental classifiers, and proposed new post-processing methods [111]. We build detectors by employing Day's program designed for face detection and the INRIA dataset with the extra training data used in HOG. However there are some factors that let us modify the dataset:

- 1) Viola and Jones' method as well as other methods developed from it, are designed for human face detection and use a  $24 \times 24$  detection window which is enough for faces. However human body images are usually more complicated and a larger size of detection windows is required for more information. Unfortunately the size of  $64 \times 128$  is much too large for this method (the program crashed in the experiment). So we scale the size of detection windows to  $32 \times 64$ , a quarter of the HOG window size.
- 2) Cascade training requires negative samples four or five times of positive samples. We add 6628 negative images to the training dataset because obviously the negative images in INRIA dataset are not sufficient.

Considering the difference between the face window and body window, we execute experiments by varying the number of Haar-like features: 10k, 12k, 20k and 30k, which are much less than Viola and Jones' because using more features costs too much memory and may cause crash due to the large samples used here. In the experiment 30k is the maximum number of features. The configuration of the process is listed in Appendix II. Day's system implements merging methods which make merging decisions according to overlapping percentages, centre distances and size ratios [111]. We configure the merging scheme as:

$$\mathbf{B\{single: overlap>0.6, sizeRatio<2\}C\{mean\}}$$

where B represents the merging strategy, and C represents the combination method. The description above means that we use single linkage, and for two windows, when their size ratio is smaller than 2, and overlapping percentage is larger than 60% of the small window, the two windows are combined in to one which takes the mean of the two windows as its location and size information.

The results are illustrated by Table 4.3.

Number of features	10k	12k	20k	30k
<b>True detections</b>	274	285	301	277
<b>False detections</b>	73	70	89	64
<b>True detection rate</b>	48.41%	50.35%	53.18%	48.94%
<b>False detection rate</b>	12.90%	12.37%	15.72%	11.31%

**Table 4.3 Results of using Haar-like features and AdaBoost**

According to Table 4.3 varying the number of features does not cause much influence on final results. Compared with the results of HOG detectors, obviously its true detection rates is much lower. It is partly because it uses a detection window in a small size during the training phase, which leads to losses of information. Another reason is that unlike face samples, body samples usually include background due to shapes of human bodies. The diversity of background causes much difficulty in detection.

Compared with the detection the Haar-AdaBoost, HOG uses a larger detection window, requires less negative training images, and achieves much better performance. The detection results reflect that in the state of art, HOG is more suitable for human detection.

## 4.4 Using Pre-processed images

When computing the gradient value of a pixel in a colour image, the original HOG approach computes gradients separately in all of the three channels, and selects the one having the maximum magnitude to represent the gradient of this pixel. As we argued in Chapter 3, this mechanism sometimes loses colour edge information. For example, considering two pixels with RGB values (0,0,0) and (255,255,255), their colour distance should be  $255\sqrt{3}$ . However the distance calculated by the mechanism above is only 255, which is as same as the distance between (0,0,0) and (255,0,0). To avoid this problem, we pre-process the data images by our density-

based colour-to-greyscale conversion introduced in Chapter 3.

There are two schemes of applying our colour-to-greyscale conversion.

- 1) Use the converted images as input for both training and testing images and execute the original HOG procedure.
- 2) Train two detectors by initial colour images and converted greyscale images respectively. At the first stage detect human bodies on colour images and crop the detection windows. At the second stage scale the cropped windows to  $64 \times 128$  and convert them to greyscale by our method. At the final stage use the greyscale detector to reject the false detections.

#### 4.4.1 HOG detection using greyscale images for both training and testing

We apply our density-based colour-to-greyscale conversion to the training and testing images differently. We set the cluster number for the positive training images and for the extra negative images (denoted as  $c_1$ ) smaller than that for other images (denoted as  $c_2$ ), as the positive training images and the extra negative images are only a local region in the source images and usually have small number of colour clusters, while the negative training images and the testing images are relatively large and have more colour clusters. In the experiment, we set  $(c_1, c_2) = (10, 15)$  and  $(15, 25)$  respectively. The configuration of the other parameters is as same as the configuration in section 4.1.

Figure 4.12 indicates the detection results compared with those of using colour detectors. Here only the “2x10l” set are used as extra training data. In the chart, “greyscale” means using greyscale images that converted by our colour-to-greyscale method as the training and testing data, “2x10l” means using the extra training images, and  $(10, 15)$  or  $(15, 25)$  means the values of the parameter set  $(c_1, c_2)$ .



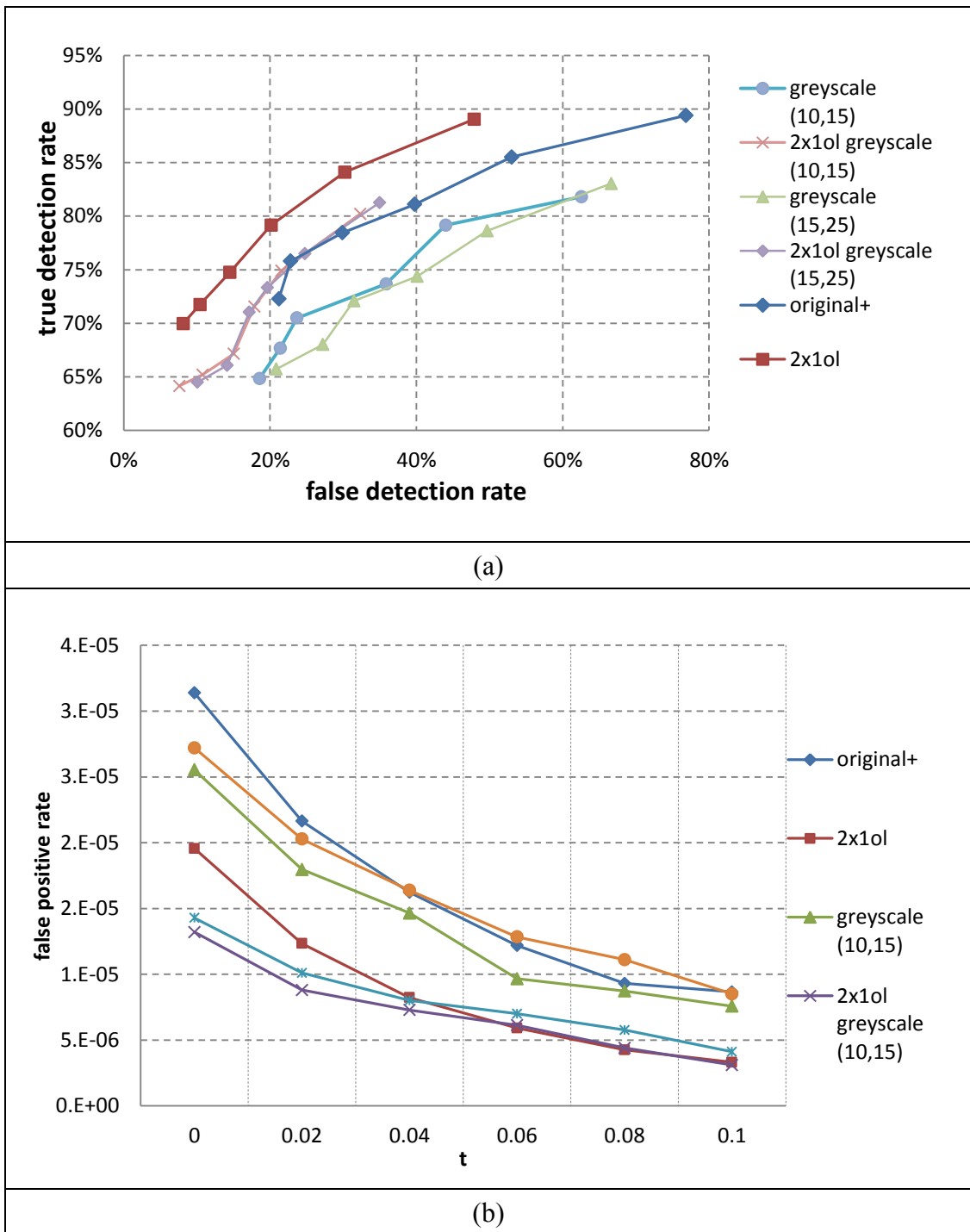


Figure 4.12 Results of the HOG detector using greyscale images

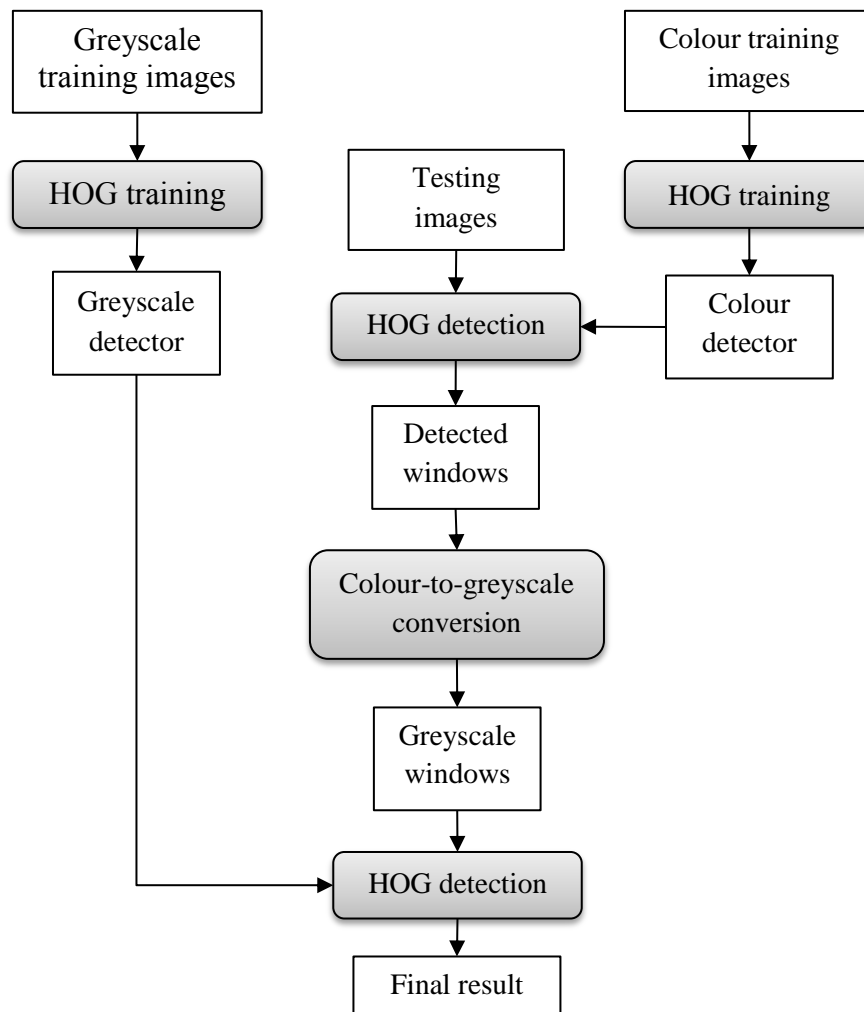
(a) true detection rates against false detection rates, (b) false positive rates. The labels with the word “greyscale” mean that we use the greyscale images converted by our colour-to-greyscale method as the experimental data.

According to Figure 4.12(a), it is clear that using greyscale images can reduce both

positive and false detections significantly. Adding extra negative training images keeps true detection rates in the same level and meanwhile reduces the false detection rate by about 30% maximally. But when  $t=0.10$ , the false detection rates are around 10% with adding “2x10l” images and around 20% without using them. Hence when  $t$  becomes large, using greyscale images does not make much influence on the false detection rate. In addition, the results are close when we alter the parameters of cluster numbers from (10,15) to (15,25). Generally speaking, according to the chart in which the curve for “2x10l” is still at the top position, we conclude that simply applying the greyscale images to an HOG detector does not improve the detection performance. However, in the chart of false positive rates (Figure 4.12(b)), using greyscale images can achieve lower false positive rates than using colour images when  $t$  is small. This is to say, compared with colour images, greyscale images lead to a stronger rejection for false detections when there are a big number of false detections.

#### **4.4.2 HOG detection using greyscale images to refine colour detections**

The reason why it is difficult to achieve a good performance by using greyscale images is partly that the positive training images are local regions while the testing images are global, which leads to a local-based colour-to-greyscale conversion on positive training images but a global-based colour-to-greyscale conversion on negative training and testing images. However doing colour-to-greyscale conversion for every scanned window is an impossible mission because it is much too computationally expensive owing to long conversion time and a huge quantity of scanned windows. An alternate scheme (called “iteratively detecting”) to use greyscale conversion is a three-stage procedure:



**Figure 4.13** Flowchart of the “iteratively detecting” scheme

- 1) Detect human bodies on colour images by the original HOG detector and crop the detection windows. To get true detections as many as possible, we do not add the extra training images.
- 2) Scale the cropped windows to  $64 \times 128$  and convert them to greyscale by our method.
- 3) Use a detector trained by greyscale images (including the extra training images) to test the greyscale windows generated in Stage 2, in order to reject false detections.

We still use  $t$  which grows from 0.00 to 0.10 with a step length of 0.02, as the threshold in the first stage. In the third stage, no threshold is needed since there is

only one detection window on each input image. Figure 4.14 indicates the results produced by this scheme. Also we used two sets of cluster numbers in the colour-to-greyscale conversion. The variation of the results generated by using the same set of images is caused by the varying threshold  $t$  in the first stage.

According to Figure 4.14(a), either configuring  $(c_1, c_2)=(10,15)$  or  $(15,25)$ , the new scheme can achieve a low false detection rate at the cost of missing a number of true detections when  $t=0.00$  and  $0.02$ , which is represented by the higher parts of the purple-triangular-node curve and the brown-circular-node curve that perform equally to the “2x1ol” curve (the red curve with rhombic nodes) in the chart. However, when  $t$  becomes larger, the false detection rates regress to the “2x1ol” curve and the true detection rates are still below that curve. In Figure 4.14(b), there is a significant improvement of false positive rates produced by the “iteratively detecting” scheme. When  $t=0.00, 0.02$  and  $0.04$ , the curves corresponding to the “iteratively detecting” scheme are below the red curve corresponding to “2x1ol” and even though  $t$  rises high, the curves of “iteratively detecting” still perform equally to the “2x1ol” curve. So a conclusion can be made that this “iteratively detecting” scheme improves the detection performance slightly especially on false positive rate when using a small threshold in its first stage, but the original HOG scheme still has the highest true detection rate.

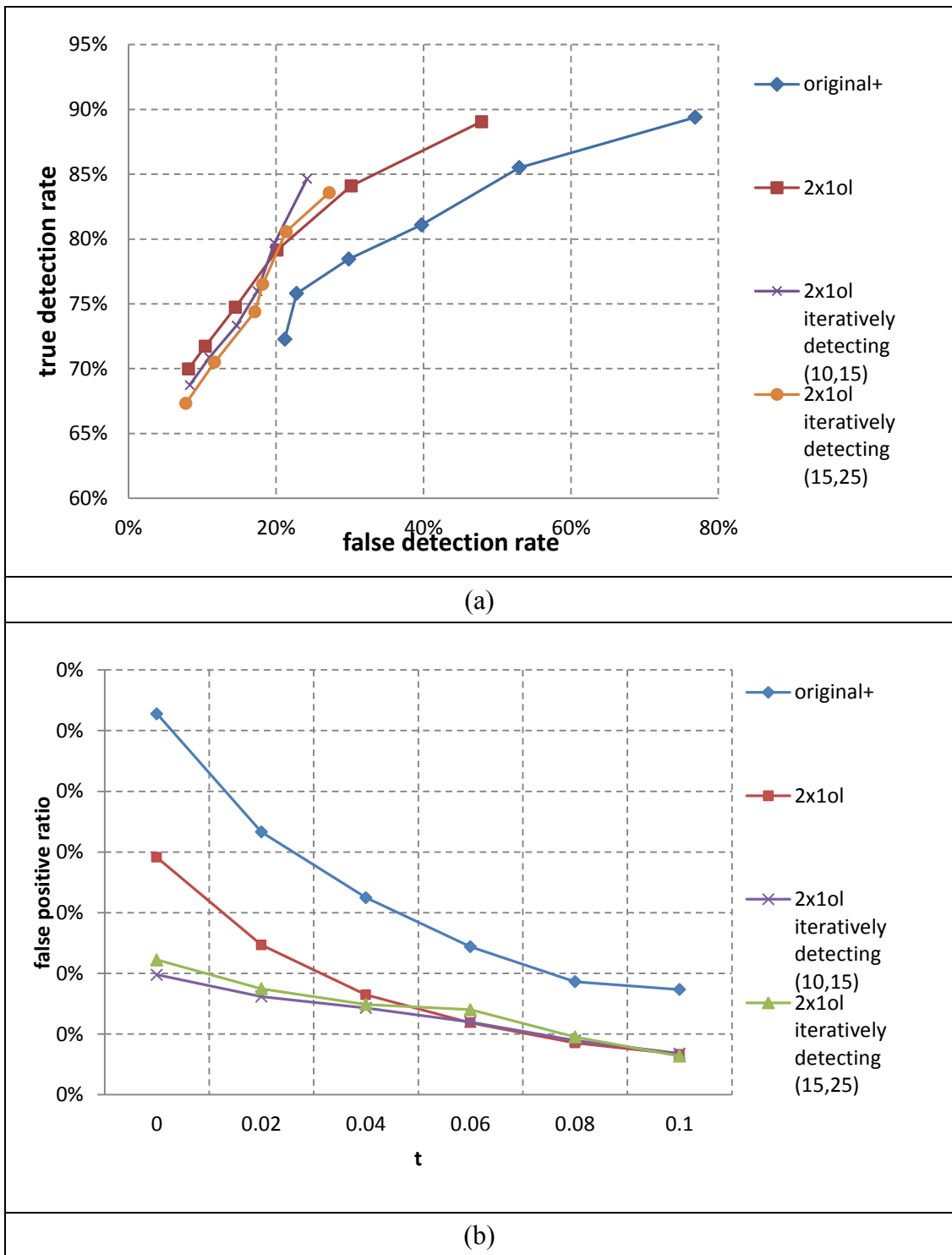


Figure 4.14 Results of the HOG detector using greyscale images and “iteratively detecting” scheme

(a) true detection rates against false detection rates, (b) false positive rates.

Now we should have a discussion about whether the colour-to-greyscale conversion improves the performance of the HOG detector. It can be argued that although the our conversion method cannot achieve a high detection rate of max-channel mechanism used by the original HOG, it reduces false detection rate significantly at the cost of not so much true detection rate. In Figure 4.14(a), our conversion method with the iterative detecting scheme reduces false detection rate by about 25% at the cost of about 5% true detection rate when  $t=0.00$ , so it can be considered as a successful scheme.

## 4.5 Conclusions

In this chapter we introduce an advanced human detection method based on histogram of orientated gradients, and compare it with another conventional method using Haar-like features and AdaBoost. The original HOG approach achieves a good true detection rate for human detection, much better than the Haar-like features and AdaBoost approach. However, the original HOG is not practical due to its high false positive rates. We improve this method by adding extra training data and pre-processing the images by colour-to-greyscale conversion. Finally the false positive rate is significantly reduced at the expense of losing some true detections. Therefore the trade-off between false positive rates and true detection rates should be considered. In practice, we need to select detection methods according to requirements of applications, and consider factors such as true detection rate, false positive rate, and processing speed as well.

# Chapter 5

## Pose Estimation

---

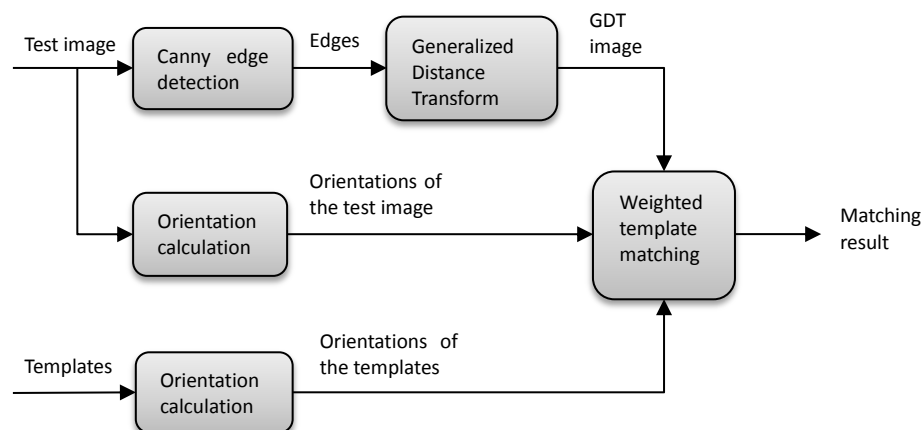
We have improved an advanced template matching method based on generalized distance transform (GDT) and orientation maps (OM) [19] which is initially designed to use contour templates to detect human bodies. As mentioned in Chapter 2, template based (signal-processing-based) detection methods usually face a problem of forming reliable template sets that distinguish human bodies from other objects and backgrounds efficiently. However, the templates used are usually manually made and subjective opinions influence detection results. For example, selecting different outline templates to represent human bodies leads to different matching results. Therefore the performance of a detector depends on the opinion of the person who selects and makes templates. Meanwhile, making a set of reliable, robust templates is very difficult because the appearances of human bodies vary wildly due to poses, body shapes and clothes.

In the proposed system, we use this template matching method to estimate human poses. After we capture human bodies from an image by human detection method (HOG), we apply a set of pose templates to the detected windows to find a best match by employing GDT&OM. Since human bodies have already been detected, interference from backgrounds and other objects will be cancelled efficiently. Meanwhile, we only need to build a range of contour templates representing certain human poses, which is much easier than to build a huge number of templates indicating all kinds of human body appearances.

GDT is applied to give larger weights on the strong edge pixels, which highlights the edges in image and leads the matching process to focus on the edges. OM increases accuracies of matching results by adding gradient information for all pixels in a test image and pixels on the outline of a template. In the matching phase, we calculate the average differences between the matching areas and the templates based on the orientation map of a test image, work along with the weights of the pixels of its GDT image, and finally obtain a best match.

For an input image, the procedure of a GDT&OM has the following steps:

- 1) Produce the Canny edge of the image.
- 2) Apply the Generalized Distance Transform by using the edge image produced in step (1).
- 3) Compute the Orientation Map.
- 4) Do weighted matching between the templates and the union of the GDT image and Orientation Map.



**Figure 5.1** The diagram of GDT&OM

In Section 5.4, another pose estimation method that is based on human body part locating and uses machine learning methods to classify objects, is introduced.



## 5.1 Theory

### 5.1.1 Generalized Distance Transform

Generalized Distance Transform is an algorithm that computes distances of each pixel to the nearest edge. We generate a binary edge image by a Canny operator and perform GDT according to the Canny edges. The formula of GDT can be represented as follows according to Felzenszwalb and Huttenlocher [38]:

$$D_{\Psi}(p) = \min_{q \in \zeta} \{d(p, q) + \Psi(q)\} \quad (5.1)$$

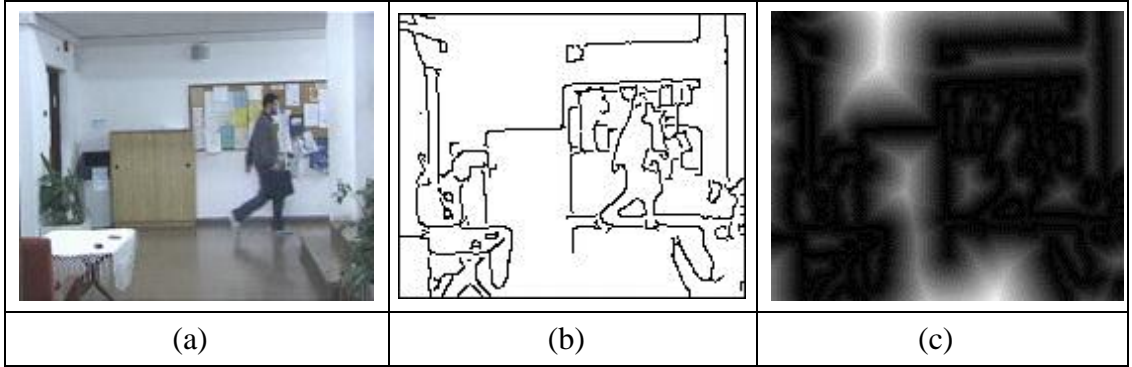
Where  $d(p, q)$  is the Euclidean distance between point  $p$  and  $q$ :

$$d(p, q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$$

$\zeta$  is defined as a regular grid and  $\Psi(q)$  represents a function on the grid. Here  $\Psi(q)$  is defined as:

$$\Psi(q) = \begin{cases} \frac{1}{\sqrt{I_x^2 + I_y^2}}, & \text{if } (q) \in e \\ \infty, & \text{otherwise} \end{cases} \quad (5.2)$$

where  $e$  is the binary edge pixels.  $I_x = \frac{\partial I}{\partial x}$  and  $I_y = \frac{\partial I}{\partial y}$ , are gradients in horizontal and vertical directions at point  $q$  in image  $I$ . Here  $\Psi(q)$  is different from the conventional distance transform (DT), which assigns  $\Psi(q)=0$  when  $q \in e$ . Equation 5.1 means that for a pixel  $p$ , find the nearest pixel  $q$  on edges, and assign the distance value as the Euclidean distance between  $p$  and  $q$  plus a small value based on its gradient. Figure 5.2c is a GDT image, in which white regions represent high distance values and black regions represent low distance values.



**Figure 5.2 Images of producing a GDT image**

**(a) the input image, (b) the Canny edge image, (c) the GDT image**

### 5.1.2 Orientation Maps

According to Thanh's paper [19], the orientation value of a pixel is related to the position of its nearest edge pixel. That is to say, the orientation of edge pixels will impact the non-edge pixels nearby. Considering a single pixel  $p$  in image  $I$ , assuming that  $q^*$  is the nearest edge pixel to  $p$  (which means that orientation maps are discontinuous), the orientation value at pixel  $p$  is defined as

$$O_{\Psi}(p) = \arctan(I_{x^*} / I_{y^*}) \quad (5.3)$$

Where  $I_{x^*}$  and  $I_{y^*}$  are the gradients at  $q^*$ .

### 5.1.3 Template Matching

Considering a template  $T$  and a test image  $I$ , we define  $o(t)$  as the orientation at point  $t$  in  $T$ . the dissimilarity between the image  $I$  and the template  $T$  at point  $t$ , is denoted as  $d_{T,I}(t)$ , which is defined as

$$d_{T,I}(t) = \sqrt{D_{\Psi}^2(t) + \sin^2(O_{\Psi}(t) - o(t))} \quad (5.4)$$

Let  $w_T(t)$  be the weight of point  $t$ , Thanh obtained it by using a set of positive and negative training images:

$$w_T(t) = 1 / \left\{ 1 + \exp \left[ - \frac{D_{\bar{S}(T)}(t) - D_{S(T)}(t)}{D_{\bar{S}(T)}(t) + D_{S(T)}(t)} \right] \right\} \quad (5.5)$$

Where  $S(T)$  is the positive images and  $\bar{S}(T)$  is the negative images corresponding to  $T$ . And

$$D_{X(T)}(t) = \frac{1}{|X(T)|} \sum_{x \in X(T)} d_{T,x}(t) \quad (5.6)$$

According to Equation 5.6,  $D_{\bar{S}(T)}(t)$  and  $D_{S(T)}(t)$  represent the average distances from a point  $t$  to the closest image points of all negative images  $\bar{S}(T)$  and all positive images  $S(T)$  respectively. Finally, the matching score between the template  $T$  and the image  $I$  is

$$D(T, I) = \sum_{t \in T} w_T(t) d_{T,I}(t) \quad (5.7)$$

#### 5.1.4 Templates

Templates are human body contours extracted from images that display typical human poses. Figure 5.3 indicates a few templates for dancing skipping, walking. Considering that human bodies can be seen in various orientations and because of symmetrical possibilities for articulation of movement, usually we make a pair of templates one of which is the mirror image of the other. A type of pose needs at least two pairs of templates: a pair of the front/back view and a pair of the side view.

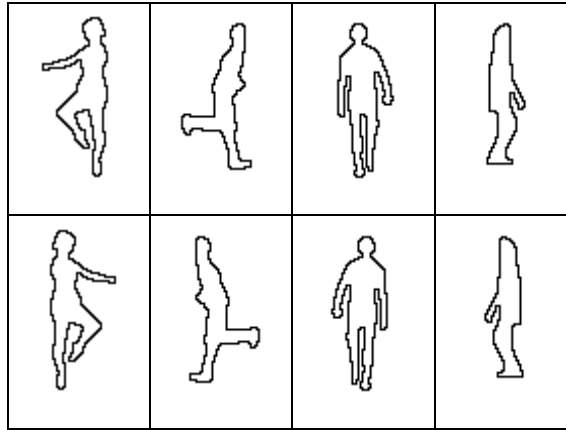


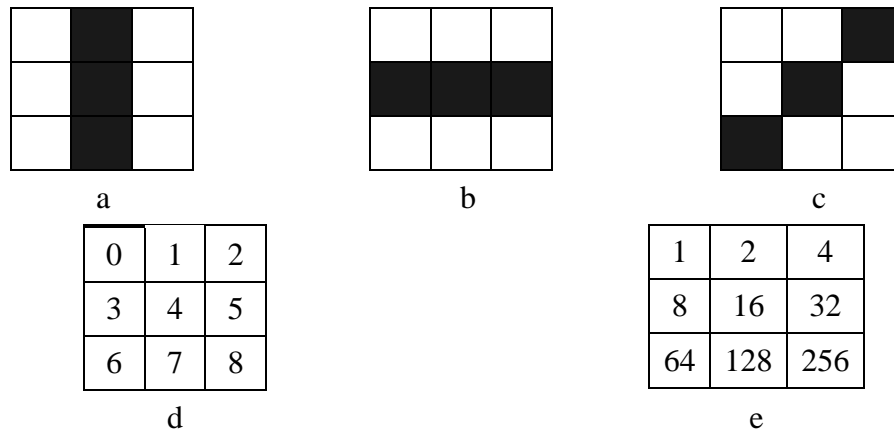
Figure 5.3 Examples of templates

To compute orientations of a template, a  $3 \times 3$  operator is used. We list all possible  $3 \times 3$  windows which are combinations of binary pixels (pixels only having values of 0 or 1). Then we assign an orientation value for each combination. For an edge pixel  $p$  on the templates, we consider the  $3 \times 3$  box, whose centre pixel is  $p$ , and decide the orientation of  $p$  according to the pixel values around. Examples are displayed in Figure 5.4. For efficient processing, we employ a lookup table mechanism, which is:

- 1) Label each pixel in the  $3 \times 3$  box and denote the labels as  $n$  ( $n=0,1,2,3,\dots$ , see Figure 5.4d).
- 2) Assign each pixel with  $2^n$  ( $n$  is its corresponding label, see Figure 5.4e).
- 3) Build a lookup table whose inputs are sums of  $2^n$  and outputs are corresponding orientations.
- 4) Considering a pixel  $t$  in a binary template  $T$  in which 1 stands for edge pixels and 0 for non-edge ones, let the coordinate of  $t$  be  $(p,q)$ . Denote the  $3 \times 3$  box as  $B$ . The LUT input (LI) for  $t$  is:

$$\text{LUT input} = \sum_{i=0}^3 \sum_{j=0}^3 T(p-1+i, q-1+j) B(i+j) \quad (5.8)$$

Hence each combination of edge/non-edge pixels has a unique LI value which corresponds to an orientation value. The orientation can be achieved according to the LUT quickly. For example, for Figure 5.4c,  $\text{LI} = 84$ . In the LUT,  $\pi/4$  ( $45^\circ$ ) is assigned as the output for  $\text{LI} = 84$ .



**Figure 5.4 Orientations of template pixels**

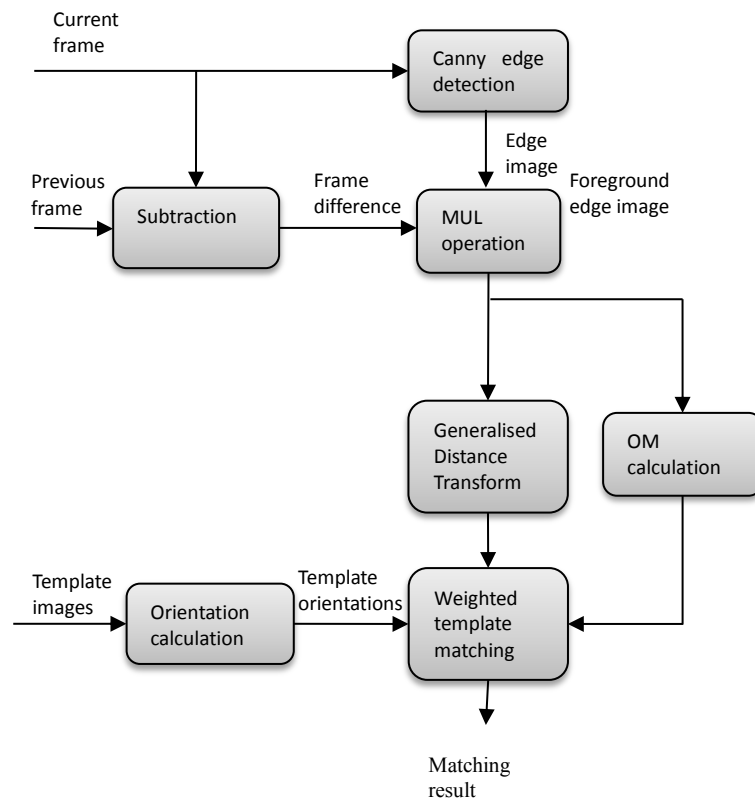
**(a) 90°; (b) 0°; (c) 45°; (d) labels of the operator; (e) assignment of the operator**

## 5.2 Detection Experiments for Frame Sequences

Practically the accuracy of GDT&OM detection is poor, especially for images with complicated backgrounds. We implement two schemes of pre-processing before applying GDT&OM. One is background cancellation for frame sequences, the other is modification for templates.

### 5.2.1 Background Cancellation

For frame sequences with static backgrounds and moving objects, to minimise the interference of background, we propose to employ a background cancellation mechanism in the work to improve the performance. When processing each frame, we extract its foreground (moving objects) by computing the difference between two continuous frames (the current frame and the previous frame). The flowchart of the whole procedure is shown in Figure 5.5.



**Figure 5.5 The flowchart of template matching**

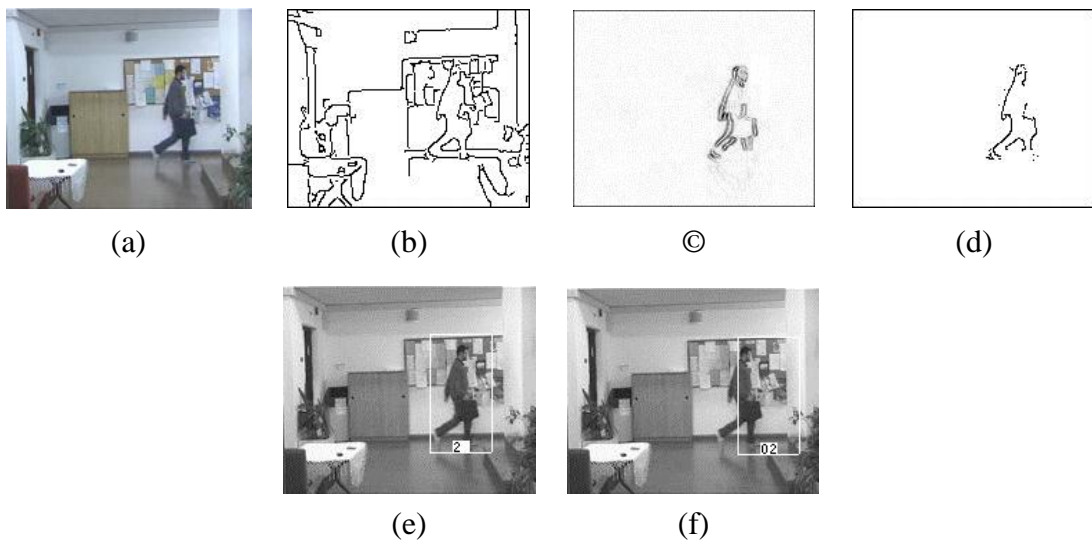
Figure 5.5 differs from Figure 5.1 by subtracting the current frame by its previous one, and multiplying the subtracted frame by the edges of current frame. Hence the only edges of moving objects are left and edges of static content (mainly the background) can be cancelled.

When we process a frame, called image  $I$ , the Canny algorithm [112] is applied to generate the edge image of  $I$ , denoted as image  $E$ . Meanwhile, we subtract  $I$  from its previous frame, to get a frame difference image, denoted as image  $D$ . Generally, a high pass filter is applied to the frame difference image  $D$ , so that the problem of luminance difference between frames can be avoided. We obtain an image that indicates the differences caused by movement between the two images. Since we are using videos captured by static cameras, it can be considered that the difference between two frames is caused by moving objects. Then the edge image  $E$  and the frame difference image  $D$  are used to produce a foreground image  $F$  by the function as Equation 5.9

$$F(p) = \begin{cases} E(p), & \text{if } D(p) = \text{BLACK} \\ \text{WHITE}, & \text{if } D(p) = \text{WHITE} \end{cases} \quad (5.9)$$

Here we treat the images as binary. In other words, the edge pixels are treated as *BLACK* while the non-edge pixels are treated as *WHITE*. In practice, for frame difference images, the values of pixels have a range between 0 and 255. We define that the pixels whose values over 127 are *WHITE* and others are *BLACK*. This operation is similar to quantifying the frame difference image into two levels and multiplying it with the edge image. So we call it a MUL operation.

After the foreground image  $F$  is generated, we apply GDT&OM template matching algorithm to  $F$ . Since edges of the background have been cancelled, its GDT image mainly contains distance information on pixels to edges of moving objects. The orientations of the templates are pre-computed as well. Figure 5.6 shows all of the results of the entire process.



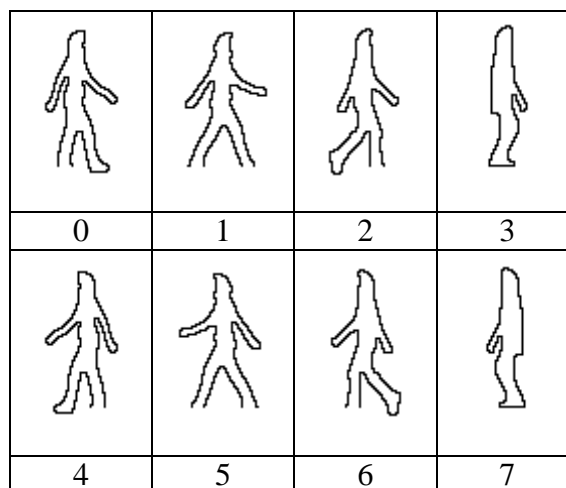
**Figure 5.6 GDT&OM result of each step**

**(a) the current frame; (b) the edge image of a, produced by Canny edge detection; (c) the subtracted image between current and previous frames; (d) the foreground edge image, produced by b and c with a MUL operation; (e) the final matching result, the matched area is boxed by a white rectangle; (f) matching result using the edge image without background cancellation.**

The background of this image sequence is complicated, especially the part behind the walking person. As Figure 5.6f indicates, it is difficult to match the person if the background cancellation is not applied. After the background cancellation process above, there is only the contour of the moving object left, and the location of the person is matched correctly.

### 5.2.2 Template Modification

Since we applied background cancellation before the template matching, the static edges in the sequences are all erased, including part of the foreground. For example, in Figure 5.5d, the person's left foot that contacts the floor is also cancelled in the foreground edge image, because it does not move between current and previous frames. Considering the motion of human, normally some parts of the body will keep still in a short period, and these parts will be cancelled in the background cancellation. Hence, these body parts in the templates also need to be cancelled to reduce the matching differences. Figure 5.7 indicates some examples of the modified templates.



**Figure 5.7 Modified templates**

Our template matching method is considered as an approach that can detect human bodies and recognize their poses in the same time. A human motion, for example walking, will be decomposed into  $n$  states. Note that motions such as walking and running are repetitive, so here ‘ $n$  states’ means a number of different poses in a single iteration. Such as the set of templates shown in Figure 5.7, walking is



decomposed into 4 states (another 4 as reflections). If we matching the images by using the sets of templates like that, not only the position of human bodies can be found, but the motion state in every frame can be pointed out as well. This template set is used in the detection of Figure 5.6, in which the pose of the person is matched correctly to Template 2. When multi sets of motion styles of templates are employed in the process, the state information can be used in state based motion recognition approaches, such as HMM or Bayesian networks.

As same as other detection methods, we move a window on a test image, extract the content in each window, scale it to the same size as the template, and compute its matching score. We make a set of templates for each motion category, and pool all of the templates together. For an extracted window, since the test images only have a single person, we match it with every template and collect the match that has the smallest score. If test samples have multiple human bodies, thresholds are required to select a number of good matches, and then a merging method, such as described in Chapter 4, is required to determine final detections.

In the experiments, two datasets are mainly used as testing samples. One is the KTH dataset which includes 6 motion categories, 25 people in each motion category and 4 different camera footage clips for every person. There are 600 frame sequences in total. We split the data into two groups: the first group includes walking, jogging and running in which most body parts are moving; the second group includes boxing, hand waving and hand clapping in which most body parts are stationary. The other dataset is the Weizmann dataset, including 10 motion categories and 9 people. We also divide them into two groups: walking, running, jacking, jumping, pjumping (jumping at a same place), skipping and galloping for group one, and bending, one-hand waving and two-hand waving for group 2. These categories are: jumping in a same place, one-hand and two-hand waving, bending. Finally 56 sequences are used (some persons have two videos for a motion).

To make a judgment that a matching is correct or not, we follow the rule: if the whole human body is in the result rectangle and the body matches the template that represents the same pose, as displayed under the detection window in Figure 5.6e,

we determine it is a correct match. Otherwise, an incorrect match is issued. In practice a human pose may be between poses represented by two serial templates. In that case we will count a correct match when the frame matches either of the two templates.

As mentioned above, a comparison result between modified and non-modified templates is shown as below:

<b>Dataset</b>	<b>Accuracy of Using Modified Templates</b>	<b>Accuracy of Using Non-modified Templates</b>
KTH group 1	83.78%	78.12%
KTH group 2	5.47%	80.91%
Weizmann group 1	85.11%	83.52%
Weizmann group 2	9.64%	79.72%

**Table 5.1 Matching results of using modified and non-modified templates**

The reason why the accuracies for KTH group 2 and Weizmann group 2 is so low when using modified templates is that the actions of these two groups require most part of human bodies to be stationary. Therefore the modified templates lose too much information of human outlines. It can be seen that for motions that full bodies have displacement, the matching accuracies are raised by using modified templates. This is because the templates are more close to the contour of the moving bodies.

However, this human detection and pose estimation is constrained, as the problems of signal-processing based detection methods discussed at the beginning of this chapter. Besides, there are a number of weaknesses:

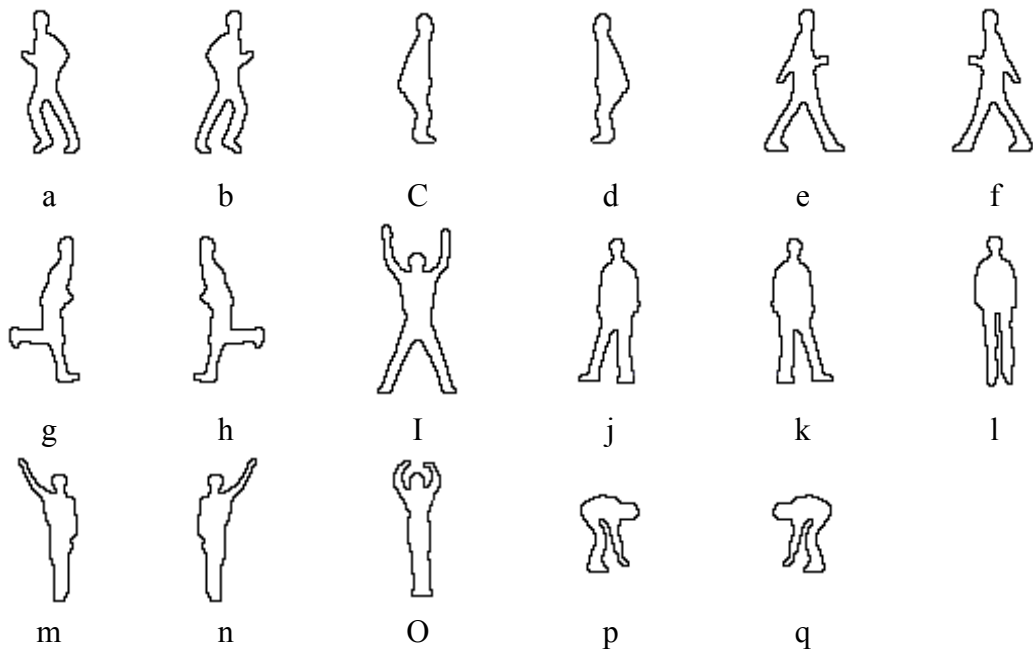
- 1) This scheme requires the input to be a frame sequence taken by a static camera so that the background can be cancelled. Otherwise the accuracy will be poor.
- 2) Human bodies can only be detected when most part of them are moving.
- 3) It is difficult to detect multiple persons with overlapping regions, because the situations of overlapping regions are unpredictable for building templates.

Therefore, we do not use GDT&OM to detect human bodies, but to estimate human poses after a relatively efficient and reliable detection.

### **5.3 Template Matching for Pose Estimation**

In the proposed system, we detect human bodies by applying an HOG detector which is depending on neither frame sequences nor moving objects. Meanwhile, the HOG detector can solve the overlapping problem well because we train it by images under overlapping situation. In this section we will introduce applying the template matching method to pose estimation, assuming that human bodies have already been correctly extracted.

Given a window which contains a detected human body, we scale it to the same size of the templates, and perform a GDT&OM template matching on it. Then we select the match with the smallest score as the best match. So the pose that the template of the best match represents is determined as the estimation result. A part of the templates are shown in Figure 5.8. Since the background cancelation mechanism is not applied, we use complete templates without erasing part of their structures. The templates of jacking are not paired due to their symmetrical structures.



**Figure 5.8** Template set for GDT&OM template matching:

**(a) and (b) running; (c) and (d) jumping; (e) and (f) walking; (g) and (h) skipping; (i) jacking; (j) and (k) galloping sideways; (l) jumping at a same place; (m) and (n) one-hand waving; (o) two-hand waving; (p) and (q) bending.**

We use 18000 frames from KTH dataset and 5016 frames from Weizmann dataset as test images. The human bodies in these frames can be correctly detected by the improved HOG detector. For each frame sequence, after performing template matching, we select the top  $n$  best matched frames, i.e. the frames have minimum matching scores, and execute a simple voting algorithm that count the numbers of frames according to their matched action type. The type that has the maximum number of corresponding frames will be determined as the action type of the frame sequence. If two or more action types have the same number of corresponding frames, the system is designed to compare the sums of the matching scores of the corresponding frames. The action type has the minimum sum will be selected as the final result. Table 5.2 indicates the result action classification.

$n$	1	3	5	7
KTH	81.50%	83.67%	91.00%	89.83%
Weizmann	78.89%	85.56%	94.44%	91.11%

**Table 5.2** The result of action recognition with different  $n$

Table 5.2 indicates that when  $n=5$ , the system reaches the best classification accuracy, for both of the two datasets.

Table 5.3 displays the recognition results for each action category respectively. Since human bodies are well located, the accuracies of pose estimation are relatively good. The jumping category in Weizmann gets the lowest accuracy as the jumping frames are influenced by templates of other categories strongly. We also see that using more motion categories will pull down the total estimation accuracy. The reason is that there are interferences between templates of different categories. For frames of jumping which have relatively simple edges, the interference from other motion categories are much more significant. On the other side jacking frames are recognised best because jacking is the most different from other motions and has the least interferences. That is a reason why we argue that selecting templates is very difficult, because it is difficult to predict interferences when we add a set of templates. In addition, using a huge number of templates reduces processing speed as well. So we conclude that in practice, it is better to approximately predict which motion categories will be used and select templates for those categories. If an exception occurs, just simply decline it as it is not concerned in the application.

KTH		Weizmann	
Action type	True classifications	Action type	True classifications
Walking	94	Walking	9
Jogging	86	Running	9
Running	89	Jumping	6
Boxing	95	Skipping	8
Waving	92	Jacking	8
Clapping	90	Galloping	9
		Pjumping	9
		One-waving	9
		Two-waving	9
		Bending	9

**Table 5.3 True numbers of classifications for each action type ( $n=5$ )**

We compare the result with two recent work on action recognition: Poppe's method [27] which uses the Weizmann dataset, as well as Jhuang's method [76], which uses both the KTH and Weizmann datasets. In Table 5.4 the three methods are compared. Table 5.4 shows that compared with other method, the method proposed in the thesis can achieve a same level of accuracy.

	KTH	Weizmann
Poppe	/	95.56%
Jhuang	91.67%	98.78%
Ours	91.00%	94.44%

**Table 5.4 A comparison with other methods**

## 5.4 Estimating Poses by Locations of Body Parts

Another idea of estimating human poses is using location information of human body parts. In the point of view that poses are essentially distinguished by differences of body parts locations and orientations, we argue that if the locations of all body parts of a person are given, the pose of the person can be recognised. The recognition procedure is designed to have 3 stages: 1) detecting human bodies, 2) locating human body parts, and 3) classifying the pose.

For Stage 1, human detectors such as HOG can be employed as we do in Chapter 4. For Stage 2, the candidate is Ramanan's iterative parsing method [58]. However, this method does not work very successfully. The biggest factor limiting its performance is that hidden body parts cannot be accurately located. As known to all, 2D images sometimes do not display all parts of a human body and it is not possible to get the information about hidden parts from images themselves. The problem is that body parts estimators do not know which part is hidden in the images and false estimation will be made. However for Stage 3, accurate body locations are needed. That is the reason why this type of approach is not so popular at present.

We have made contributions to Stage 3, under the assumption that all body parts are

accurately located. We use several machine learning methods to learn models from training data, which are a set of body parts locations extracted from training samples, and classify test data which are also body parts locations extracted from test samples.

#### 5.4.1 Preparing experimental data

We locate human body parts manually to ensure reliable experimental data. We use a stickman model to mark 11 key points of each human body. These key points are: head, chest, hip, two elbows, two wrists, two knees and two ankles, as Figure 5.9 indicates.



**Figure 5.9 An example of marking body part locations**

Then the head location is used as the datum point, and the relative locations of other key points can be calculated. Except the head, these body parts are connections or terminals of a body, so these relative locations can represent locations and orientations of all body parts. We put each set of relative locations into a vector in the same order, so each vector has 20 members (a relative location is a two dimensional value  $(x,y)$ ), and is called a location sample.

We use the Weizmann dataset in the experiment. The six action categories are the same as the experiment in Section 5.3: running, walking, jumping, skipping, jacking and galloping. 2038 images are selected as training data and 560 images are used as test data.

#### 5.4.2 Model training and testing

We employ the “Spider” as the experiment tool. Spider [113] is a Matlab-based

machine learning tool box that implements and integrates many popular machine learning methods. The experiment in this section is a multi-class case (in contrast of two-class classification, multi-class classification means the number of object classes is more than two), therefore we employ three machine learning methods:  $k$ -nearest-neighbourhood ( $k$ - $NN$ ) [114], one-vs-rest [115], multi-class SVM [115].

The method of estimating distance between two samples is a key factor of SVM. We use linear, Gaussian and RBF kernels [60] for one-vs-rest and multi-class SVM which are SVM based methods. Meanwhile both Gaussian and RBF kernels have a parameter  $\sigma$ . We let  $\sigma=1, 2$ , which are the most common values. For  $k$ - $NN$ , we test the parameter  $k=1, 3, 5, 10$ . The result is shown in Table 5.3.

k-nearest-neighbourhood				
k	1	3	5	10
class-loss	0.0018	0.0009	0.0098	0.0473

(a)

One-vs-rest					
Kernel	linear	Gaussian		RBF	
Parameter $\sigma$		1	2	1	2
class-loss	0.3946	0.7366	0.5134	0.0768	0.0263

(b)

Multi-class SVM					
Kernel	linear	Gaussian		RBF	
Parameter $\sigma$		1	2	1	2
class-loss	0.2687	0.7634	0.7741	0.0554	0.0054

(c)

**Table 5.5 The result of body-part based pose estimation**

The term class-loss which used in Spider in Table 5.3 is the false rate of classification.  $K$ - $NN$  performs well especially when  $k$  is small. It means that test samples usually find the nearest neighbourhoods which are in the same categories in



the training set. But when concerning more neighbourhoods, influence from other categories becomes larger and causes more false decisions.

For one-vs-rest and multi-class SVM, it is proved that the linear kernel is not good and the Gaussian kernel is totally unsuitable. However the RBF kernel appears to be appropriate for these two methods.

## 5.5 Conclusions and Discussion

GDT&OM template matching, pre-processed by background cancellation, combines human body detection and pose/state estimation in a single step, which can simplify the state-based motion recognition.

Background cancellation is used on the sequences with static background. This mechanism cancels the negative impact brought by the complicated backgrounds. Only the edges of moving objects, considered as foreground, are left. After GDT, these edges will be strongly emphasised. Working with the background cancellation mechanism, templates need to be modified. The still parts of bodies are erased in order to achieve smaller matching differences. Processed by the methods above, the performance of the matching program achieved a relative improvement for the videos of full-body-moving people, which is proved by the experiment.

However, this method is constrained. First, this method can only apply to frame sequences. Second, it is hugely depends on the selection of the templates. Meanwhile, along with the increasing number of templates, the interferences between templates grow quickly. More mismatched result will come out when we employ more templates, and the processing speed will be slow down as well. Therefore, some other mechanisms, such as histograms of oriented gradients (HOG), are used to pre-process for detection. In addition, the modified templates do not work for the videos showing persons whose bodies are partly stationary.

When we use GDT&OM for pose estimation after a reliable human detector, it can achieve a good performance for either static images or frame sequences in the experiment. However, the problems of multiple motion categories and interferences between templates still exist. Hence a prediction of using template set is the key.

In conclusion, although GDT&OM can only be applied to a limited range of human detection and pose estimation, it is relatively simple and fast. The applications of GDT&OM will be introduced in Chapter 6.

We have also implemented a body-part-based pose estimator. Assuming that all body parts can be accurately located, it is possible to do highly accurate pose classification by using multi-class machine learning methods. This research direction has a large potential and now the key factor is to solve the problem of hidden parts localisation. Therefore developing a reliable body part estimator will be a topic in the further work. It is complicated and difficult to solve the problem of locating hidden parts of human bodies, and there are two issues to work out. First, since hidden body parts are unpredictable, we have to use a full body model to locate body parts. Therefore a method that determines which parts are hidden is required. Second, we have to train an observed-hidden model to estimate locations of hidden parts according to observed parts, and use a fault-tolerant classifier to determine action categories. Essentially, training an observed-hidden model equals to training a pose estimation model using observed body parts, so poses can be estimated by only using locations of observed body parts.

# Chapter 6

## Practical Applications

---

As mentioned before, human motion analysis systems can be widely applied to a lot of areas in computer vision and image processing. In this chapter a key-frame extraction method and a video logging system employing the proposed detection and estimation methods are introduced.

### 6.1 Key-frame Extraction System

Key-frame extraction, a topic developing computer vision methods that select one or a small number of frames to represent a long frame sequence, is usually used in movie producing, video summarisation and storyboard making. For example, DVDs or online videos usually represent the content of a long video by displaying a few pictures. That is what a key-frame extractor is designed to do. Generally, one key-frame is selected for each shot. Conventional methods of automatic key-frame extraction analyse motion factors between frames, such as moving velocities or accelerations of image regions (not objects), and select key-frames based on these factors. These methods are considered as non-object-based because they do not concern objects or contents represented by videos. Thus sometimes these methods will extract key-frames which have strong motion factor features but are not representative for contents, in other words, are not understandable for people. Considering that key-frame extraction in practical applications is primarily used to help users to understand video contents, object-based methods are required. Our

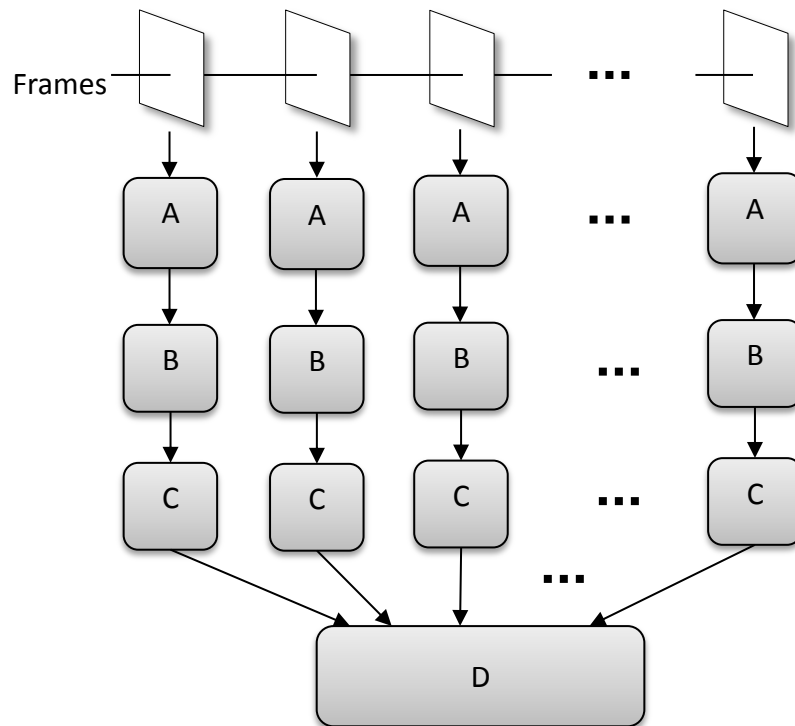
method, however, aims to select a frame which has a human pose being able to represent the human action best. Meanwhile, the action type of a frame sequence can also be determined because every selected key-frame reflects a class of action. In addition, this method can also be extended to other object classes. So our method can be defined as an object-based method.

Of course, “key-frame” is a concept that has a lot of subjective factors. In other words, determining whether a key-frame is a success or failure strongly depends on individual opinions. In this chapter, we also introduce a conventional key-frame extraction method as a comparison. We compare the results between this method and our method according to the rule that for a frame sequence the key-frame which can represent the content of a frame better is the winner.

Our system which is designed to process videos representing human actions, contains four procedures for extracting the key-frame from an input frame sequence:

- 1) Each frame is processed by HOG human detector to detect human bodies.
- 2) Detected human bodies (in fact these are image regions containing human bodies) are scaled to the same size as the prepared templates.
- 3) Each human body is matched with the template set by GDT&OM and the best match is found (smallest matching score).
- 4) All frames’ best matching scores are compared and the smallest is chosen. The corresponding frame is selected as the key-frame.

In the experiments, we use the Weizmann dataset [74] as the test dataset. This dataset contains videos of full human bodies without overlapping, which is suitable for our detection and matching methods. Each video in the dataset represents a single shot and a single person performing a single action type. Besides, it contains not only some typical common action types, such as walking and running, but some abnormal action types such as jacking and skipping as well.



**Figure 6.1** The diagram of the key-frame extraction system

**A:** HOG human detection; **B:** Scaling the detected windows to the same size of the templates; **C:** GDT&OM template matching to find the best match; **D:** select the frame with the smallest matching score as the key-frame.

### 6.1.1 HOG human detection

We used an HOG human detector to locate human bodies in frames. There are some benefits of using an HOG detector:

- 1) Detection performance is relatively high – high true detection rate against low false rate.
- 2) Scaling and merging problems can be solved.
- 3) Since we used a large number of training images, the detector's compatibility for multiple action classes is high, which means that human bodies in abnormal or even weird poses can be detected.

However the use of the HOG detector is still conditional. We use vertical human bodies as training images. Therefore sitting or lying bodies cannot be detected. Besides, bodies smaller than the size of training images cannot be detected either.

Training data consists of training images of the INRIA dataset and part of human body windows from the Weizmann dataset, because this dataset includes some abnormal poses. Finally we have 2812 training images. The test data includes 9 persons and 6 action classes, totally 3278 frames. Because the test images are simple (simple backgrounds, single person, no overlapping), we set the merging threshold  $t$  to 0.00, in order to achieve a maximum true positive detection number. Table 6.1 shows the result.

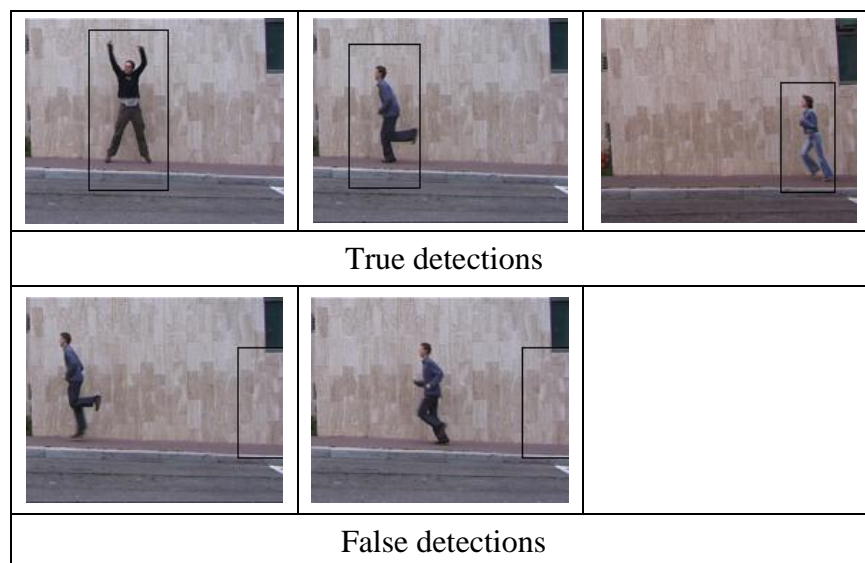
	Jacking	Jumping	Running	Skipping	Walking	Gallop ing
Number of frames	729	458	405	494	711	481
Detected bodies	691	419	397	413	694	468
Detectio n rate	94.79%	91.48%	98.02%	83.60%	97.61%	97.30%
	Pjumping	One- waving	Two- waving	Bending		Total
Number of frames	454	438	445	401		5016
Detected bodies	431	408	423	337		3082
Detectio n rate	93.50%	93.15%	95.06%	84.04%		93.12%

**Table 6.1 The HOG result for the Weizmann dataset**

According to Table 6.1, most action classes can be well detected. The detection for skipping and bending images is the worst detection because skipping is an abnormal pose which has only a small number of training images. In addition, there are only 13 false positive detections in all in the final result, which is not indicated in Table 6.1. Compared with the detection results presented in Chapter 4, the performance is significantly better for the Weizmann dataset, due to its simple backgrounds and no

overlapping, and such detection rate can satisfy the further processing steps. Figure 6.2 shows examples of true and false detections.

Figure 6.2 indicates a few detecting results. False detections which happen rarely, classify the background region with texture similar to a human body as a person. It should be pointed out that the second true detection and the first false detection have similar bodies. It indicates that even very small variation (such as shape, luminance) between two image region may cause different detection results.



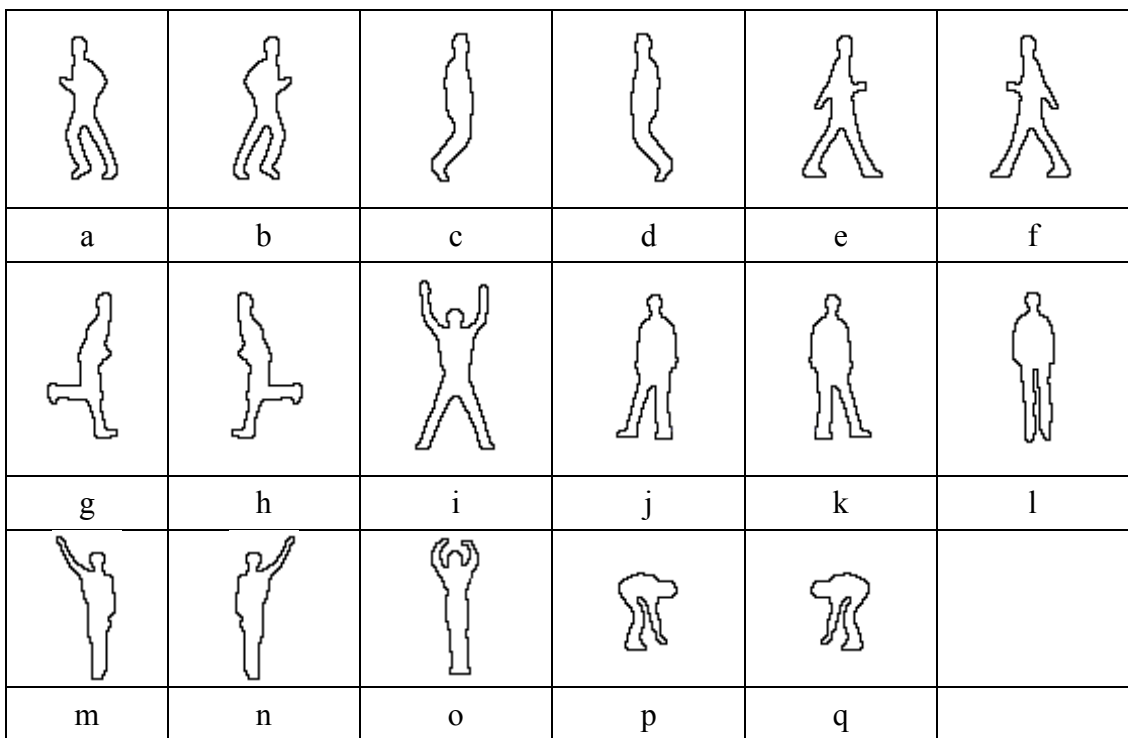
**Figure 6.2 Examples of true and false detections**

### 6.1.2 GDT&OM template matching

Since there is no clear definition of what a key-frame of a sequence should be, selecting a frame that understandable for a human can also be considered as a choice. That is to say, people can understand what happens represented by a sequence only by seeing its key-frame. For applications such as movie producing and storyboard making, template matching can be a good approach. Our method is mainly applied for the sequences that contain moving human bodies.

We use the Weizmann dataset as the test dataset, which includes ten action classes. We select six of them which are running, walking, jumping, skipping, jacking and

side moving, because these six classes are stronger action types. We select a set of templates from all poses of the six action classes. Each action class has a pair of templates to represent two different moving directions, except jacking, whose templates are all symmetrical. The selected templates are considered to be significantly different from each other so that each of them can represent its action class uniquely. In the experiment, we built the template set as Figure 6.3:



**Figure 6.3** Template set for key frame selection

**(a) and (b) running; (c) and (d) jumping; (e) and (f) walking; (g) and (h) skipping; (i) jacking; (j) and (k) galloping sideways; (l) jumping at a same place; (m) and (n) one-hand waving; (o) two-hand waving; (p) and (q) bending.**

The detected image regions are scaled to the same size of the templates. In practice the detection windows have some margins between human bodies and window boundaries, but the templates don't, so the detection window will be slightly larger than the templates. We execute a quick scan inside the detection window by the templates. The reason why we do such procedure is that sometimes the body of a detection result is not just at the centre position of the window due to different body poses and the merging process. If we used the same sized templates, the edges of the



templates and the bodies in the windows would have a small dislocation, which leads to inaccurate matching results. If we employ smaller templates, scan inside the detection window, and finally select the best match, the result will be much more accurate. False detections have little influence on matching results, because these detections are far different from human body shapes and will get poor matching scores. Since the set of templates are only a part of all poses, it is pointless to discuss matching accuracy. What we want to see is that how well selected key frames represent their corresponding videos. The final result will be illustrated in Section 6.4.

### 6.1.3 Perceived motion energy

Before we present the final result, we introduce a comparison method. We use a method based on perceived motion energy model (PME) proposed by Liu, Zhang and Qi [20] as the comparison method. This method implements a conventional idea that the system selects frames in which the objects have largest moving speed and smallest acceleration, in other words, the largest motion energy, as the key-frame. They calculated the PME of a frame by accumulating the motion vectors between two serial frames, and used a triangle model to determine the maximum PMEs.

According to the paper [20], PME is defined as “a combined metric of motion intensity and motion characteristics with emphasis on dominant motion”. In other words, in a video all motion is projected onto the dominant motion and the sum is computed as the measurement of motion energy. Since the PME method is based on MPEG video compression, each backward frame (B frame) in a MPEG video is segmented into a number of macro blocks with a certain size (e.g.  $8 \times 8$  or  $16 \times 16$ ). Each macro block is used to compute motion vectors and then to form motion vector fields. The magnitude of a macro block reflects the motion intensity of the block and the accumulation of the magnitudes can reflect the global motion intensity of the entire frame.

Denote  $\text{Mag}(t)$  as the average magnitude of motion vectors in a frame, and it can be calculated by:

$$\text{Mag}(t) = \frac{\left( \frac{\sum \text{MixFEn}_{i,j}(t)}{N} + \frac{\sum \text{MixBEn}_{i,j}(t)}{N} \right)}{2} \quad (6.1)$$

where  $N$  is the number of micro blocks in the frame,  $\text{MixFEn}_{i,j}(t)$  represents mixture energy values of forward motion vectors and  $\text{MixBEn}_{i,j}(t)$  indicates those of backward motion vectors.  $(i,j)$  is the position of a macro block in the frame.  $\text{MixFEn}_{i,j}(t)$  and  $\text{MixBEn}_{i,j}(t)$  are generated from vector magnitudes  $\text{Mag}_{i,j}$ , by passing a set of filters to remove noisy and atypical vectors. First, we use a spatial window with the size of  $W_s$  ( $W_s$  by  $W_s$  macro blocks) to filter the magnitudes by removing small values:

$$\text{Mag}_{i,j} = \begin{cases} \text{Mag}_{i,j}, & \text{if } \text{Mag}_{i,j} \leq \text{Max4th}(\text{Mag}_k) \\ \text{Max4th}(\text{Mag}_k), & \text{if } \text{Mag}_{i,j} > \text{Max4th}(\text{Mag}_k) \end{cases} \quad (6.2)$$

where  $\text{Max4th}(\text{Mag}_k)$  represents the fourth largest value of magnitudes in the window. Then a temporal filter is applied as:

$$\text{MixEn}_{i,j} = \frac{1}{(M - 2 \times \beta M) \times W_t^2} \sum_{m=\beta M+1}^{M-\beta M} \text{Mag}_{i,j}(m) \quad (6.3)$$

The equation above illustrate a filter that uses a temporal window with the size of  $W_t$  ( $W_t$  by  $W_t$  macro blocks) to smooth the magnitudes into mixture energies.  $M$  indicates the number of magnitudes in the window and  $\beta$  ( $0 \leq \beta \leq 0.5$ ) is the trimming parameter. All magnitudes in the window are sorted, and  $\text{Mag}_{i,j}(m)$  represents the  $m$ th largest one.

After mixture energies of the F frame and B frame of a frame are achieved, then PME can be computed by:

$$\text{PME}(t) = \text{Mag}(t) \times \alpha(t) \quad (6.4)$$

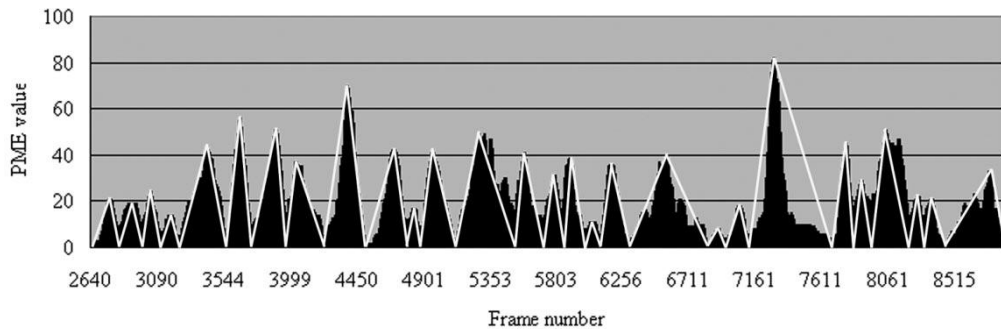
where  $\alpha(t)$  indicates the proportion of dominate motion direction. Liu et al. evenly quantised angles within  $(0, 2\pi]$  into  $n$  bins, and counted the number of angles in each bin, denoted as  $AH(t,k)$ , and  $k \in [1, n]$ . The bin with maximum value of  $AH(t,k)$  is

selected as the dominant direction. So  $\alpha(t)$  can be computed by:

$$\alpha(t) = \frac{\max(AH(t, k))}{\sum_{i=1}^n AH(t, i)} \quad (6.5)$$

Since the videos used are uncompressed, for a frame, we set its previous frame as the backward frame and its next frame as the forward frame (F frame). We implemented the method by setting the macro block size to  $8 \times 8$ , the filter window sizes  $W_f=4$ ,  $W_s=4$  and the number of angle bins  $n=18$ .

In Liu's work, they represented PME values in diagrams and perform a triangle mechanism to estimate peak PMEs. Then corresponding frames of these peak PMEs are selected as key frames, as Figure 6.4 indicates. However, we aim to select a key frame for each shot, so we just select the frame with the largest PME of a frame sequence.



**Figure 6.4 Triangle mechanism, from the paper [20]**

#### 6.1.4 Experimental results

When processing a frame sequence, after we achieve the best matches for each frame, we select the frame with smallest matching score, in other words, the most similar to its matching template, as the key-frame. Meanwhile, the action class of the frame sequence can be determined as what the best matching template represents. Here is a comparison result between PME and our method in Figure 6.5. Only two class

actions – walking and running, which have significant differences between the results for the PME method and our method – are presented. For other motion categories, the differences of the results between the two methods are very small, so the results are not listed below.






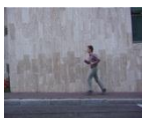





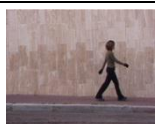
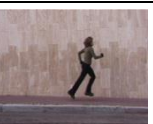




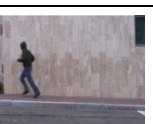
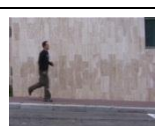
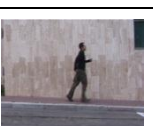
Sequences	PME	Our method	Sequences	PME	Our method
Daria walk			Daria run		
Denis walk			Denis run		
Eli walk			Eli run		
Ido walk			Ido run		
Ira walk			Ira run		
Lena walk1			Lena run1		
Lena walk2			Lena run2		
Lyova walk			Lyova run		
Moshe walk			Moshe run		
Shahar walk			Shahar run		

Figure 6.5 Comparison of key-frames extracted by PME and template matching

As argued above, deciding whether a key-frame is good or bad is subjective. According to Figure 6.5, most key-frames of the two methods are satisfying. However, under the rule that “a key-frame should represent the content of the frame sequence”, we argue that our method performs better for a few of these frame sequences.

For walking sequences, the key-frames extracted by the PME method appear to have different poses. While those extracted by our method can be considered as in the same pose. Some key-frames from template matching are easier to be recognised as walking, such as the sequences named ‘Ido walk’ ‘Lena walk1’ and ‘Lena walk2’.

For running sequences, the results between the two methods are closer. But there are still a few key-frames extracted by our method can be considered as better ones, such as ‘Moshe run’ and ‘Shahar run’.

	Jacking	Jumping	Running	Skipping	Walking	Galloping
PME	6	8	8	8	7	7
Ours	9	7	10	8	10	7
Total	9	9	10	10	10	9
	Pjumping	One-waving	Two-waving	Bending		Total
PME	9	9	9	7		78
Ours	9	9	9	9		87
Total	9	9	9	9		92

**Table 6.2 Comparison of “good” key-frames achieved**

Table 6.2 lists how many “good” key-frames achieved by both PME and our method. PME is more like a random selection in the view of frame content, and only performs better than our method for jumping images. Generally speaking, according to the table, our method achieves more understandable key-frames than PME.



**Figure 6.6 Some “bad” key-frames**

Figure 6.6 presents three “bad” key-frames produced by our method. We realise that “bad” key-frames are caused by incorrect classifications. That is to say, if the result is a mismatch, which means the key-frame matches an incorrect template, a bad select will usually occur. Considering Figure 6.6 A and B, the two poses appear very similar to the template of running. This is a major problem of template matching. It is unavoidable that when the number of classes increases, some templates will match poses of other action types well, and frames incorrectly matched will be selected as key-frames which cannot represent the contents of frame sequences. Meanwhile as we integrate more and more templates of different actions, the interference between templates is becoming stronger and stronger. That is to say, it is becoming easier to get a mismatched result as the number of templates increases. A bad template selection will lead to a bad extraction result.

A solution for the mismatching problem of template matching is to select a few top-matching frames, and get the key-frame through a kind of voting mechanism. For example, if the first matching is running, but the second and third are walking, a voting mechanism with weights for the top three matching frames can allow an accurate decision to be. Figure 6.6C illustrates an absolutely incorrect match, in which obviously the matching of one of the legs is lost but this frame is still selected as the best match. This error result reflects another factor that has significant influence on our method. It appears that the template matches a part of the human body well and then the system determines the frame as a good match. This problem is caused by using edges in templates to judge matching performance. However it is

impossible to judge matching performance according to edges of test images, because extra edges from backgrounds or other textures will cause more harm on matching results. We call it a “judgement paradox”. Using templates similar to images like Figure 6.6C can relieve this problem. However, only a limited number of templates can be used due to the characteristic of this key-frame extraction system.

## 6.2 Video Logging

The key-frame extraction system above identifies and represents one type of significant action per shot. The system can also be applied to longer footages that contains multiple action types to do video logging [96]. A video logging system finds several remarkable poses, and records them to represent the content of the video. If the motion of the person consists of routine action types the log can be a precise summary of the video. Gymnastic videos meet the requirement above, and this kind of footage usually has fixed camera angles and focuses on the performance of one person.

Motion analysis methods for sports in prior art usually segment objects by their motion information. The paper [116] presented by Luo et al. illustrates a method that detects differences in a video between two continuous frames to extract moving objects, abstracts the video into a number of clusters, and uses a Bayesian Network to analysis the motion pattern. Li et al. presented a method [117] [118] that for each clip extracts human bodies by estimating motion energy between two frames, selects frames that have motion energy larger than a threshold as key-frames, and finally uses an HMM to recognise athletes’ actions. However, sports behaviours do not consist only of dynamic motions but static poses as well. Meanwhile the intensity of the motion energy of a frame cannot always represent its significance, as we have stated for the key-frame extraction method. Therefore extracting human bodies by motion features will cause losses of important information, such as static poses in some sports. Our method is object-based and focuses on detecting frames

representing important content, so both dynamic and static actions can be detected.

Here we do experiments on videos of rings and balance beam. These two types of gymnastics represent different features of this sport. In gymnastics, an element is defined as an action or a skill that is used to evaluate performance, and a routine is a series of skills in addition to graceful dance performed by a gymnast in a range of time (usually about one minute). Routines of rings are only performed by male gymnasts, and most elements of rings such as handstand and iron cross are required to be held for a few seconds. In contrast balance beam is only for female gymnasts, and most elements are dynamic, such as leap and somersault. Hence different mechanisms are employed to make video logs. These important elements in both rings and balance beam videos are called “probed elements” in this thesis. Figure 6.7 displays examples of frames in videos of rings and balance beam.

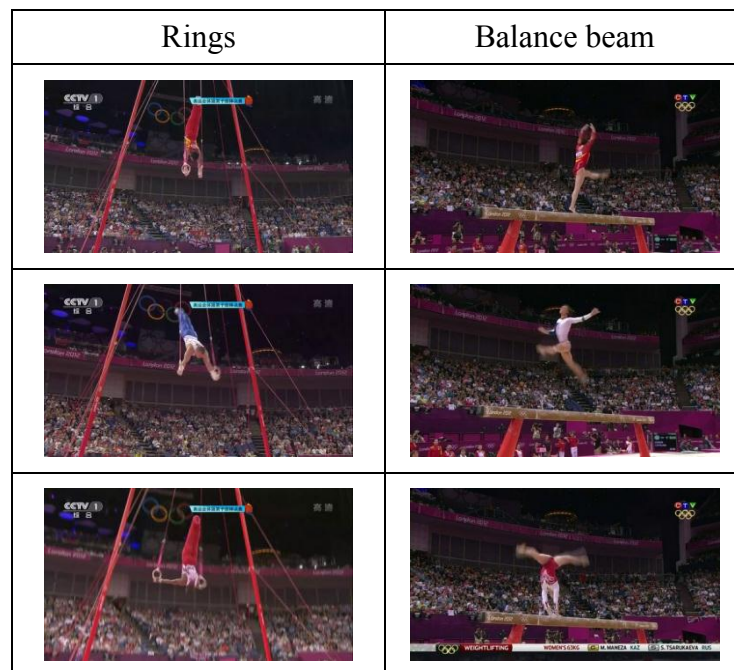


Figure 6.7 Examples of gymnastic images

### 6.2.1 HOG human detection



The procedure of human detection is as the same as that in key-frame extraction. To build a suitable detector for gymnasts, we employ training images extracted from videos of gymnastic competitions. Considering that gymnastic poses do not have a fixed ratio between width and height (for example, a body performing handstand has a height much larger than its width, performing leap may have its height smaller than the width, and performing iron cross has a similar height and width), we use a window size of  $96 \times 96$  (using a much larger window size than that costs much more memory and slows down the processing speed).

For rings, we have prepared 4102 positive training images, each of which represents a gymnast performing a pose of rings, and large 1166 negative images that represent background, non-human objects, body parts and other people. For balance beam, we have 2788 positive training images which contain gymnasts on the balance beam, and use the same negative set as rings. The test sets for both rings and balance beam contains 8 clips, each of which displays a complete routine of a gymnast. We design to use videos from 2008 Beijing Olympic as the training data, and those from 2012 London Olympic as the test data. Therefore all positive training images are extracted from videos of 2008, as are most negative training images. In the training phase, the only different parameter from the original HOG is we random extract 20 windows instead of 10 from each negative image, to ensure that sufficient negative information is included.

Then in the test phase, a problem occurs when the data of rings is applied: bodies in the poses of hanging and handstand are totally missed, even though we have included hanging and handstand poses in the training data. To solve this problem, we execute the following mechanism: pick the training images of hanging and handstand out and use them to train a separate detector (detector A), and the rest are used to train another detector (detector B). Then we apply both the two detectors to test data. In that case, the hanging and handstand bodies can be found correctly by detector A and other bodies can be located by detector B. Then we output the union of detector A and B's results. In contrast, the detector for balance beam works well. Figure 6.8 indicates the detection results.

Figure 6.8 shows the variation and relationship of FPR and TPR, both of which are influenced by the merging threshold  $t$ , which varies from 0.00 to 0.10. FPR and TPR decrease when  $t$  rises. For rings,  $t$  makes significant influence on both TPR and FPR. While for balance beam,  $t$  causes huge influence on FPR but little on TPR.

When we check the output of HOG detection, we find that most missed detections are bodies in abnormal poses. In the further process, the templates used represent regular poses, so the frames of abnormal poses are not useful and can be ignored. Hence we set  $t$  to 0.10 in order to reject as many false detections as possible.

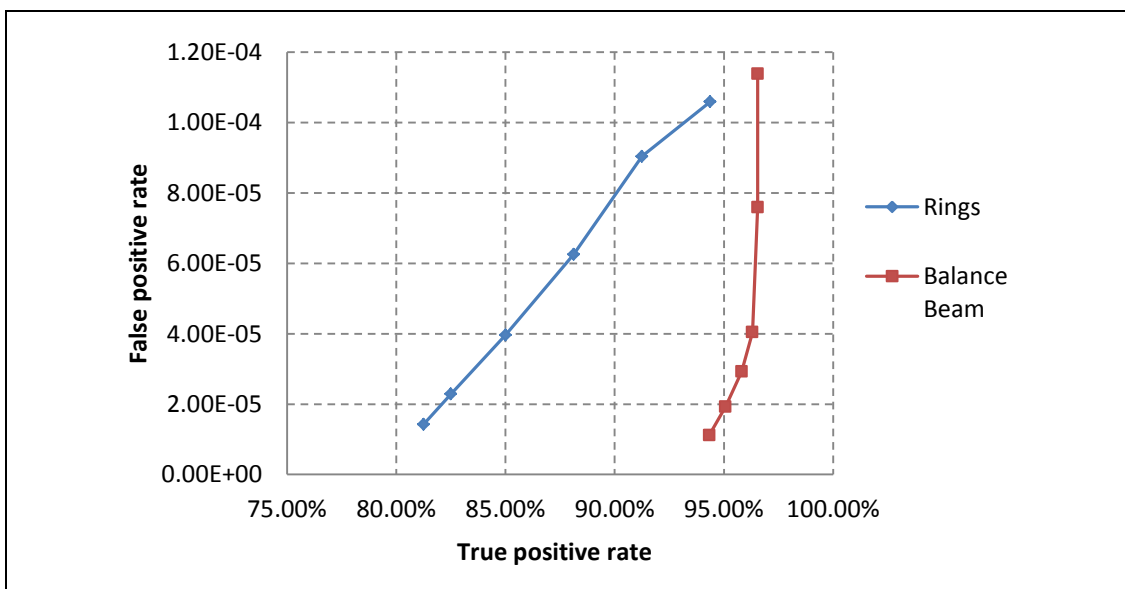



























Figure 6.8 Detection results

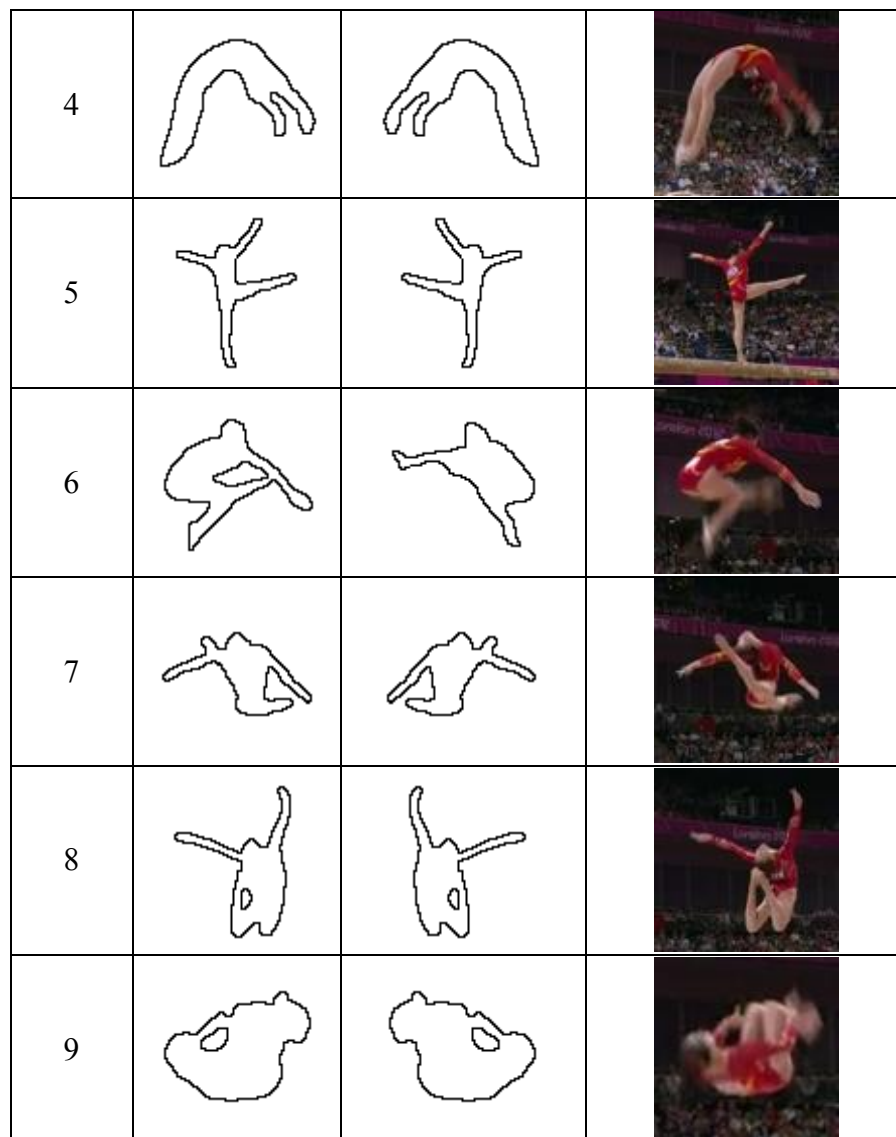
### 6.2.2 Template matching

We build template sets for rings and balance beam separately. For rings, 8 templates are used. Each template stands for a pose that is important for marking and requires gymnasts to keep for a short while in routines. While for balance beam, 10 pairs of templates are made. Each pair of templates represents a remarkable pose in two directions. Figure 6.9 shows all the templates used in the experiment. The numbers under the pictures are the labels of the templates, and will be used later in Section 6.2.3.

			
0		4	
			
1		5	
			
2		6	
			
3		7	

(a) Rings

1			
2			
3			



(b) Balance beam

**Figure 6.9 Templates used for logging gymnastic videos**

The matching procedure is as the same as key-frame extraction. Detected regions are scale to the size of HOG training images (96×96), converted to GDT and OM data, scanned by each template and finally obtain the best match. Matching labels and scores are exported for further processes. Like key-frame extraction, the matching accuracy does not make sense here, because templates used cannot represent all types of poses. Post-processing is required to summarise the matching results into video logs.

### 6.2.3 Producing video logs

As mentioned above, template matching cannot recognise all types of poses, and the goal of this section is to extract those important poses. Since the features of rings and balance beam are different, the post-processing mechanisms are also different for the two types of sport.

For rings, a routine is treated as a combination of several static elements which last for a certain period (usually more than two seconds) and are linked by swings. We focus on recognising and categorising these static elements which are more important for marking and treat the rest of the routine as swings. For balance beam, a routine consists of several dynamic elements, which is completed very quickly (less than one second), and relatively static poses such as standing on the beam, which occupy most of time. However, those dynamic elements are more important for marking, so the mechanism is designed to recognise and categorise these elements.

Having got the matching labels and scores of a rings video, we produce its log by a voting mechanism as the following steps:

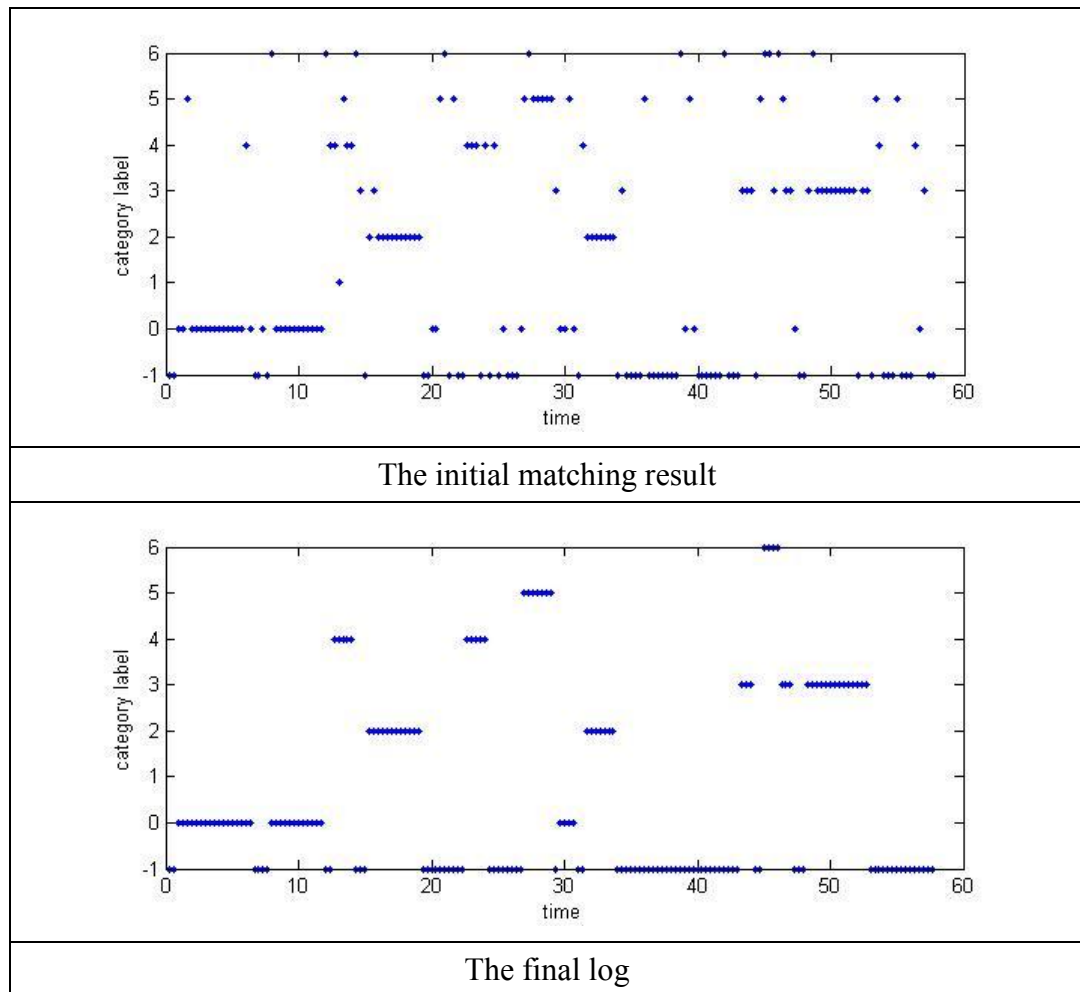
- 1) Nine categories of poses are formed. Eight of them correspond to the eight templates and are denoted by the templates' labels. The remaining one is an "unknown" category that is for frames in which no human bodies are detected or no templates matched, denoted by -1.
- 2) A threshold is used to filter the matching results that have large scores (the smaller the score, the higher the similarity to a template). Here we use the threshold  $t=0.55$ . Thus any matching results with scores higher than  $t$  will be categorised to category -1 ("unknown").
- 3) We put all the labels of the frames in to an array, in the time order. Then the following mechanism is employed to smooth the array:
  - a) The input array is denoted as  $I$ , and the output array is  $O$ . A 1D smoothing window with the width of  $w$  is defined. Here in this case  $w=\text{roundodd}(1.5*r)$ . The symbol  $r$  is the sampling rate (number of frames sampled per second). The function  $\text{roundodd}()$  returns the nearest odd integer of a float number. Define  $u=(w-1)/2$ ;
  - b) Considering a member in array  $I$ , denoted as  $I_i$ , we count the number

- of each pose category from  $I_{i-u}$  to  $I_{i+u}$ , and record them as  $c_1, c_2, c_3 \dots$ . If  $c_j$  ( $j=1,2,3 \dots$ ) is larger than a proportion of  $w$  (50% here),  $O_i$  equals to the corresponding category label of  $c_j$ , else,  $O_i$  is set to -1.
- c) Then check each member of  $O$ . If  $O_i=-1$  while  $O_{i-1}$  or  $O_{i+1} \neq -1$ , and  $I_i = O_{i-1}$  or  $O_{i+1}$ , then  $O_i$  is assigned to the same value of  $O_{i-1}$  or  $O_{i+1}$ . This step avoids losses in step b.
- 4)  $O$  is the final log of the video.

In step 3 above, a) and b) guarantee that the current frame lies in the middle of the smoothing window whose length is odd, and c) recovers the frames between probed element categories and the “unknown” category.

Figure 6.10 visualises an example of an initial matching result and its final log. The category label of each frame is plotted along the time domain. There are some frames categorised to “unknown”, which means the bodies in these frame is not detected, or classified incorrectly. For rings videos, “unknown” frames usually means some swing poses that do not match any of the templates of probed elements. So the moments of “unknown” frames can be treated as swing moments.

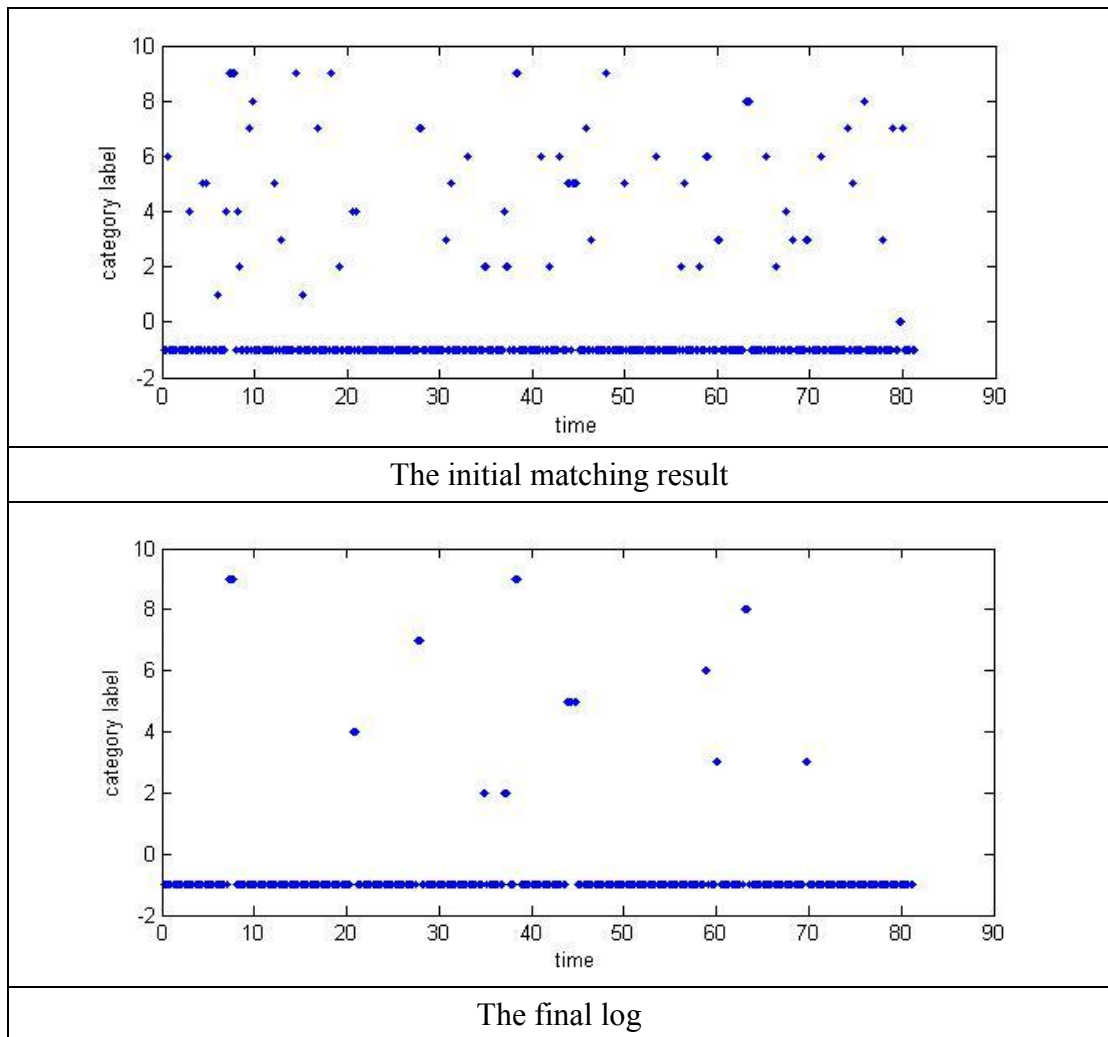
The final log clearly shows what the gymnast is doing in every moment during the performance. According to Figure 6.10, in the initial matching result there are a number of false matches which are caused by errors in template matching, or by lacking of similar templates. By using the post-processing mechanism, these false matches are solved and merged to the nearest pose categories which represent probed elements or “unknown”. However, there are still a few errors: the log shows that there is a pose “0” around the time spot of 30 seconds, and that is an error due to some false matches which are near each other and select the same category; meanwhile the pose “-1” at the time spot of 8 seconds should be pose “0”.



**Figure 6.10 A visualised rings video log**

Logging balance beam has the same procedure as logging rings. However since we aim to capture dynamic elements, the width of the smoothing window  $w = \text{roundodd}(0.3 * r)$ . In order to reject more false matches, the threshold for matching scores is set to 0.5. Figure 6.11 visualises an example of balance beam video log.

According to Figure 6.11 it can be seen that most frames are categorised to pose 0 (standing on the beam), which fit the feature of balance beam, and this pose will not be treated as an probed element. However since the ratio of false matches is high, while the smoothing window is small, more errors occur in logging balance beam. In the example above, two dynamic elements are missed and another one is incorrectly categorised.



**Figure 6.11 A visualised balance beam video log**

We applied the logging system to all 16 test videos. To evaluate the performance of the system, we count the total number of probed elements of both rings and balance beam, as well as the true and false categorised elements, to make a comparison, as shown in Table 6.3. Since those “unknown” actions are not probed elements, the numbers of total probed elements do not contain the number of the “unknown” category, and the true classified elements do not contain the “unknown” either. The false classified actions include three types of incorrect classification: a probed element being classified to another probed element, a probed element being classified to the “unknown” category, and an “unknown” action being classified to a probed element.



	Total probed	True	False
Rings	89	80	17
Balance beam	93	66	39

**Table 6.3 Performance of the video logging system**

Table 6.4 indicates incorrect detections of both rings and balance beam. The vertical labels stand for ground-truth probed elements, and the horizontal labels represent the classified probed elements. For example, in the table for rings, an action of element 3 is classified to element 7, so the value is 1 at the cell whose vertical label is “3” and horizontal label is “7” (the cell with bold border). According to Table 6.4, most false classification results are caused by categorising probed elements to “unknown” or “unknown” to probed elements. That is because “unknown” actions include various poses which are difficult to predict and some of them may similar to probed elements. Meanwhile the interference between probed elements is not very strong, according to the table.

The result shows that the logging system performs much better on rings videos. The reason is: to deal with the false matches, we have to employ the smoothing mechanism. If the elements last for a relatively long period, they will be easily recognised. However if the time of an element is too short, it can hardly be identified from false pose matches. Since no pose estimators can recognise all types of poses because of variations of human body shapes and behaviours, as well as exceptions (such as errors or faults in gymnastics), false pose classification results cannot be avoided. As false results appear occasionally and last briefly, it is difficult to identify them from short-term probed actions. Thus there is a paradox between false pose categorisation results and short-term probed actions: on one hand, false results need to be filtered, while on the other hand, short-term term actions which have similar characteristics, are required to be kept and emphasised.

Rinags									
labels	0	1	2	3	4	5	6	7	-1
0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	1	2
4	0	0	0	0	0	0	0	0	1
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0	0	2
-1	1	0	2	2	0	0	1	3	0

Balance beam										
labels	1	2	3	4	5	6	7	8	9	-1
1	0	0	0	0	0	0	2	0	0	2
2	0	0	0	0	0	0	0	0	0	3
3	0	1	0	0	0	0	0	0	0	2
4	0	0	0	0	0	0	0	0	1	1
5	0	0	0	0	0	0	0	0	0	2
6	0	0	0	0	0	0	1	0	0	3
7	0	0	0	0	0	0	0	0	0	1
8	0	0	1	0	0	0	1	0	0	2
9	0	0	0	0	0	1	0	1	0	2
-1	1	2	0	1	3	1	1	1	2	0

**Table 6.4 Incorrect classification results**

Here we should state that accurately determining when an action starts and ends is very difficult, because there are some frames representing the link-up poses between two actions and these poses can be classified to either of the two actions. That is to say, the system can only approximately record athletes' actions along timelines. Meanwhile, this system cannot evaluate the quality of an athlete's performance. Therefore it can be only used as an assistant for referees and commentators to summarise and replay athletes' performance. Evaluating and marking still depend on human's judgement.

### 6.3 Conclusions

In this chapter, we have introduced a practical application using our action analysis method: object and content based key-frame extraction, which differs from conventional methods in aiming to extract key-frames understandable for users. The entire procedure can be concluded to two phases: HOG human body detection and GDT&OM template matching.

The HOG detection method performs well for simple data, such as the Weizmann dataset. However, for more complicated data, such as the INRIA dataset used in Chapter 4, it still requires improvement. Besides, each HOG detector can only be applied for one type of object, such as full body human, upper body human, animals, vehicles etc. This means that in practice users need to build a number of HOG detectors and switch them according to what object types are in input videos. Therefore users must have prior knowledge of what an input video represents. In addition, users need to take care of exceptional cases, such as abnormal poses of humans, to avoid losing detections.

The GDT&OM template matching method leads to understandable key-frames when good matches have been performed. However, interference and judgement problems constrain the system to have only a small number of action classes. Meanwhile, since human styles and poses vary hugely, building appropriate templates is a difficult job. We are planning to employ the pose classification method based on body parts/joints locations introduced in Chapter 5 instead of the contour template matching method, but locating body parts on 2D images is still a big challenge at present.

Another problem is processing videos with multiple bodies: when multiple bodies perform different actions in a video, how to determine the domain action or the key-frame? We argue that the system can take the largest body as the main body and determine the key-frame according to it.

Compared with a conventional method called PME, the entire key-frame extracting system achieves better experimental result. We argue that extracting key-frames

representative for content of videos is more meaningful and useful in practice. However, such systems are more complicated and expensive, due to the difficult issues of detection and matching. The further work is to improve performance of human detectors in complex situations, and to develop more reliable pose estimators.

The video logging system provides a solution of automatically recording content of videos, and can extract most probed elements for gymnastic videos. However the system is also constrained by the camera angle and the number of people. If the viewpoint of the camera is changed in the footage, a lot more templates are needed. However it will increase false rate of matching. Hence for rings, balance beam or bars which are using a fixed camera in gymnastics, our method will work. However for floor videos whose camera angle often changes, the solution is difficult. If we apply this system to a video with multiple people, when multiple human bodies are detected, it is also difficult to determine the main content that the video represents. But this issue is more about a higher level of video understanding and artificial intelligence. For the work in this section, solving the paradox between false pose classification results and short-term probed actions is the key in the future. Meanwhile, as the same as key-frame extraction, body-part based pose estimation can be used to solve the problem of the varied viewpoint.

# Chapter 7

## Conclusions and Further Work

---

### 7.1 Conclusions

In this thesis, the research area of Computer-Vision-based human motion analysis has been introduced, techniques associated to this area have been investigated, and a human motion analysis system which parses images or videos (image sequences) to achieve human poses has been proposed. We have built a human motion analysis system that integrates a novel colour-to-greyscale converter, an optimised HOG human body detector, and an improved GDT&OM pose estimator. Each stage of the system has been tested separately and a key-frame extractor which is an application of the system is also illustrated.

This thesis proposed a novel colour-to-greyscale conversion method that converts RGB images to chrominance-edge-enhanced greyscale images. In the experiments, the novel method achieves good performance for most images within the testing data. When compared with Color2Grey [88] qualitatively, the novel method achieves similar results for complicated images, but is far less computational expensive. When compared with Ren's method [105] quantitatively, the novel method produces less noise and obtains even more "true" edges when "thin" ground truth images are used. The weakness of the novel method is that it is still parameter dependent (the number of clusters) and has a problem of losing edges between two objects which are similar in chrominance. (i.e. the "walker" image in Figure 5.13).

Histogram of Orientated Gradients has also been investigated and improved. A new training scheme significantly improves the FPR of human detection at the cost of only a very small proportion of losses of true positive detections. Using pre-processed data produced by the novel colour-to-greyscale convertor, can also improve the FPR. We compare the improved HOG detector with a Haar/AdaBoost cascade detector (a classifier that classifies input windows to human or nonhuman, so it can be treated as a detector), and the HOG detector achieves higher TPR and lower FPR than the Haar/AdaBoost cascade detector.

A pose estimator based on GDT&OM template matching has been developed. We improved the original GDT&OM human detector by using background cancellation, and modified it to a pose estimator by employing an HOG detector and pose templates. This method can successfully estimate (classify) human poses when use a small set of pose categories. Besides a pose estimator method using locations of body parts is also proposed. This method can achieve high pose estimation (classification) accuracy by applying a k-nearest neighbourhood or a multiple-class SVM classifier, under the assumption that body parts are correctly located.

Finally an integrated system applied to key-frame extraction is proposed. This key-frame extractor is more intelligent than conventional approaches as it is designed to select frames representing content of videos. However it is an object-dependent system that needs users to manually select a target object, and the performance is limited by the human body detector and pose estimator. Then the system is extended to log gymnastic videos. Rings and balance beam videos are tested. We build human detectors by using separate positive training data and shared negative training data for rings and balance beam, and two sets of templates are made separately as well. We employ a voting mechanism to smooth the pose estimation result. The final results indicates that the system performs better on rings than balance beam, as rings videos contain static elements while balance beam videos focus on dynamic elements.

## 7.2 Further Work

The methods and system proposed in this thesis are still not practical, for each part of the system has weakness and requires improvement. Overall, we are aiming to improve the system to be more reliable and efficient. There are several points to be improved on the system:

- 1) The accuracy and efficiency of the human detector.
- 2) The robustness of the system, especially the pose estimator.
- 3) The extension of the system to high-level behaviour description.

The further work can be carried out by three stages:

### 7.2.1 Stage one: improvement in HOG human detection

The novel colour-to-greyscale converter proposed in Chapter 3 is relatively successful. However, when we apply the converted greyscale images to the HOG method, the improvement is not very significant. The reason is probably that when colour edges are enhanced, luminous edges are weakened at the same time. The next step is to do experiments by combining both colour edges and luminous edges. In practice for an image, the gradients of its colour and converted greyscale will be both computed and combined to a gradient union.

At present images used in the experiments are mostly representing standing or walking people. Experiments on applying more human bodies poses, such as sitting, lying and other abnormal poses is in the plan. In addition, the conventional SVM for classifying a relatively large detection window in high resolution images, e.g. a  $64 \times 128$  window in an image of  $1600 \times 1200$ , is too computationally expensive for real-time systems. Even when the image size is scaled to  $400 \times 300$ , it is still difficult to achieve real-time processing. For real-world applications, a more efficient classifier is required. We plan to implement classifiers such as AdaBoost-cascading and random forests to improve the detection efficiency.

### 7.2.2 Stage two: improvement in the robustness

The template matching method used is not a robust way to estimate poses, since it is constrained by the templates and the viewpoint of the camera. The HOG detection method performs with high accuracy and robustly according to Chapter 6, so the pose estimation method can still be based on the output of the human detection stage. To build a robust pose estimator, the following two ideas can be applied:

Classifying poses by using locations of body parts is proved to be in high accuracy. We consider that it has a larger potential than template matching. The problem at present is that existing methods of locating body parts are not ideal, especially when there are overlaps between people or some parts of a body are hidden. An estimation mechanism is required to estimate the body parts which do not appear. Meanwhile a fault-tolerant mechanism is needed so that false estimation of hidden body parts can be handled, and the system is still able to classify a human body to a correct pose category.

Alternatively the system will apply a local feature method instead of global outline matching. Many local methods do not need detection as the pre-stage. However we argue that detecting human bodies before pose estimation can solve the multi-person problem: when an interest point is detected, if there a more than one person in the picture, it is difficult to determine which person the point belongs to. Besides, human detection can help to reject false interest points. Of course tracking is also required to identify different persons. Alternatively a spatial-temporal model combines global human shapes and motions can be designed, such as the global spatial-temporal method introduced by the paper [74], so for videos the system can capture human locations and motions within a single detection procedure. Using local features is able to deal with the problem caused by different viewpoint of the camera, because some interest points will still be detectable even if the viewpoint changes. Thus these detectable interest points can also be used to estimate the relative positions between the viewpoints.

There is another issue that the system proposed by the thesis is under a low noise



assumption. In the real world there are various types of noise to be considered. For example, indoor sport pictures and videos usually have flashes which change illumination condition significantly, or in outdoor sport pictures and videos have noise from weather. Therefore the pre-processing phase also needs to measure and cancel noise. A normalisation and noise cancellation step will be added to the system.

### **7.2.3 Stage three: development in high-level behaviour recognition**

Then for the entire system of key-frame extraction, its performance and efficiency will be enhanced by the improvement of each stage. Besides, the thesis has stated above that for multiple types of objects, the target object needs to be manually selected. A general plan is to make the extractor more intelligent – the system can automatically decide the key object in a video among a number of object candidates. In the preliminary scheme, multiple detectors can be trained for multiple types of interest objects, and use state estimators respectively for each type. The system will combine the factors of objects' sizes, locations and similarities to detection models to determine a main object, or the weight of each interest object, and compute a score to evaluate the importance of each frame. Finally the key-frame is the one with the highest importance.

Finally for video logging the system needs to be extended to more applications, as now it can only process limited videos each of which has a fixed camera viewpoint and a single person. An advanced system like this requires not only a better human detector and a pose estimator, but also a mechanism that analyses group interaction after behaviours are recognised. Such a system can be applied to more complicated group sports such as football and basketball, as well as surveillance videos and movies. The ideas to solving problems on multiple persons and multiple viewpoints have been mentioned in Stage 2. There is another issue that the system needs to recognise high-level behaviours which are more important than basic actions for practical applications. Considering that complicated behaviours usually consist of several basic actions, a state-based method such as Hidden Markov Model, Bayesian Networks or Neural Networks can be employed. Therefore the system will be

designed to detect human bodies first, then estimate poses as motion states by a robust method, and finally classified by a state-based model. Besides, an even higher level of behaviours—interactions between persons, will be an important topic afterwards.

In conclusion, human motion analysis is an interesting research area with large a market potential. There are various advanced methods in theory but not many practical approaches in the real world. Developing a reliable, robust and efficient system to recognise human actions and understand human behaviours still needs a lot of further effort.

# Appendix I

---

## Shell script for HOG training

```
#!/bin/sh
WIDTH=96; export WIDTH
HEIGHT=96; export HEIGHT

HardOption=" --poscases 754 --negcases 7140 "

BinDIR=/home/zw514/work/OLTbinaires/bin
DumpRHOG=$BinDIR/dump_rhog
ClassifyRHOG=$BinDIR/classify_rhog
dump4mc=$BinDIR/dump4svmlearn
svm_learn=$BinDIR/svm_learn

Option=" -W $WIDTH,$HEIGHT \
  -C 8,8 -N 2,2 -B 9 -G 8,8 -S 0 --wtscale 2 --maxvalue 0.2 --epsilon 1 \
  --fullcirc 0 -v 3 --proc rgb_sqrt --norm l2hys "

ExtraOption2=" -t 0.1 -m 0 "
ExtraOption1=" -p 1,0 --no_nonmax 0 -z 8,16,1.3 --cachesize 128 \
  --scaleratio 1.05 --winstride 6 --margin 4,4 --avsize 0,96 "
ExtraOption=$ExtraOption1$ExtraOption2
OutDir=HOG

OutFile=$OutDir/record
CMDLINE=$OutDir/record

CMD="mkdir -p $OutDir"
$CMD

# get features on positive images
CMD="$DumpRHOG $Option -s 1 train/pos.lst $OutDir/train_pos.RHOG "
echo $CMD >> $CMDLINE
$CMD >> $OutFile
```

```

# get features on negative images
CMD="$DumpRHOG $Option -s 20 train/neg.lst $OutDir/train_neg.RHOG "
echo $CMD >> $CMDLINE
$CMD >> $OutFile

# dump to svmdense format
CMD="$dump4mc -p $OutDir/train_pos.RHOG -n $OutDir/train_neg.RHOG \
  $OutDir/train_BiSVMLight.blc -v 4 "
echo $CMD >> $CMDLINE
$CMD >> $OutFile

# learn
CMD="$svm_learn -j 3 -B 1 -z c -v 1 -t 0 $OutDir/train_BiSVMLight.blc \
  $OutDir/model_4BiSVMLight.alt "
echo $CMD >> $CMDLINE
$CMD >> $OutFile

echo First iteration complete

# create hard examples
HardDir=$OutDir/hard
CMD="mkdir -p $HardDir"
echo $CMD >> $CMDLINE
$CMD >> $OutFile

CMD="$ClassifyRHOG train/neg.lst $HardDir/list.txt \
  $OutDir/model_4BiSVMLight.alt -d $HardDir/hard_neg.txt -c $HardDir/hist.txt \
  -m 0 -t 0 --no_nonmax 1 --avsize 0 --margin 0 --scaleratio 1.2 -l N $Option "

echo $CMD >> $CMDLINE
$CMD >> $OutFile

echo Hard examples created

## now second iteration
# dump hard examples
CMD="$DumpRHOG $Option -s 0 $HardDir/hard_neg.txt \
  $OutDir/train_hard_neg.RHOG $HardOption \
  --dumphard 1 --hardscore 0 --memorylimit 1700 "
echo $CMD >> $CMDLINE
$CMD >> $OutFile

# dump positive, negative and hard examples
CMD="$dump4mc -p $OutDir/train_pos.RHOG \
  -n $OutDir/train_neg.RHOG -n $OutDir/train_hard_neg.RHOG \
  $OutDir/train_BiSVMLight.blc -v 4 "
echo $CMD >> $CMDLINE
$CMD >> $OutFile

```

```

# learn, second iteration
echo Doing second learning
CMD="$svm_learn -j 3 -B 1 -z c -v 1 -t 0 \
  $OutDir/train_BiSVMLight.blc \
  $OutDir/model_4BiSVMLight.alt "
echo $CMD >> $CMDLINE
$CMD >> $OutFile

echo Second iteration complete

```

## Shell script for HOG testing

```

#!/bin/sh
if [ $# -lt 3 ]
then
  echo "<usage>: runonimage.sh <image name/image directory/list file> "
  echo "      <out text file> <out image file/out image dir>"
else
  InFile=$1
  OutFile=$2
  ImageFile=$3

  WIDTH=96; export WIDTH
  HEIGHT=96; export HEIGHT

  BinDIR=/home/zw514/work/OLTbinaires/bin
  Classifier=$BinDIR/classify_rhog

  Option=" -W $WIDTH,$HEIGHT \
    -C 8,8 -N 2,2 -B 9 -G 8,8 -S 0 --wtscale 2 --maxvalue 0.2 --epsilon 1 \
    --fullcirc 0 -v 3 --proc rgb_sqrt --norm l2hys "

  #####
  for t in 0.10 #0.02 0.04 0.06 0.08 0.10
  do
    echo t = $t
    Margin=4
    ExtraOption2=" -t $t -m 0 "
    ExtraOption1=" -p 1,0 --no_nonmax 0 -z 8,16,1.3 \
      --cachesize 128 --scaleratio 1.05 --winstride 8 \
      --margin $Margin,$Margin --avsize 0,96 "
    ExtraOption=$ExtraOption1$ExtraOption2

```

```
echo Running hog on image list
ResultDir=$OutFile"_t"$t".txt"
ImageDir=$ImageFile"_t"$t
echo $ImageDir
CMD="mkdir $ImageDir"
$CMD
CMD="$Classifier $Option $ExtraOption \
  $InFile $ResultDir HOG/model_4BiSVMLight_1.alt -i $ImageDir "
$CMD
CMD="cp $ResultDir results/$ResultDir"
$CMD
done
fi
```

# Appendix II

---

## Cascade configuration

Window Width	32
Window Height	64
Detector Minimum Detection Rate	0.88
Detector Maximum False Positive Rate	0.000001
Elemental Classifier Learner	AdaBoost
Elemental Classifier Type	Perceptron
Scan Step	1
Scan Scale	0.8
Node Maximum False Positive Rate	0.55
Performance Trade OffFactor	0.1
Calibrate To Validation Data	0
Validation Scan Step	4
Validation Scan Scale	0.8
Max Nodes	35
Max Classifiers Per Node	300
Use Rect Diff Features	1
Bootstrap Scan Step	12
Bootstrap Scan Scale	0.65
Classifiers Per Node	3 10 25 25 50 50 50 80 80 130 130 200 200 300

# Bibliography

---

- [1] J. K. Aggarwal and Q. Cai, "Human Motion Analysis: A Review," *Computer Vision and Image Understanding*, Vol,73, No.3, pp. 428-440, March 1999.
- [2] G. Johansson, "Visual motion perception," *Sci. American*, 232(6), pp. 76-78, 1975.
- [3] D. Marr and H. K. Nishibara, "Representation and recognition of the spatial organization of three-dimensional shapes," *Proceedings of the Royal Society of London. Series B, Biological Science, Volume 200, Issue 1140*, pp. 269-294, Feb. 1978.
- [4] C. Wren, A. Azarbajani, T. Darrell and A. Pentland, "Pfinder: Real-time Tracking of the Human Body," in *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, Killington, Vermont, USA, 1996, pp. 51-56.
- [5] H. Hienz, K. Grobel and G. Offner, "Real-time Hand-arm Motion Analysis using a Single Video Camera," in *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, Killington, Vermont, USA, 1996, pp. 323 - 327.
- [6] J. Triesch and C. von der Malsburg, "Robust Classification of Hand Postures Against Complex Backgrounds," in *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, Killington, Vermont, USA, 1996, pp. 170-175.
- [7] D. Kang, Y. Onuma and J. Ohya, "Estimating complicated and overlapped human body postures by wearing a multiple-colored suit using color information processing," in *Proceedings of Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, , Seoul, Korea, 2004,



- pp. 687 - 692.
- [8] S. Perrin, A. Cassinelli and M. Ishikawa, "Gesture Recognition using Laser-based Tracking System," in *Proceedings on Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, 2004, pp. 541-546.
- [9] J. Gao and J. Shi, "Multiple Frame Motion Inference using Belief Propagation," in *Proceedings of Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea, 2004, pp. 875 - 880.
- [10] C. Chen, R. Ugarte, C. Wu and H. Aghajan, "Discovering Social Interactions in Real Work Environments," in *2011 IEEE International Conference on Automatic Face & Gesture Recognition and Workshops*, Santa Barbara, USA, 2011, pp. 933 - 938.
- [11] G. Chittaranjan, O. Aran and D. Gatica-Perez, "Exploiting Observers' Judgements for Nonverbal Group Interaction Analysis," in *2011 IEEE International Conference on Automatic Face & Gesture Recognition and Workshops*, Santa Barbara, USA, 2011, pp. 734 - 739.
- [12] C. J. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery, Volume 2, Issue 2*, p. 121-167, Jun. 1998.
- [13] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), Volume 1*, Kauai, HI, USA, 2001, pp. I-511 - I-518.
- [14] L. Breiman, "Random forests," *Machine Learning, Volume 45, Issue 1*, pp. 5-32, 2001.
- [15] F. De la Torre, J. Hodgins, J. Montano, S. Valcarcel, R. Forcada and J. Macey., "Guide to the Carnegie Mellon University Multimodal Activity (CMU-MMAC) Database," Tech. report CMU-RI-TR-08-22, Robotics Institute, Carnegie Mellon University, 2009.
- [16] R. Girshick, J. Shotton, P. Kohli, A. Criminisi and A. Fitzgibbon, "Efficient

- Regression of General-Activity Human Poses from Depth Images,” in *2011 IEEE International Conference on Computer Vision (ICCV)*, Barcelona, Spain, 2011, pp. 415-422.
- [17] L. Breiman, J. Friedman, C. J. Stone and R. A. Olshen, *Classification and Regression Trees*, Chapman and Hall/CRC, 1984.
- [18] N. Dalal and B. Triggs, “Histograms of Oriented Gradients for Human Detection,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 1*, San Diego, CA, USA, 2005, pp. 886 - 893.
- [19] N. D. Thanh, W. Li and P. Ogunbona, “A Novel Template Matching Method for Human Detection,” in *16th IEEE International Conference on Image Processing (ICIP)*, Cairo, Egypt, 2009, pp. 2549-2552.
- [20] T. Liu, H.-J. Zhang and F. Qi, “A Novel Video Key-Frame-Extraction Algorithm Based on Perceived Motion Energy Model,” *IEEE Transactions on Circuits and Systems for Video Technology, Volume 13, Issue 10*, pp. 1006-1013, Oct. 2003.
- [21] M. Kumar, “Key Frame Extraction From Consumer Videos Using Sparse Representation,” in *2011 18th IEEE International Conference on Image Processing (ICIP)*, Brussels, Belgium, 2011, pp. 2437-2440.
- [22] J. W. Mateer and J. A. Robinson, “A Vision-Based Postproduction Tool for Footage Logging, Analysis and Annotation,” *Journal of Graphical Models, Volume 67, Issue 6*, pp. 565-583, Nov. 2005.
- [23] M. Day, “Towards Optimised Cascade Classifiers for Face Detection,” PhD thesis, University of York, 2008.
- [24] R. Jain and H. Nagel, “On the Analysis of Accumulative Difference Pictures from Image Sequences of Real World Scenes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 1, Issue 2*, pp. 206-214, Apr. 1979.
- [25] V. Ferrari, M. Marin-Jimenez and A. Zisserman, “Progressive Search Space Reduction for Human Pose Estimation,” *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008.*, pp. 1-8, 2008.
- [26] J. Gall, A. Yao, N. Razavi, L. V. Gool and V. Lempitsky, “Hough Forests for

- Object Detection, Tracking, and Action Recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 33, Issue 11, pp. 2188-2202, Nov. 2011.
- [27] R. Poppe, “A survey on vision-based human action recognition,” *Image and Vision Computing*, Volume 28, Issue 6, p. 976–990, Jun. 2010.
- [28] L. Wang, W. Hu and T. Tan, “Recent Developments in Human Motion Analysis,” *Pattern Recognition*, Volume 36, Issue 3, pp. 585-601, 2003.
- [29] A. J. Lipton, H. Fujiyoshi and R. Patil, “Moving Target Classification and Tracking from Real-time Video,” in *1998 Proceedings of IEEE Workshop on Applications of Computer Vision*, Princeton, New Jersey, USA, 1998, pp. 8-14.
- [30] R. T. Collins, A. J. Lipton and T. Kanade, “A system for video surveillance and monitoring,” *VSAM final report, CMU-RI-TR-00-12, Technical Report, Carnegie Mellon University*, 2000 2000.
- [31] Y. Kuno, T. Watanabe, Y. Shimosakoda and S. Nakagawa, “Automated Detection of Human for Visual Surveillance System,” in *Proceedings of the 13th International Conference on Pattern Recognition*, Brisbane, Australia, 1996, pp. 865-869.
- [32] R. Cutler and L. Davis, “Robust Real-time Periodic Motion Detection, Analysis, and Applications,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 22, Issue 8, pp. 781-796, Aug. 2000.
- [33] A. J. Lipton, “Local Application of Optic Flow to Analyse Rigid Versus Non-rigid Motion,” in *Proceedings of IEEE International Conference on Computer Vision Workshop on Frame-Rate Applications*, Corfu, Greece, 1999, pp. 797-804.
- [34] D. G. Sim, O. K. Kwon and R. H. Park, “Object Matching Algorithms Using Robust Hausdorff Distance Measures,” *IEEE Transactions on Image Processing*, Volume 8, Issue 3, pp. 425-429, Mar. 1999.
- [35] X. Yu and M. K. Leung, “Shape Recognition using Curve Segment Hausdorff Distance,” in *18th International Conference on Pattern Recognition*, Volume 3,

- Hong Kong, China, 2006, pp. 441-444.
- [36] Z. Lin, L. S. Davis, J. D. Doermann and D. DeMenthon, "Hierarchical Part-template Matching for Human Detection and Segmentation," in *IEEE 11th International Conference on Computer Vision*, Rio de Janeiro, Brazil, 2007, pp. 1-8.
- [37] D. M. Gavrila, "A Bayesian, Exemplar-Based Approach to Hierarchical Shape Matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *Volume 29, Issue 8*, p. 1408–1421, Aug. 2007.
- [38] P. F. Felzenszwalb and D. P. Huttenlocher, "Distance transforms of sampled functions," Tech.Rep., Cornell Computing and Information Science, 2004.
- [39] J. Pearl, "Bayesian Networks: A Model of Self-Activated Memory for Evidential Reasoning," in (*UCLA Technical Report CSD-850017*). *Proceedings of the 7th Conference of the Cognitive Science Society*, University of California, Irvine, CA, 1985, p. 329–334.
- [40] E. Seemann, B. Leibe and B. Schiele, "Multi-Aspect Detection of Articulated Objects," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 2*, New York, NY, USA, 2006, p. 1582–1588.
- [41] B. Leibe, E. Seemann, and B. Schiele, "Pedestrian detection in crowded scenes," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 1*, San Diego, CA, USA, 2005, p. 878–885.
- [42] H. Kruppa, M. C. Santana and B. Schiele, "Fast and Robust Face Finding via Local Context," *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 157-164, 2003.
- [43] R. Lienhart, A. Kuranov and V. Pisarevsky, "Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection," *Lecture Notes in Computer Science, Volume 2781/2003*, pp. 297-304, 2003.
- [44] R. Lienhart and J. Maydt, "An Extended Set of Haar-Like Features for Rapid Object Detection," *2002 International Conference on Image Processing. 2002. Proceeding, Volume 1*, pp. I-900 - I-903, 2002.
- [45] Q. Liu and Y. Qu, "HOG and Color Based Adaboost Pedestrian Detection," in

- 2011 Seventh International Conference on Natural Computation (ICNC)*, Shanghai, China, 2011, pp. 584-587.
- [46] I. A. Karaulova, P. M. Hall and A. D. Marshall, "A Hierarchical Model of Dynamics for Tracking People with a Single Video Camera," in *11th British Machine Vision Conference*, Bristol, UK, 2000, pp. 352-361.
- [47] L. Jin, J. Cheng and H. Huang, "Human Tracking in the Complicated Background by Particle Filter Using Color-histogram and HOG," in *2010 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, Chengdu, China, 2010, pp. 1-4.
- [48] M. Isard and A. Blake, "CONDENSATION-conditional Density Propagation For Visual Tracking," *International Journal of Computer Vision, Volume 29, Issue 1*, pp. 5-28, Aug. 1998.
- [49] M. Tong and H. Bian, "3D Human Tracking by using Shared Latent Dynamical Model," in *2010 International Conference on Artificial Intelligence and Computational Intelligence (AICI)*, Nanjing, China, 2010, pp. 345 - 349.
- [50] L. Zhang, J. Chen, Z. Zeng and Q. Ji, "A Generic Framework for 2D and 3D Upper Body Tracking," in *Machine Learning For Human Motion Analysis: Theory and Practice*, Hershey, New York, USA, Medical Information Science Reference, IGI Global, 2010, pp. 133-151.
- [51] K. Murphy, "Dynamic Bayesian Networks: Representation, Inference and Learning," in *PhD Thesis*, Computer Science Division, UC Berkeley, 2002.
- [52] S. McKenna, S. Jabri, A. Rosenfeld and H. Wechsler, "Tracking Interacting People," in *Proceedings of Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, Grenoble, France, 2000, pp. 348-353 .
- [53] R. Polana and R. Nelson, "Low Level Recognition of Human Motion," in *Proceedings of IEEE CS Workshop on Motion of Non-Rigid and Articulated Objects*, Austin, TX, USA, 1994, pp. 627-632.
- [54] D. Jang and H. I. Choi, "Active Models for Tracking Moving Objects," *Pattern Recognition, Volume 33, Issue 7*, no. 1135-1146, p. 1135–1146, Jul.

- 2000.
- [55] T. Tung and T. Matsuyama, “Human Motion Tracking in Video: A Practical Approach,” in *Machine Learning for Human Motion Analysis: Theory and Practice*, Medical Information Science Reference, IGI Global, 2010, pp. 1-13.
  - [56] D. Ramanan, D. A. Forsyth and A. Zisserman, “Strike a Pose: Tracking People by Finding Stylized Poses,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 1*, San Diego, CA, USA, 2005, pp. 271-278.
  - [57] N. Ikizler, R. G. Cinbis, S. Pehlivan and P. Duygulu, “Recognizing Actions from Still Images,” in *19th International Conference on Pattern Recognition, 2008. ICPR 2008*, Tampa, Florida, USA, 2008, pp. 1-4.
  - [58] D. Ramanan, “Learning to parse images of articulated bodies,” in *Advances in Neural Information Processing Systems*, Vancouver, B.C., Canada, 2006.
  - [59] Y. Wang, H. Jiang, M. S. Drew, Z. Li and G. Mori, “Unsupervised Discovery of Action Classes,” in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 2*, New York, NY, USA, 2006, pp. 1654-1661.
  - [60] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*, MIT Press, 2002.
  - [61] T. Tangkuampien and D. Suter, “KSM Based Machine Learning for Markerless Motion Capture,” in *Machine Learning for Human Motion Analysis: Theory and Practice*, Hershey, New York, USA, Medical Information Science Reference, IGI Global, 2010, pp. 74-106.
  - [62] B. Schölkopf, A. J. Smola and K. R. Müller, “Kernel Principal Component Analysis,” in *Fourth International Conference on Artificial Neural Networks*, Lausanne, Switzerland, 1997, pp. 583-588.
  - [63] L. K. Saul and S. T. Roweis, “Nonlinear Dimensionality Reduction by Locally Linear Embedding,” *Science, Volume 290*, p. 2323–2269, 2000.
  - [64] R. Poppe, “Common Spatial Patterns for Real-Time Classification of Human

- Actions,” in *Machine Learning for Human Motion Analysis: Theory and Practice*, Hershey, New York, USA, Medical Information Science Reference, IGI Global, 2010, pp. 55-73.
- [65] A. Poritz, “Hidden Markov Models: a guided tour,” in *1988 International Conference on Acoustics, Speech, and Signal Processing, (ICASSP-88), Volume 1*, New York, New York, USA, 1988, pp. 7-13.
- [66] W. Li, Z. Zhang, Z. Liu and P. Ogunbona, “Human Action Recognition with Expandable Graphical Models,” in *Machine Learning for Human Motion Analysis: Theory and Practice*, Hershey, New York, USA, Medical Information Science Reference, IGI Global, 2010, pp. 187-212.
- [67] Wikipedia, “Hidden Markov model,” [Online]. Available: [http://en.wikipedia.org/wiki/Hidden\\_Markov\\_model](http://en.wikipedia.org/wiki/Hidden_Markov_model).
- [68] J. Yamato, J. Ohya and K. Ishii, “Recognizing Human Action in Time-sequential Images using Hidden Markov Model,” in *Proceedings of 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Champaign, IL, usa, 1992, pp. 379-385.
- [69] X. Feng and P. Perona, “Human action recognition by sequence of movelet codewords,” in *First International Symposium on 3D Data Processing Visualization and Transmission*, Padova, Italy, 2002, pp. 717-721.
- [70] M. Brand, N. Oliver and A. Pentland, “Coupled hidden Markov models for complex action recognition,” in *1997 IEEE Conference on Computer Vision and Pattern Recognition*, San Juan, PR, 1997, pp. 994-999.
- [71] K. Sage and H. Buxton, “Joint Spatial and Temporal Structure Learning for Task Based Control,” in *Proceedings of the 17th International Conference on Pattern Recognition, Volume 2*, Cambridge, England, UK, 2004, pp. 48-51.
- [72] I. Ben-Gal, *Encyclopedia of Statistics in Quality and Reliability*, John Wiley & Sons, 2007.
- [73] P. Remagnino, T. Tan and K. Baker, “Multi-agent visual surveillance of dynamic scenes,” *Image and Vision Computing, Volume 16, Issue 8*, pp. 529-532, 1998.

- [74] L. Gorelick, M. Blank, E. Shechtman, M. Irani and R. Basri, "Actions as Space-Time Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, Volume 29, Issue 12, p. 2247–2253, 2007.
- [75] I. Laptev, B. Caputo, C. Schuldt and T. Lindeberg, "Local Velocity-Adapted Motion Events for Spatio-Temporal Recognition," *Computer Vision and Image Understanding Volume 108*, Issue 3, pp. 207-229, 2007.
- [76] H. Jhuang, T. Serre, L. Wolf and T. Poggio, "A Biologically Inspired System for Action Recognition," in *IEEE 11th International Conference on Computer Vision*, Rio de Janeiro, Brazil, 2007, pp. 1-8.
- [77] O. T. Oshin, A. Gilbert, J. Illingworth and R. Bowden, "Learning to Recognise Spatio-Temporal Interest Points," in *Machine Learning for Human Motion Analysis: Theory and Practice*, Hershey, New York, USA, Medical Information Science Reference, IGI Global, 2010, pp. 14-30.
- [78] M. Ozuysal, P. Fua and V. Lepetit, "Fast Keypoint Recognition in Ten Lines of Code," in *2007 Proceedings of IEEE Conference on Computer Vision and*, Minneapolis, MN, USA, 2007, pp. 1-8.
- [79] H. Zhang, "The Optimality of Naive Bayes," in *Florida AI Research Society (FLAIRS) Conference 2004*, Miami Beach, Florida, USA, 2004, pp. 562-567.
- [80] P. Dollar, V. Rabaud, G. Cottrell and S. Belongie, "Behavior Recognition via Sparse Spatio-temporal Features," in *Proceedings of 2nd Joint IEEE International Workshop on Visual Surveillance and Performance*, Beijing, China, 2005, pp. 65-72.
- [81] A. Gilbert, J. Illingworth and R. Bowden, "Action Recognition using Mined Hierarchical Compound Features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 33, Issue 5, pp. 883-897, May 2011.
- [82] M. Grundland and N. A. Dodgson, "The Decolorize Algorithm for Contrast Enhancing, Color to Grayscale Conversion," *Technical Report UCAM-CL-TR-649*, Computer Laboratory, University of Cambridge, 2005.
- [83] K. Rasche, R. Geist and J. Westall, "Detail Preserving Reproduction of Color Images for Monochromats and Dichromats," *IEEE Computer Graphics and*



- Applications, Volume 25, Issue 3*, pp. 22-30, 2005.
- [84] K. Rasche, R. Geist and J. Westall, “Re-coloring Images for Gamuts of Lower Dimension,” *Computer Graphics Forum, Volume 24, Issue 3*, p. 423–432, 2005.
- [85] A. Alsam and Ø. Kolås, “Grey Color Sharpening,” in *Fourteenth Color Imaging Conference: Color Science and Engineering Systems, Technologies, Applications*, Scottsdale, Arizona, USA, 2006, pp. 263-267.
- [86] D. A. Socolinsky, “A Variational Approach to Image Fusion,” *PhD Thesis, Johns Hopkins University*, 2000.
- [87] R. Bala and R. Eschbach, “Spatial Color-to-grayscale Transform Preserving Chrominance Edge Information,” in *Twelfth Color Imaging Conference: Color Science and Engineering Systems, Technologies, and Applications*, Scottsdale, AZ, USA, 2004, pp. 82-86.
- [88] A. A. Gooch, S. C. Olsen, J. Tumblin and B. Gooch, “Color2Gray: Saliency-Preserving Color Removal,” *Proceedings of ACM SIGGRAPH 2005, Volume 24 Issue 3*, pp. 634-639, 2005.
- [89] M. S. Drew, D. Connah, G. D. Finlayson and M. Bloj, “Improved Colour to Greyscale via Integrability Correction,” in *Proceedings of SPIE 7240, Human Vision and Electronic Imaging XIV*, San Jose, CA, USA, 2009.
- [90] V. Ferrari, M. Marin-Jimenez and A. Zisserman, “Pose Search: Retrieving People Using Their Pose,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Anchorage, Alaska, USA, 2009, pp. 1-8.
- [91] C. Rother, V. Kolmogorov and A. Blake, “Grabcut: Interactive Foreground Extraction Using Iterated Graph Cuts,” *ACM Transactions on Graphics (SIGGRAPH)*, pp. 309-314, Aug. 2004.
- [92] International Color Consortium, “Specification ICC.1:2004-10 (Profile version 4.2.0.0) Image technology colour management — Architecture, profile format, and data structure,” 2006.
- [93] L. Horvat, in *Digital Imaging: Essential Skills*, Focal Press, 2003, p. 74.
- [94] S. Jennings, “Artist's Color Manual: The Complete Guide to Working with

- Color,” Chronicle Books LLC, 2003, p. 21.
- [95] M. Stokes, M. Anderson, S. Chandrasekar and R. Motta, “A Standard Default Color Space for the Internet - sRGB,” HP, Microsoft, Nov. 1996.
- [96] R. Johnson and D. Wichern, “Clustering, Distance Methods, and Orination,” in *Applied Multivariate Statistical Analysis*, Prentice-Hall press, 2007, pp. 726-799.
- [97] J. B. MacQueen, “Some Methods for Classification and Analysis of Multivariate Observations,” in *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, Volume 1*, University of California Press, 1967, p. 281–297.
- [98] S. P. Lloyd, “Least Squares Quantization in PCM,” *IEEE Transactions on Information Theory, Volume 28, Issue 2*, p. 129–137, Apr. 2009.
- [99] D. Arthur and S. Vassilvitskii, “k-means++: the Advantages of Careful Seeding,” in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, Louisiana, USA, 2007, p. 1027–1035.
- [100] J. Foley, A. van Dam, S. K. Feiner and J. F. Hughes, *Computer Graphics: Principles and Practice*, Addison-Wesley, 1990, p. 1174.
- [101] F. Meyer, “Topographic Distance and Watershed Lies,” *Signal Processing - Special issue on mathematical morphology and its applications to signal processing, Volume 38, Issue 1*, pp. 113-125, Jul. 1994.
- [102] J. Kruskal, “Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis,” *Psychometrika, Volume 29, Issue*, pp. 1-27, 1964.
- [103] A. Webb, in *Statistical Pattern Recognition*, Oxford University Press, 1999, 1st Edition, pp. 275-284.
- [104] D. Shepard, “A Two-dimensional Interpolation Function for Irregularly-spaced Data,” in *Proceedings of the 1968 23rd ACM National Conference*, 1968, p. 517–524.
- [105] J. Ren, J. Jiang, D. Wang and S. Ipson, “Fusion of Intensity and Inter-component Chromatic Difference for Effective and Robust Colour Edge Eetection,” *IET Image Processing, Volume. 4, Issue 4*, p. 294–301, Aug. 2010.

- [106] J. Ren, *jinchang.ren@eee.strath.ac.uk*, Apr. 2012, Private Communication.
- [107] L. R. Dice, in *Measures of the Amount of Ecologic Association Between Species Ecology* 25 (3), JSTOR 1932409, 1945, pp. 297-302.
- [108] N. Dalal, "Finding People in Images and Videos," *PhD Thesis, Institut National Polytechnique De Grenoble*, 2006.
- [109] D. G. Lowe, "Distinctive Image Features from Scale-invariant Keypoints," *International Journal of Computer Vision, Volume 60, Issue 2*, p. 91–110, Nov. 2004.
- [110] D. Comaniciu, "An Algorithm for Data-driven Bandwidth Selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 25, Issue 2*, p. 281–288, Feb. 2003.
- [111] M. Day and J. A. Robinson, "On Output-Processing in Face Detectors," in *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, Kuala Lumpur, Malaysia, 2011, pp. 46-51.
- [112] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 8, Issue 6*, p. 679–698, Nov. 1986.
- [113] J. Weston, A. Elisseeff, G. BakIr and F. Sinz, "The Spider," [Online]. Available: <http://people.kyb.tuebingen.mpg.de/spider/main.html>.
- [114] D. Coomans and D. Massart, "Alternative k-nearest Neighbour Rules in Supervised Pattern Recognition : Part 1. k-nearest Neighbour Classification by using Alternative Voting Rules," *Analytica Chimica Acta, Volume 136*, p. 15–27, 1982.
- [115] J. Weston and C. Watkins, "Multi-class Support Vector Machines," Royal Holloway, University of London, 1998.
- [116] Y. Luo, T. D. Wu and J. Neng, "Object-based analysis and interpretation of human motion in sportsvideo sequences by dynamic bayesian networks," *Computer Vision and Image Understanding, Volume 92, Issues 2–3*, p. 196–216, Nov.-Dec. 2003.
- [117] H. Li, S. Lin, Y. Zhang and K. Tao, "Automatic Video-based Analysis of

Athlete Action,” in *14th International Conference on Image Analysis and Processing*, Modena, Italy, 2007, pp. 205-210.

- [118] H. Li, S. Lin, Y. Zhang and K. Tao, “Automatic Detection and Analysis of Player Action in Moving Background Sports Video Sequences,” *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 351-364, March 2010.