國立臺灣師範大學
資訊工程研究所碩士論文

指導教授：李忠謀 博士

以視覺為基礎之即時指揮手勢追蹤系統

A Vision-based Real-time Conductor Gesture Tracking System

研究生：莊謹萍 撰

中華民國 九十七 年 七 月

摘 要

以視覺為基礎之即時指揮手勢追蹤系統

莊謹萍

隨著網路視訊的普及，網路攝影機品質日趨優良、價格也相對低廉，本研究旨在提出一個「指揮者手勢追蹤系統」，代替鍵盤與滑鼠作為輸入單元，讓使用者能透過視訊攝影機(Webcam)及個人電腦、運用基本的指揮動作，能夠即時追蹤使用者手勢的軌跡與方向變化、偵測音樂節拍所在的時間點。

本研究可分為兩個主要階段：第一階段為目標物追蹤，採用 CAMSHIFT 演算法來實現物件追蹤。CAMSHIFT 演算法為平均位移演算法的改良，此演算法利用使用者所感興趣的顏色機率分佈特性，經由平均位移迭代的方式，找出其機率分佈圖的峰值，此峰值即為可能性最高之影像區塊並得到物體移動路徑。第二階段則利用兩種方法計算：K-曲率法則以及垂直分量低點偵測。K-曲率法則利用物體移動路徑各點之區率並計算找出其方向轉變；而垂直分量低點偵測則是找出物體移動的垂直低點，將此低點定義為音樂的節拍點。

本研究所開發之系統可以讓使用者自行選定偵測目標（如指揮棒）並準確偵測移動的軌跡，將使用者的指揮動作上方向的改變，轉變成音樂檔的節拍事件，其準確率平均可達 86.46%以上。

ABSTRACT

A Vision-based Real-time Conductor Gesture Tracking System

By Chin-ping Chuang

In recent years, interaction between humans and computers is becoming more important. “Virtual Orchestra” is an Human Computer Interface (HCI) software which attempts to authentically reproduce a live orchestra using synthesized and sampled instruments sounds. Compared with the traditional HCIs, using vision-based gesture can provide a touch-free interface which is less bounding than mechanical instruments. In this research, we design a vision-based system that can track the hand motions of a conductor from webcam and extract musical beats from motions.

The algorithm used is based on a robust nonparametric technique for climbing density gradients to find the mode of probability distributions. For each frame, the mean shift algorithm converges to the mode of the distribution. Then, the CAMSHIFT algorithm is used to track the moving objects in a video scene. After acquiring the target center point continuously, we can form the trajectory of moving target (such as baton, conductor’s hand…etc). By computing an approximation of k-curvature for the trajectory, and the angle between these two motion vectors, we can compute the point of the change of direction.

In this thesis, a system was developed for interpreting a conductor’s gestures and translating theses gestures into musical beats that can be explained as the major part of the music. This system does not require the use of active sensing, special baton, or other constraints on the physical motion of the conductor.

謹獻給我最親愛的家人

致　謝

兩年的研究所時光，隨著這篇論文的完成告一段落。感謝我的指導教授李忠謀教授，無論在專業或是日常生活中有任何問題，老師都能給予我幫助與指導，使我在做研究的方法與態度上獲益良多、在對未來的規劃上也更加明確。

感謝VIP實驗室的全體成員：謝謝富豪學長、政杰學長以及育慈學姐對於我的研究給了我相當多的寶貴建議，讓我的論文更加完善；感謝在趕論文期間陪我寫程式、一起DEBUG的定翔，沒有你的幫忙，我的系統肯定無法完成；感謝實驗室的學弟妹們，因為有你們讓實驗室充滿歡笑與活力。

感謝育妏幫我完成系統所需的器材以及郁樺適時地打氣，能在研究所期間認識你們兩個好室友真的很棒！還要謝謝大學時期的好友們，無論是快樂或是難過的時候，都有你們的共享與陪伴，讓我的研究生歲月充實且無悔。

最後，更要感謝陪我度過每個重要時刻的父母以及和我同時接受畢業煎熬的哥哥，無時無刻給我精神上的鼓勵、支持與生活上的照顧、體諒，是我能完成研究所學業最重要的支柱。

再次，感謝你們給我的支持與幫助。

謹以最誠摯的心，願大家心想事成、一切順利。

莊謹萍

於國立台灣師範大學資訊工程研究所

中華民國九十七年七月二十三日

TABLES OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Overview of the Problem

In recent years, with the rapid development of computer technology, the computer is being used in more applications and the computer processing power is growing. At this stage, the interaction between humans and computers becomes more and more important. A user interface, such as a GUI, is how a human interacts with a computer, and Human Computer Interaction (HCI) goes beyond designing screens and menus that are easier to use [1][2]. Many researchers tried to develop a new kind of HCI and there are many new HCI technologies used widely in face tracking, hand tracking, face recognition and gesture recognition field.

"Virtual Orchestra", as the name implies, is one of the HCI software which attempt to authentically reproduce a live orchestra using synthesized and sampled instruments sounds. We can describe the scenario as an example: If in concert a singer takes a tempo different from the rehearsal tempo, the Virtual Orchestra should adapt in real time based on the information providing by the system procedure [3]. Such a system would offer increased usability in a public space, and it would also be a very useful training tool for student conductor.

Keyboards and mice are the standard devices that people typically use to interact with a computer. These standard devices might be convenient for office work, but for some other areas better alternatives are needed. For instance, gestures are more natural and convenient to express our intention. In the area of conducting music, there exists a language of conducting gestures that enable a conductor to communicate with an orchestra. These gestures are more formal than gestures used in daily life but they are still very expressive in their own application domain, so they can become a very useful gesture input [4]. Using vision-based gesture as the inputs, we can provide a touch-free interface which is less bounding than mechanical instruments.

1.2 Challenges

Designing a system that can track the motions of an orchestra conductor and at the same time extract a musical beat and other information (such as volume) from those motion is not a trivial problem. While new interaction methods for digital music have became hot topics in today’s research community, much of the work has remained. Researchers have examined several different input devices to achieve our goal. Based on previous researches, these approaches can be classified into two categories: One is the “instrumented baton”, and the other is “vision-based system” [5].

Within the “instrumented baton” approach, the researchers always use sensors to monitor body measures (such as heart rate, temperature) or use special batons fitted with infer-red sources to monitor their locations. But these instrumented batons are fatiguing in prolonged use, do not feel like "the real thing", and need to attach some special hardware. In order to reduce the device cost and provide the more approachable interface, the other vision-based approach is needed.

These vision-based systems sense the motion of conductor’s baton and hand with video cameras. This kind of mechanism provides an advantage that the conductor or musician does not need to carry any other device or sensor. The conductor’s motion is captured from the video camera aiming at the conductor. What we are concerned are to detect human motions and extract musical beats correctly, furthermore, control the music playback via these information.

In order to detect and track these objects correctly, the lighting environment needs to be build properly to make the baton and the conductor’s hands clearly visible. The conductor’s baton needs to be coated with some material whose color is different from the environment at the tip and the conductor needs to wear the special-colored glove to overcome environmental problem and deduce the processing complexity. So that we can assure the moving targets are uniformly bright against a reasonably contrasting background.

We can obtain the location of the baton’s tip or the centroid of the hand while we are tracking the baton or hands. In order to play the music dynamically, some special points (we called it “beat”) need to be mapped to parameters that generate the sound. It poses the question, “What is the way that a real musician interprets the conductor?” We tried to use heuristic rules to extract the main beat of an up-down movement or some features, such as the change of direction, to solve this problem.

Another problem is the discrete sampling of the baton locations due to the limited frame-rate of the video sequence [6]. This procedure must run fast and efficiently with consuming as few system resources as possible, so that the object could be tracked in real time.

1.3 Objectives

In this thesis, we design an HCI system for interpreting a human conductor’s gestures and translating theses gestures into musical beats that can be explained as the major part of the music. This system enables conductors to conduct electronic music using natural gestures of a baton or hands which are acquired and translated by a video recognition system into beat and tempo control. It does not require any use of active sensing, special baton, or other constraints on the motion of the conductor. The goal of our work is using the technology to support these methods of multimedia interaction, and to encourage people exploring music more interactively.

1.4 Thesis Organization

Following the Introductory chapter, Chapter 2 presents an overview of existing conductor gesture tracking system. We also discuss the performances of the previous methods. Chapter 3 describes our system framework and details the algorithms we proposed for conductor’s gesture tracking. Chapter 4 is the experimental result and discussions. Finally, conclusions and future works are given in Chapter 5.

Chapter 2

Literature Review

2.1 Key Terms in Music Conducting

This section describes some music conducting terminologies for those users with no prior music experience. It is based on several conducting on-line course documents [7] and the use of Wikipedia encyclopedia [8][9].

2.1.1 Beat

In written music, *beats* and *notes* are grouped into *measures*. The *beat* of the music is typically indicated with the conductor's right hand, with or without a baton. The hand traces a shape in the air in measure and indicates each beat with a change from downward to upward motion. The instant at which the beat occurs is usually indicated by a sudden click of the wrist or change in baton direction (as Figure 2.1).

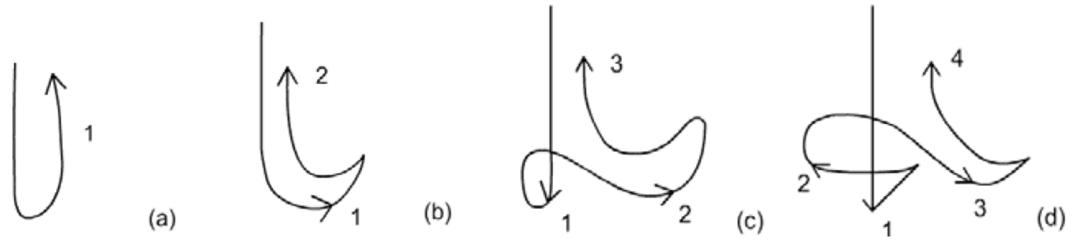


Figure 2.1 The conductor has to “write” in the air to create different beats [9]

2.1.2 Tempo

Tempo, the Italian word for “time”, is the speed of the fundamental beat and should stay even from beat to beat. The tempo of a piece will typically be written at the start of a piece of music, and in modern music is usually indicated in beats per minute (BPM). The greater the tempo, the larger the number of beats that must be played in a minute is and, therefore, the faster a piece must be played.

2.1.3 Dynamics (Volume)

In music, dynamics normally refers to the softness or loudness a sound or *note*. It always be communicated by the size of the conducting gestures: the larger the shape, the louder the sound. Changes in dynamic may be signaled with hand that is not being used to indicate the beat: an upward motion indicates a *crescendo* and a downward motion indicates a *diminuendo* or *decrescendo*. In general, loud dynamic levels are conducted with larger beat patterns, and soft dynamic levels with small beat patterns.

2.1.4 Other Musical Elements

Time signatures are figures written on the score at the start of the composition. Each *measure* is assigned a *meter* that tells the musician how many rhythmic beats there are within the *measure* and what type of *note* equals one *beat* [3]. For instance, a *measure* with a $\frac{2}{4}$ *meter* tells the musician there are two rhythmic *beats* in the *measure* (from the “2” in the numerator) and quarter *notes* are the fundamental *beats* for this *measure* (from the "4" in the denominator). The number of *beats* per *measure* and the *time signature* usually stay the same from the start of a song to the end, but it may vary on different *time signature*.

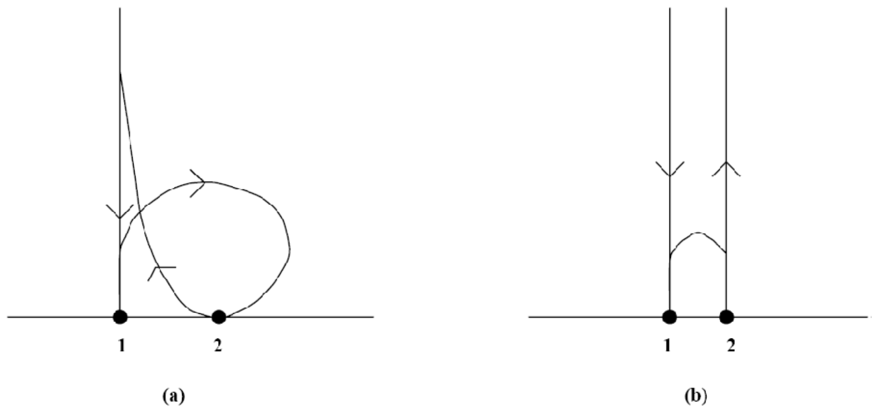


Figure 2.2 Two kinds of 2-beat conducting patterns [3]
(a) A legato style pattern (b) a staccato style pattern

Every *meter* can have a range of conducting patterns depending on the style of the piece and the type of mood that the conductor is trying to present to the musicians and audience. The conductor may use flowing gestures as a means of expressing the *legato* style. Alternatively, a piece like a "march" that is in the same meter as the *legato* piece would be conducted in a much more angular manner which we called it *staccato* style. These two terms *legato* and *staccato* indicate how much silence is to be left between notes played one after another.

2.2 Reviews of Conductor Gesture Tracking Systems

There are many approaches to design a system which can understand a human conductor gesture [10]. In this section, we classified these systems into two categories in conductor’s gesture tracking. The reviews of instrumented baton will be introduced in Section 2.2.1, and the other categories of vision-based conductor’s gesture tracking systems will be mentioned in Section 2.2.2.

2.2.1 Instrumented Batons

Some approaches introduce special instrumented batons as the input of the system. Max Mathews, who is called one of the fathers of computer music, created the first conductor’s gesture tracking system. His system, *Radio Baton*, consisted of two batons that omitted radio waves from their tips and a plate which was equipped with antennas to receive the signals emitted by the batons [11].

Buchla Lightning Baton Series (see Figure 2.3) is another instrumented baton that senses the position and movement of wands and transforms this information to MIDI signals to control musical instrumentation more expressively [12]. The baton has been used in many systems, which is like *Adaptive Conductor Follower* [13] and *Personal Orchestra* [14].

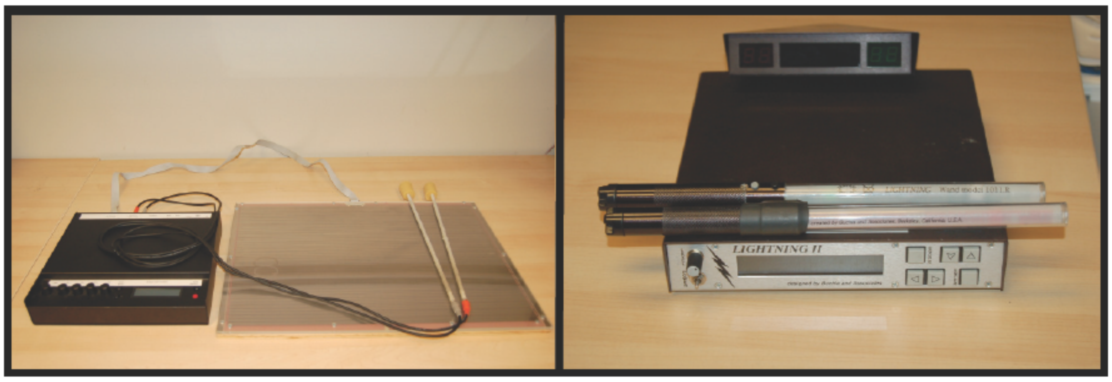


Figure 2.3 Radio Baton (left) and Buchla Lightning II (right) [14]

Besides using Lightning Baton as an input, *Adaptive Conductor Follower* [13] provided three methods of tracking and predicting tempo at beat analysis stage. The most important achievement of the system was that it produced the first attempt using Neural Networks for recognition purpose. In the *Extraction of Conducting Gestures in 3D Space* [16], it included two Lightning batons to track the baton's movement in 3D space and extracted information including tempo, dynamic, beat pattern and beat style.

Teresa Marrin Nekra *et al.* presented *Digital Baton* in 1997 [17]. They developed an input device which included acceleration sensors to measure baton's movement and other pressure sensors to obtain the every finger pressure values of the hand holding the baton with an infrared LED at the tip of the baton. A position-sensitive photodiode was placed behind a camera lens to track position of the infrared LED. But there was no beat information was derived from the input to control the tempo of the piece. In 1998, they created *the Conductor's Jacket system* [18][19] used a multitude of sensors built into a jacket to record physiological and motion data. *The Conductor's Jacket* consisted of four muscle tension (EMG) sensors, heart rate and respiration monitor, temperature sensor, and skin response sensor (see Figure 2.4(a)).

Tommi Ilmonen *et al.* created a system to track conducting gestures with neural networks and it was part of a system *DIVA* (see Figure 2.4(b)) to extract rhythm data from conductor's movements [20][21]. It used magnetic motion tracker as the input to collect positional information and used the neural networks to classify and predict beats. But the system has some limitations: The system only understand the standard conducting techniques, so users must have some prior knowledge about conducting.

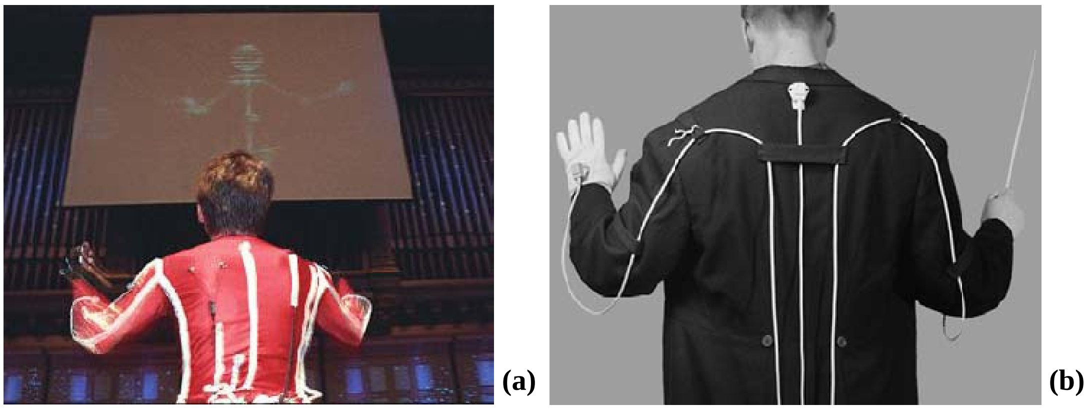


Figure 2.4 (a) Conducting an orchestra with the *Conductor's Jacket system* [10]
(b) The magnetic sensors for motion tracking at *DIVA* [21]

In 2002, Jan Borchers et al. designed the *Personal Orchestra* system [14] using a Buchla Lightning baton [12]. Users can interact not with a synthetic, but an original audio/video recording of a real orchestra. A beat was detected each time the baton changed from going down to going up and vertical coordinates of the left hand was dynamic indicators. Its audio stretching algorithm which rendered audio and video at variable speed without time-stretching artifacts, such as pitch changes, made a notable contribution in further systems. Moreover, it did not contain complex rules to extract beats and tempo. So the system could be a more general system for those users with little or no conducting experience. It is also the first system that allows users to control an audio/video recording in real time, using natural conducting gestures.

2.2.2 Vision-based Conductor Gesture Tracking Systems

In 1989, Morita *et al.* built a system, *A Computer Music System that Follows a Human Conductor.* This system tracked either a white marker attached to the baton or the hand wearing a white glove, using a CCD camera and special feature extraction hardware that passed two-dimensional position values to a PC. The computer derived tempo and volume information from upper and lower turning points of the trajectory. It is the first project to use a CCD camera as an input device [22].

Light Baton created in 1992 by Graziano Bertini *et al.* was another system using a CCD camera and the baton with a LED light on its tip [22]. The position of the baton was analyzed by a special image acquisition board and the playback of pre-recorded score is adjusted in terms of tempo and intensity of notes.

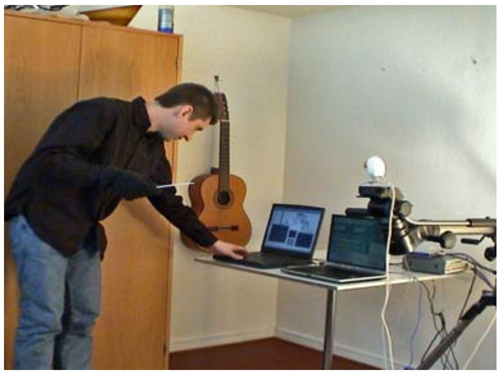
Michael T. Driscoll [3] proposed *A Machine Vision for Capture and Interpretation of an Orchestra Conductor's Gestures* in his master thesis in 1999. This work involves the design and implementation of a real-time Vision-based HCI that analyzes and interprets a music conductor's gestures to detect the beat. It used several basic image processing methods, such as rapid RGB color thresholding, multiple contour extraction, and center of mass calculations to understand the time location of beats. But there were performance constraint due to the insufficient time resolution of the system. This issue might create a “choppy" response by the Virtual Orchestra.

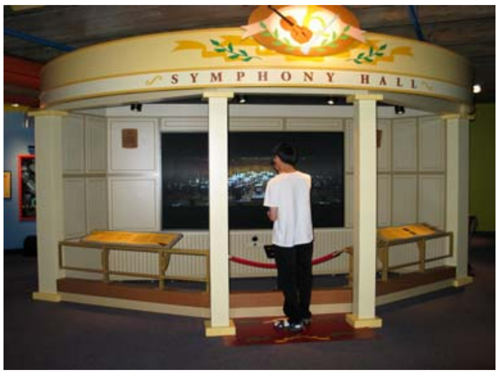
In 2000, Jakub Segen *et al.* created their *Virtual Dance and Music system* [24] and *Visual Interface for Conducting Virtual Orchestra* [25]. It used two synchronized cameras to acquire a 3D trajectory of the baton and beats were placed at the locally lowest trajectory points. It aloes presented an simple scenario in beat following: If the music sequencer had already played all notes corresponding to the current beat, the sequencer should wait until the user conducted the next beat, and if the user had already conducted the next beat before the music sequencer finished playing the notes corresponding to the last beat, the sequencer should increase the tempo slightly to catch up with the conductor.

Declan Murphy *et al.* presented a conducting gesture recognition system which was able to control the tempo of an audio file playback through standard conducting movements in 2003 [29]. It worked with one or two cameras as input sources for front and side view of the conductor. Computer vision techniques were then used to extract position and velocity of the tip of the baton or of the conductor's right hand.

In Section 2.2.1, we mentioned about T. Marrin and J. Borchers who involved in this field several years. They worked with Eric Lee to design another museum exhibit for the Children's Museum in 2004. The final system was called *You're the Conductor* [26]. Instead of employing a *Buchla Lightning* baton for input, a rugged baton-like device was developed, which was mainly a light source and could stand heavy use.

Movement of baton was translated into playback speed and volume, so that children for all ages could use the system. If the child started moving the baton faster or slower, the orchestra sped up or slowed down, respectively, and if the child stopped moving the baton, the orchestra slowed to a halt. Compared with *Personal Orchestra*, it used a real-time, high-quality time stretching algorithm for polyphonic audio and allowed the system to respond more instantaneously and accurately to user input.


(a)



(b)

Figure 2.5 (a) Declan Murphy using his conducting system [29]
(b) *You're the Conductor* exhibit at Children's Museum of Boston [26]

R. Behringer presented another system *Conducting Digitally Stored Music by Computer Vision Tracking* in 2005 [6]. At this stage, Computer vision methods are used to track the motion of the baton and to deduce musical parameters (volume, pitch, expression) for the time synchronized replay of previously recorded music notation sequences. Combined with acoustic signal processing, this method can provide the automatic playing which the conductor conducts both this instrument as well as the human musicians.

In 2007, Terence Sim *et al.* presented a *vision-based, interactive music playback* *system*, called VIM, which allows anyone, even untrained musicians, to conduct music [31]. Because of the assumption above, they did not recognize any specific musical gestures which mean any kind of motion will suffice and used a webcam to capture the movements. This system applied the Intel OpenCV Library as a useful programming tool, to decide the speed and area of the moving object. This system also provided some visualization to project colorful patterns that respond to the user.

2.2.3 Summary of Conductor Gesture Tracking Systems

This section presents a summarized description of all systems described in this chapter in table format [10] (see Table 2.1). To simplify our expression, we usually have the name of the latest system as the representative of the entire series which were created by the same research group unless there are major changes between two systems.

Table 2.1 The summary of the conductor gesture tracking system

Year	Name	Authors	Input Device	Tracked Info.	Control Var.	Output	Features
1989	Computer Music System that Follows a Human Conductor	Hideyuki Morita, Shuji Hashimoto, Sadamu Ohteru	CCD camera, White glove Baton with marker	Baton 2D position, Trajectory,	Tempo, Dynamics	Prerecorded MIDI	Computer vision system
1991	Radio Baton	Max Mathews	two radio-wave batons, a plate with antennas	2 batons positions above the plate	Tempo, Dynamics, Voice balance	Prerecorded MIDI	First baton conducting system for computers, Limit: small work area
1992	Light Baton	Bertini Graziano, Paolo Carosi	CCD camera Baton with LED	Baton 2D position,	Tempo, Dynamics,	Prerecorded MIDI	Use image acquisition board to get the baton position
1991-1996	Buchla Lightning Series	Buchla and Associates	Batons with tiny infrared transmitters	Batons' coordinates (four values total)	MIDI notes	32 voice synthesizer	Expensive commercial equipments
1992-1995	Conductor Follower	Bennett Brecht, Guy Garnett	Buchla Lightning Baton, Mattel Power Gloves	Baton 2D position,	Tempo, Dynamics,	Prerecorded MIDI	First system to use neural network for beat analysis

Table 2.1 The summary of the conductor gesture tracking system (Cont.)

Year	Name	Authors	Input Device	Tracked Info.	Control Var.	Output	Features
1995-1996	Extraction of Conducting Gestures in 3D Space	Forret Tobey, Ichiro Fujinaga	Two Buchla Lightning Batons	Baton 3D position,	Tempo, Dynamics, Beat patterns, Beat style,	Prerecorded MIDI	First system to acquire 3D position coordinates
1996	Digital Baton	Teresa Marrin, Joseph Paradiso	Digital Baton	Pressure, Acceleration LED coordinates,	Tempo, Dynamics,	Prerecorded MIDI	Combined several sensors onto one baton
1998	Conductor's Jacket	Teresa Marrin, Rosalind Picard	Physiology sensors, Motion sensors	Muscle tension, Breath speed, Heart rate, Skin response	Tempo, Dynamics, Articulation, Vibrato…etc	Prerecorded MIDI	Record the gestural motion and musical gesture, but Jacket is weird
1999	Conductor Following with Artificial Neural Network	Tommi Ilmonen, Tapio Takala	Data dress suit with 6-dof sensors (motion trackers)	3D positional information for the body	Tempo, Dynamics, Vibrato…etc.	Prerecorded MIDI, 3D graphics	ANN analysis system with 6-dof positional sensors
1999	A Machine Vision System of an Conductor's Gestures	Michael T. Discoll	1 video camera	Right hand 2D position	Tempo,	Prerecorded MIDI,	Use basic IP methods to extract the hand's position.

Table 2.1 The summary of the conductor gesture tracking system (Cont.)

Year	Name	Authors	Input Device	Tracked Info.	Control Var.	Output	Features
2000	Visual Interface for Conducting Virtual Orchestra	Jakub Segen Senthil Kumar Joshua Gluckman	2 video cameras	Right hand 3D position	Tempo, Dynamics,	Synchronized prerecorded MIDI / Video	Do synchronization between music and video
2002	Personal Orchestra	Jan O. Borchers, Wolfgang Samminger, Max Mühlhäuser	Buchla Lightning Baton	Right hand 2D position	Tempo, Dynamics, …etc.	Synchronized prerecorded Audio / Video	No recognition of beat patterns, just up or down movements
2003	Conducting Audio Files via Computer Vision	Declan Murphy, Tue Haste Anderson, Kristoffer Jensen	1 or 2 CMOS cameras (direct / profile)	Right hand 3D position	Tempo	Prerecorded Audio, 3D graphics	Robust and easy obtainable hardware
2004	You're the Conductor	Eric Lee, Teresa Marrin, Jan O. Borchers	Infrared, rugged baton-like device	Right hand 2D position	Tempo, Dynamics, …etc.	Synchronized prerecorded Audio / Video	The input device could stand heavy use
2005	Conducting Digitally Stored Music by Computer Vision Tracking	ReinHold Behringer	CMOS camera	Right hand 2D position	Tempo, Dynamics, Pitch, Expression	Synchronized prerecorded Audio	Is more Intuitive, and provide more applications which can be implemented
2007	VIM: Vision for Interactive Music	Terence Sim, Dennis Ng, etc	A simple webcam	Hand 2D position	Tempo, Dynamics,	Synchronized prerecorded MIDI	Using the Intel OpenCV as a tool.

2.3 Backgrounds on Object Tracking

In general, many previously approaches for motion detection and tracking are conceptually ambiguous. According to the articles written by W. Hu *et al.* [33] and W. Yao [34], we can classify these two issues and introduce them separately in Section 2.3.1 and 2.3.2.

2.3.1 Motion Detection

Identifying moving objects from a video sequence is a fundamental and critical task in many computer-vision applications. Motion detection aims at detecting regions corresponding to moving objects from the rest of an image. Detecting moving regions provides a focus of attention for later processes such as tracking procedure. Here we introduce basic methods as examples: *background subtraction*, *temporal differencing*, and *optical flow*.

(1) Background Subtraction

It is a popular method for motion detection, especially under those situations with a relatively static background. It detects moving regions in an image by identifying moving objects within the current image and the reference background image. But it is highly dependent on changes in dynamic scenes derived from lighting and extraneous events. Therefore, an active construction and updating of the background model are indispensable to reduce the influence of these changes.

(2) Temporal Differencing

Temporal differencing makes use of the pixel-wise differences between two or three consecutive frames in an image sequence to extract moving regions. Temporal differencing is very adaptive to dynamic environments, but it may not work well for extracting all relevant pixels. That is, there may be holes left inside moving entities.

(3) Optical Flow

Optical-flow-based algorithms was used to calculate the optical flow field from a video sequence attempt to find correlations between adjacent frames, generating a vector field showing where each pixel or region in one frame moved to in the next frame. Typically, the motion is represented as vectors originating and terminating at locations in consecutive video sequences. However, most optical-flow-based methods are computationally complex and sensitive to noise, and they cannot be applied to streams in real time without specialized hardware.

2.3.2 Motion Tracking

The tracking algorithms usually have some intersection with motion detection during processing. Although there are many researches trying to deal with the motion tracking problem, existing techniques are still not robust enough for stable tracking. Tracking methods are divided into four major categories: *region-based tracking*, *active contour-based tracking*, *feature-based tracking*, and *model-based tracking*.

(1) Region-Based Tracking

Region-based tracking algorithms track objects according to variations of the image regions corresponding to moving objects. In these algorithms, the background image is maintained dynamically, and motion regions are detected by subtracting the background from the current image. They work well in scenes containing only a few objects, but they may not handle occlusion between objects reliably. So they cannot satisfy the requirements in a cluttered background or with multiple moving objects.

(2) Active Contour-Based Tracking

Active contour-based tracking algorithms track objects by representing their outlines as bounding contours and updating these contours dynamically in successive frames. In contrast to region-based algorithms, these algorithms describe objects more simply and more effectively and reduce computational complexity. Even under partial occlusion, these algorithms may track objects continuously. However, a difficulty is that they are highly sensitive to the initialization of tracking, making it more difficult to start tracking automatically.

(3) Feature-Based Tracking

Feature-based tracking algorithms do the objects tracking by extracting elements, clustering them into higher level features and then matching the features between images. These are lots of features that can help us in tracking objects, like edges, corners, color distribution, skin tone and human eyes. However, the recognition rate of objects based on 2D image features is low, and the stability of dealing effectively with occlusion is generally poor.

(4) Model-Based Tracking

Model-based tracking algorithms track objects by matching projected object models, produced with prior knowledge, to image data. The models are constructed off-line with manual measurement, or other computer vision techniques. Compared with other tracking algorithms, the algorithms can obtain better results even under occlusion (including self-occlusion for humans) or interference between nearby image motions. Ineluctably, model-based tracking algorithms have some disadvantages such as the necessity of constructing the models, high computational cost, etc.

Chapter 3

Vision-based Conductor Gesture Tracking

3.1 Overview

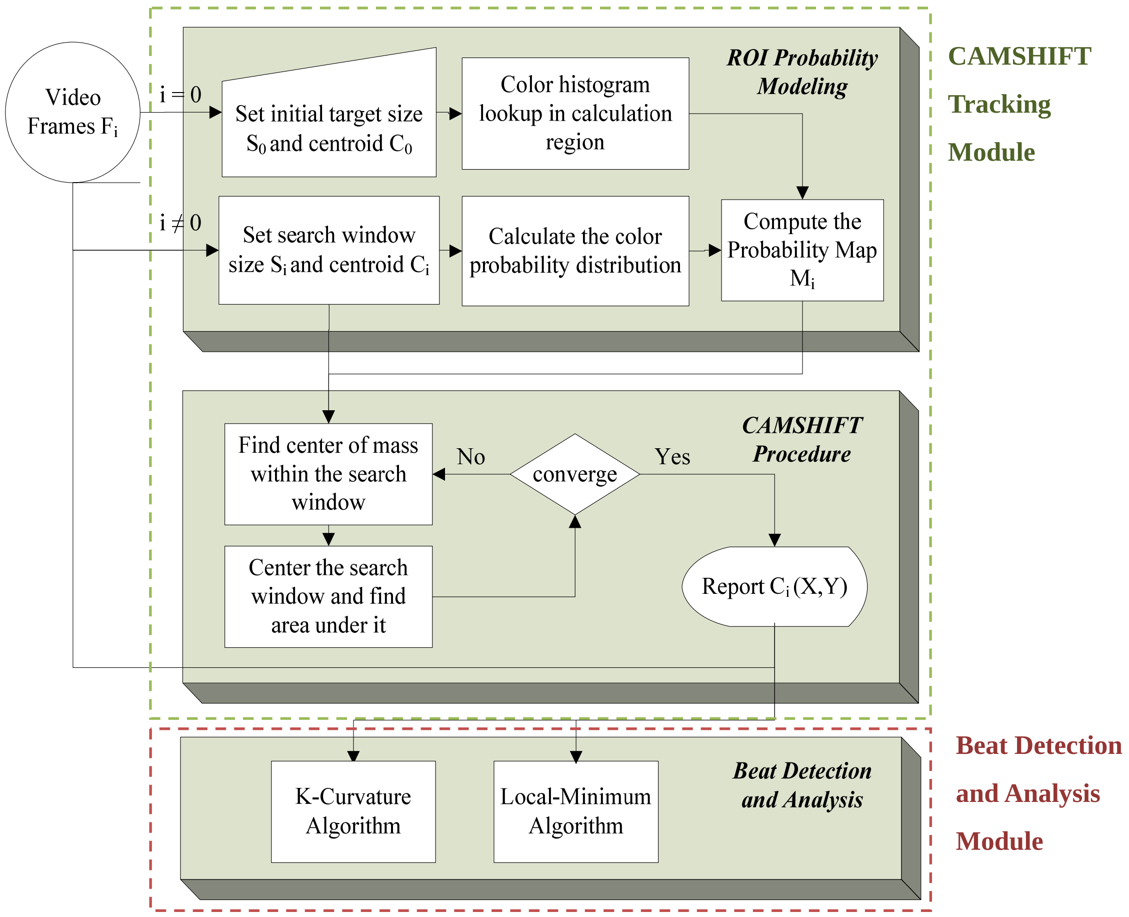
Most of conductor gesture tracking systems presented before focused on technical issues and did not mention how to organize a framework to build a complete HCI. The framework we present here is based on tracking the target that user defined, and the output must be the timing of musical beat after we analyzed. Inspired by face tracking approaches to track the center of our target, the position data must be detected via a real-time algorithm. After acquiring the successive position data, our algorithm was formulated to detect the time point when the target changed its direction. Our proposed framework can be divided into two independent modules, CAMSHIFT tracking and beat detection and analysis module. The diagram of our system is shown in 1Figure 3.1, and the details of these two modules will be discussed in Section 3.2 and 3.3.

(1) CAMSHIFT Tracking Module

We created the *ROI (Region of Interest) probability map* using the 1D histogram from the Hue channel in Hue Saturation Value (HSV) color system that corresponds to projecting standard RGB color space. Then we computed Histogram Back-projection algorithm to calculate the ROI probability map. The major consideration is the correct rate in detecting the moving target, which is a critical issue to the following module.

(2) Beat Detection and Analysis Module

Our system used the movement of the target that we detected in the last module to determine the change of direction. After selecting the WAV file and other parameters, we also display the beat detection result in the visualization waveform of WAV file and calculate the precision and recall rate.



Video Frames F_i
$i = 0$
$i \neq 0$
Set initial target size S_0 and centroid C_0
Color histogram lookup in calculation region
ROI Probability Modeling
Set search window size S_i and centroid C_i
Calculate the color probability distribution
Compute the Probability Map M_i
CAMSHIFT Tracking Module
Find center of mass within the search window
No
converge
Yes
CAMSHIFT Procedure
Center the search window and find area under it
Report C_i (X,Y)
K-Curvature Algorithm
Local-Minimum Algorithm
Beat Detection and Analysis
Beat Detection and Analysis Module

Figure 3.1 Diagram of the framework we proposed

This system was designed in this way so that all algorithms within one module can be changed at will, without affecting the functionality of the other module.

3.2 User-defined Target Tracking Using CAMSHIFT

User-defined target tracking module is the first stage of the system we proposed, which separates the target from the background and extracts position information. In our system, we applied the *CAMSHIFT (Continuously Adaptive Mean Shift)* Algorithm [35][36],which is an efficient and simple colored object tracker, as the kernel of this module.

The target area from the user are sampled by mouse, so the target can be user's head/hand, baton and other objects which color are different from the background. Using the *Histogram Back-projection* method, we can measure the characteristics of target which we are interested in to build *ROI probability model* which is also the first step of the. At the same time, color distribution derived from video change over time, so we utilize the Mean Shift algorithm to *calculate the center of mass* of the color probability within its 2D window of calculation, re-centers the window, then calculates the area for the next window size, until convergence (or until the mean location moves less than a threshold which means there is no significant shift). The details are described in as follows.

3.2.1 The CAMSHIFT Algorithm

In the latest years, the tracking algorithm of the Continuously Adaptive Mean Shift Algorithm is being concerned because of its practicability and robustness. For object tracking, CAMSHIFT is an adaptation of the Mean Shift algorithm which is a robust non-parametric technique that climbs the gradient of a probability distribution to find the mode (peak) of the distribution [37].

The primary difference between CAMSHIFT and the Mean Shift algorithm is that CAMSHIFT uses continuously adaptive probability distributions (That is, distributions may be recomputed at each frame) while Mean Shift is based on static distributions, which are not updated unless the target experiences significant changes in shape, size or color [36].

For each video frame, the raw image is converted to a probability distribution image via a color histogram model of the color being tracked. We use the zeroth and first-order moments of target color probability model to compute the centroid of an area of high probability which is the main idea in the Mean Shift algorithm. So, the center and size of the color object are found via the mean shift algorithm operating. The current size and location of the tracked object are reported and used to set the size and location of the search window in the next frame. This process is repeated for continuous tracking [35].

The procedure can be summarized in the following steps [35]:

1. Set the region of interest (ROI) of the probability distribution image (PDF) to the entire image.
2. Select an initial location of the Mean Shift search window.
3. Calculate a color probability distribution of the region centered at the Mean Shift search window.
4. Iterate Mean Shift algorithm to find the centroid of the probability image. Store the zeroth moment (distribution area) and centroid location.
5. For the following frame, center the search window at the mean location found in Step 4 and set the window size to a function of the zeroth moment. Go to Step 3.

Figure 3.2 CAMSHIFT Algorithm

Because CAMSHIFT uses the color feature to track the moving object, the change of the color feature is very small when the object is moving. So it is a robust method. Moreover, much time is economized because the moving object is searched around the position where the object possibly appears. So CAMSHIFT also has real time feature.

3.2.2 Color Space Used for ROI Probability Model

The ROI probability model, also called probability distribution image (PDF), can be determined by methods which associate a pixel value with a probability that the given pixel belongs to the target. In order to create the probability distribution image of the desired color, we first create a model using a color histogram.

The color model that we capture from an image stream is within the standard Red, Green, and Blue (RGB) color model. However, the RGB color model is influenced by illumination easily. Hence, we adopt the Hue Saturation Value (HSV) color model to the RGB one, because of its hue (color) separated from saturation (how concentrated the color is) and from brightness. The RGB color model is additive, defining a color in terms of the combination of primaries, whereas the HSV color model encapsulates the information about a color in terms that are more familiar to humans. Figure 3.3 shows the RGB and HSV color model respectively. Descending V axis in Figure 3.3 (b) gives us smaller hexcone corresponding to smaller (darker) RGB subcube in Figure 3.3 (a).

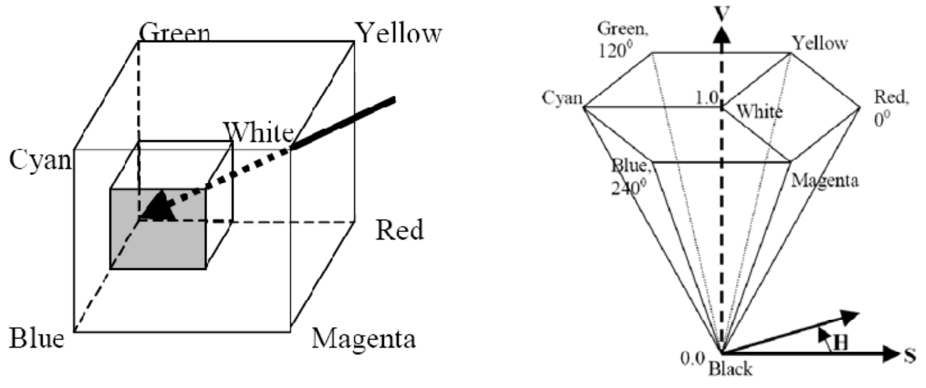


Figure 3.3 RGB color cube (left) and HSV color hexcone (right)

The formula from the RGB color model to the HSV one is described below:

$$H = \begin{cases} \frac{\pi}{3}\left(0 + \frac{b - r}{\max(r,g,b) - \min(r,g,b)}\right) & \text{if } g = \max(r,g,b) \\ \frac{\pi}{3}\left(2 + \frac{b - r}{\max(r,g,b) - \min(r,g,b)}\right) & \text{if } b = \max(r,g,b) \\ \frac{\pi}{3}\left(4 + \frac{b - r}{\max(r,g,b) - \min(r,g,b)}\right) & \text{if } r = \max(r,g,b) \end{cases} \tag{3.1}$$

$$S = \frac{\max(r,g,b) - \mathrm{minx}(r,g,b)}{\max(r,g,b)} \tag{3.2}$$

$$V = \max(r,g,b) \tag{3.3}$$

Where H, S and V represent hue, saturation and luminescent components in HSV color model, the r, g, b represents red, green and blue component in RGB color model.

To generate the PDF, an initial histogram is computed from the initial ROI of the filtered image. We create our color models by taking 1D histogram from the H (Hue) channel in HSV space. When sampling is complete, the histogram is used as a lookup table, to convert incoming video pixels to a corresponding probability of ROI image.

When using real cameras with discrete pixel values, a problem can occur when using HSV space. When brightness is low (V near 0), saturation is also low (S near 0). Hue value becomes quite noisy, since in such a small hexcone, the small number of discrete hue pixels cannot adequately represent slight changes in RGB. To overcome this problem, we simply ignore hue pixels that have very low brightness values. In our system, we set minimum brightness value of target as the lower threshold of brightness value at first. Users also can adjust these two threshold values by themselves.

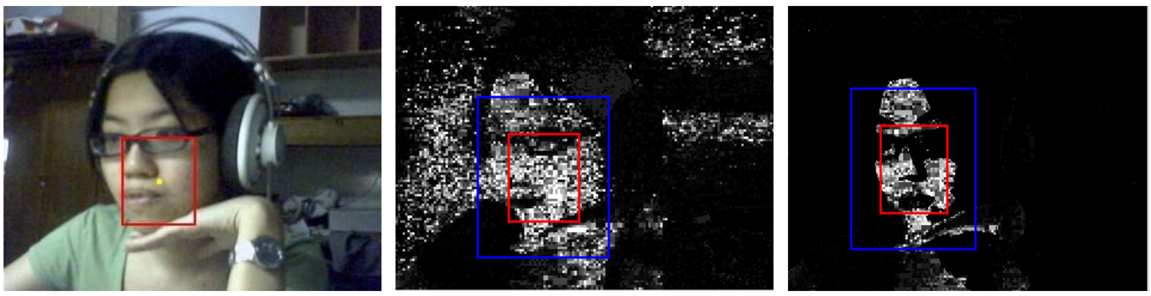


Figure 3.4 (a) The Original image and ROI
(b) The Probability Map with Vmin = 0
(c) The Probability Map with Vmin = 108

3.2.3 Histogram Back-projection

Histogram Back-projection is used to answer the question "Where are the colors that belong to the object we are looking for (the target)?" The back-projection of the target histogram with any consecutive video frame can generate a probability image where the value of each pixel characterizes probability that the input pixel belongs to the histogram that was used.

Suppose we have a target image and try to locate it in a crowded image. We use the 1D Hue histogram of the target image we calculate in the last section. Given that *m*-bin histograms are used, we define the *n* image pixel locations $\{x_i\}_{i=1\ldots n}$ and $\{\hat{q}\}_{u=1\ldots m}$ are the histogram value of the elements. We also define a function $c:\Re^2 \rightarrow \{1 \ldots m\}$ that associates to the pixel at location x_i^* and the histogram bin index $c(x_i^*)$. The unweighted histogram is computed as:

$$\hat{q}_u = \sum_{i=1}^{n} \delta\left[c\left(x_i^*\right) - u\right] \tag{3.4}$$

, where $\delta[x]$ is the unit impulse function.

Using the formula above, we can derive the unweighted histogram, which is also the probability distribution image (PDF) $\hat{q}$ of the target.

The formula of histogram normalization is as:

$$\left\{\hat{p}_u = \min\left(\frac{255}{\max(\hat{q})}\hat{q}_u, 255\right)\right\}_{u=1\ldots m} \tag{3.5}$$

That is, the range of histogram bin values in $[0, \max(\hat{q})]$ can be normalized to the interval of [0, 255] based on Eq. (3.5), where pixels with the highest probability of being in the sample histogram will map as visible intensities. Then the image element value of image corresponding to the value of the histogram for each color can be linked up by back-projection.

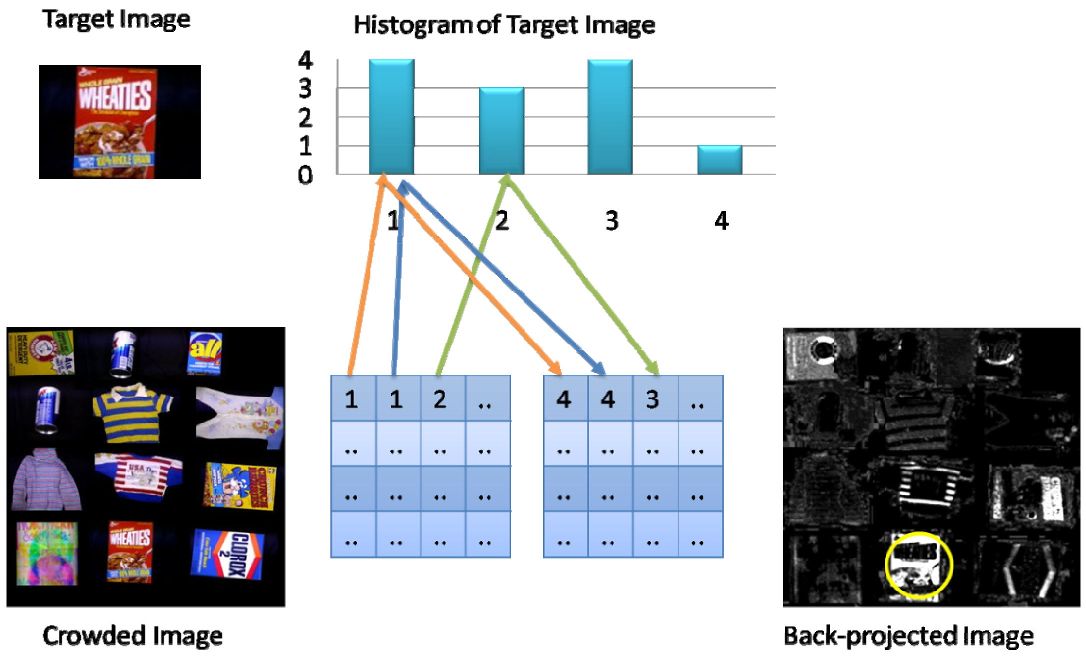


Figure 3.5 Histogram Back-projection Procedure

Looking up the normalized histogram table, we can produce the back-projected image. The center of the target object within the crowded image must be the point which has most neighbors with high value. So the final task is simple, we find the peak after applying the convolution and that will be the point where the object lies.

Figure 3.6 (a) shows the original image and our target ROI is bound by yellow rectangle; Figure 3.6 (d) shows the continuous trajectory. Figure 3.6 (b)(c) are results we tracked, and Figure 3.6 (e)(f) are the probability Image of (b)(c) respectively.

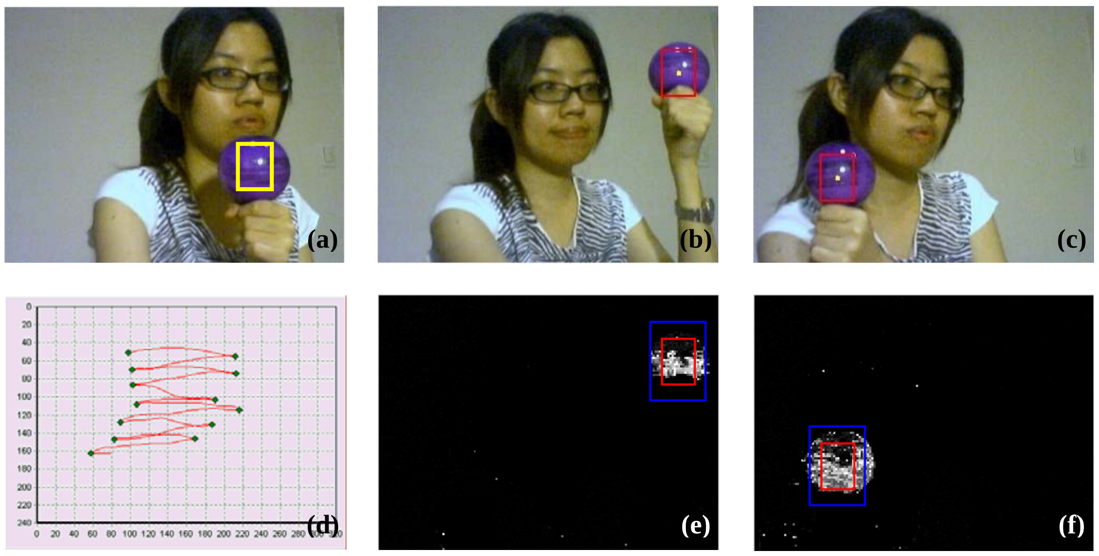


Figure 3.6

(a) Original Image and ROI (b) A Tracked Frame #1 (c) A Tracked Frame #2

(d) The trajectory of ROI (e) Probability Image of #1 (f) Probability Image of #2

3.2.4 Mass of Center Calculation

Image moments is a global image descriptor and have been applied the shape analysis of binary image before. The moments in one image sum over all pixels, so the moment are robust against small pixel value changes. The centroid, mean location, within the search window of the discrete probability image computed found using moments [35].

Given *I(x, y)* is the intensity of the discrete probability image at *(x, y)*, the Mean Shift location can be found as follows:

(1) Compute the zeroth order moment

$$M_{00} = \sum_{x}\sum_{y} I(x, y) \tag{3.6}$$

(2) Find the first order moment for x and y

$$M_{10} = \sum_x \sum_y x \times I(x, y) \tag{3.7}$$

$$M_{01} = \sum_x \sum_y y \times I(x, y) \tag{3.8}$$

(3) Compute the mean search window location

$$\text{Center of Mass} = \left(\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}}\right) \tag{3.9}$$

where *M_{mn}* denotes the *m_{th}* and *n_{th}* moments in *x* and *y* directions. By definition, *M_{00}* is the area of a two dimensional object. Using these Equations, we can descend the pixel data to an object position.

The target of the area in the image is reflected by the moments. The color probability distribution is discrete gray image whose maximum is 255, therefore the relation of the size of search window S and 255 is

$$\text{Search window size} = 2 \times \sqrt{\frac{M_{00}}{256}} \tag{3.10}$$

We also need to set the ratio of window width and length according to the interested probability distribution in search window. For our object tracking, we set the window width to S and window length to $S \times \frac{\text{height}}{\text{width}}$, where *height* and *width* are the height and width value of target selected by user.

3.3 Beat Detection and Analysis

After extracting the target location by user-defined target tracking module, the next step is the beat detection and analysis module. Compared with the exactly hand sign language recognition, the detection of beat events for analyzing a musical time patterns and tempo does not have to be so accurate [39]. Compared with slight posture or hands movement in the hand sign language represents an independent and important meaning, only salient features like beat transition point of hand motion are the most important information in the conducting gesture.

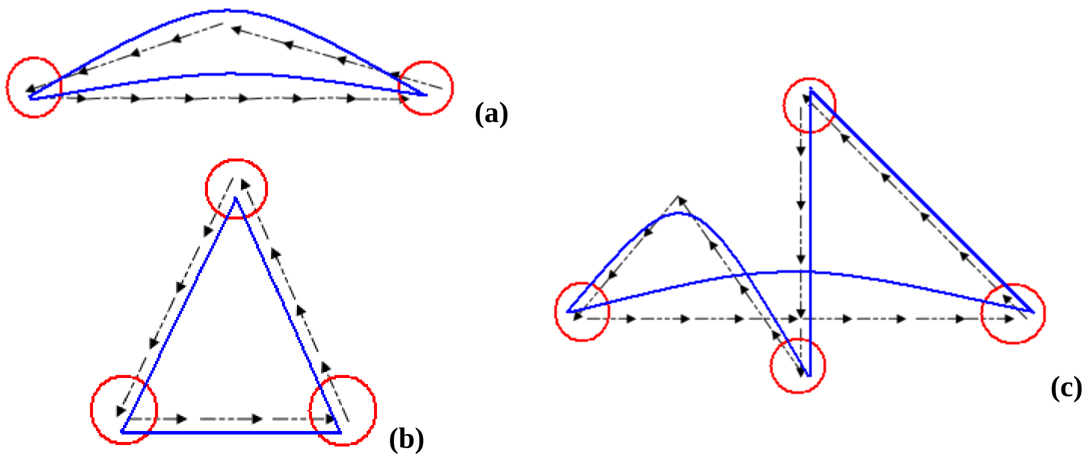


Figure 3.7 The trajectories and the approximated directions of each conducting pattern. (solid line - trajectory, dashed line - motion direction, red circle- direction change point)

Figure 3.7 illustrates the representative features which are called as direction change points (red circle) of each musical time pattern. During the detection of these points (DCP), we can detect the beat event more easily. That is, we can calculate the motion of consecutive extracted target region and generates the gesture features, direction change point, which is the point of sudden motion direction changes.

In order to measure every DCP happened exactly, we compute an approximation of k-curvature, which is defined by the angle between the two vectors [P(i-k), P(i)] and [P(i), P(i+k)], where k is a constant and P(i)=(x(i), y(i)) is the list of trajectory points. When the angle between these two vectors is less than the threshold we defined, we can imply the change of direction point which is also a beat event in conducting gesture.

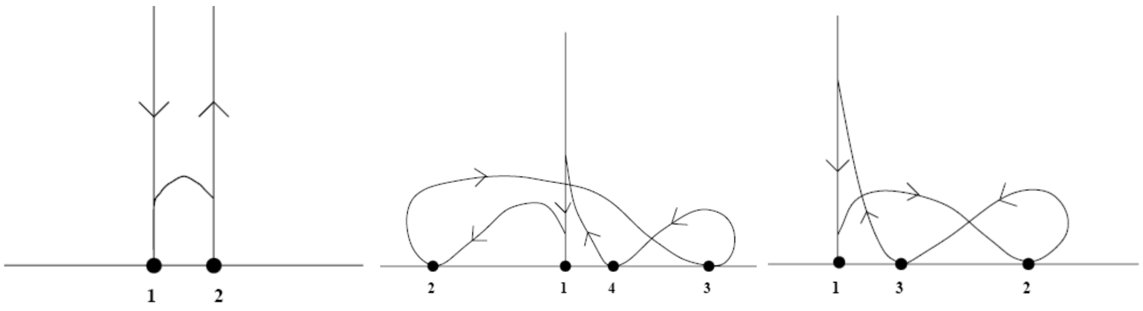


Figure 3.8 Another representative of conducting patterns

Based on another conducting style [25], the beat always occurs at locally lowest points which we called *local-minimum points* of trajectory. A typical 2-beat, 3-beat and 4-beat conducting pattern is shown in Figure 3.8. A beat event is defined to be a local minimum of the vertical component of target's trajectory. We evaluate the trajectory of the target and seek a local minimum of its vertical component, $y(t)$. Thus, we obtain the current timing of beat and tempo of the playing music via these two methods.

3.3.1 K-curvature Algorithm

The k-curvature algorithm was proposed in [40] and this method was used in chain-code corner detection field widely. In this thesis, we use this method to measure the angle between two consecutive motion vectors. The details of this algorithm are described as follow [41]:

A point p_i one the curve is defined in terms of the chain-code corner vectors v_i as

$$p_i = p_0 + \sum_{j=0}^{i-1} v_j$$

, where p_0 is the initial point on the trajectory and v_j is a vector from the set {(1, 0), (1, 1), (0, 1), (-1, 1), (-1, 0), (-1, -1), (0, -1), (1, -1) }. The k-curvature θ_i^k at p_i is given by

$$\theta_i^k = \cos^{-1}\left(\frac{V_i^1 \cdot V_i^2}{|V_i^1||V_i^2|}\right)$$

, where $V_i^1 = \sum_{j=1}^{k} w_j v_{i-j}$ and $V_i^2 = \sum_{j=1}^{k} w_j v_{i+j-1}$. We chose the weight w_j to be 1. For this case, the angle is illustrated in Figure 3.9.

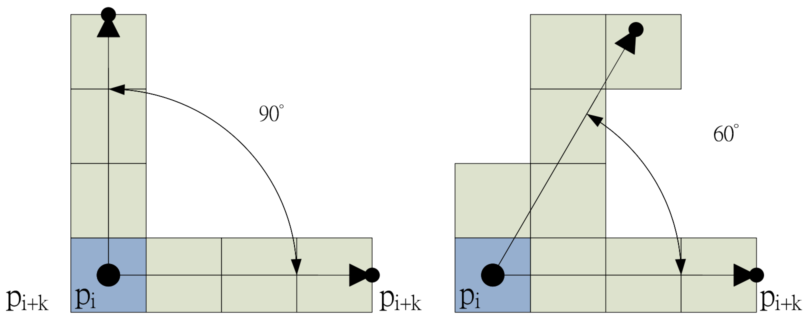


Figure 3.9 Angle measured by the k-curvature algorithm

More specifically, we define the object trajectory as a time sequential list P, $P = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots (x_t, y_t)\}$, where t is the frame number we finished tracking and θ is the angle we want to measure . We also defined two vectors:

$$\overrightarrow{V_a} = (x_{i-k} - x_i, y_{i-k} - y_i) \tag{3.11}$$

$$\overrightarrow{V_b} = (x_{i+k} - x_i, y_{i+k} - y_i) \tag{3.12}$$

$$\begin{cases} \overrightarrow{V_a} \cdot \overrightarrow{V_b} > 0 & 0^\circ < \theta < 90^\circ \\ \overrightarrow{V_a} \cdot \overrightarrow{V_b} = 0 & \theta = 90^\circ \\ \overrightarrow{V_a} \cdot \overrightarrow{V_b} < 0 & 180^\circ > \theta > 90^\circ \end{cases} \tag{3.13}$$

So we can compute the inner product of $\overrightarrow{V_a}$ and $\overrightarrow{V_b}$, $\overrightarrow{V_a} \cdot \overrightarrow{V_b}$, to decide the angle between these two vectors. Because we cannot get the trajectory information at future time, what we can do is to decide whether the point (x_{t-k}, y_{t-k}) is a direction change point or not at frame t. That means we need to allow the detecting delay k. To deuce the impact of detecting delay, we use 1 as the K factor in this thesis.

3.3.2 Local-Minimum Algorithm

Figure 3.8 also confirms the fact that a beat always directly corresponds with a downward-to-upward change in direction. While the y-axis data can be searched for a change in direction from downward-to-upward movement, the x-axis data we collected can be completely ignored.

We assume the situation that the y-axis scale increases from top to bottom. So, if the y-axis waveforms are rising, subtracting the previous y position value from the current y position value produces a negative value. If the waveform is unchanging, the result is zero. If the waveform is falling, this produces a positive value. Mathematically, this can be expressed as:

$$\mathrm{sign}(t) = \begin{cases} 1 & \text{if } y(t) - y(t-1) \geq 0 \\ -1 & \text{if } y(t) - y(t-1) < 0 \end{cases} \qquad (3.14)$$

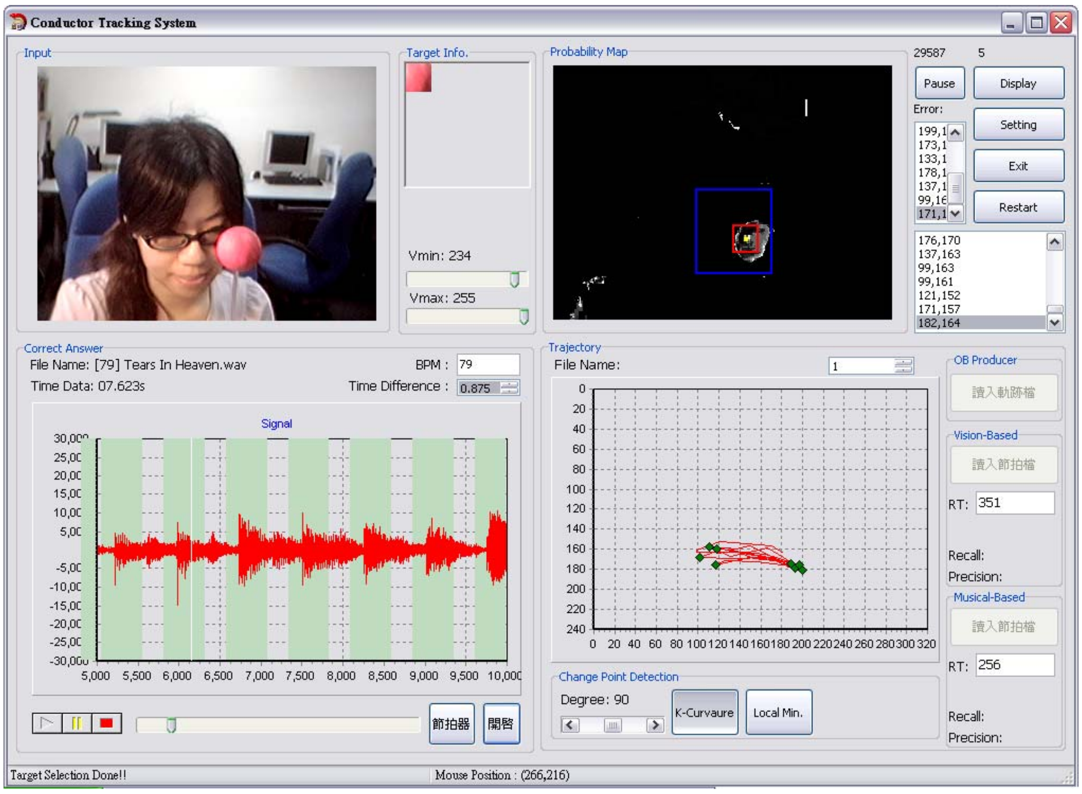
According to the reference scale used, a maximum is characterized by a change from a negative to a positive $\mathbf{sign(t)}$. Using similar idea, a minimum is characterized by a change from a positive to a negative $\mathbf{sign(t)}$. Since the beat directly corresponds with the minima, the system only has to search for a change in $\mathbf{sign(t)}$ from positive to negative. A minimum cannot be detected until the first rising point after the minima has been acquired, so a slight delay will always be present.

Chapter 4

Experimental Results and Discussions

4.1 Overviews

We designed a conductor gesture tracking system to help users training their conducting gesture. This system which realizes the algorithms described in the previous chapter will track the target and detect the beat event to calculate the relative correction rates.



Conductor Tracking System

File Name: [79] Tears In Heaven.wav

BPM : 79

Time Data: 07.623s

Time Difference : 0.875

Vmin: 234

Vmax: 255

29587 5

Error:

199,1
173,1
133,1
178,1
137,1
99,16
171,1

176,170
137,163
99,163
99,161
121,152
171,157
182,164

File Name: 1

RT: 351

Recall:

Precision:

RT: 256

Recall:

Precision:

Degree: 90

Target Selection Done!!

Mouse Position : (266,216)

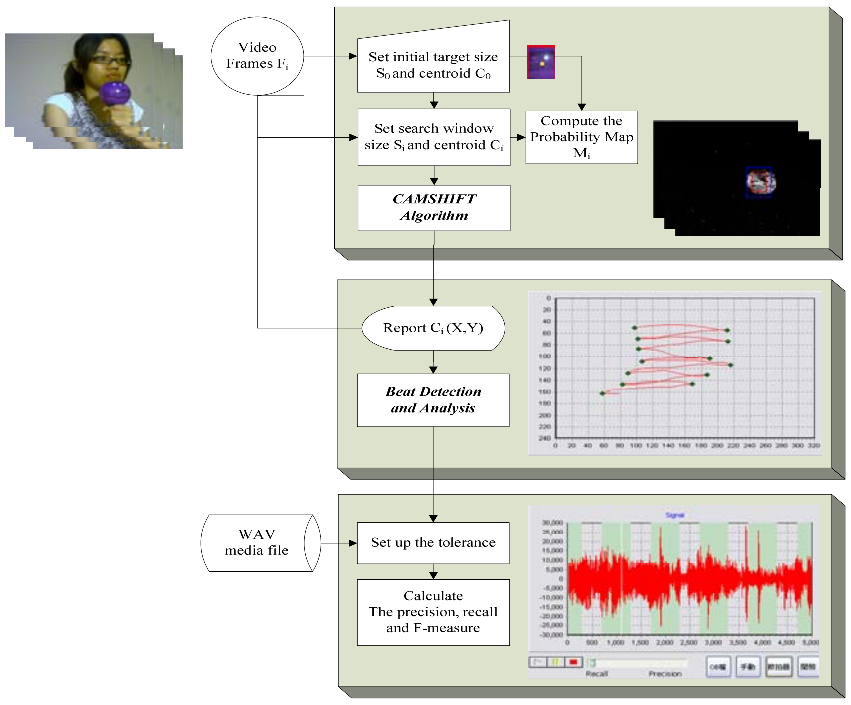
Figure 4.1 A snapshot of our conductor gesture tracking system

Figure 4.1 shows a snapshot of our conductor gesture tracking system. In our experiments, the video inputs are captured from the *Logitech QuickCam* webcam with the resolution of 320×240 pixels and the music files are sampled from compact discs, 10 constant-speed songs with different Beats per Minute (BPM). The BPMs of the music are from 60 to 124 beats per minute and the lengths of music are from 54s to 85s. All the software of our real-time conductor's gesture tracking system is run on a personal computer with 1G RAM. The software includes the CodeGear C++ Builder 2007 and Windows XP Home SP2. The throughput obtained is from 5-8 frames per second. In section 4.2 and 4.3, we will explain our experimental procedure and demonstrate the results and analysis afterward.

Combined with Figure 3.1 and Figure 4.1, we can figure out the relation between the processing procedure and user interface demonstration in Figure 4.2. Our system tracks a target from an image sequence and provides a robust tracking result. However, computer vision-based tracking is extremely difficult in an unconstrained environment, and many situations may affect the accuracy of tracking such as interfering with other objects with the same color in the background, variety of illuminant condition, too small tracking targets. In order to simplify tracking we assume there is no other object with the same color to interfere with the system.

After choosing the tracking target, our system starts producing probability map at

Right-UP column of our system UI. Upon the probability map image, we also can see the centroid and rectangle region of the target we tracked via CAMSHIFT Algorithm. At the same time, we can see the trajectory of the target and DCPs at Right-Bottom column of our system UI.



Video Frames F_i
Set initial target size S_0 and centroid C_0
Set search window size S_i and centroid C_i
Compute the Probability Map M_i
CAMSHIFT Algorithm
Report C_i (X,Y)
Beat Detection and Analysis
WAV media file
Set up the tolerance
Calculate The precision, recall and F-measure
Signal
Recall
Precision

Figure 4.2 Experimental Flowchart

To demonstrate the beat events in real-time, we drew the time information when beat events happened on the WAV form data of music file. We defined the correct interval is $\text{Correct\_Time}_i - \text{RT} < T < \text{Correct\_Time}_i + \text{RT}$, where T is the time we detected beat events, Correct_Time_i which came from the ground truth (via calculation at different BPM or men-make file) is the correct time of beat event i and the RT is based on *reaction time* we will mention later. If the time of beat event we detected is in the correct interval we defined, we called it a *correct beat* detection.

4.2 Experimental Results

We use *Precision*, *Recall* and *F-measure* rate to evaluate the correctness of the experiment output. In an Information Retrieval scenario, *Precision* is the fraction of the documents retrieved that are relevant to the user's information need and *Recall* is the fraction of the documents that are relevant to the query that are successfully retrieved [8]. With similar thought, we can define *Precision*, *Recall* more specifically.

$$\text{Precision Rate} = \frac{\{\text{\#of correct beat we detected}\}}{\{\text{\# of beat we detected}\}} \tag{4.1}$$

$$\text{Recall Rate} = \frac{\{\text{\#of correct beat we detected}\}}{\{\text{\# of correct beat from ground truth}\}} \tag{4.2}$$

That means, *Recall* parameter denotes the percentage of correct detection by the detection algorithm with respect to the overall beat events and the *Precision* is the percentage of correct detection with respect to the overall declared beat events.

Precision and Recall usually trade off against each other. As precision goes up, recall usually goes down (and vice versa). In order to take these two parameters into account, we use F-measure to combine precision and recall.

$$\mathrm{F}_\alpha = \frac{(1+\alpha)\times \text{precision} \times \text{recall}}{\big((\alpha \times \text{precision}) + \text{recall}\big)} \tag{4.3}$$

And F_1-measure combines recall and precision with an equal weight as follows:

$$\mathrm{F}_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{4.4}$$

We experiment our system using the same video and audio inputs with these two methods, K-Curvature and Local-Minimum algorithm, respectively. In K-Curvature algorithm, there exists a factor θ, the degree between two consecutive motion vectors, to control in K-Curvature algorithm. Therefore, we compute our experiment twice with two different factors, $\theta = 60^{\circ}$ and $\theta = 90^{\circ}$, for comparison purpose.

The experiment contains two parts: Music-based and Vision-based evaluations. The ground truth for Music-Based evaluation was determined by the constant BPM of music. If BPM =80, the time interval between two consecutive beats should be 0.75s and the continuous beat point series should be { 0.00, 0.75, 1.50, 2.25, …} until the music ends. The ground truth for the Vision-based evaluation was determined by men-make file whose time interval may not be so as stable as the Music-based one.

In Vision-based evaluations, we set the ground truth manual when we saw the beat events (the time point of direction change); in music-based evaluation, we conducted when we heard the beat events. Because people always need some time to react the visual or sound stimulus, it is overconstrained to limit the time point of beat event we detected must be exactly match with the ground truth. So we need to join the idea of *reaction time* (RT), which is usually defined as the time that an observer might be asked to press a button as soon as a light or sound appears. In the Master thesis of Lain [42], mean RT is 351±44 milliseconds to detect visual stimuli, whereas for sound it is 256±41 milliseconds.

Table 4.1 The overall experimental results with different methods

Methods	Music-Based Evaluation (RT=256)			Vision-Based Evaluation (RT=351)		
	Precision	Recall	F_1-measure	Precision	Recall	F_1-measure
K-Curvature ($\theta = 60^{\circ}$)	90.59	84.43	87.34	89.18	84.03	86.46
K-Curvature ($\theta = 90^{\circ}$)	85.43	92.97	89.02	84.69	92.67	88.48
Local Minimum	95.91	86.98	91.21	93.70	85.30	89.29

Table 4.1 shows the overall experimental results using different methods. Figure 4.3 and Figure 4.4 show the F_1-measures in two different evaluations. In Music-based evaluation, the lowest F_1-measure is 78.29% and highest F_1-measure is 96.31%. The lowest and highest F_1-measure is 77.26% and 94.81% in Vision-based evaluation.

We further analyze the vision-based evaluation more specifically: Compared with the F_1-measure rate in $\theta = 60^{\circ}$ and $\theta = 90^{\circ}$, we find that the F_1-measure at $\theta = 90^{\circ}$ is higher than the F_1-measure at $\theta = 60^{\circ}$. When $\theta = 60^{\circ}$, the range that *precision* rate decreased is higher than the range that the *recall* rate increased. Also, there is no significant impact between the BPM (the music speed) and F_1-measure and we can handle the music whose BPM is between 60 and 120 without any further support. Moreover, the *Reaction Time* (RT) can be adjusted dynamically based on the reacting ability of user. If we set RT more widely, it can help to raise the recall and precision rate efficiently, but it may decrease the accuracy of the system.

The following Table 4.2, Table 4.3 and Table 4.4 show the precision rates, recall rates and F_1-measure in every test sequence respectively.

Table 4.2 The precision, recall rates and F_1-measures using K-Curvature ($\theta = 90^{\circ}$)

Sequence No.	BPM	Music-Based Evaluation (RT=256)			Vision-Based Evaluation (RT=351)		
		Precision	Recall	F_1-measure	Precision	Recall	F_1-measure
1	60	89.91	97.01	93.33	86.37	93.94	90.00
2	65	81.69	87.81	84.64	79.32	84.86	82.00
3	68	89.14	96.30	92.58	77.38	85.00	81.05
4	79	75.21	82.40	78.64	77.60	86.14	81.65
5	88	90.29	92.89	91.57	87.12	89.51	88.30
6	93	87.17	94.84	90.84	87.17	95.99	91.37
7	100	84.07	93.66	88.60	88.00	97.20	92.37
8	103	86.66	98.39	92.16	86.29	98.16	91.84
9	120	83.28	88.79	85.95	89.81	97.02	93.28
10	124	86.82	97.64	91.92	87.80	98.83	92.99

Table 4.3 The precision, recall rates and F_1-measures using K-Curvature ($\theta = 60^{\circ}$)

Sequence No.	BPM	Music-Based Evaluation (RT=256)			Vision-Based Evaluation (RT=351)		
		Precision	Recall	F_1-measure	Precision	Recall	F_1-measure
1	60	94.73	87.06	90.74	89.54	83.33	86.33
2	65	93.32	91.76	92.53	82.44	81.31	81.87
3	68	90.48	88.89	89.68	78.45	78.33	78.39
4	79	78.31	78.28	78.29	82.74	83.52	83.13
5	88	94.12	84.27	88.92	91.02	81.27	85.87
6	93	84.17	79.76	81.91	89.36	85.14	87.20
7	100	86.25	85.45	85.85	91.68	90.19	90.93
8	103	97.80	87.59	92.41	90.62	87.13	88.84
9	120	94.24	82.18	87.80	98.65	86.86	92.38
10	124	92.48	79.06	85.24	97.27	83.22	89.70

Table 4.4 The precision, recall rates and F_1-measures using **Local-Minimum**

Sequence No.	**BPM**	**Music-Based Evaluation (RT=256)**			**Vision-Based Evaluation (RT=351)**		
		Precision	**Recall**	**F_1-measure**	**Precision**	**Recall**	**F_1-measure**
1	60	99.40	86.57	92.54	92.17	81.31	86.40
2	65	99.62	89.61	94.35	87.03	78.44	82.51
3	68	97.79	90.94	94.24	79.67	75.00	77.26
4	79	85.51	79.40	82.34	88.70	82.40	85.43
5	88	100.00	92.89	96.31	94.69	88.02	91.23
6	93	91.12	82.54	86.62	98.26	89.56	93.71
7	100	92.86	86.86	89.76	98.25	91.60	94.81
8	103	99.24	89.89	94.33	98.50	89.20	93.62
9	120	95.88	84.07	89.59	100.00	88.64	93.98
10	124	97.68	87.02	92.05	99.68	88.84	93.95

Figure 4.3 and Figure 4.4 shows the Vision-based evaluation results in chart forms. We will discuss the reasons of failed detection in vision-based evaluation in the next section.

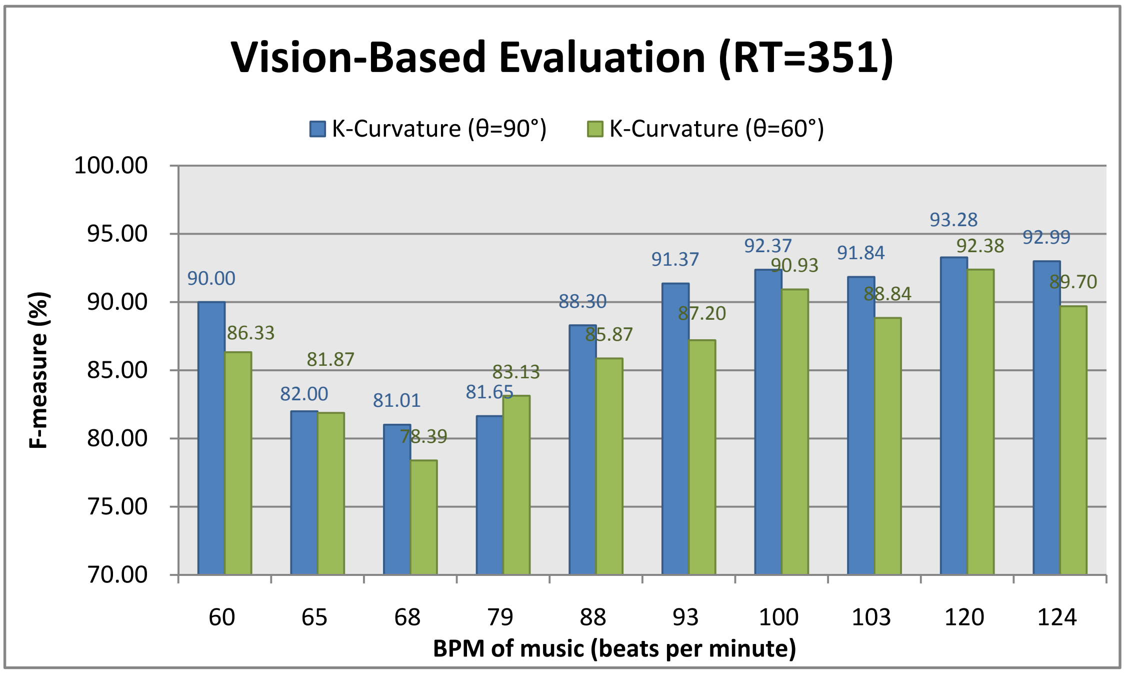


Figure 4.3 F-measure results of Vision-Based Evaluation using K-curvature algorithm

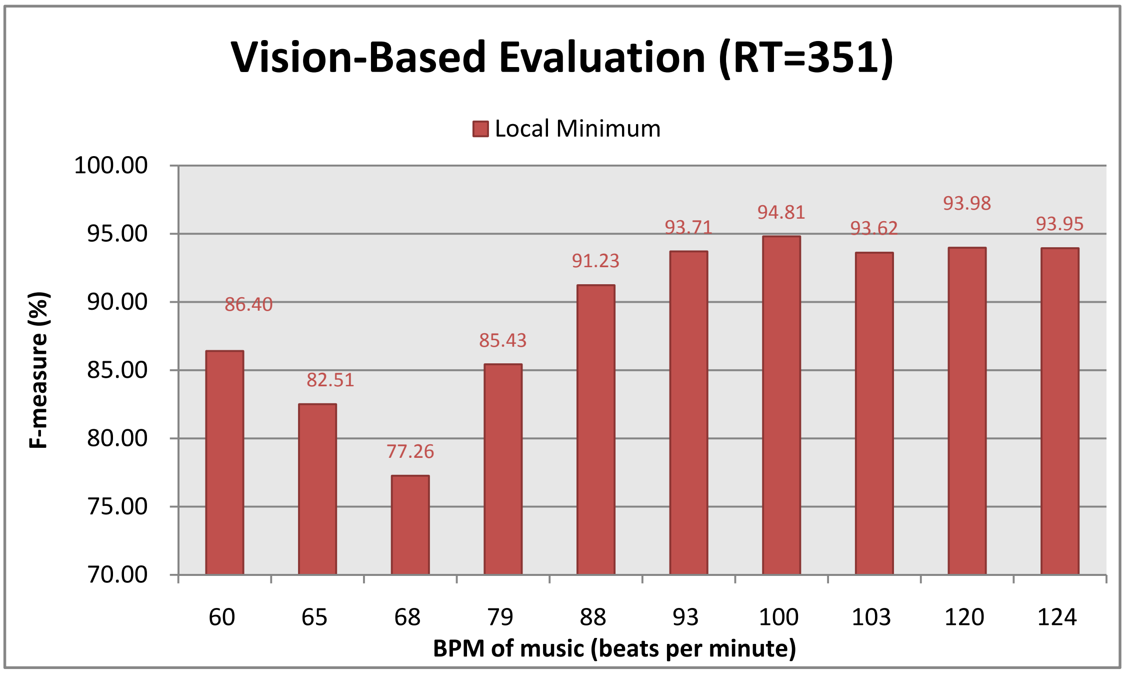


Figure 4.4 F-measure results of Vision-Based Evaluation using local-minimum algorithm

4.3 Analysis of the Experimental Results

In this section, we classified our beat detection errors into two categories based on Vision-based evaluation. They are known as false positives and false negatives.

(1) **False Positive Error:** A false positive is the error which normally means that a test claims something to be positive, when that is not the case. In our experiment, our beat detection with a positive result (indicating that here is a beat event) has produced a false positive in the case where there is no beat event. We separate the errors into two situations: detected the beat events when there was no beat event and detected the beat events when the correct beat event was already detected.

(2) **False Negative Error:** A false negative is the error of failing to observe when in truth there is one. In our case, our beat detection without a positive result has

produced a false negative where there is a beat event at that moment.

Table 4.5 The False Positive and False Negative Error Rates

Type of Error / **Method**	**False Positive**		**False Negative**
	Non-beat but detected	**duplicate detected**	**beats but not detected**
K-Curvature $(\boldsymbol{\theta} = \mathbf{60}^{\circ})$	4.72%	6.10%	15.97%
K-Curvature $(\boldsymbol{\theta} = \mathbf{90}^{\circ})$	5.37%	9.93%	7.34%
Local Minimum	4.79%	1.51%	14.70%

When the false positive errors occurred, we need to separate the errors into two kinds of situations: detected the beat events when there was no beat event and detected the beat events when the correct beat event was already detected. We detected non-beat events falsely due to the trajectory of user including some non-beat change of direction. The duplicate detection always occurred due to the tracking lost when the target left the scene or the vibration of the target. This kind of situation might be solved by the design of dynamic beats filter to eliminate those successive wrong beats while the time interval between two consecutive beats is too close.

The false negative errors always occurred if $\theta = 60^{\circ}$(in K-Curvature algorithm) or in Local-Minimum algorithm. It is due to the frame lost because of the performance of the program. If we can increase the maximum frame rates we processed, this kind of mistake might be solved.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this thesis, we have presented an efficient real-time target tracking system for conductor gesture tracking based on CAMSHIFT algorithm. Also, in order to extract beat events of trajectory of the trajectory of the target, we used K-Curvature algorithm and Local-Minimum algorithm to interpret different kinds of conductor gesture.

The major part of our framework is based on CAMSHIFT algorithm which is a very simple, computationally efficient colored object tracker. It is usable as a visual interface and it can be incorporated into our system that provides the conductor gesture tracking. The CAMSHIFT algorithm handles computer-vision problems as follows:

- **Irregular object motion:** CAMSHIFT scales its search window to object size thus naturally handling perspective–included motion irregularities, so it is suitable for our purpose to detect the change of direction.
- **Distracters:** CAMSHIFT ignores objects outside its search window so other objects do not affect CAMSHIFT's tracking.

- **Lighting Variation:** Using only hue from the HSV color space and ignoring pixels with high/low brightness gives CAMSHIFT wide lightness tolerance.

In other words, we design an HCI system for interpreting a conductor’s gestures and translating theses gestures into musical beats that can be explained as the major part of the music. This system does not require the use of active sensing, special baton, or other constraints on the physical motion of the conductor. Thus, this framework can also be used for human analysis and many other applications such as interactive virtual worlds that allow user to interact with computer systems.

5.2 Future Work

Since CAMSHIFT relies on color distribution alone, errors in color (color lighting, dim illumination, too much illumination…etc) will cause errors in tracking procedure. More sophisticated trackers use multiple modes such as feature tracking and motion analysis to compensate for this, but more complexity would undermine the original design criterion for CAMSHIFT. Other possible improvements include:

- **Improve tempo following:** the current system cannot react to some complex and subtle movement of professional conductor, not only for the direction change. We can replace our *beat detection and analysis module* with some more sophisticated gesture recognition algorithms, so that we can adjust our module according to the different level of users conducting skill.

- **Include time stretching algorithm:** time stretching algorithm is the process of changing the speed or duration of an audio signal without affecting its pitch. While our system can adjust the music playback speed according to the beat event, it can help users to understand the conducting speed he/she performed.
- **New application area:** we can use our framework to several different areas in interface with vision and some other multimedia. We not also can estimate the accuracy of beat events for conducting gesture, but also the movement of the dancer. Based on the previous future work, we can design another system for a dancer whose routine is no longer constrained by the tempo of a recording and the music would spontaneously react to his/her movements.

In conclusion, since the system we proposed is a framework which combines the video and audio processing areas, the applications of this technology can help us to examine unexplored area in interfaces with music and other multimedia. We can build an “interactive karaoke” system where a user could sing a song along to a recording, but have the recording adjusting to the user’s tempo. Some other applications can be implemented following these rules, including conductor training, live performance and music synthesis control, and so forth. We hope that the flexible and interchangeable modules would make the further researches easier in the future.

References

[1] T. T. Hewett, et al., "ACM SIGCHI Curricula for Human-Computer Interaction", *ACM Press*, New York, NY, 1992, ACM Order Number: 608920.

[2] B. A. Myers, "A Brief History of Human-Computer Interaction Technology", *Interactions,* vol. 5, pp. 44-54, 1998.

[3] M.T. Driscoll, "A Machine Vision System for Capture and Interpretation of Orchestra Conductor’s Gestures", *M. S. Degree Thesis*, May, 1999.

[4] E. Lee, I. I. Grüll, H. Kiel and J. Borchers, "Conga: A Framework for Adaptive Conducting Gesture Analysis", *NIME '06: Proceedings of the 2006 Conference on New Interfaces for Musical Expression,* pp. 260-265, 2006.

[5] D. Murphy, “Tracking a Conductor's Baton” , *Søren I. Olsen, Editor, Proceedings of the 12th Danish Conference on Pattern Recognition and Image Analysis,* volume 2003/05 of DIKU technical report series, pp. 59-66, Copenhagen, Denmark, August 2003.

[6] R. Behringer, "Conducting Digitally Stored Music by Computer Vision Tracking", *AXMEDIS '05: Proceedings of the First International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution,* pp. 271, 2005.

[7] The Church of Jesus Christ of Latter Day Saints. ***Conducting Course***.

[8] Wikipedia, The free Encyclopedia, http://en.wikipedia.org

[9] M. Lambers, "How Far is Technology from Completely Understanding a Conductor?”, *4th Twente Student Conference on IT, Enschede,* January 30, 2006.

[10] Paul Kolesnik, "A Conducting Gesture Recognition, Analysis and Performance System", *M. S. Degree Thesis*, McGill University, June, 2004.

[11] R. Boulanger and M. Mathews, “The 1997 Mathews Radio-baton and Improvisation Modes”, *Proceedings of the 1997 International Computer Music Conference*, pp.395-398, Thessaloniki, Greece, 1997.

[12] Buchla Lightning II. <http://www.buchla.com>

[13] B. Brecht and G. Garnett, “Conductor Follower” , *Proceedings of the 1995 International Computer Music Conference*, pp. 185-186, Banff, Canada, 1995, Available: http://cnmat.berkeley.edu/publication/conductor_follower.

[14] J. Borchers, W. Samminger and M. Mühlhäuser, "Personal Orchestra: Conducting Audio/Video Music Recordings", *Proceedings of the Second International Conference on WEB Delivering of Music (WEDELMUISC’02),* 2002.

[15] Carmine Cascaito and Marcelo M. Wanderley, “Lessons from Long Term Gestural Controller Users”, in *Proceedings of the 4th International Conference on Enactive Interfaces (ENACTIVE'07),* pp. 333-336, Grenoble, France, 2007.

[16] F. Tobey and Ichiro Fujinaga, "Extraction of Conducting Gestures in 3D Space", *Proceedings of the 1996 International Computer Music Conference*, pp. 305-307, San Francisco, 1996.

[17] T. Marrin and J. Paradiso, “The Digital Baton: a Versatile Performance Instrument”, *Proceedings of the 1997 International Computer Music Conference*, pp.313-316, Thessaloniki, Greece, 1997.

[18] T. Marrin and R. Picard, “The Conductor's Jacket: a Device for Recording Expressive Musical Gestures”, *Proceedings of the 1998 International Computer Music Conference*, pp.215-219, Ann Arbor, MI, 1998.

[19] T. Marrin, "Inside the Conductor's Jacket: Analysis, Interpretation and Musical Synthesis of Expressive Gesture", *Ph.D. Dissertation*, MIT Media Lab, February, 2000.

[20] T. Ilmonen, “Tracking Conductor of an Orchestra Using Artificial Neural Networks”, *M. S. Degree Thesis,* Helsinki University of Technology, Espoo, Finland, 1999.

[21] T. Ilmonen and T. Takala, "Conductor Following with Artificial Neural Networks", *Proceedings of the 1999 International Computer Music Conference,* pp. 367-370, Beijing, China, October, 1999.

[22] H. Morita, "A Computer Music System that Follows a Human Conductor," *Computer*, vol. 24, pp. 44-53, 1991.

[23] Light Baton. < http://web.tiscali.it/pcarosi/Lbs.htm>

[24] J. Segen, A. Majumder, and J. Gluckman, "Virtual Dance and Music Conducted by a Human Conductor", *Eurographics*, vol. 19(3), EACG, 1999.

[25] J. Segen, S. Kumar and J. Gluckman, "Visual Interface for Conducting Virtual Orchestra", *Proceedings of the 15th International Conference on Pattern Recognition (ICPR'00)*, vol.1, pp. 276-279, 2000.

[26] E. Lee, T. Marrin and J. Borchers, "You're the Conductor: A Realistic Interactive Conducting System for Children", *NIME '04: Proceedings of the 2004 Conference on New Interfaces for Musical Expression*, pp. 68-73, Hamamatsu, Japan, June 3-5, 2004.

[27] E. Lee, M. Wolf and J. Borchers, "Improving Orchestral Conducting Systems in Public Spaces: Examining the Temporal Characteristics and Conceptual Models of Conducting Gestures", *Proceedings of the CHI 2005 Conference on Human Factors in Computing Systems*, pp. 731-740, Portland, Oregon, April 2-7, 2005.

[28] E. Lee and J. Borchers, "The Role of Time in Engineering Computer Music Systems", *NIME '05: Proceedings of the 2005 Conference on New Interfaces for Musical Expression*, pp. 204-207, Vancouver, Canada, May 26-28, 2005.

[29] D. Murphy, T. H. Andersen, and K. Jensen, "Conducting Audio Files via Computer Vision", *Gesture-Based Communication in Human-Computer Interaction: Selected Revised Papers from the 5th International Gesture Workshop*, volume 2915 of LNAI, pp. 529-540, Genoa, Italy, April, 2003.

[30] D. Murphy, "Live Interpretation of Conductors' Beat Patterns" , *Proceedings of the 13th Danish Conference on Pattern Recognition and Image Analysis*, Copenhagen, Denmark, pp. 111-120, 2004.

[31] T. Sim, D. Ng, and R. Janakiraman, "VIM: Vision for Interactive Music", *Proceedings of IEEE Workshop on Applications of Computer Vision (WACV '07)*, pp.32-32, February, 2007.

[32] K. C. Ng, "Music via Motion: Transdomain Mapping of Motion and Sound for Interactive Performances", *Proceedings of the IEEE*, vol.92, no.4, pp. 645-655, April, 2004.

[33] W. Hu, T. Tan, L. Wang and S. Maybank, "A survey on visual surveillance of object motion and behaviors", *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews,* vol. 34, pp. 334-352, 2004.

[34] Wen-Han Yao, "Mean-Shift Object Tracking Based On A Multi-blob Model", *M. S. Degree Thesis*, National Tawan Chiao Tung University, June, 2006.

[35] G. Bradski, "Computer vision face tracking for use in perceptual user interface", *Intel Technology Journal*, vol. 2nd Quarter, 1998.

[36] G. John Allen, Y. D. Richard Xu and S. Jin Jesse, "Object Tracking Using CamShift Algorithm and Multiple Quantized Feature Spaces", *Inc. Australian Computer Society*, vol.36, 2004.

[37] Dorin Comaniciu and Peter Meer, "Mean Shift: A robust approach toward feature space analysis", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 24(5):603-619, May, 2002.

[38] Xia Liu, "Research of the Improved Camshift Tracking Algorithm", *International Conference on Mechatronics and Automation*, ICMA 2007, pp. 968-972, 2007.

[39] Hongmo Je, Jiman Kim and Daijin Kim, "Vision-Based Hand Gesture Recognition for Understanding Musical Time Pattern and Tempo", *The 33rd Annual Conference of the IEEE Industrial Electronics Society (IECON)*, pp. 2371-2376, , Taipei, Taiwan, November 5-8, 2007.

[40] W.S. Rutkowski, A. Rosenfeld, "A comparison of corner-detection techniques for chain-coded curves", TR-623. Computer Science Center, University of Maryland, 1978.

[41] T. Peli, "Corner extraction from radar images", *1988 International Conference on Acoustics, Speech, and Signal Processing*. ICASSP-88, pp. 1216-1219 vol.2, 1988.

[42] Huang-Yu Lian, "The Effects of Human Factors on Reaction Speed to Visual and Auditory Signals ", *M. S. Degree Thesis,* National Kaohsiung First University of Science and Technology, Kaohsiung, Taiwan, 2000.