

UiO • **Department of Informatics**
University of Oslo



Abstract

Today, there are several toolboxes which can work on audio, motion, or other sensor data. These toolboxes are very useful to provide characteristic analysis of audio and motion. Unfortunately, the analysis is done separately by different toolboxes. This results in inconvenience when we want to work on these data simultaneously. So developing a toolbox which integrates the existing toolboxes is necessary. The main goal of the project is to integrate these toolboxes in Matlab and provide video analysis combined with audio and motion capture data. This would be important for our interdisciplinary research on music and motions through fourMs as well as for external work on e.g. analyzing video recording for early child diagnosis of cerebral palsy. This project presents the development of a toolbox for Matlab entitled “Musical Gestures (MG) Toolbox”. This toolbox is aimed for solving pressing needs for the video analysis of music-related body motion since video source recorded by regular video camera is a very good option for studying motion. The term music-related body motion refers to all sorts of body motion found in music performance and perception. It has received a growing interest in music research and behavioral science over the last decades. Particularly, with the rapid development of modern technology, various motion capture systems make it possible to further study music-related body motion.

Matlab has been chosen as the platform since it is readily available, and there are already several pre-existing toolboxes to build on. This includes the “Motion Capture (MoCap) Toolbox” [1] developed for the analysis and visualization of Motion Capture data, which is aimed specifically for the analysis of music-related body motion. The “Music Information Retrieval (MIR) Toolbox” [2] is another relevant toolbox, which is developed for the extraction of musical features from audio data and the investigation of relationships between sound and music features.

While the two above mentioned toolboxes are useful for studying motion capture data and audio, respectively, they are very differently designed, and it is not possible to make combined analysis of audio and motion capture data. Furthermore, there is no integration with video analysis. The MG Max toolbox [3] has been developed for music-related video analysis in the graphical programming environment Max/Msp/Jitter, with a number of novel visualization techniques (motiongrams, motion history images, etc.). These techniques are commonly used in music research, but are not currently available in Matlab. The main contributions of this project consist of two following things. One is to integrate the MoCap toolbox and MIR toolbox, and provide simple preprocessing on different input data. Another is to provide several video analysis techniques to study music-related body motion in the toolbox. These video analysis techniques include motiongram, optical flow, eulerian video magnification. With these techniques, the developed MG toolbox for Matlab could provide reliable and quantitative analysis of music-related body motion based on video.

Contents

1 Introduction	10
1.1 Objective of thesis	11
1.2 Description of the related dataset	12
1.3 Thesis outline	12
Part 1 Theory and techniques	14
2 Music and music-related body motion	16
2.1 Music cognition history	16
2.2 Motion or movement	16
2.3 Music-related body motion	17
2.4 Summary	18
3 Techniques	20
3.1 Introduction	20
3.2 Motiongram	20
3.2.1 Motion image	21
3.2.2 Theory of motiongram	21
3.3 Optical flow	22
3.3.1 Theory of optical flow	22
3.3.2 Optical flowgram	23
3.3.3 Advantage of optical flow	23
3.3.4 Comparison with motiongram	23
3.4 Eulerian video magnification	25
3.5 Motion Capture system	27
3.6 Summary	28
Part 2 Design and implementation	30
4 Design	32
4.1 Framework	32
4.2 Memory management	32
4.3 Data structure	33
4.4 Data flow	33
4.5 Summary	34
5 Implementation	37
5.1 Import	37
5.2 Preprocessing	37
5.2.1 mgvideocrop	37
5.2.2 mgvideoadjust	38
5.2.3 mgvideorotate	38

5.2.4 mgvideomagnify	39
5.3 Feature extraction	39
5.3.1 Quantity of motion	40
5.3.2 Centroid of motion	40
5.3.3 Area of motion	41
5.3.4 Statistics descriptors	41
5.3.5 Some related audio features	42
5.4 Visualization tool	42
5.4.1 mgvideoplot	43
5.4.2 mgwaveplot	44
5.5 Summary	44
 Part 3 Using the MG Toolbox	 46
6 Using the MG Toolbox	48
6.1 Case 1	48
6.1.1 Importing and preprocessing	48
6.1.2 Analysis	49
6.2 Case 2	52
6.2.1 Importing and preprocessing	52
6.2.2 Analysis	53
6.3 Discussion	60
7 Summary and future work	62
7.1 Conclusion	62
7.2 Future work	63
 References	 64
 Appendix	 66
Function list	68
Function reference	69

Abbreviations

AoM Area of Motion

CoM Centroid of Motion

CP Celebral Palsy

IrMoCap Optical infrared marker based motion capture

MI Motion Image

Max Max/MSP/Jitter programming environment

MHI Motion History Image

MGT Musical Gestures Toolbox

MIRT Music Information Retrieval Toolbox

LMA Laban Movement Analysis

QoM Quantity of Motion

Acknowledgement

This thesis could not have been written without the inspiration from Robin-group and Department of Musicology at the University of Oslo. Especially I would like to thank my supervisor Jim Tørresen and co-supervisor Alexander Refsum Jensenius. Jim was greatly helpful to read this whole thesis and provided valuable feedback. He organized monthly meeting and kept following the progress of the thesis. Alexander never closed his door and he was always open for my questions. He provided lots of suggestions and comments during the development of the musical gestures toolbox. Without his great help, I could not have completed the development. My family deserves my gratitude for always supporting me. My mother gives me a lot of inspiration life and she is my mentoring teacher. Lastly my best friend, Xiao encouraged me a lot and willingly shared both the great frustrations and happiness. Thank you all.

Chapter 1 Introduction

Music-related body motion has gained increasingly attention in musical research. The “Motion Capture (MoCap) Toolbox” [1] and “Music Information Retrieval (MIR) Toolbox” [2] have been developed for different purposes. The MoCap toolbox is a Matlab toolbox providing the analysis and visualization of the *motion capture* data recorded by motion capture system, which has been developed for the analysis and visualization of music-related motion. The MoCap toolbox works with recordings made with the infrared marker-based optical motion capture system [4]. The MIR toolbox is a Matlab toolbox as well and contains a set of functions for the analysis of *audio* and *music*, which mainly provides functions computing various features related to audio and music. However, it is designed with different data structure from the MoCap toolbox. Lastly, the existing MG Max toolbox [3] is largely different from other two toolboxes. It has been developed for music-related *video* analysis, but in the graphical programming environment Max/Msp/Jitter. It is not currently available in Matlab. So developing a new toolbox for Matlab which integrates the MoCap toolbox and MIR toolbox combined with video analysis is necessary. In this project, we present a new toolbox entitled “Musical Gestures (MG) Toolbox” for Matlab, which integrates the MoCap toolbox and MIR toolbox, and provides video analysis of music-related body motion with several techniques. Figure 1 gives a high-level overview of the developed MG Matlab toolbox. The details of the development will be discussed in *chapter 4* and *chapter 5*.

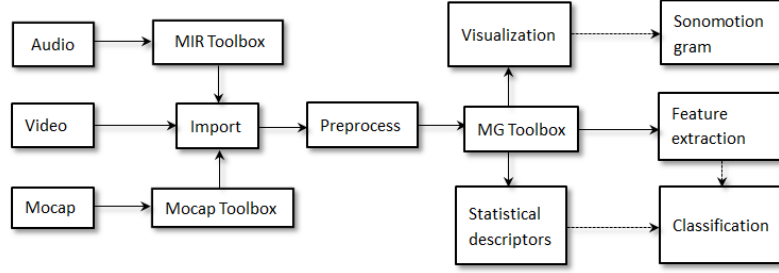


Figure 1: Framework of MG toolbox, dotted line means possible extension in the future work.

Even though Matlab has offered a large range of audio and video functionalities, they are not for music-related body motion analysis, and they only work on separate input data. The MG toolbox for Matlab is developed specially for video analysis of music-related body motion and combined with audio and motion capture data. Currently, the developed toolbox consists of four basic parts: importing, preprocessing, analysis and visualization, with each basic part containing a set of functions. The MG toolbox requires the Matlab pre-built toolboxes, for example, the Signal Processing Toolbox and Image Processing Toolbox provided by MathWorks. In addition, it needs a multi-scale image processing toolbox: *matlabPyrttools* [5]. Users themselves can make further extension of the toolbox and make use of additional functionalities provided by Matlab.

Importing. The MG toolbox provides several functions for reading different raw data (video, audio/sound, mocap data) into Matlab workspace. These functions can operate on any one of them or any combination of them, and import specified *temporal segment* of the raw data. These functions are extremely useful when we deal with large video files. For one thing, importing the whole data would need lots of memory, and it would be time-consuming. For another thing, the signal of interest is often hidden in the temporal domain. Extracting the temporal segment of the input data is important during importing.

Preprocessing. After importing, preprocessing of the data is important when the raw data is of poor quality. For instance, the video has low contrast or improper vision angle. The MG toolbox contains various functions for video preprocessing. Such preprocessing operations include adjusting the contrast of a video, rotating the view angle of a video, cropping the region of interest of a video, down sampling a video with high resolution, magnifying micro motion of a video, etc.

Analysis. The MG toolbox provides several techniques for video analysis, such as motiongram, optical flow, eulerian video magnification. The features computed by these techniques can reveal the relationship of body motion and music. Motiongram and optical flow make it possible to see spatial temporal information of the movements of an object. From these features, we can find the correspondence with music. Eulerian video magnification reveals the subtle movements in a video which are difficult or impossible to see with naked human eyes.

Visualization. The MG toolbox has two visualization tools. One is to show the motiongram, optical flow field and bounding box over time. It mainly shows two types of motiongrams: the vertical motiongram and horizontal motiongram. Another is to show waveform and spectrum of music or sound, together with quantity of motion.

1.1 Objectives of thesis

The main topic of this thesis is music-related body motion based on video analysis. The goal is to develop a new toolbox entitled “Musical Gestures (MG) Toolbox” for Matlab which combines the video analysis with audio and MoCap data. So there are actually two threads in this thesis. From exploration view, there are three questions needed to be answered during the writing:

- What is music-related body motion?
- Which terminology should be used to describe music-related body motion?
- Which features from video can be used to measure body motion?

From exploitation view, there are two questions:

- Which techniques and tools are used to study music-related body motion?
- How to develop the new toolbox?

As the questions show above, the aims and objectives are formed in this thesis. Similarly, from exploration view, they are to:

- clarify the definitions and terminology which are used to describe music-related body motion.
- introduce the features for studying body motion and measuring body motion.

From exploitation view, they are to:

- introduce techniques and tools applied to study music-related body movement.
- develop the toolbox to study music-related body movement based on video analysis.

Furthermore, in terms of developing the toolbox, the new MG toolbox for Matlab aims to be a swiss-army knife for music researchers. It builds on the above mentioned toolboxes, and also adds new functionalities as well as a consistent namespace and documentation. It consists of the following main detailed structures:

- tools for importing and exporting data.
- transformation and data processing tools (trimming, cropping, rotating, etc.).
- visualisation tools (motiongram, spectrogram, mocapgram, etc.).
- middle and higher level feature extraction tools (quantity of motion, sound level, etc.).

1.2 Description of the related data

Together with the submission of the thesis, the attachments contain the Matlab source codes of the developed MG toolbox and related MG toolbox manual as well as the four related datasets used in this thesis:

- the video data: baby.mp4, which is used in *section 3.4*.
- the video data: dancer.mov, which is used in *section 3.2* and *section 3.3*.
- the dancing dataset containing dance.mp4, dance.c3d and dance.wav, which is mainly used in case 1 in *section 6.1*.
- the pianist dataset containing pianist.mp4, pianist.wav and pianist.tsv, which is mainly used in *chapter 5* and *section 6.2*.

1.3 Thesis outline

Part 1 Theory and techniques

Chapter 2 introduces the relevant theory and concepts of music-related body motion. The two concepts movement and motion are clarified in this chapter. The common analysis methods will be introduced as well. This chapter lays the theoretical foundation of the thesis.

Chapter 3 briefly introduces techniques for video analysis, motion image, motiongram, optical flow, eulerian video magnification, and motion capture system. The comparison between motiongram and optical flow is illustrated by an example.

Part 2 Design and implementation

Chapter 4 starts with an overview of the new toolbox, MG toolbox for Matlab and introduces the framework and the memory management of the MG toolbox. The data structure and data flow of the MG toolbox are described as well. This chapter mainly relates to design details.

Chapter 5 mainly discusses the implementation details. Firstly, it introduces the various importing operations handling the different data and preprocessing operations based on video file. Then the feature extraction from the video is presented. The calculations of the features are shown mathematically. Lastly, the two visualization tools are introduced.

Part 3 Using the MG Toolbox

Chapter 6 explains how to apply MG toolbox to analyze music-related body motion. Two cases which correspond to two types of music-related body motion: sound-producing movements and sound-accompanying movements are investigated by the toolbox.

Chapter 7 makes the summary and conclusion, as well as proposes promising future work.

References lists the references in this thesis.

Appendix provides the function table and function references of the MG toolbox involved in this thesis.

PART 1
THEORY AND TECHNIQUES

Chapter 2 Music and music-related body motion

***Chapter abstract:** This chapter will introduce some basic concepts of music and music-related body motion. Since music and music-related body motion are highly related to the title of the thesis, this specific chapter will introduce them respectively. The first part of this chapter will describe the music cognition history. In the second part, the terminologies and difference of movement and motion are discussed. With knowledge of the first two parts, music-related body motion is presented in the third part of this chapter.*

2.1 Music cognition history

In the early western classic music tradition, the most typical way of experiencing music is to be seated, in silence, not moving. Even though the conductor may exaggeratedly gesticulate and move while performing, all audiences sit still and quiet. This is considered as the only possible way to experience with music in the 19th century. The certain class social aspect causes this special social phenomenon. Richard Sennet [6] pointed out that restraint of emotion in the theater became a way for middle-class audiences to mark the line between themselves and the working class. A respectable audience could control its feelings through silence by the 1850s. People have limited knowledge about how body motion involved in music may enhance the experience of tradition music. This situation has gradually changed in the 20th century with diverse music genres development. These music genres include the jazz, swing, rock and disco, etc. It is necessary to note that the most radical change happened in the 20th century, which is that music could be experienced without any performer's present. Starting with swing jazz in 1920s and 1930s aiming for making people move, the traditional concepts of music experiences have gradually changed. Because of the various types of the jazz, the tempo could be fast or slow. Many improvised parts appeared in the music. Gradually, it becomes conventions for a jazz concert that the seated audiences should applaud after solos and nod their head or tap their feet to the beat. In the 1950s, the rock music evolved from the African American rhythm's blues became very popular. This type of the new music has a close connection with movement. From then on, the music seemed to be danceable and encourage the audiences out of their chairs to participate with dancing, moving, singing along. Funk music, disco, hip hop coming out after rock music emphasize further the link between music and body motion.

In the contemporary popular music, bodily engagement movement has played a key role on enhancing the music experience. Body motion has received an increasing interest in behavioral research over the last decades. Some researchers like Alf Gabrielsson, Eric Clarke and Jane Davidsson have been studied body motion in music from the music performance view.

2.2 Motion or movement

Motion and movement are two concepts which are confusing and hard to tell the difference. It seems that movement may be more acceptable and used in everyday life by ordinary people when talking about displacement of the object happening and position of the object changes in space over time. The literal meaning of the movement itself contains a series of actions. However, movement is used as a simple concept or phenomenon as well. For instance, when it comes to the "feminist movement" of the 20th century, here the term movement is an implicit concept. Motion is often used as an academic and scientific term, describing the physics of movement. When it comes to description of the technology, motion is often used. For instance, motion detection, motion capture, etc. From figure 2¹, there is an interesting trend which movement seems to be more popular and acceptable by people.

¹Type movement, motion in this website: <https://books.google.com/ngrams/> to see the result.

So to avoid misunderstanding, motion is used more often to describe the academic terms in this thesis.

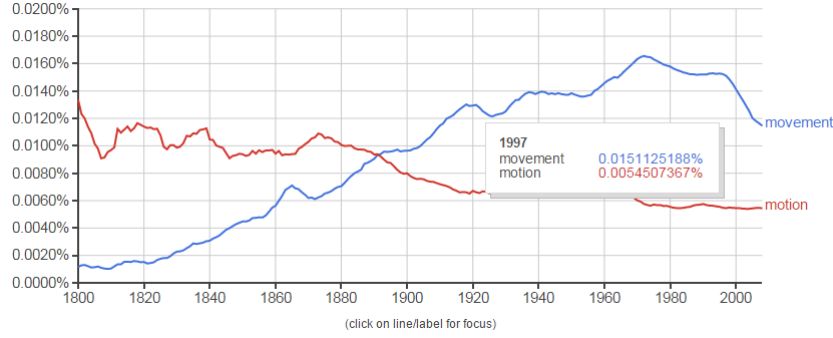


Figure 2: Usage frequency of movement and motion, which appear in Google books.

2.3 Music-related body motion

Music related-body motion as a particular phenomenon in human beings opens a wide field to us. It has not gained attention because such a phenomenon happens in our daily life, and people have for long got used to that. Music-related body motion is not only a cognitive process of music, but also objective bodily reflection to music. In terms of the first aspect, human beings cognitive process involves multi-channels sensing, namely multimodality [7]. From the second aspect, reflection to music relates to specific object. This is somewhat called affordance [7]. As the main goal of this thesis is to develop MG toolbox, these two concepts are not much discussed in this thesis.

An embodied approach is suggested that the mental processing is inseparable from the body [8]. With such approach, the embodied music cognition stresses that the whole body plays an important role in the perception of music. This opposes the traditions which uphold a separation of body and mind. The music can be rhythmic, repetitive and danceable. Music has always been considered as a movement based phenomenon. The connection between music and body motion is obvious in the light of how music is made. Body motion not only produces sounds, but also responds to specific features in the music. Body motion is considered as the essential part of musical behavior, but it receives little attention in traditional music research. This is probably because the meaning of the body motion itself is harder to be understood directly than that of hand-writing and speech.

Music-related body motion refers to any type of body motion in a musical context. Then, how to study music-related body motion? Generally, the two kinds of methods being used for carrying out research are qualitative methods and quantitative methods. The former is often exploratory, aiming to reveal and explain phenomena. Such methods include Labanotation [9] and Laban Movement Analysis (LMA), named after the dancer and movement analyst Rudolf Laban (1879-1958). Quantitative methods tend to use measurements through numerical methods. In this thesis, quantitative methods are used based on video analysis.

Music-related body motion is a broad concept even though it is limited in the musical context, because there are several kinds of movements [10] which are involved in music-related body motion. The most direct one is sound-producing movements. These movements produce sound directly. For instance, a pianist attacks piano keys and a drummer hits a drum. The second is sound-modifying movements. These movements happen to modify the sound. For instance, damping the strings of the guitar and tapping the pedal of the piano are sound-modifying movements. The third is sound-accompanying movements. This kind of movements happens when we move to music. For instance, in club, people tend to dance to music and track the tempo of the music. In this thesis, sound-producing and sound-accompanying movements are mainly studied. Of course, there are other kinds

of movements, such as sound-communicative movements. One representative example is conductor in concert. Even though the conductor does not produce sound directly, the conductor still plays a key role in performance. Figure 3 shows a pianist's performance, where three imaginary boxes indicate three types of movements happening in space.

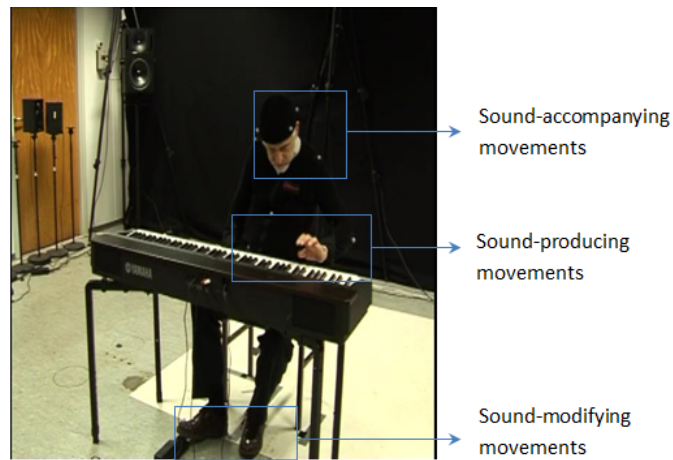


Figure 3: Three types of movements happening in space are indicated by the imaginary boxes, including sound-producing, sound-modifying and sound-accompanying movements.

2.4 Summary

In this chapter, we have given a brief introduction to the music cognition history. Terminology, such as motion and movement were introduced as well, and the difference between them was explained. In methodology section, two types of methods, qualitative methods and quantitative methods were described, and four kinds of music-related movements were introduced. This chapter answered the questions: what is music-related body motion and which terminology should be used to describe music-related body motion. It stands alone and is not highly related to design and develop the musical gestures toolbox, but the target of this chapter is to help better understand music-related body motion. In next chapter, we will start to introduce some techniques used to study the body motion.

Chapter 3 Techniques

Chapter abstract: *This chapter will give a brief insight into three techniques: motiongram, optical flow, and eulerian video magnification. The reason of choosing these three techniques is that they are powerful and have its respective advantages. The motiongram approach has very low computing cost and can provide spatial temporal movement information of an object, whereas the optical flow approach can provide accurate movement analysis by estimating both orientation and magnitude of the motion. The eulerian video magnification approach is able to reveal micro movements and magnify the signal of interest, which is very useful in music research. Firstly, we will explain how the motion image and motiongram are created. Next, the optical flow as a powerful motion estimation technique will be introduced as well. Introducing optical flow to music research is the first attempt in this thesis. Then the eulerian video magnification will be described. At last, motion capture system will be briefly introduced.*

3.1 Introduction

Recording and visualizing the human body movement have puzzled human being for centuries. Tracking the movement could be considered as a four dimensions problem because the movement happens in time and space. With technique and industrial manufacture development, some photographic techniques motion capture systems were developed in 19th century. British-American photographer Eadweard James Muybridge (1830-1904) observed movement through pictures of motion of object [11]. He used a series of time lapse photographs of horses and observed the movement of horse in different time slot. At the same time, Jules Marey (1830-1904) developed types of pictures of time [12].

100 years later, Swedish psychologist Gunnar Johansson (1911-1998) used point light displays to capture the markers of the main joints of the body [13]. He found it possible to recognize various movements from point light displays. More interestingly, it was possible to analysis both temporal and spatial characteristics of the movement from a single picture. This technique was applied in some behavioral-related research.

In the 1940s, the American psychologist James J. Gibson (1904-1979) introduced the concept of optical flow to describe the visual stimulus of animals moving. From then on, many researchers worked on optical flow, and proposed a number of optical flow techniques. The methods of calculating the motion between two image frames which are taken at time interval δt are called differential. Such methods include Lucas-Kanade, Horn-Schunck [14], etc. In the MG toolbox, Horn-Schunck has been implemented to perform a global motion estimation.

Michael Rubinstein and his fellows proposed eulerian video magnification method [15] to reveal subtle variations in videos which are hard or impossible to see with human eyes. Michael Rubinstein has systematically described eulerian video magnification in his Ph.D. thesis [16]. This could be interesting to investigate the performer's and audience's emotional changes in musical research.

Now, motion capture systems using both marker or sensor-based system and camera-based system provide high resolution in time and space. These systems can provide feature analysis of human movement. However, high prices of such systems and usage inconvenience somewhat limit their wide application. Several years ago, Alexander Refsum Jensenius used regular video cameras for recording motion. He has explored various visualization techniques to display music-related body motion [17, 18], which was quite interesting. He first created motion image to represent the motion happening between two successive frames. Then movement of motion image itself was created to display the motion over time. He called it motion history image.

3.2 Motiongram

3.2.1 Motion image

The most common technique in motion analysis of a video is to create *motion image*. The motion image represents the motion happening between two successive video frames, and it is usually created by taking the absolute pixel difference between each two successive frames. As such, the motion image indicates the pixels that have changed between frames with respect to spatial domain. Considering two video frames $I(x,y,t)$, $I(x,y,t + \delta t)$ at time t and $t + \delta t$, respectively, then the motion image is computed by following equation,

$$I_{mo}(t + \delta t) = |I(x,y,t + \delta t) - I(x,y,t)| \quad (1)$$

where (x,y) denotes the location of the image domain. More specifically, t can be represented in video frame number after discretizing the time domain. Figure 4 shows the motion image and motion history image of a dancer with duration of 4 seconds.

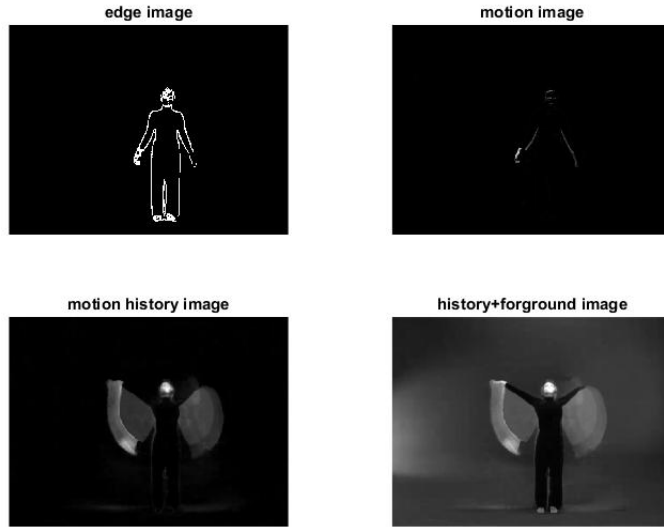


Figure 4: Edge image (top left) was created by applying edge detector. Motion image (top right) was created by taking the absolute difference of each two successive frames. Taking the sum of multiple motion images results in motion history image (bottom left). The bottom left shows the motion history image added to foreground image.

Several factors can affect the quality of the motion image. Firstly, the quality of the raw video stream, background, and foreground of images and camera movement may all influence the quality of the motion image. In practice, it is necessary to preprocess the video frames before computing the motion image. Such processing operations include adjustment of brightness, application of low pass filter, segmentation, rotation of the video, etc. Meanwhile, filter, such as median filter, average filter can be applied on the motion image to remove the noise. Furthermore, the average motion image and motiongram can be created from the motion image.

3.2.2 Theory of motiongram

The concept of the *motiongram* is used by Alexander Refsum Jensenius in his Ph.D.thesis [7]. The motiongram is calculated from the motion image, which is considered as 1D projection into horizontal and vertical direction. As a matter of fact, taking the average of the motion image in horizontal and

vertical direction gives two types of motiongrams: the horizontal and vertical motiongram. Following two equations show the calculation mathematically,

$$I_{gramx}(x, t) = \frac{\sum_{y=1}^n I_{mo}(x, y, t)}{n} \quad (2)$$

$$I_{gramy}(y, t) = \frac{\sum_{x=1}^m I_{mo}(x, y, t)}{m} \quad (3)$$

where (x,y) denotes the location of the image domain, m, n denote the width and the height of the image, respectively. Figure 5 shows the two types of the motiongrams of a dancer with duration of 20 seconds. From the motiongram, it is possible to see the trajectories of the dancer's movements both in horizontal and vertical direction. As such, the types of the movements can be further analyzed.

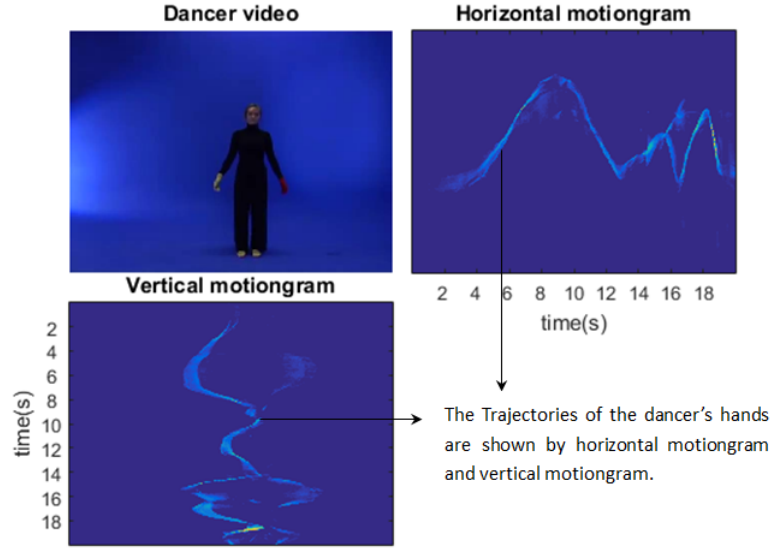


Figure 5: Two types of motiongrams are created from the dancer video. The top left is the dancer video. The top right is the horizontal motiongram. The bottom left is the vertical motiongram. From the motiongrams, it is clear to see the trajectories of the dancer's motion both in horizontal and vertical direction.

Motiongram makes it possible to see both the location of motion and the number of the local moving objects in a video. Several features can be calculated from motiongram. Feature extraction will be discussed in *chapter 5*. The motiongram has been applied in music-related research and medical research. There are two types of medical research, which are attention deficit disorder in the animal experiments [19] and the study of young infants at risk of developing cerebral palsy [17]. When combines the motiongram with spectrogram of the corresponding sound, this is very useful to study the relationships between motion and sound so that it can be used to investigate music-related motion in performance studies.

3.3 Optical flow

3.3.1 Theory of optical flow

The previous section has introduced the motiongram. This section will describe an alternative technique, which is applied intensively in motion estimation. Optical flow is defined as [20]: image flow is the velocity field in the image plane due to the motion of the observer outside of the image, the motion of objects in the scene, or apparent motion which is a change in the image intensity between frames that mimics object or observer motion.

An image sequence is represented by a real valued image intensity function $I(x,y,t)$ that is continuous in space and time. The variable (x,y) denotes the location within a rectangular image domain, and t labels the corresponding frame at time t . The most frequent assumption within these approaches is that the observed intensity $I(x,y,t)$ is conserved over time. This means that the intensity at position (x,y) at time t will be the same as the intensity at time $t + \delta t$ at position $(x + \delta x, y + \delta y)$ for a small δt . Using the intensity function $I(x,y,t)$ along with $u_1(x,y)$ and $u_2(x,y)$, which are the two components of the optical flow vector $u = (u_1, u_2)$, this results in the equation,

$$I(x + u_1\delta t, y + u_2\delta t, t + \delta t) = I(x, y, t) \quad (4)$$

where $u_1\delta t = \delta x$ and $u_2\delta t = \delta y$. Assuming that the brightness varies smoothly over time, the term on the left hand side of the equation 4 can be approximated by a first-order Taylor expansion at the point (x, y, t) .

$$I(x, y, t) + \delta t u_1 \partial_x I + \delta t u_2 \partial_y I + \delta t \partial_t I + \epsilon(\delta x^2, \delta y^2, \delta t^2) = I(x, y, t) \quad (5)$$

Dividing by δt and throwing the higher order terms $\epsilon(\delta x^2, \delta y^2, \delta t^2)$ for $\delta t \rightarrow 0$, then obtaining the following constraint equation.

$$\partial_t I + u \cdot \nabla I = 0 \quad (6)$$

The equation 6 is known as Optical Flow Constraint Equation (OFCE) [14]. It indicates that the intensity of an object keeps constant along the orientation of motion happening. As such, equation 6 is often used when there are small displacements between image sequences. When dealing with large displacements, a coarse-to-fine strategy [21] is often used to compute optical flow, which computes the optical flow in an image pyramid. The reason for this is to make sure the motion only happens between two consecutive pixels.

3.3.2 Optical flowgram

Since optical flow field could be used to estimate the motion, it should have the similar characteristics as the motiongram. Because optical flow image contains the information of magnitude and orientation of motion, taking 1D projection as motiongram should give the similar results. I prefer to call it *optical flowgram* which reveals the spatial temporal information of the movement as the motiongram does. The way to compute the optical flowgram is quite similar as that of motiongram. Assuming that the optical flow image $I_{op}(x, y, t)$ at time t is found, then the optical flowgram can be computed with the following equations,

$$I_{gramx}(x, t) = \frac{\sum_{y=1}^n I_{op}(x, y, t)}{n} \quad (7)$$

$$I_{gramy}(y, t) = \frac{\sum_{x=1}^m I_{op}(x, y, t)}{m} \quad (8)$$

where (x,y) denotes the location of the image domain, m, n denote the width and the height of the image. Figure 6 shows the similar movement trajectories as motiongram, but more clear than motiongram.

3.3.3 Advantage of optical flow

Optical flow field gives the informative amount of the movement from one video frame to the next, and tells us if the movement at the certain location has happened and at what speed by detecting the direction and velocity at all pixels. This characteristic is quite different from the motiongram, and gives us an opportunity to extract proper features from optical flow field as well. In fact, optical flow has been applied on medical research. It has been turned out to be a promising approach for diagnosing infants with Cerebral Palsy [22]. Feature extraction using optical flow will be discussed in detail in the *section 5.3*.

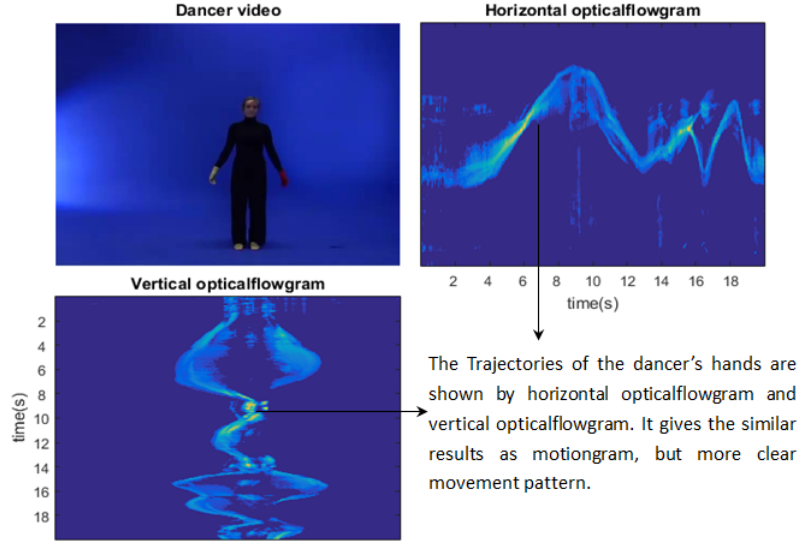


Figure 6: Two types of optical flowgrams are created from the dancer video. The top left is the dancer video. The top right is the horizontal optical flowgram. The bottom left is the vertical optical flowgram. From the optical flowgrams, it is more clear to see the trajectories of the dancer’s motion both in horizontal and vertical direction.

3.3.4 Comparison with motiongram

Since both motiongram and optical flow field can be used for motion estimation, it is interesting to compare both of them. Here an example is given to show the difference between motiongram and optical flow field. Following results are calculated from the pianist’s performance video, which is temporally extracted from 10 seconds to 15 seconds. Figure 7 shows the pianist video, and figure 8 shows the quantity of motion (QoM) and centroid of motion (CoM) of the video that both motiongram and optical flow provide the similar results. This can be observed from the locations of the peaks as well. The optical flow actually provides more details of motion by estimating the motion orientation of the object. Figure 9 and figure 10 show the optical flow field of the pianist, where the arrows point mainly the moving orientation of the pianist’s hands and the length of the arrows indicates the magnitude of velocity of the moving hands.

The optical flow approach assumes constant illumination and velocity smoothness, however, this is not usual case in real images. Even though illumination will influence the results computed by the motiongram as well, it has less influence than optical flow. In addition, the motion of a homogeneous object is locally ambiguous, which is somewhat called aperture problem. Within aperture, different physical motions are indistinguishable. To avoid aperture problem, several improved optical flow techniques were proposed including feature-based optical flow [23] and coarse-to-fine optical flow, etc. At last, computing optical flow field itself involves the gradients in 3D (dx , dy , dt), which has higher computation cost than that of motiongram. For instance, considering a video with the length of 20 seconds, frame rate 30 and frame width 480, frame height 640, elapsed time of the motiongram method is often less than 25 seconds, whereas the optical flow method takes more than 40 seconds.



Figure 7: The pianist video

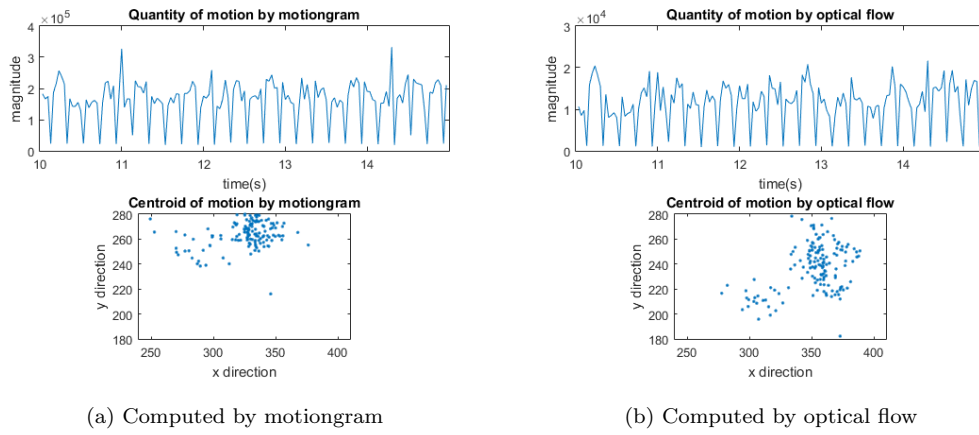


Figure 8: (a) shows the quantity of motion (QoM) and centroid of motion (CoM) computed by motiongram; (b) shows the QoM and CoM computed by optical flow. We see that both methods give the similar QoM.

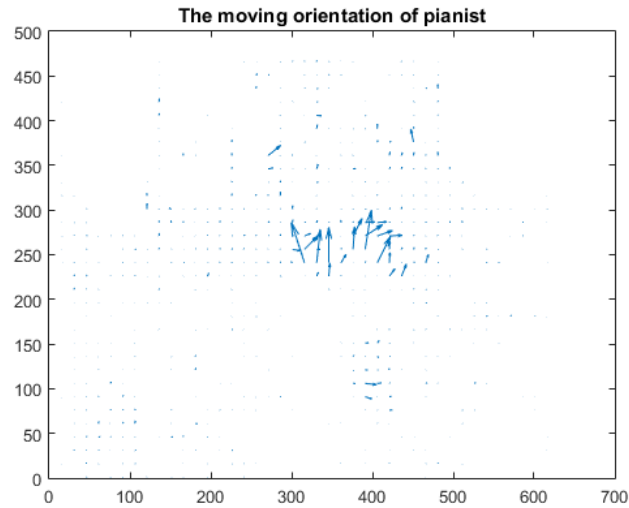


Figure 9: Optical flow field, the arrows indicate the moving orientation of the object. The length of the arrow indicates the magnitude of velocity.



Figure 10: The background image with optical flow field. Large arrow indicates the moving orientation of the pianist's hand.

3.4 Eulerian video magnification

Eulerian video magnification [15] can amplify small variations in a video. The basic idea of the eulerian video magnification is to apply spatial and temporal filters to magnify subtle temporal changes in a video. Firstly, the input video sequence is decomposed into different spatial frequency bands, which are normally created by Laplacian pyramid. Then temporal processing is applied on each spatial band. This procedure is normally performed by applying a bandpass filter on each frequency band. After the temporal processing, the frequency bands of interest are found and magnified by multiplying a magnification factor α . Lastly, the magnified frequency band is added to the original and collapse the spatial pyramid to generate the output.

The theory behind eulerian video magnification is the same as optical flow, which uses the first-order Taylor series expansions. Let $I(x,y,t)$ denote the image intensity at position (x,y) and time t . Given a displacement function $\delta(t)$, the observed intensities after displaced motion can be expressed as $I(x,y,t) = f(x + \delta(t), y + \delta(t))$ and $I(x,y,0) = f(x,y)$. Assuming the image can be approximated by the first-order Taylor series expansion with respect to x,y , then

$$I(x,y,t) \approx f(x,y) + B(x,y,t) \quad (9)$$

where

$$B(x,y,t) = \delta(t) \frac{\partial f(x,y)}{\partial x} + \delta(t) \frac{\partial f(x,y)}{\partial y} \quad (10)$$

If $B(x,y,t)$ is magnified by a magnification factor α and added back to $I(x,y,t)$, then we have

$$\bar{I}(x,y,t) = I(x,y,t) + \alpha B(x,y,t) \quad (11)$$

Combining above three Equations, we obtain

$$\bar{I}(x,y,t) \approx f(x,y) + (1 + \alpha)\delta(t) \frac{\partial f(x,y)}{\partial x} + (1 + \alpha)\delta(t) \frac{\partial f(x,y)}{\partial y} \quad (12)$$

The goal is to magnify the displacement function $\delta(t)$ by a magnification factor α , such that

$$\tilde{I}(x,y,t) = f(x + (1 + \alpha)\delta(t), y + (1 + \alpha)\delta(t)) \quad (13)$$

Assuming the first-order Taylor expansion holds for the amplified larger perturbation, $(1 + \alpha)\delta(t)$. Then we end up with

$$\bar{I}(x,y,t) \approx f(x + (1 + \alpha)\delta(t), y + (1 + \alpha)\delta(t)) \quad (14)$$

The above equation shows the displacement $\delta(t)$ of the local image $f(x,y)$ at time t has been magnified by a factor α .

To see how the EVM combined with the motiongram works, a test video was magnified by the EVM method, and then motiongram method was applied on the magnified video. The video shows a sleeping baby in which the breathing movements of the baby were not possible to see by human eyes. Figure 11 shows the original video and its motiongram. As a matter of fact, it is not possible to observe the subtle breathing movements of the baby in the original video, and motiongram could not obviously reveal spatial temporal information as well. Figure 12 shows the magnified video and its motiongram, which is clear to see periodic breathing movements of the baby in both vertical and horizontal motiongram.

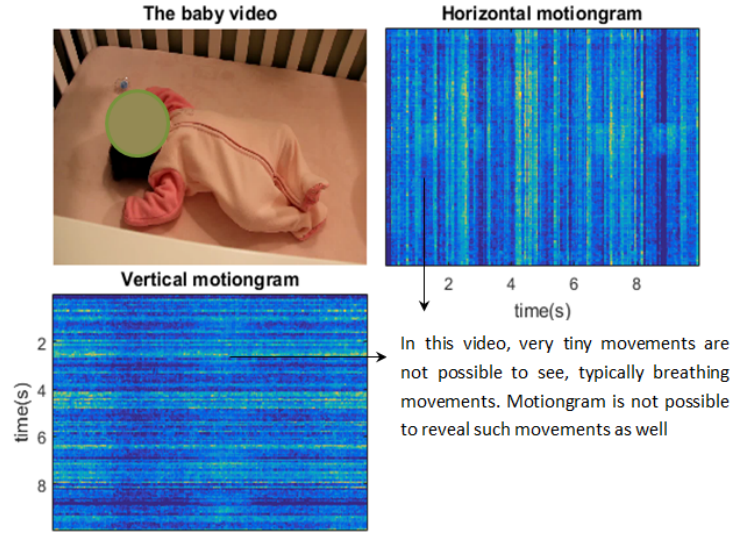


Figure 11: The original baby video, and its motiongram. Motiongram could not reveal useful information because of subtle movements.

One problem of the EVM is that there are many parameters needed to be tuned, such as coefficients of the filters, magnification factor, and spatial cutoff frequency, etc. However, some reference parameters were given in [15], which could satisfy various applications. The EVM could be very helpful in music research as well. In some typical cases, i.e., standing still analysis in a musical context, it would be interesting to study performer's and perceiver's emotion changes corresponding to music. In such cases, applying the motiongram or optical flow approach on the original video directly may not get ideal results, whereas applying the motiongram or optical flow approach on the video processed by the EVM approach would give the expected results. Another problem of the EVM is that the EVM can only work well on a video with small variations. The phase-based EVM [24] supports the large amplification factors and is significantly less sensitive to noise.

3.5 Motion Capture system

Motion capture (MoCap) often refers to describe the process of recording human body motion. There are many existing MoCap technologies [4, 25]. Generally, they are divided into two different groups: optical and non-optical systems. Inertial sensor systems based on sensors such as gyroscopes, accelerometers and magnetometers are most affordable and popular among the non-optical systems. However, such systems have lower spatial accuracy and precision than optical systems. Optical systems can be divided into two categories as well: marker-less systems and marker based systems.

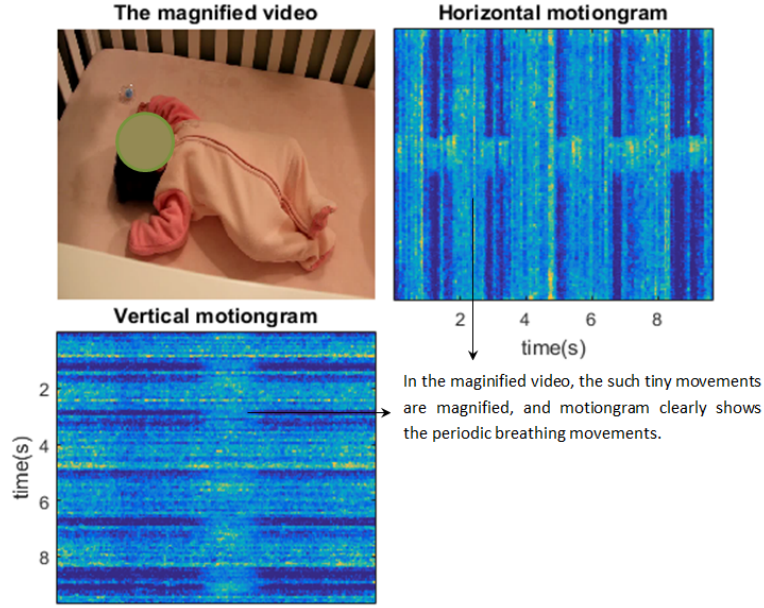


Figure 12: The magnified video, and its motiongram. It is now possible to observe the periodic breathing movements of the baby.

Normally, marker based systems provide more accurate partial tracking of body motion. Optical infrared marker based motion capture (IrMoCap) systems are kind of the state of the art among the motion systems due to their high spatial temporal precision and accuracy. Optical infrared marker based motion system consists of a group of cameras, surrounding the persons to be tracked. The infrared light emitted from cameras is bounced off reflective markers attached on the body of the person, which is observed and captured by the cameras again. IrMoCap can provide the most precise, reliable and fast MoCap solution. In the MG toolbox and MoCap toolbox, the mocap data which is generated by IrMoCap is used as input for the analysis of music-related body motion.

However, some drawbacks limit wide application of motion capture systems. On the one hand, their price is high, because infrared systems require expensive equipment and need to be installed in a controlled environment. On the other hand, sensor-based motion capture systems forcing the user to wear sensors and cables on the body may make the user somewhat uncomfortable and result in unnatural performance. With development of computer vision techniques, video based visualization techniques could also provide promising solution. Especially, video cameras with higher resolution become more affordable, and make them a good starting point for those researchers, artists without access to expensive motion capture systems. This is also one of reasons why we decide to develop the MG toolbox based on video analysis.

3.6 Summary

In this chapter, we have described the details of the three techniques. In the first two sections, the steps of creating the motiongram were described. We have also introduced how the motiongram could be applied in music-related body motion analysis. As a common method for video analysis, the motiongram actually provides us spatial temporal information of the movement of an object. Combining motiongram with other tools, such as spectrogram of sound, in fact, allows us to analyze the relationship between sound and movement. In the *section 3.3*, optical flow was presented, and the comparison with motiongram was illustrated by an example. We showed that both motiongram and optical flow provided similar results with respect to quantity of motion and centroid of motion. In the

section 3.4, we introduced eulerian video magnification, which is used to reveal micro movements and magnify the signals of interest in a video. EVM is very useful to perform emotion analysis in music research. At last, as a powerful motion analysis tool, motion capture system was briefly introduced as well. Generally, video-based motion analysis and motion capture system are complementary in practice. This chapter answered the question: which techniques and tools are used to study music-related body motion by introducing the three techniques of video analysis and motion capture systems, and formed the technological foundation for the implementation of the toolbox.

PART 2
DESIGN AND IMPLEMENTATION

Chapter 4 Design

Chapter abstract: *This chapter will describe the framework, the memory management, data structure and data flow of the MG toolbox. Generally, the framework gives an insight to software being developed. To start the development of the MG toolbox, it is necessary to discuss its framework. Memory management for any software and system development is critical, especially dealing with large files. As the MG toolbox aiming to deal with the analysis of large data file as well, the memory management is especially discussed in the section 4.2. Then the section 4.3 gives an in-depth discussion of data structure of the MG toolbox. It is actually based on memory management scheme. Followed by data structure, data flow of the MG toolbox is introduced in section 4.4.*

4.1 Framework

The general framework was shown in the figure 1. There are three different inputs: video, audio and motion capture data. It is necessary to mention that all three inputs should be recorded from the same object. With no exception, the first step is to import the different recordings into Matlab workspace. So the first part of the framework is to import data. Considering dealing with large data file, particularly, video file, preprocessing the data file is necessary. Such preprocessing operations include resampling, rotating, extracting and cropping the video. After preprocessing, the various features are computed by the feature extraction part. Finally, the toolbox provides two visualization tools to plot the motiongrams, motion image over time, as well as audio waveform and QoM. In general, the framework includes four main parts, importing, preprocessing, feature extraction, visualization. The dotted line in figure 1 indicates possible extension of the MG toolbox in the future work, including applying advanced machine learning algorithms to classify the data set and implementing the sonomotiongram, which is the inverse of Fourier Transform of the motiongram. They would be interesting and promising extensions of the toolbox. As this part is not the main task in the current MG toolbox, it is not yet implemented. However, related future work will be mentioned in *chapter 7*.

4.2 Memory management

Memory management is important in software and system development as improper memory management usually results in memory overflow problem, and influences the computation efficiency as well. Matlab has provided powerful functions in video processing. It used object-oriented programming to create a VideoReader object to read video files. However, after reading video file into Matlab workspace, there are two strategies to continue preprocessing: strategy A is to store all frames of the video file, which is memory-consuming when we deal with large video files. On the other hand, passing the parameters including video data across the functions influences the computation efficiency as well. Strategy B is to write processed data back to disk, only storing and passing a constructed object. It reads video data from disk when it needed, i.e., a down sampled video is written back to the disk when it needs to do contrast adjustment. This is with no doubt more efficient when we process the large video files. In the MG toolbox, the similar smart approach is applied. A musical gestures data structure is created which contains three fields: video, audio, mocap, respectively, but only the object is stored and passed. Furthermore, after preprocessing, the preprocessed results are written back to the disk and stored for further usage. Referring to the memory management of the MIR toolbox as well, only the computed features are stored in musical gestures data structure. These two main memory management approaches make the MG toolbox possible in dealing with large data files.

As a comparison, we showed the elapsed time of adjusting contrast of a video with the function

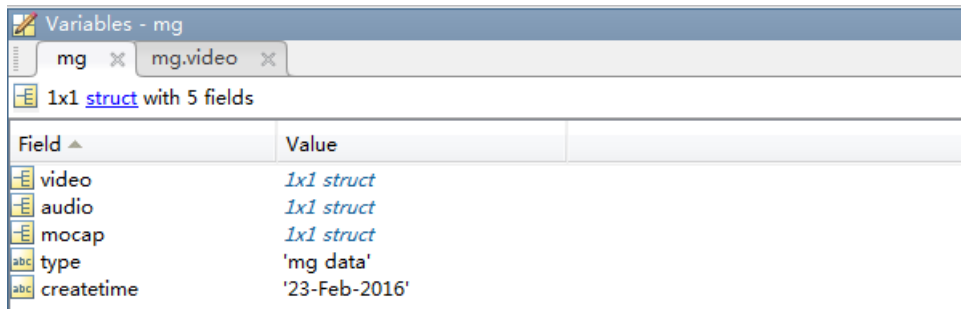
mgvideoadjust in the MG toolbox. The test video is the pianist video, with frame rate 30, frame width 640 and frame height 480. To test more generally, we took average of elapsed time of 10 runs. The results are shown in table 1. We see that strategy A (storing data in Matlab) has obvious advantage when processing short input video, usually less than 90 seconds, whereas dealing with long input video, strategy B (storing data in disk) is better than strategy A.

test function:mgvideoadjust,test video:pianist video		
video length(s)	elapsed time(s)	
	strategy A	strategy B
20	16.36	35.44
40	31.18	43.95
60	42.35	52.12
70	47.27	61.48
80	55.84	72.32
90	91.56	84.51
100	150.82	93.76

Table 1: Elapsed time of strategy A (storing data in Matlab) and strategy B (storing data in disk). It is tested on computer with windows 7 operating system, Inter(R) i5 CPU 2.5GHz, 4GB RAM.

4.3 Data structure

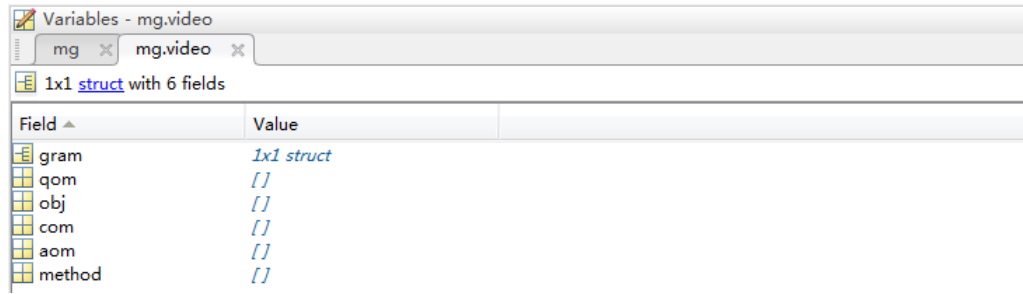
The MG toolbox aims to deal with three different types of media files: video, audio, mocap data file. A proper data structure is of course extremely important. Firstly, the data structure should provide easy operations across the different media files. Secondly, it should be convenient to access its content and variables. Based on these two principles, structure array becomes the first option. The data structure of the MG toolbox is called *musical gestures data structure* which corresponds to the title of the toolbox. A musical gestures data structure can be created by the function, *mginitstruct* (see Appendix) in the MG toolbox, or can be constructed during import and preprocessing. For instance, *mg = mginitstruct* initiates a musical gestures data structure called *mg*. Figure 13 shows three fields of *mg*, which the video field stores a VideoReader object and parameters, audio field keeps a MIR object from the MIR toolbox, mocap contains a mocap structure from the MoCap toolbox, respectively. The additional fields are the type and creating time of *mg*.



Field	Value
video	1x1 struct
audio	1x1 struct
mocap	1x1 struct
type	'mg data'
createtime	'23-Feb-2016'

Figure 13: Musical gestures data structure: *mg*

Figure 14 shows 5 sub-fields of the video, which the field *gram* is used to store the motiongrams, the field *qom* stores the quantity of motion, *com* keeps the centroid of motion, *aom* is for the area of motion, *obj* is kept for VideoReader object, *method* indicates the method of computing motion.



Field	Value
gram	1x1 struct
qom	[]
obj	[]
com	[]
aom	[]
method	[]

Figure 14: The video sub-struct of mg

4.4 Data flow

To understand the MG toolbox well, the data flow is shown in this section. There are actually two routes to do analysis in the toolbox, which correspond to two main flows are shown in figure 15, with each part of the framework being marked by the red imaginary box. This figure lists almost all the functions in the toolbox. With the first flow from *mgvideoreader* to *mgsave*, it will import only video and preprocess it first, and then corresponding audio and mocap data are imported later with the function *mgmap*. The function *mgmotion* calculates the motion image, motiongram, quantity of motion, centroid of motion, area of motion, etc. It provides two options to do motion analysis, which are motiongram and optical flow. After that, two functions *mgvideoplot* and *mgwaveplot* are helpful for visualization. Furthermore, some functions: *mgautocor*, *mgsimilarity*, *mgstatistics* provide additional analysis. Finally, *mgsave* stores the data structure in the disk. The second flow marked with blue dotted line is from *mgreadsegment* or *mgread* to *mgsave*. The function *mgread* or *mgreadsegment* can read any type or any combination of three different recordings. As such, this route does not need *mgmap*. Others are the same as the first one.

4.5 Summary

This chapter has presented the framework, memory management, data structure and data flow of the toolbox. The framework showed an overview and the main parts of the MG toolbox. In next chapter, these main parts will be introduced in detail. How the toolbox manages the memory was introduced as well. The memory management scheme of the toolbox gave an inspiration how to construct the MG data structure. Data structure played a critical role in the development of systems. The data structure of the toolbox is so simple that it provides easy operations on any types of three recordings. Data flow gives an overview of functions in toolbox, and it extends the framework to individual functions. This chapter has laid design foundation for the implementation of the MG toolbox. The next chapter will start to introduce its implementation.

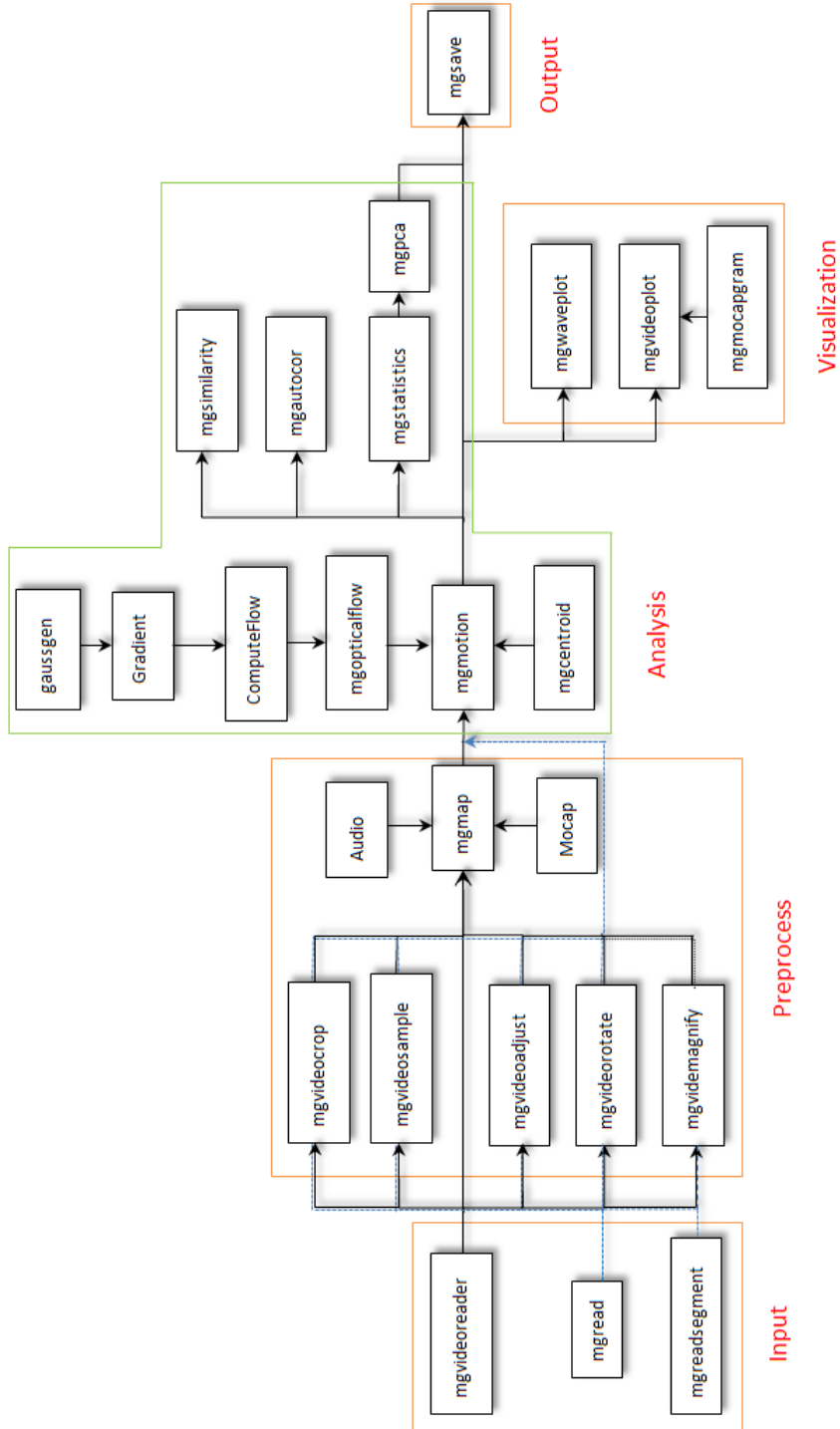


Figure 15: An overview of data flow in MG toolbox. Dotted blue line shows the second flow. The corresponding parts are marked by orange boxes.

Chapter 5 Implementation

Chapter abstract: In this chapter we will have a thorough introduction of import, preprocessing, feature extraction, and visualization tool in the MG toolbox. Section 5.1 introduces the import, that is how the toolbox reads different data into Matlab workspace. In section 5.2 preprocessing of the data is discussed and four basic preprocessing functions in the toolbox will be illustrated. Import and preprocessing, in fact, correspond to first two parts of the framework shown in the figure 1. In section 5.3, we show the feature extraction using motiongram and optical flow. At last, visualization tools will be presented in section 5.4.

5.1 Import

There are often three types of data recordings: video, audio, mocap data, and two main operations involved in the importing phase. One is to import a single data stream or any combination of data recordings without extracting temporal segment. Other one is to extract a temporal segment of any single data stream or any combination when importing. The MG toolbox provides three importing functions: *mgread*, *mgvideoreader*, *mgreadsegment* (see Appendix), which can handle the two operations above and finally returns a musical gestures data structure. The musical gestures data structure was introduced in chapter 4. For instance, *mgread* reads any type and any combination of data recordings into Matlab workspace by opening a file selection dialog and returns a musical gestures data structure. With *mgvideoreader*, it reads a video file, and user can set the parameters to extract the temporal segment. The function *mgreadsegment* can read a temporal segment from a musical gestures data structure, no matter what type of data recording the structure contains.

5.2 Preprocessing

After importing, in order to deal with large data files, particularly video files, preprocessing data is necessary. It not only influences the speed of computing, but also the quality of the computed features. To implement this procedure, user has to balance both of them. The MG toolbox provides four basic functions, which aim to resample, crop, rotate the video and adjust the contrast of the video. Note that all these operations should be done on video recordings. After these operations, the processed video recordings will be written back to the disk and stored for further usage. Writing back is a smart design in the toolbox. Because preprocessing large data recordings itself is time-consuming, storing the processed results is very helpful. For more details of these functions, please refer to the MG toolbox manual. Here four basic preprocessing functions: *mgvideocrop*, *mgvideorotate*, *mgvideoadjust*, *mgvideomagnify* (see Appendix) are selected for illustration.

5.2.1 mgvideocrop

The function *mgvideocrop* crops the region of interest of a video in the time domain. This function is further built on *imcrop* in Image Processing Toolbox provided by MathWorks, which crops an image. In the MG toolbox, *mgvideocrop* works on video stream data and crops the region of interest according to the user's selection. This operation is very useful when we want to focus on one local object in the video. Even though the positions of the object may be different in every video frame, the spatial range of the movement is not so hard to find in some special cases, for instance, pianist's performance. There are three main types of music-related motion, sound-producing movement which happens in hands attacking piano keys, sound-accompanying movement which happens in upper body moving during performance and lastly sound-modifying movement which happens in foots stepping the pedal of the piano. Generally, the hands of the pianist are the most interesting local objects. Cropping only the

region of the hands may give us directly analysis of music-related motion. The function *mgvideocrop* provides two options when cropping. It can crop the region according to the given coordinates. If the coordinates is not given to the function, it will first plot the first frame of the video, and allow user to select the region of interest. Then it crops the selected region in the video. The function *mgvideocrop* stores the cropped video in the disk and returns a musical gestures data structure which contains parameters of the cropped video. Figure 16 shows the original video and cropped video which only shows the upper body of the pianist. This would be helpful to analyze pianist's hands and head movements.

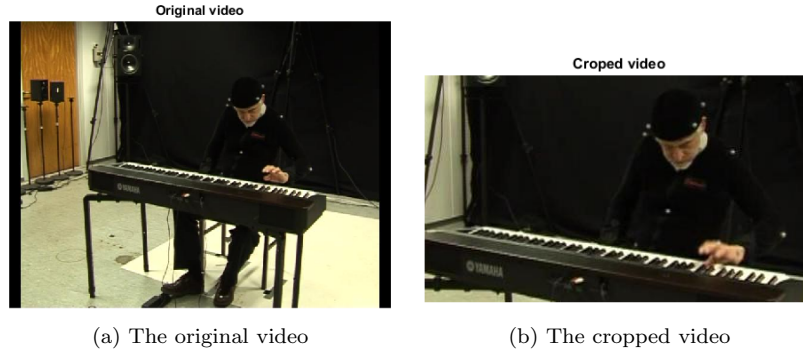


Figure 16: The left shows the original video, the right shows the cropped video. The cropped video only shows the upper body of the pianist.

5.2.2 *mgvideoadjust*

The function *mgvideoadjust* adjusts the contrast of a video. Matlab has provided the function *imadjust* adjusting the image intensity value or colormap. In the MG toolbox, the function *mgvideoadjust* is further built on *imadjust*, as such, it could adjust the contrast of a video. It will be applied when a video has low contrast. The video with low contrast usually has no sharp differences between black and white, resulting in ambiguous motion. To change the contrast of the video, *mgvideoadjust* performs a contrast stretching, which means pixel values below a specified value are displayed as black, pixel values above a specified value are displayed as white and pixel values between these two values are linearly mapped into the entire range of graylevels. Finally it stores the adjusted video in the disk and returns a musical gestures data structure containing the parameters of the adjusted video. Figure 17 shows the original video frame and adjusted video frame, respectively. After contrast adjusting, it highlights the local details of the pianist, such as hands and face.

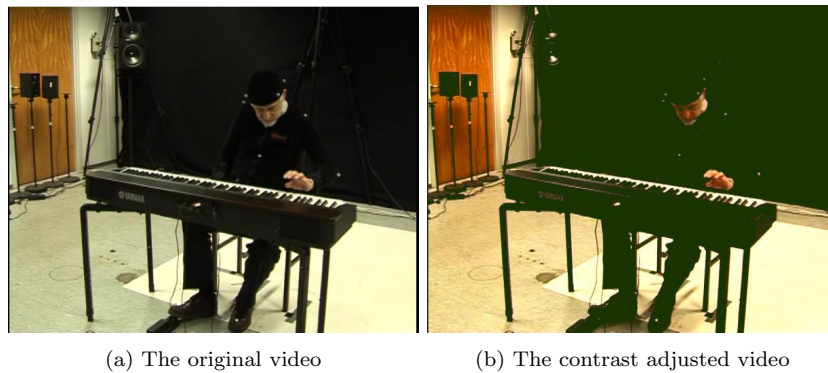


Figure 17: The left shows the original video, the right shows the contrast adjusted video. The contrast adjusted video can highlight the some details of the pianist's face and hands, suppress the details of the cloths.

5.2.3 mgvideorotate

The function *mgvideorotate* rotates a video by a specific angle. This function is built on *imrotate* in Image Processing Toolbox provided by MathWorks as well, which is applied to rotate an image. In the MG toolbox, the function *mgvideorotate* rotates a video according to certain angles. This is very useful when the video is recorded by the camera from different angles. Rotating the video could provide us proper viewing angle. Since the motiongrams are created by projecting motion image in horizontal and vertical direction, so viewing angle is important for the creation of the motiongrams as well. Poor viewing angle results in motiongrams not revealing movement trace of the object. After rotating the video, it stores the rotated video in the disk and returns a musical gestures data structure containing the parameters of the rotated video. Figure 18 shows the original video frame and rotated video frame.

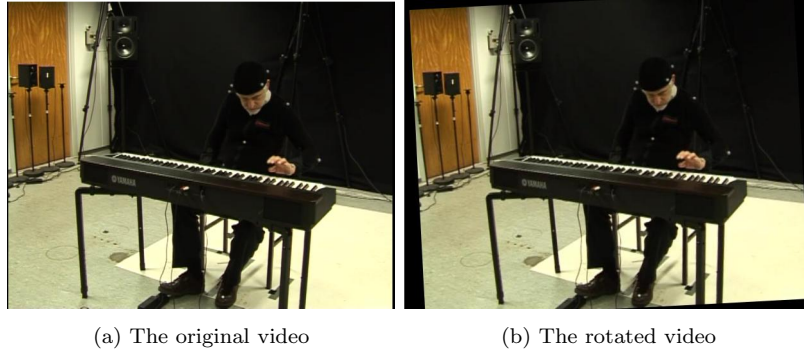


Figure 18: The left shows the original video, the right shows the rotated video with rotation angle 3 degrees in counterclockwise direction.

5.2.4 mgvideomagnify

The function *mgvideomagnify* magnifies the signals of interest of a video. This is very helpful when we want to analyze micro movements and emotion. This function applies eulerian video magnification (EVM) to amplify small variations in videos (see section 3.4). The basic idea of the EVM is to apply spatial and temporal filters to magnify subtle temporal changes in a video. Currently, the function *mgvideomagnify* provides two types of the filters, Butter filter and IIR filter, and allows users to select coefficients of the filter and magnification factor. As the EVM aims to magnify subtle variations of the video, it will be sensitive to noise if the video has large variations. Figure 19 shows the original video and magnified video, respectively.

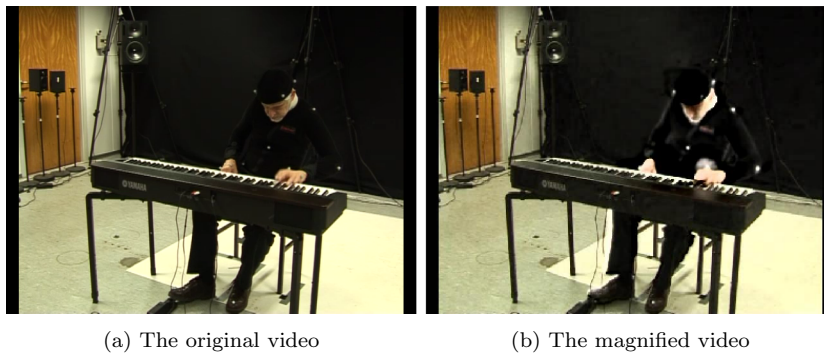


Figure 19: The left shows the original video and the right shows the magnified video by the function *mgvideomagnify*. We see the movements of the local objects like hands and head are magnified.

5.3 Feature extraction

As future additions of the MG toolbox will include more advanced machine learning techniques for feature extraction and classification across the different types of data recordings, feature extraction plays a key role in this thesis. How are the features generated? It concerns the feature generation stage of the design of a classification system. The basic rule of feature extraction is that extracted features should contain discriminant information. In this section, we will present the feature extraction from the video recording. Three features, quantity of motion (QoM), centroid of motion (CoM), area of motion (AoM) will be described, respectively.

5.3.1 Quantity of motion

The most direct feature of motion image is the amount of motion, which is how much the pixel has changed. So the value of quantity of motion for each motion image is calculated by taking the sum of motion image. There are two routes to calculate the quantity of motion: motion image and optical flow field.

QoM from motiongram. The creation of motion image is discussed in *chapter 2*. Given a motion image $I_{mo}(x, y, t)$ at time t , the quantity of motion is computed by following equation [18],

$$QoM(t) = \sum_{x=1}^m \sum_{y=1}^n I_{mo}(x, y, t) \quad (15)$$

where m, n are the width and height of motion image. More specifically, t can be represented in frame number.

QoM from optical flow. Optical flow method will generate two matrices U and V , which contain the horizontal components and vertical components of the velocity vector. By taking the magnitude of the velocity vector, a scalar matrix A is created by equation 16 which contains the magnitude of the velocity vector of each pixel [22].

$$A = \sqrt{U^2 + V^2} \quad (16)$$

Then, the value of the QoM is computed by taking the total sum of the matrix A .

5.3.2 Centroid of motion

Centroid of motion represents the central moments of the motion. Figure 20 shows a pianist's centroid of motion (CoM). The centroid of motion shows the spatial information of the motion. There are two routes to calculate the CoM as well: motiongram and optical flow.

CoM from motiongram. Given the motiongrams I_{gramx} and I_{gramy} , the way to compute the CoM is with the following equations,

$$\bar{x} = \frac{\sum_{i=1}^n x_i I_{gramx}(x_i, t)}{\sum_{i=1}^n I_{gramx}(x_i, t)} \quad (17)$$

$$\bar{y} = \frac{\sum_{i=1}^m y_i I_{gramy}(y_i, t)}{\sum_{i=1}^m I_{gramy}(y_i, t)} \quad (18)$$

where I_{gramx} , I_{gramy} are calculated by *equation 2* and *equation 3*. Furthermore, the position invariant moments can be further calculated by the equation 19,

$$u_{p,q,t} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q I_{mo}(x, y, t) \quad (19)$$

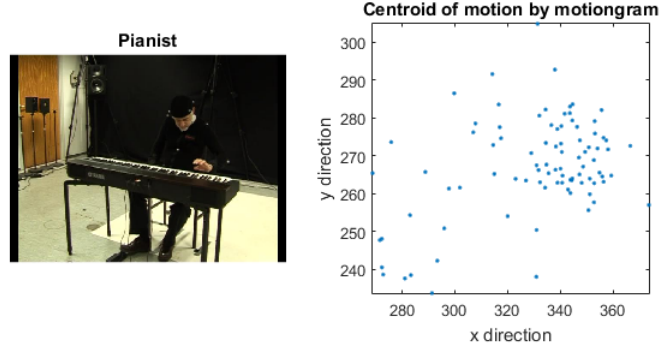


Figure 20: The left shows a pianist, the right shows the centroid of motion.

where p, q indicate the order of moments.

CoM from optical flow. Given the scalar matrix computed by equation 8, taking the mean in both horizontal and vertical direction, results in one $1 \times n$ vector \bar{A}_x and $n \times 1$ \bar{A}_y vector, respectively. Then the CoM is found by using equation 12 and 13 [22],

$$\bar{x} = \frac{\sum_{i=1}^n \bar{A}_x(x_i) x_i}{\sum_{i=1}^n \bar{A}_x(x_i)} \quad (20)$$

$$\bar{y} = \frac{\sum_{i=1}^n \bar{A}_y(y_i) y_i}{\sum_{i=1}^n \bar{A}_y(y_i)} \quad (21)$$

5.3.3 Area of motion

The feature area of motion is one kind of features which is invariant to rotation. Generally, the area is defined as:

$$Area = \int_x \int_y I(x, y) dx dy \quad (22)$$

where $I(x, y) = 1$ if the pixel is within the object, 0 otherwise. In digital images, the area of motion can be defined in the same way.

$$Area_{mo} = \sum_x \sum_y I(x, y) \quad (23)$$

5.3.4 Statistics descriptors

The basic statistics descriptors are the mean, standard deviation, skewness, kurtosis, etc. In Musical Gestures Toolbox, it includes two types of the descriptors: the first order descriptors and the second order descriptors. The second order descriptors contain the spatial information which are computed by using gray level co-occurrence matrices (GLCM) [26]. With GLCM, contrast, correlation, energy, and homogeneity can be computed. Here the details for calculation of the second order descriptors

are not discussed. Reader can refer to function *mgstatistics* in *Appendix*.

$$\begin{aligned}
\text{mean} : u &= \sum_{i=0}^{G-1} iP(i) \\
\text{variance} : \sigma^2 &= \sum_{i=0}^{G-1} (i - u)^2 P(i) \\
\text{skewness} : \gamma_3 &= \frac{1}{\sigma^3} \sum_{i=0}^{G-1} (i - u)^3 P(i) \\
\text{kurtosis} : \gamma_4 &= \frac{1}{\sigma^4} \sum_{i=0}^{G-1} (i - u)^4 P(i) - 3
\end{aligned} \tag{24}$$

where $P(i)$ is graylevel density histogram.

5.3.5 Some related audio features

MIR toolbox has provided many functionalities of extracting audio features, including dynamics, rhythm, timbre feature extractors, etc. We will not describe all of them. In this thesis, we mainly focus on root-mean-square (RMS), spectrum, flux, and tempo of audio.

Root-mean-square energy. Since RMS presents a measurement of the energy of the signal, it would be highly related to QoM. Considering the audio as a signal represented by x , then the global energy of the signal x can be computed by taking the root average of the square of the amplitude with the following equation,

$$x_{rms} = \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2} = \sqrt{\frac{x_1^2 + x_2^2 + \dots + x_n^2}{n}} \tag{25}$$

If the audio is frame-decomposed, taking the RMS of each successive frame results in dynamic energy measurement of the audio, which would be related to QoM.

Spectrum. Spectrum representing a decomposition of the energy of a signal can be calculated using Discrete Fourier Transform (DFT). It is computed by following equation,

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn}, k = 0, \dots, N \tag{26}$$

where N is the length of the signal.

Flux. Flux as a distance measurement between successive frames is kind of similar with QoM as well. To what extent flux has close relationship with QoM. In the MG toolbox, flux is computed based on spectrogram, measuring spectrum distance of successive frames.

Tempo. Tempo refers to the pace of a piece of audio and is usually given in beats per minute. The tempo estimation is based on detecting periodicities in a range of beats per minute (BPM), and choosing the maximum periodicity score for each frame separately. Generally, tempo estimation includes three major steps, onset detection, periodicity estimation and beat location estimation. The onset detection is normally performed based on the spectral flux of the audio. Autocorrelation is often used to estimate periodicity.

5.4 Visualization tool

In this section, two kinds of visualization tools in MG toolbox will be introduced. The first one is *mgvideoplot*, which plots the video frame providing several options to make plots over time. Such options include “Motiongram”, “Boundingbox”, “Opticalflow” and “Gradient”. The second one is *mgwaveplot*, which shows the waveform of audio, spectrum of waveform, and spectrum of quantity of motion.

5.4.1 mgvideoplot

The first visulization tool in the MG toolbox is the function *mgvideoplot*. This function provides user four options, which are “Motiongram”, “Bounding box”, “OpticalFlow” and “Gradient”. With “Motiongram” option, it shows the horizontal and vertical motiongram, motion image and mocapgram over time. The plot gives directly visual comparison between the motion image and motiongrams. With “Boundingbox” option, *mgvideoplot* shows bounding box based on motion area in each video frame. If multiple motion areas happen, then the largest three of them are chosen to plot. With “Opticalflow” option, *mgvideoplot* plots the optical flow field in each video frame. Lastly, with “Gradient” option, it plots the horizontal and vertical gradient of each video frame. Motiongram and optical flow have been introduced in *chapter 3*. Here figure 21 and figure 22 show how the bounding box and gradient work.



Figure 21: *mgvideoplot* with “Boundingbox” option. The red bounding box in the video frame shows the largest motion area.



Figure 22: *mgvideoplot* with “Gradient” option. It shows both horizontal and vertical gradient of the video frame.

5.4.2 mgwaveplot

The second visualization tool in the MG toolbox is *mgwaveplot*. This function shows the waveform of the audio, rooted mean square (RMS) of the frame-decomposed audio, spectrogram, and spectrum of the RMS in the same plot. Figure 23 shows the results computed from pianist's audio data. Furthermore, if QoM is computed, the function will show QoM and RMS in another plot. The relationship between QoM and RMS is quite close as both of them represent the energy of the signals. Figure 24 shows the results.

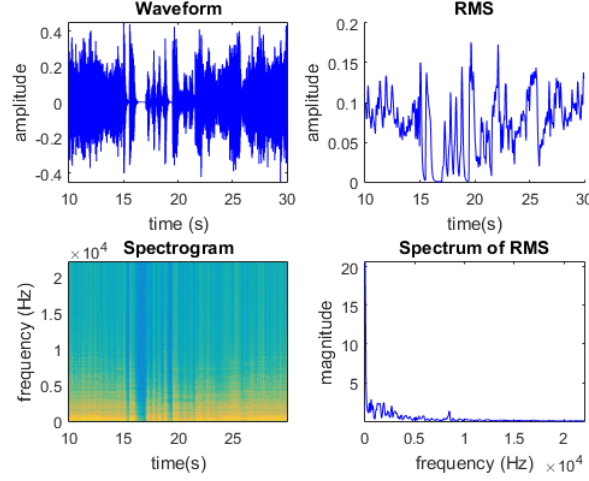


Figure 23: The top left shows waveform of the audio, top right shows RMS of frame-decomposed audio; The bottom left shows spectrogram of the audio, the bottom right shows the spectrum of the RMS.

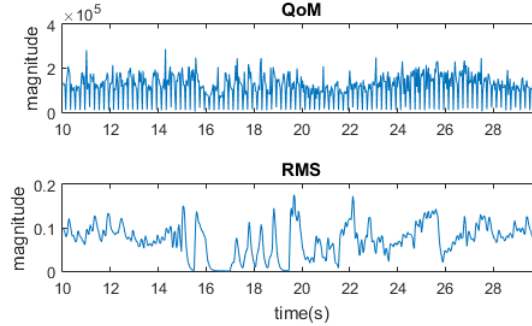


Figure 24: The top shows QoM of the video, the bottom shows RMS.

5.5 Summary

In this chapter, importing and preprocessing operations in the toolbox were described briefly. Four typical preprocessing functions were selected for illustration. The MG toolbox provided actually various operations, such as concatenating two videos, down sampling a video, etc. They are not introduced here. For any details of these functions, reader can refer to the MG toolbox manual or *Appendix*. Preprocessing plays an important role in feature calculation as it affects the quality of the features. After preprocessing, feature extraction was discussed in *section 5.4*. Various features extracted from motion image, motiongram and optical flow field were described. The calculations of most of them were shown

mathematically. The MG toolbox manual illustrates the functions to compute them in Matlab. These features, especially, QoM and CoM represent the basically quantitative measurements of the motion and play an important role in video analysis. With future conditions of the toolbox, these features will be applied in classification problem across the different media data.

PART 3
USING THE MG TOOLBOX

Chapter 6 Using the MG Toolbox

Chapter abstract: In the previous chapter, various features were extracted from motiongram and optical flow field. The question is what the relationship between these features and music is. In this chapter, two cases are shown in order to investigate music-related body motion. The first one is dancing dataset, which is related to sound-accompanying movements. The periodicity of movements and tempo of audio will be analyzed. Furthermore, a comparison will be made with motion capture view. The second one is pianist's performance data set, which is related to sound-producing movements. In this example, the periodicity of movements, RMS and spectrogram flux of the audio will be used to analyze the music-related body motion.

6.1 Case 1

The first case is dancing data set. In the dancing video, nine persons, with one marker on each person's head, dance to music in a club-like environment. Meanwhile, the crowd's bodily activity was recorded with an infrared, marker-based motion capture system. In this example, it is interesting to investigate how they track the tempo of the music and which person has rhythmic movements responding to music.

6.1.1 Importing and preprocessing

First of all, the video file can be imported into Matlab using the function `mgvideoreader` returning a musical gestures data structure, i.e.,

```
mg = mgvideoreader('dance.mp4','Extract',5,25);
```

It extracts a temporal segment of the input video file from 5 seconds to 25 seconds. From the `mg` structure, we can find the video information, including file name, time index, file path, frame width, frame height, etc. In terms of the dance video, video frame with the dimensions of 1920×1080 has very high resolution. This would need lots of memory and influence the computation speed. Usually, down sampling the video with proper sampling rate is performed before further processing. This could be done by the function `mgvideosample`, i.e.,

```
mg = mgvideosample(mg,[2,4]);
```

The first parameter is the musical gestures data structure returned by `mgvideoreader`, and the second parameter means the row sampling rate and column sampling rate. After down sampling, the video is reduced to the dimensions of 480×540 , and the sampled video will be written back to the disk. Figure 25 shows the down sampled video with the heads being covered because of the permission issue.

The video can be further rotated with the function `mgvideorotate` if view angle of the video is improper or the contrast of the video can be further adjusted with the function `mgvideoadjust`. In this case, since the original video has right view angle and proper contrast, both video rotation and contrast adjustment are not required anymore. At last, recall all operations above only working on video data, one more step is to map the temporal video segment to motion capture and audio data with the function `mgmap`, i.e.,

```
mg = mgmap(mg,'Both','dance.wav','dance.c3d');
```

The input file `dance.c3d` is the motion capture data file recorded by Motion Capture system (see section 3.5). Then the musical gestures data structure `mg` contains three fields: `video`, `audio`, `mocap`.

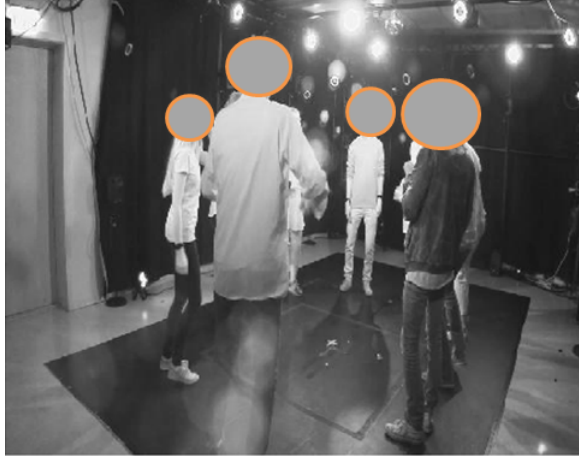


Figure 25: The down sampled video with the dimensions of 480×540 . There were nine people in scene dancing to music in a club-like environment. Every people's head had one marker. The crowd's bodily activity was recorded with an infrared, marker-based motion capture system.

Next, we apply the MG toolbox to do analysis.

6.1.2 Analysis

Quantity of motion analysis. We have constructed the musical gestures data structure *mg* storing *video*, *audio*, *mocap* data. It will be easy to compute the motiongram, QoM and CoM with the function *mgmotion*, i.e.,

```
mgmo = mgmotion(mg, 'Diff');
```

The option “*Diff*” indicates the method using absolute difference of the two successive frames, this is the exact way of computing motiongram. The function returns a new musical gestures data structure storing the computed features and motiongram, or we can use optical flow method with the function, i.e.,

```
mgop = mgmotion(mg, 'OpticalFlow');
```

Figure 26 shows the results of QoM and CoM of the dancing video. We see the periodic movements from the QoM of the dancing video. Even though the QoM here measures the overall movements of the nine persons, it still shows their regular movements in general. Figure 27 shows the results of QoM and CoM of the dancing video computed by optical flow method, in which the QoM is similar with the QoM computed by motiongram. In this case, it is not possible to analyze the individual movements only using the video data. However, recalling that the musical gestures data structure contains the mocap data as well, it is possible to perform the analysis of the individual movements of the partial body.

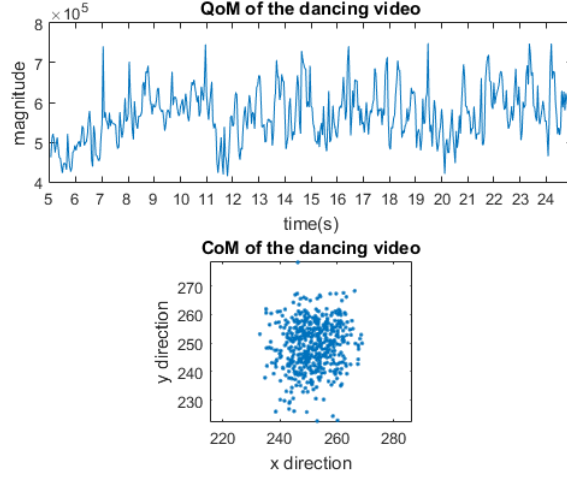


Figure 26: The QoM and CoM of dancers with 20 sec duration from 5 sec to 25 sec. QoM and CoM are computed using the function *mgmotion* with “Diff” option in MG toolbox. Here QoM of the dancing video measuring the overall movements of the nine persons shows the periodic movements of them.

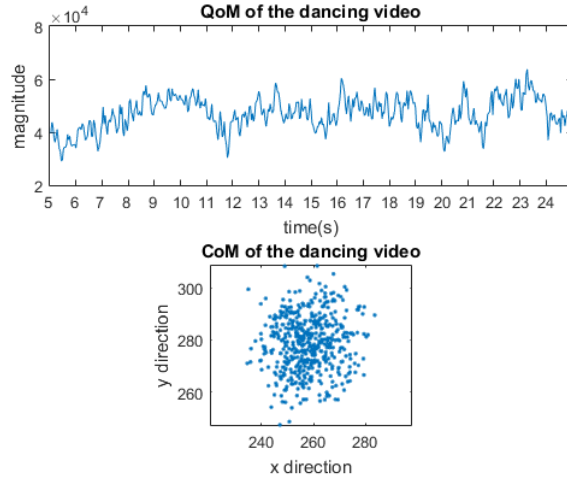


Figure 27: The QoM and CoM of dancers with 20 sec duration from 5 sec to 25 sec. QoM and CoM are computed using the function *mgmotion* with “OpticalFlow” option in the MG toolbox. Here QoM cuputed by optical flow is similar with the QoM computed by motiongram.

Rhythm analysis. With the QoM of the video, the periodicity of the movement can be further estimated using the function *mgautocor* in the MG toolbox, which is based on autocorrelation. The first non-zero lag or the highest peak of the autocorrelation function is taken as periodicity estimation. Next, periodicity estimation is performed using the QoM of the dancing video computed by the function *mgmotion* with “Diff” option. For instance,

$$[per, ac, eac, lag] = mgautocor(mgmo, 'video', 2);$$

Figure 28 shows the periodicity estimation results. From the enhanced period plot, the first non-zero lag is located at 0.5 sec, which is taken as periodicity estimation. This could be found in the returned variable *per*: 0.5017 sec as well. To see how the periodicity of the movement relates to the

music, the tempo of the audio could be computed using the function *mgtempo* in the MG toolbox. For instance,

```
tempo = mgtempo(mgmo);
```

The returned value is 122 bpm, namely around 2 beats per second, which is very close to the periodicity estimation. It is interesting to note that people tend to track the tempo of the sound while dancing to music.

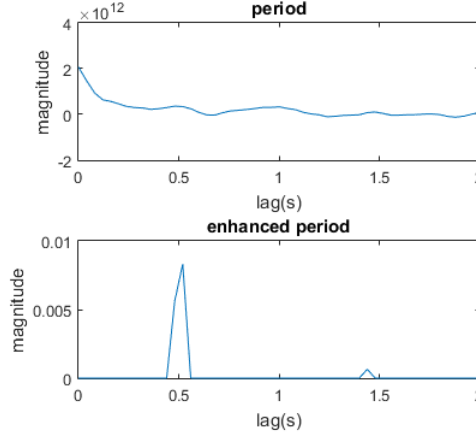


Figure 28: The periodicity estimation computed with the function *mgautocor* based on the QoM of the dancing video, which is calculated by *mgmotion* with “Diff” option. From the enhanced period plot, the first non-zero lag is located at 0.5 sec, which corresponds to the periodicity 0.5 sec. This is almost the same as the tempo of the audio, 122 bpm, namely around 2 beats per second.

One limitation is that the QoM is not possible to tell the individual movement when there are multiple objects in the video. However, one possible way to analyze the individual is to use motion capture data. With motion capture data, it is possible to investigate which person dances to music actively and rhythmically. Recall that the musical gestures data structure *mgmo* contains such data. To perform the individual periodicity estimation of the movement, the same function *mgautocor* can be used, but with “mocap” option, for instance,

```
[per,ac,eac,lag] = mgautocor(mgmo,'mocap',2);
```

The way to estimate periodicity is based on autocorrelation as well, however, the QoM of the motion capture data is calculated before applying the autocorrelation function. Because each marker is represented by a three dimensions vector, it needs to further take square norm after computing the difference between two successive frames. Figure 29 shows each person’s periodicity estimation, and the returned variable *per*=[0.4722 0.1868 0.6579 1.0130 0.4839 0.4946 0.2934 0.4895 0.7346] gives each person’s periodicity estimation. We see that person 1, person 3, person 5, person 6 and person 8 have obvious response to the tempo of the audio, suggesting that in this case they are able to move to music rhythmically. Next, we show similarity matrix of the QoM of the motion capture data, which is computed with the function *mgmocapqom*. Similarity matrix measures the similarity of the movements of the nine persons. Figure 30 shows both similarity matrix of the QoM of the mocap data and similarity matrix of the QoM spectrum of the mocap data. From the similarity matrix, we can find the similarity of each pair of persons. We see that the movements of the person 1-5 and person 7 are similar to that of person 6. It seems that person 6 acts as a dancing leader. It reflects a fact that people tend to imitate each other while dancing. It can be calculated with the function *mgsimilarity* in the MG toolbox, i.e.,

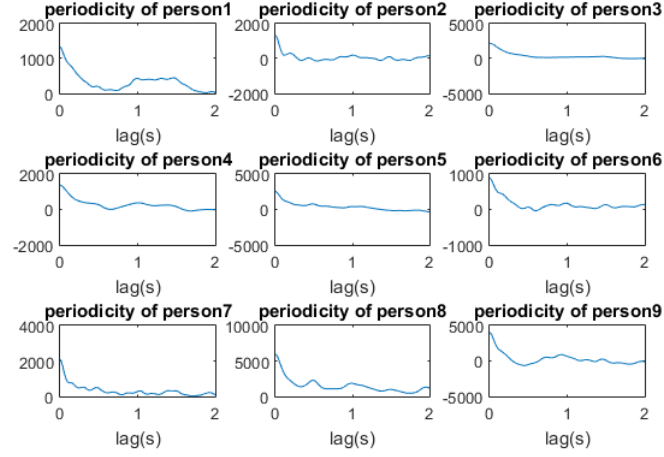


Figure 29: The periodicity estimation of each person. It shows that person 1,5,6,8 have obvious response to the tempo of the audio, and person 4 moves in roughly half of the tempo of the audio.

```
s = mgmocapqom(mgmo);
sm = mgsimilarity(s.mocap.data');
```

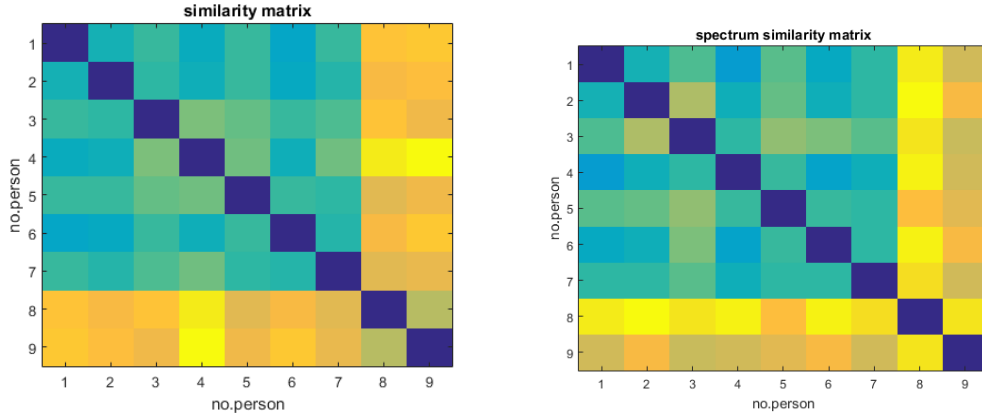


Figure 30: The left shows the similarity matrix of the QoM of the motion capture data, the right shows the similarity matrix of the QoM spectrum of the motion capture data.

6.2 Case 2

The second case is the pianist data set which was used for illustration in the *chapter 5*. This data set contains video, audio, mocap data. Pianist's performance contains several types of movements, but the most direct and important type is sound-producing movements happening when the pianist attacks the piano key. In this case, we apply the MG Toolbox to crop the video resulting in the cropped video only contains sound-producing movements, and analyze such movements relating to audio data. We show both relationship between body motion and mocap data, and relationship between body motion and audio features.

6.2.1 Importing and preprocessing

We firstly import all data with the function *mgread* and extract a temporal segment from the imported data without mapping step. To preprocess data, a temporal segment from 10 sec to 30 sec is extracted with the function *mgreadsegment*,

```
mg = mgread('Folder');  
mg = mgreadsegment(mg,10,30);
```

The function *mgread* opens a file selection dialog and allows user to select input files. In this case, the file “pianist.mp4”, “pianist.wav” and “pianist.tsv” are selected, where “pianist.tsv” is the motion capture data file recorded by Motion Capture system (*see section 3.5*). The most obvious movements of the pianist’s performance are sound-producing movements, and they could be analyzed by cropping such region of interest with the function *mgvideocrop* returning a new musical gestures data structure *mgcrop*, i.e.,

```
mgcrop = mgvideocrop(mg);
```

The function *mgvideocrop* allows user to select the region of interest if the cropping position vector is not given. Or, if prior knowledge about the spatial information of the region of interest is known, the cropping position vector could be given directly, i.e.,

```
mgcrop = mgvideocrop(mg,[100 450;200 350]);
```

If necessary, the video could be magnified with the function *mgvideomagnify*. This function aiming to magnify and reveal subtle movements often work on small variations of the video. The pianist video has large variation of the movements, which would be sensitive to noise when using this function.

```
mgmag = mgvideomagnify(mg,'Butter',3.6,6.2,60,90,30,0.2);  
mgcropmag = mgvideomagnify(mgcrop,'Butter',3.6,6.2,60,90,30,0.2);
```

6.2.2 Analysis

Quantity of motion analysis. Now all preprocessed data are stored in the musical gestures data structures: *mg*, *mgcrop* (contains the cropped video data), *mgmag* (contains magnified video data). Then the QoM and motiongram can be computed using the function *mgmotion* with “Diff” or “OpticalFlow” option, corresponding to motiongram method and optical flow method, respectively. The calculations based on the data structure *mg* are done like this,

```
mgmo = mgmotion(mg,'Diff');  
mgop = mgmotion(mg,'OpticalFlow');
```

The returned musical gestures data structure *mgmo* contains the computed motiongram, QoM, CoM, AoM, whereas *mgop* contains the computed optical flowgram, QoM, CoM. Figure 31 shows the QoM computed by motiongram and optical flow method. It is obvious to note that the whole pianist’s performance consists of a set of repeatable movements. Both methods give the similar results in terms of the movement trajectories. It might be interesting to see the QoM of the cropped video with only the pianist’s hands, which is shown in figure 32. This gives almost the same movement trajectories as well.

The results could be filtered as well. The filtered results are shown in figure 33. In the MG toolbox, the function *mgmotionfilter* could help filter the results. The spatial temporal information of the movement can be observed in the motiongram, which is shown in figure 34. It is clear to see the

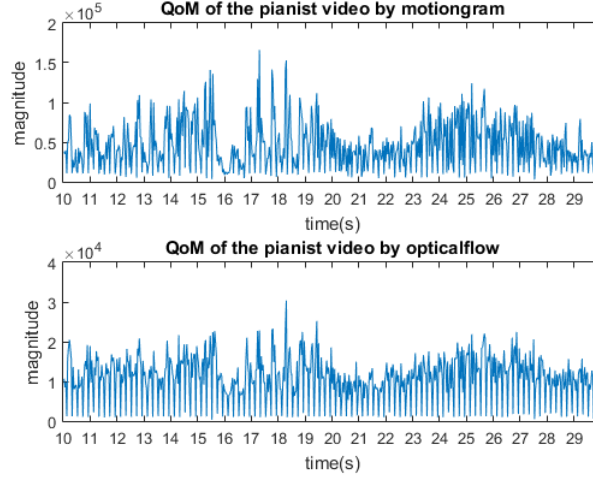


Figure 31: The QoM of the pianist video from 10 seconds to 30 seconds. The top was computed using the function *mgmotion* in the MG toolbox with “Diff” option, and the bottom was computed with “OpticalFlow” option. We see that both give the similar results.

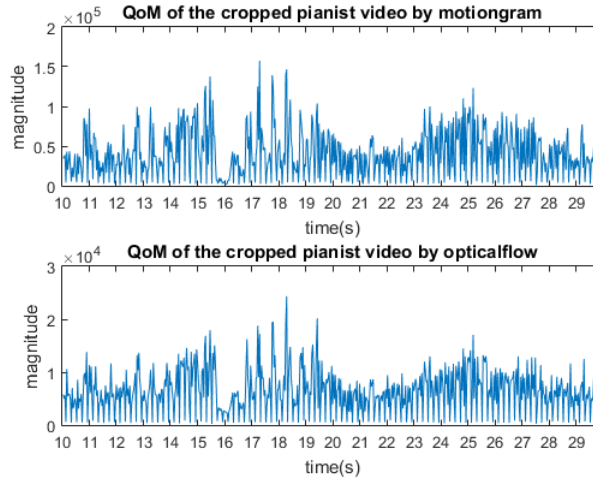


Figure 32: The QoM of the cropped pianist video from 10 seconds to 30 seconds. The top was computed using the function *mgmotion* in the MG toolbox with “Diff” option, and the bottom was computed with “OpticalFlow” option. The QoM of the cropped video only measures the movement of the pianist’s hands. We see that both give almost the same movement pattern.

repeatable movement trajectories. From the vertical motiongram, the movement of the local objects (the pianist’s hands) could be observed, including regularly repeatable movements, a break between 16 seconds and 17 seconds, and the pianist’s left hand playing the piano from 19 seconds to 23 seconds.

Figure 35 shows the two types of optical flowgrams, which are similar as the results of the motiongram, but a little bit sensitive to noise. This is probably because the optical flow approach is more sensitive to illumination change. To get better results, the video need be further preprocessed, for instance, denoising the illumination effects or the filter is applied during computing optical flow field.

To analyze the sound-producing movements, the motiongrams computed from the cropped pianist video only shows the movements of the pianist’s hands, which are shown in figure 36. Here we could also combine the similarity matrix of audio spectrum so that it is possible to compare sound with motion. The motiongrams of the cropped video could be able to show only the movements of pianist’s

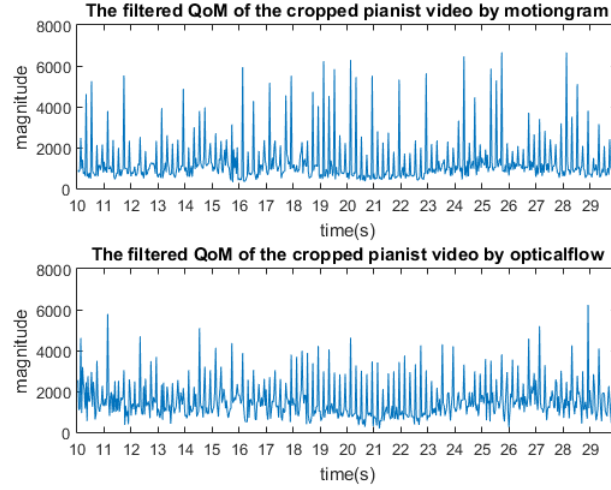


Figure 33: The filtered QoM computed with the function *mgmotion* in the MG toolbox as well, but a binary filter with threshold 0.2 is applied during computing, which the value above the threshold is converted to 1, below the threshold is converted to 0. It clearly shows the repeatable movement pattern.

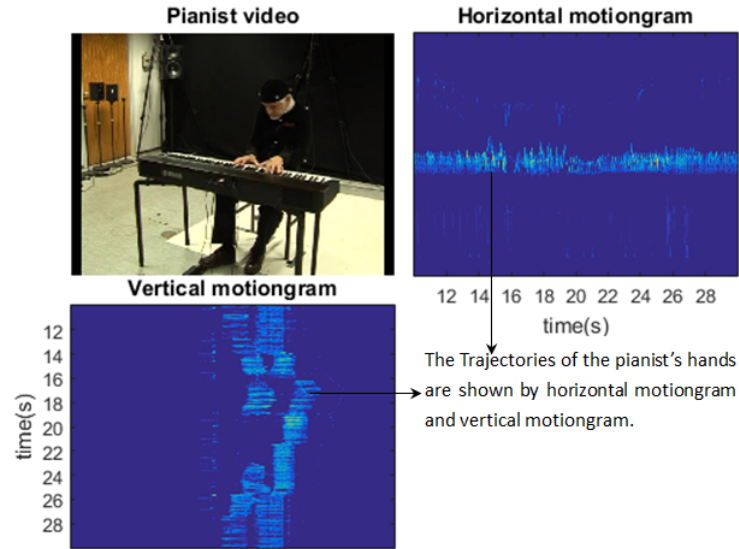


Figure 34: Two types of motiongrams are computed with the function *mgmotion* with “Diff” option in the MG toolbox. From the vertical motiongram, a break can be noted from 16 seconds to 17 seconds, and from 19 seconds to 23 seconds, there is only the pianist’s left hand playing the piano. From the horizontal motiongram, a series of repeatable actions could be observed during performance.

hands. The horizontal motiongram indicates the continuous attacks in the hands, while the vertical motiongram displays the horizontal movements of the hands.

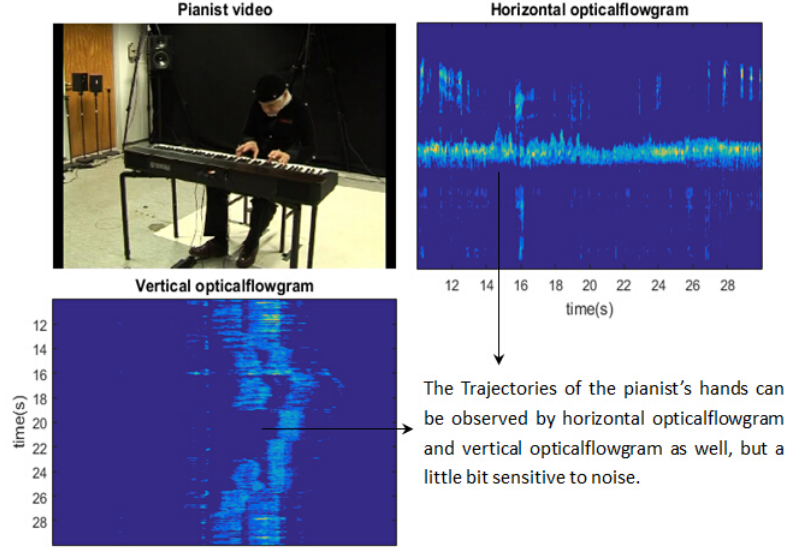


Figure 35: Two types of opticalflowgrams are computed with the function *mgmotion* with “OpticalFlow” in the MG toolbox. From the vertical optical flowgram, a break can be noted from 16 seconds to 17 seconds, and from 19 seconds to 23 seconds, there is only the pianist's left hand playing the piano. From the horizontal optical flowgram, a series of repeatable actions could be observed as well. They give the similar results as motiongram, but a little bit sensitive to noise.

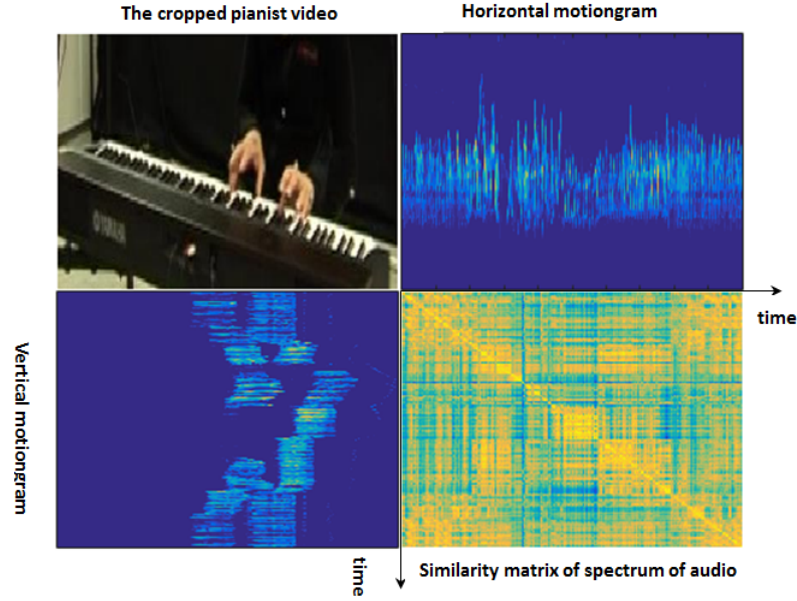


Figure 36: Two types of motiongrams are computed with the function *mgmotion* with “Diff” option in the MG toolbox based on the cropped pianist video. Bottom right shows the similarity matrix of spectrum of the audio. The motiongram of the cropped pianist video clearly shows only sound-producing movements while the pianist's hands attack the piano keys.

Rhythm analysis. The same as the first case, the periodicity of the movements is estimated by computing the autocorrelation of the QoM of the video data with the function *mgautocor*,

$$[per, ac, eac, lag] = mgautocor(mgmo, 'video', 2);$$

Figure 37 shows the periodicity estimation of the movement. It is given by the location of the first non-zero lag, which is 0.2 sec. This result can be found by computing the spectrum of the QoM

of the video as well. Figure 38 shows the spectrum of the QoM. The fundamental frequency of the QoM is given by 5Hz, which means the periodicity of QoM is 0.2 sec.

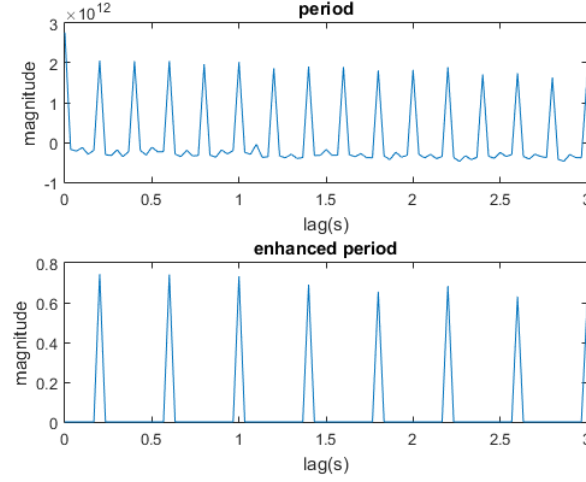


Figure 37: The periodicity estimation of the movements based on QoM of the video data. The first non-zero lag is around 0.2 sec. It is computed by the `mgautocor` in the MG toolbox.

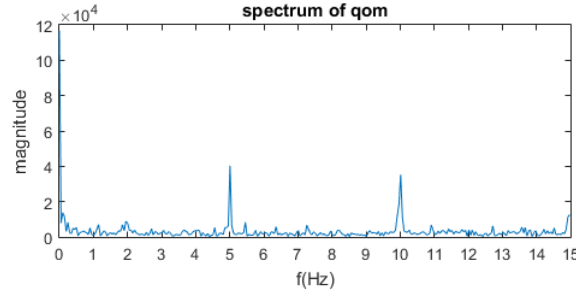


Figure 38: The spectrum of the QoM of the pianist video. It may be interesting to note that the peak at 5Hz, corresponds to the periodicity of 0.2 sec.

Relationship between body motion and mocap. Next, we analyze the association between body motion and mocap data. Similarly, the QoM of mocap data could be computed with the function `mg-mocapqom` as well. Figure 39 shows the QoM of the mocap data. The red line indicates the smoothed QoM. Figure 40 shows the QoMs of the video and mocap data. It can be observed from the plots that both QoMs are highly related. Since the QoM of the mocap data only measures the movements of the partial body, it may not represent the overall movements of the body, and it highly depends on the positions of the markers put on the body. In this case, 23 markers were put on the pianist, almost including all the key joints of the whole body, which could provide reliable motion capture analysis. This is also why both QoMs give the similar trajectories, and it proves also that the video analysis combined with the motion capture data is accurate and reliable. In addition, one significant capability of the motion capture system is that it is possible to provide 3D tracking of the body motion, which is also a significant difference from the 2D image domain.

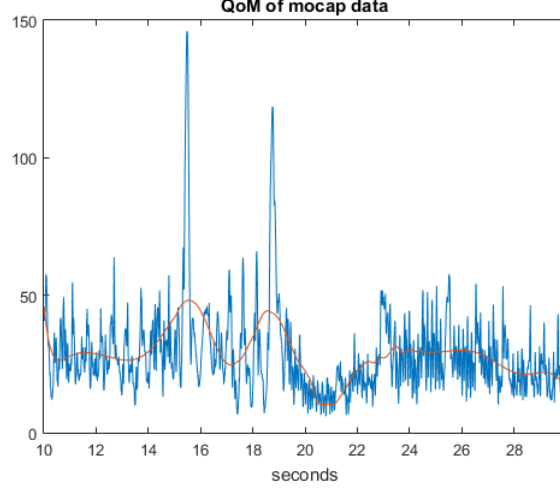


Figure 39: The QoM of mocap data. The red line shows the QoM filtered by Savitzky-Golay filter.

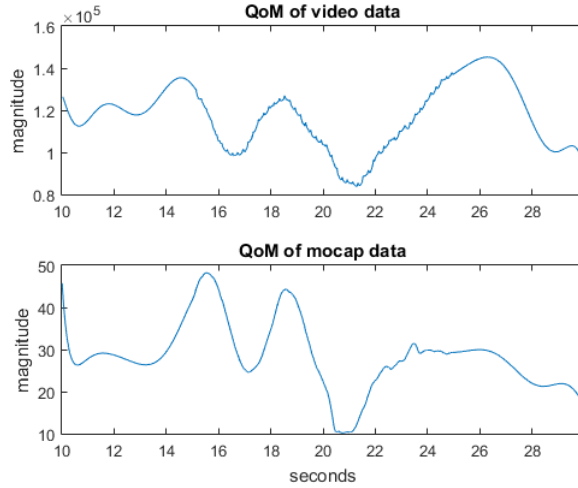


Figure 40: The above shows the QoM of video data, the bottom shows the QoM of mocap data.

Relationship between body motion and audio features. In order to investigate the associations between body motion and audio features, the audio data will be preprocessed to align timely with video. This operation results in frame-decomposed audio with frame length $\frac{1}{30}$. Here the frame length is set according to the frame rate of the video recording. We have mentioned some related audio features in *section 5.3.5*. Flux can measure the distance between two successive frames. The spectrum flux gives the distance between the spectrum of the two successive frames, which is similar as the way of computing QoM of the video. Figure 41 shows the waveform, spectrum flux of the frame-decomposed audio. In particular, the peaks and valleys of both spectrum flux plot and QoM of the video happen at almost the same time. Figure 42 shows the smoothed QoM and spectrum flux. We see how the spectrum flux changes according to the QoM, particularly, from 10 seconds to 21 seconds.

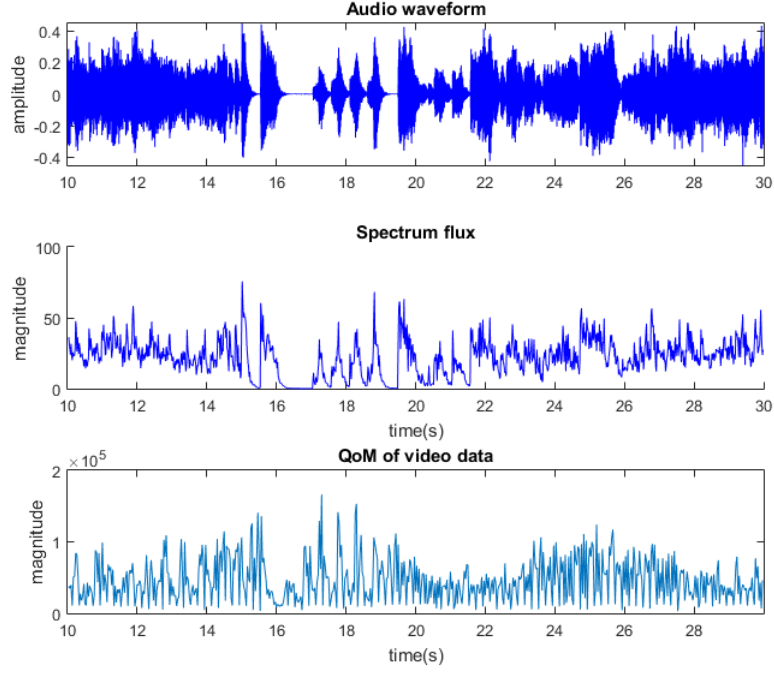


Figure 41: The top is waveform of the audio, the middle is spectrum flux, the bottom is QoM of the pianist video computed by motiongram method.

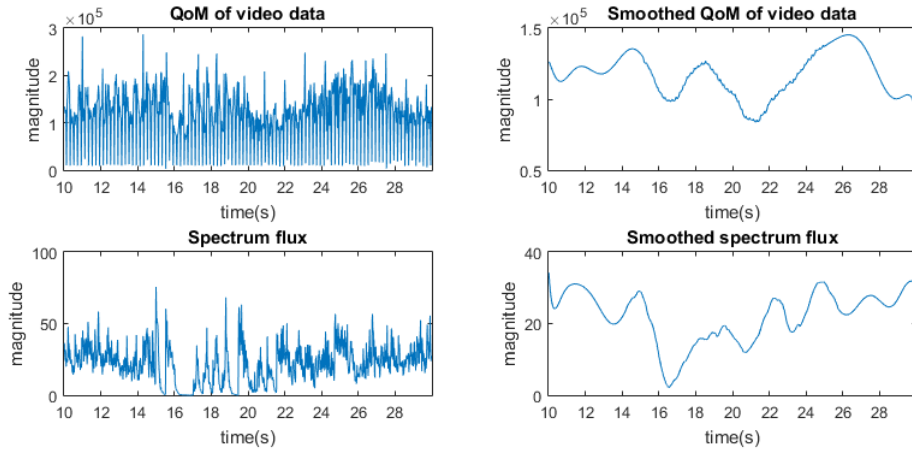


Figure 42: The top left shows the QoM, the top right shows the smoothed QoM by Savitzky-Golay filter. The bottom left show spectrum flux of the audio, the bottom right shows the smoothed spectrum flux.

Furthermore, RMS is computed based on the frame-decomposed waveform. Figure 43 shows the audio waveform, RMS and QoM of the video. We see how the RMS changes according to the QoM of the video.

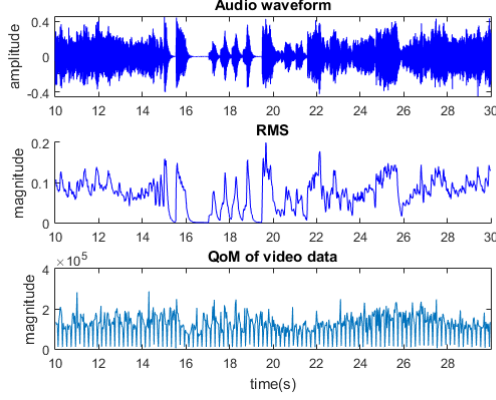


Figure 43: The top is the waveform of the audio, the middle is the RMS of the frame-decomposed audio, the bottom is QoM of the video.

6.3 Discussion

In the previous two sections, we have applied the developed MG toolbox to perform the video analysis of music-related body motion by showing the two cases. We have seen how people relate to music. In the first case, in which corresponding to sound-accompanying movements, we first preprocessed the video data by the functions provided in the MG Toolbox. Then the quantity of motion analysis was performed based on motiongram and optical flow method. We saw that both methods could provide similar results in terms of the QoM. However, we have to admit that motiongram itself has limitation when multiple objects are moving in a scene. In such situation, we could use motion capture data to do analysis. We estimated also the tempo of the audio, which was close to the periodicity estimation based on the QoM of the video. It is interesting to note that most persons were able to track the musical tempo. In addition, similarities of the movements of these persons were estimated by similarity matrix. The analysis supported our basic idea behind music-related motion. This case has shown that people tend to track musical features, especially in a club-like environment. As a matter of fact, this is true in other scenes.

In the second case, we studied sound-producing movements. We analyzed the relationship between the QoM of the video and QoM of the mocap data, and how the QoM relates to musical features (RMS, spectrogram, spectrogram flux). It has been shown that the peaks and valleys of QoM had the most obvious responses to the musical features. In this case, the motion analysis based on the preprocessed video could provide more precise results with respect to the motiongram and QoM. We showed also how the motiongram or optical flowgram was used to analyze the music-related body motion. From the motiongram or optical flowgram, it is possible to observe the types of the movements in a musical context. Both motiongram and optical flow have its respective advantage of motion analysis. The motiongram has very low computation cost in comparison to optical flow, whereas the optical flow provides orientation estimation of the body motion. In the MG toolbox, we have implemented both techniques as well as the EVM providing users the flexible options. At last, even though audio, video, and mocap data represent the different characteristics, they all have close relationships. Since the music-related motion itself involves multi-data processing, this is also why we choose the audio, video, and mocap data to study the music-related motion.

Chapter 7 Conclusion and future work

7.1 Conclusion

This thesis mainly presented the developed “Musical Gestures (MG) Toolbox”. The main idea behind the toolbox is to integrate several existing toolboxes and provide music-related body motion analysis based on video. The main goal is to understand more about music-related motion and how body motion enhances our perception of music. In this thesis, we have described in detail the design and implementation of the MG toolbox, as well as application of the toolbox. The MG toolbox has been developed for video analysis of music-related body motion by using motiongram, optical flow method in this project. In addition, the toolbox provides some powerful importing and preprocessing functionalities, such as reading a single data stream or any combination of three input data (video, audio/music, mocap data) and extracting a temporal segment of them, and cropping, rotating, magnifying a video, adjusting the contrast of a video for preprocessing, etc. The functionalities of feature extraction and statistical descriptors have been implemented in the toolbox, as such, providing the quantitative analysis of music-related motion. The developed toolbox contains also two types of visualization tools, plotting the motiongram, motion image and mocapgram in a single figure, and plotting audio features and QoM of a video in another figure as well. These visualization tools could directly give visual representation of music-related motion. The MG toolbox has also integrated MIR toolbox and MoCap toolbox so that the integrated toolbox could provide audio or music analysis as well as the analysis and visualization of the motion capture data. More importantly, the integrated toolbox could perform the analysis of music-related motion based on the multi-data streams. This is whichever the MIR toolbox or MoCap toolbox could not perform.

We have introduced how to use the toolbox to perform video analysis of music-related motion by illustrating the two cases in *chapter 6*, which mainly include two types of movements: sound-producing movements and sound-accompanying movements. It has been shown that these two types of movements have close relationship with music. We did quantity of motion (QoM) analysis, and estimated periodicity of the movement based on QoM. We showed also the QoM of the video computed by both motiongram and optical flow. Even though both methods give the similar QoMs, the motiongram method has lower computation cost than that of optical flow method. The optical flow method has its advantage of accurately estimating the orientation of the movement of an object. To show the spatial temporal information of the movement of the object, both motiongram and optical flowgram were shown. We see that both of them could be able to reveal the trajectory of the object. To investigate what the relationships between the video data and the mocap data or audio data are, the QoM of the video and the QoM of the mocap data, or RMS and spectrogram flux were shown as well. All the analysis support our idea that video analysis of music-related motion could also provide accurate, reliable analysis. In addition, video analysis is the most low cost approach comparing with using a Motion Capture system. With respect to this point, it reflects the value of the MG toolbox.

However, motiongram and optical flowgram could not work well when multiple objects are moving in a scene. In such situation, we could use the mocap data to do analysis. The MG toolbox provides additional functions to compute the QoM of the mocap data and the similarity matrix. In case 1, we have shown that Motion Capture system makes it possible to perform individually motion analysis of parts of the body, and estimate the similarities between them. Furthermore, the periodicity of the movement of each person was estimated based on mocap data. It is interesting to note that the periodicity of the movement is highly related to the tempo of the music. It says that people tend to track musical features when they are moving to music. Combined with mocap and video data, the MG toolbox is able to perform comprehensive analysis of music-related motion.

At last, getting back to the questions proposed at the beginning of the thesis, we have answered these questions in different chapters, i.e., the answer of the question: *how to develop the new toolbox*

could be found in *chapter 4* and *chapter 5* by introducing the design and implementation of the MG toolbox. In *chapter 5*, we have described various features extracted from the video, which could be used to quantitatively measure motion. We have also introduced various terminologies and research methods to help us better understand music-related motion in *chapter 2*.

7.2 Future work

This thesis has explored theoretically and technologically the MG toolbox development. As discussed in this thesis, music is a movement based phenomenon. It consists of both sides, music generating movements and movements creating music. We can apply the MG toolbox to analyze music-related motion. Conversely, it might be interesting to study motion-controlled musical sound generation. Pattern recognition techniques make it possible to map motion to sound [27, 28]. As such, we could understand music and motion from both directions. In addition, there are still many open questions left. One interesting question is how to explore higher level features from video, which can represent more useful information to analyze music-motion correspondences. The scale-invariant features (SIFT) [29] could be good solutions. We have shown that eulerian video magnification is a useful technique to preprocess the videos, but it often works on the videos which have small variations. To improve its performance, phase based eulerian video magnification [16, 24] might be a promising approach. Even though it has higher computation overhead than the eulerian video magnification, it supports large amplification factors which can work on the video with large variations. It would be interesting to investigate how the emotion relates to music by combining phase based eulerian video magnification and motiongram.

Another important question is how to apply machine learning algorithms to solve classification problem of music-related motion [30]. This is a complex question. Many machine learning algorithms could be applied on different data streams, depending on the classification task, for instance, musical style classification on audio data, movement style on the video or motion capture data or combined both of them [31]. Such algorithms include Bayesian classifier, linear classifier, and neural networks, etc. However, to classify movements becomes difficult. This is simply because significantly different movements may have the same meaning. If our research limits in the musical context, classification of movement style becomes less complicated.

In the current vision of the MG toolbox, it still has some error handling problems unsolved, for instance, when improper parameters are passed in functions, no clear error messages are returned in some functions. The error handling will be improved in the future.

Lastly, visualization techniques are important to analyze music-related motion as well. For this issue, it can continue studying more advanced motiongram techniques for visualization. Three-dimensional motiongrams might be a good solution, and the sonomotiongram will be created by taking the inverse of Fourier transform of the motiongram. In the future work, some additional functionalities will be also added in the toolbox, i.e., some functions for creating the motion history images and average motion images.

References

- [1] Birgitta Burger and Petri Toiviainen. Mocap toolbox—a matlab toolbox for computational analysis of movement data. In *Proceedings of the Sound and Music Computing Conference 2013, SMC 2013, Logos Verlag Berlin, Stockholm, Sweden. ISBN 978-3-8325-3472-1*. Logos Verlag Berlin, 2013.
- [2] Olivier Lartillot and Petri Toiviainen. A matlab toolbox for musical feature extraction from audio. In *International Conference on Digital Audio Effects*, pages 237–244, 2007.
- [3] Alexander Refsum Jensenius, Rolf Inge Godøy, and Marcelo M Wanderley. Developing tools for studying musical gestures within the max/msp/jitter environment. 2005.
- [4] Ståle Andreas van Dorp Skogstad, Alexander Refsum Jensenius, and Kristian Nymoen. Using ir optical marker based motion capture for exploring musical interaction. 2010.
- [5] EP Simoncelli. matlabpyrtools—matlab source code for multiscale image processing. *Matlab, Natick, MA*, 2003.
- [6] Richard Sennett. *The fall of public man*. WW Norton & Company, 1992.
- [7] Alexander Refsum Jensenius. Action-sound: Developing methods and tools to study music-related body movement. 2007.
- [8] Esther Thelen. Time-scale dynamics and the development of an embodied cognition. *Mind as motion: Explorations in the dynamics of cognition*, pages 69–100, 1995.
- [9] Ann Hutchinson Guest. *Your move: A new approach to the study of movement and dance*, volume 2. Psychology Press, 1983.
- [10] Alexander Refsum Jensenius, Marcelo M Wanderley, Rolf Inge Godøy, and Marc Leman. Musical gestures. *Musical gestures: Sound, movement, and meaning*, 12, 2009.
- [11] Brian Clegg. *The Man Who Stopped Time:: The Illuminating Story of Eadweard Muybridge-Pioneer Photographer, Father of the Motion Picture, Murderer*. Joseph Henry Press, 2007.
- [12] Marta Braun. *Picturing time: the work of Etienne-Jules Marey (1830-1904)*. University of Chicago Press, 1995.
- [13] Gunnar Johansson. Visual perception of biological motion and a model for its analysis. *Attention, Perception, & Psychophysics*, 14(2):201–211, 1973.
- [14] Berthold K Horn and Brian G Schunck. Determining optical flow. In *1981 Technical symposium east*, pages 319–331. International Society for Optics and Photonics, 1981.
- [15] Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John V Guttag, Frédo Durand, and William T Freeman. Eulerian video magnification for revealing subtle changes in the world. 2012.

- [16] Michael Rubinstein. *Analysis and visualization of temporal variations in video*. PhD thesis, Citeseer, 2013.
- [17] Alexander Refsum Jensenius. Some video abstraction techniques for displaying body movement in analysis and performance. *Leonardo*, 46(1):53–60, 2013.
- [18] Alexander Refsum Jensenius. Motion-sound interaction using sonification based on motiongrams. 2012.
- [19] Alexander Refsum Jensenius. From experimental music technology to clinical tool. 2014.
- [20] Ramesh Jain, Rangachar Kasturi, and Brian G Schunck. *Machine vision*, volume 5. McGraw-Hill New York, 1995.
- [21] Roberto Battiti, Edoardo Amaldi, and Christof Koch. Computing optical flow across multiple scales: an adaptive coarse-to-fine strategy. *International Journal of Computer Vision*, 6(2):133–145, 1991.
- [22] Harald Kirkerød. Optical flow applied to infant movement. 2010.
- [23] Chi-Cheng Cheng and Hui-Ting Li. Feature-based optical flow computation. *International Journal of Information Technology*, 12(7):82–90, 2006.
- [24] Neal Wadhwa, Michael Rubinstein, Frédo Durand, and William T Freeman. Phase-based video motion processing. *ACM Transactions on Graphics (TOG)*, 32(4):80, 2013.
- [25] http://en.wikipedia.org/wiki/motion_capture.
- [26] Fritz Albregtsen et al. Statistical texture measures computed from gray level cooccurrence matrices. *Image processing laboratory, department of informatics, university of oslo*, pages 1–14, 2008.
- [27] Frédéric Bevilacqua, Jeff Ridenour, and David J Cuccia. 3d motion capture data: motion analysis and mapping to music. In *Proceedings of the workshop/symposium on sensing and input for media-centric systems*, 2002.
- [28] Gang Qian, Feng Guo, Todd Ingalls, Lowell Olson, Jody James, and Thanassis Rikakis. A gesture-driven multimodal interactive dance system. In *Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on*, volume 3, pages 1579–1582. IEEE, 2004.
- [29] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.
- [30] Rolf Inge Godøy, Alexander Refsum Jensenius, Arve Voldsund, Kyrre Harald Glette, Mats Erling Høvin, Kristian Nymoen, Ståle Andreas van Dorp Skogstad, and Jim Tørresen. Classifying music-related actions. 2012.
- [31] Roger B Dannenberg, Belinda Thom, and David Watson. A machine learning approach to musical style recognition. 1997.

APPENDIX

Function table

Function Table		
	Function	Synopsis
Import	mgread	Import three types media data
	mgvideoreader	Import or extract a temporal segment of video
	mgreadsegment	Import or extract a temporal segment of three types media data
Preprocess	mgvideoadjust	Adjust the contrast of the video
	mgvideocrop	Crop the region of interest of the video
	mgvideocat	Concatenate two videos
	mgvideorotate	Rotate a video by a specific angle
	mgvideomagnify	Magnify small variations in a video
	mgvideofilter	Filter a video
	mgmotionfilter	Filter a motion image or optical flow field
	mgmap	Map the temporal video to other two types of media data
Analysis	mgmotion	Compute motiongram or optical flow field
	mgstatistics	Compute first order or second order features
	mgautocor	Estimate the periodicity
	mgpca	Perform principle component analysis
Visulization	mgvideoplot	Plot motiongram and motion image over time
	mgwaveplot	Plot waveform and RMS and their spectrums
Other	mgsave	Save the musical gestures structure
	mginitstruct	Initiate a musical gestures structure
	mgopticalflow	Helpful function for computing Qom and Com by optical flow
	mgcentroid	Compute Qom and Com by motion image
	ComputeFlow	Compute optical flow field
	Gradient	Compute dx,dy,dt
	gaussgen	Generate a gauss kernel

Function reference

mginitstruct

synopsis

Initializes Musical Gestures data structure.

syntax

```
mg = mginitstruct;
```

output

mg:musical gestures data structure with three fields, audio, video, and mocap.

examples

```
mg = mginitstruct;
```

mgread

synopsis

Reads any one of three types of data file,mocap,video,audio data file and returns a musical gestures data structure.

syntax

```
mg = mgread('Video');  
mg = mgread('Audio');  
mg = mgread('Mocap');  
mg = mgread('Folder');
```

input parameters

Video, Audio, and Mocap indicate three types of input data. With any one of them,a file open dialog opens and provides user to choose corresponding input file. With option folder, mgread reads any combinations of three types of data. For instance, it could only read Mocap and Video,etc.

output

mg:musical gestures data structure with corresponding field filled by parameters and data.

examples

comments

Currently mgread provides .avi,.mp4,.m4v,.mpg,.mov file formats.

also see

mgvideoreader

mgvideoreader

synopsis

Reads specified video file with or without extracting video and returns a data structure containing the video parameters and data. If input is a musical gestures data structure containing the video data, mgvideoreader will extract the section according to the extraction parameters.

syntax

```
mg = mgvideoreader(filename);  
mg = mgvideoreader(mg);  
mg = mgvideoreader(filename,'Extract',starttime);  
mg = mgvideoreader(filename,'Extract',starttime,endtime);  
mg = mgvideoreader(mg,'Extract',starttime,endtime);
```

input parameters

filename:specifies the name of the video file
mg:musical gestures data structure containing the video data
Extract:indicates that mgvideoreader reads a certain segment from the video
starttime:start of extracted segment
endtime:end of extracted segment

output

mg:a struct data structure containing the video frames from starttime to endtime and other video parameters

examples

```
mg = mgvideoreader(filename);  
mg = mgvideoreader(filename,'Extract',10,15) mg = mgvideoreader(mg,'Extract',10,15);
```

comments

also see

mgreadsegment

mgreadsegment

synopsis

Extracts a temporal segment from the video, audio, mocap data file.

syntax

```
mg = mgreadsegment(filename);  
mg = mgreadsegment(filename,starttime,endtime);
```

input parameters

filename:can be any type of three different input data files
starttime:start of extracted segment
endtime:end of extracted segment

output

mg:a struct containing video, or audio, or mocap data from starttime to endtime

examples

```
mg = mgreadsegment(videofile);  
mg = mgreadsegment(audiofile);  
mg = mgreadsegment(mocapfile);  
mg = mgreadsegment(videofile,10,15);  
mg = mgreadsegment(audiofile,10,15);  
mg = mgreadsegment(mg,10,15);  
mg = mgreadsegment(mocapfile,10,15);
```

comments

Filename.tsv is a type of the mocap data file. Other types of mocap data file can be filename.c3d,filename.mat,filename.wii.

also see

mgread,mgvideoreader,mctrim,miraudio

mgvideocrop

synopsis

Crops video recording according to the position given by user. If the position is not given in the function, it will show first frame of video, user can draw any shape, any region of interest, then

function will create the mask of this area in every frame. Finally it writes back the cropped video into disk and returns a musical gestures of cropped video.

syntax

```
mg = mgvideocrop(mg,pos);
mg = mgvideocrop(mg);
mg = mgvideocrop(filename,pos);
mg = mgvideocrop(filename);
```

input parameters

mg:musical gestures data structure contains video parameters

pos:2 by N position matrix,first row indicates the index of column of the vertex;second row indicates the index of row of the vertex,N is the number of vertexes

filename:if the input is a video, the name of video should be given

output

mg:a musical gestures data structure contains the cropped video parameters

examples

```
mg = mgvideocrop(mg,[20 30 30 20;30 30 40 40]);
mg = mgvideocrop(filename,[20 30 30 20;30 30 40 40]);
mg = mgvideocrop(filename);
mg = mgvideocrop(mg);
```

mgvideorate

synopsis

Rotates the video by angle degrees in a counterclockwise around its center point. To rotate the video clockwise, specify a negative value for angle. It will write the rotated video in the disk and return a musical gestures data structure contains the rotated video.

syntax

```
mg = mgvideorate(file,angle);
mg = mgvideorate(file,angle,method);
mg = mgvideorate(mg,angle);
mg = mgvideorate(mg,angle,method);
```

input parameters

file:the video file which needs to be rotated

mg:musical gestures data structure contains a video information

angle:specify the angle of rotation, positive value rotates in counterclockwise;Negative value rotates in clockwise

method:gives the interpolation method

output

mg:a musical gestures data structure contains rotated video

examples

```
mg = mgvideorate(file,5,'bilinear','crop');
mg = mgvideorate(file,5,'bilinear');
mg = mgvideorate(mg,5,'bilinear');
mg = mgcrop(mg,-5,'bilinear','crop');
```

comments

also see

imrotate

mgvideoadjust

synopsis

Adjust the contrast of the video,for RGB video, mapping value of three channels must be given. Finally, it writes the adjusted video in the disk and returns a musical gestures data structure which contains the adjusted video.

syntax mg = mgvideoadjust(mg);
mg = mgvideoadjust(file);
mg = mgvideoadjust(file,mapin,mapout);

input parameters

file:the name of video file

mg:a musical gestures data structure contains the video

mapin:a matrix contains range of mapping in, for instance, [*low_in*; *high_in*] for graylevel video; [*low_in1*, *low_in2*, *low_in3*] for RGB video

mapout:a matrix contains range of mapping out, for instance, [*low_out*; *high_out*] for graylevel video; [*low_out1*, *low_out2*, *low_out3*; *high_out1*, *high_out2*, *high_out3*] for RGB video

output mg:musical gestures data structure contains the adjusted video

examples

```
mg = mgvideoadjust(file,[0.1;0.8],[0.2;0.9]);  
mg = mgvideoadjust(file,[0.1 0.2 0.05;0.8 0.9 1],[0.3 0.4 0.2;0.9 0.8 0.9]);  
mg = mgvideoadjust(mg,[0.1;0.8],[0.2;0.9]);  
mg = mgvideoadjust(mg,[0.1 0.2 0.05;0.8 0.9 1],[0.3 0.4 0.2;0.9 0.8 0.9]);
```

comments

also see

imadjust

mgvideocat

synopsis

Concatenates two videos into one,finally writes the concatenated video in the disk and returns a musical gestures data structure which contains concatenated video.

syntax

```
mg = mgvideocat(file1,file2);  
mg = mgvideocat(mg1,mg2);
```

input parameters

file1,file2:the name of video file

mg1,mg2:musical gestures data structures contain the videos

output

mg:a musical gestures data structure contains the concatenated video

examples

```
mg = mgvideocat(file1,file2);  
mg = mgvideocat(mg1,mg2);
```

comments

also see

mgvideomagnify

synopsis

Magnify the subtle variations of a video which are difficult or impossible to see by naked human eyes, providing two types of temporal spatial filters:IIR,Butter. This function requires matlabPyrttools toolbox.

syntax

```
mg = mgvideomagnify(videofile,'IIR',f1,f2,alpha,lambda_c ,attenuation);  
mg = mgvideomagnify(videofile,'Butter',f1,f2,alpha,lambda_c ,samplingrate,attenuation)
```

input parameters

videofile: input video file
med: filter method with option 'IIR','Butter'
f1,f2: coefficients of second-order filter
alpha: magnification factor
lambda_c : spatial cutoff frequency
samplingrate: spatial resolution
attenuation: attenuation factor

output mg: musical gestures data structure containing magnified video

examples

```
mg = mgvideomagnify('video.mp4','IIR',0.4,0.05,10,16,0.1);  
mg = mgvideomagnify('video.mp4','Butter',3.6,6.2,60,90,30,0.3);
```

comments

also see

mgvideofilter

synopsis

Filter a video with 'Temporal', 'Spatial', 'Both' options. With 'Temporal' option, perform temporal filter; with 'Spatial' option, perform spatial filter by median, or average filter.

syntax mg = mgvideofilter(videofile,type,med,h);

input parameters

videofile: input video file
type: filter type
med: filter method
h: size

output mg: musical gestures data structure containing filtered video

examples

```
mg = mgvideofilter(videofile,'Spatial','Median',[3,3]);  
mg = mgvideofilter(videofile,'Spatial','Average',[3,3]);
```

comments

also see

mgmotionfilter

synopsis

Filter the motion image or optical flow field with 'Regular', 'Binary', 'Blob' option.

syntax

```
mg = mgmotionfilter(f,med,thres);
```

input parameters

f: input motion image or optical flow field
med: 'Regular', turns all values below thres to 0; 'Binary' truns all values below thres to 0, above thres to 1; 'Blob' removes individual pixels with erosion method.
thres: for 'Regular', 'Binary' option, thres is a value of threshold [0,1]; for 'Blob', thres is element structure created by strel.

output

mg: musical gestures data structure containing the filtered video.

examples

```
mg = mgmotionfilter(f,'Regular',0.2); mg = mgmotionfilter(f,'Blob',[1 1 1;1 1 1;1 1 1]);
```

comments

also see

mgmap

synopsis

Maps the same section from audio and mocap data file according to the parameters of video data file.

syntax

```
mg = mgmap(mg,'Audio');  
mg = mgmap(mg,'Audio',filename);  
mg = mgmap(mg,'Mocap',filename);  
mg = mgmap(mg,'Both',filename);  
mg = mgmap(mg,'Both',filename,filename);
```

input parameters

mg:musical gestures data structure containing parameters of video file. Such parameters contain the specified duration, starttime, endtime, etc

type:Audio,Mocap indicate the type of mapping data

filename:if map the mocap data file, then the filename of mocap file must be given;the filename of audio is not necessary to be given. The first filename is audio name, the second is mocap name

output

mg:returns a musical gestures data structure containing the same segment from video, audio, mocap

examples

```
mg = mgmap(mg,'Audio');  
mg = mgmap(mg,'Audio',audiofilename);  
mg = mgmap(mg,'Mocap',mocapfilename);  
mg = mgmap(mg,'Both',audiofilename,mocapfilename);
```

comments

also see

mgmotion

synopsis

Computes the motion, motion gram, quantity of motion, centroid of motion according to the method given by user.

syntax

```
mg = mgmotion(mg,'Diff');  
mg = mgmotion(mg,'OpticalFlow');
```

```
mg = mgmotion(filename,'Diff',starttime,endtime);
mg = mgmotion(filename,'OpticalFlow',starttime,endtime);
```

input parameters

mg:musical gestures data structure containing the parameters and data of video
method:'Diff', 'OpticalFlow' specify the method used to compute the motion and features. 'Diff' method takes the absolute difference value between two successive frames. 'OpticalFlow' method uses optical field to estimate the motion.

filename:the video file should be given,when the musical gestures data structure is given

starttime:compute from the specified start time of video file

endtime:compute motion and features until the specified end time of video

output

mg:return a gestures data structure containing the parameters and data of video, the motion gram,quantity of motion and centroid of motion will be stored in the corresponding fields

examples

```
mg = mgmotion(mg,'Diff');
mg = mgmotion(filename,'Diff',5,10);
```

comments

When specify OpticalFlow method,the motion gram field in the data structure is empty.

also see

mgcentroid

synopsis

Computes the centroid of the motion image. If the input image is rgb image, mgcentroid will first change it to a gray image.

syntax

```
[com,qom] = mgcentroid(im);
```

input parameters

im:an image or motion image

output

com:centroid of the given image

qom:quantity of motion of the given image

examples

```
[com,qom] = mgcentroid(im);
```

comments

also see

mgautocor

synopsis

Estimates the period of motion for the quantity of motion.

syntax

```
[per,ac,eac,lag] = mgautocor(mg);
[per,ac,eac,lag] = mgautocor(mg,maxperiod,method);
```

input parameters

mg:musical gestures data structure containing the parameters and data of the video

maxperiod:maximum period is considered

method:sets if 'first' or 'highest' maximal value of the autocorrelation function is taken as periodicity estimation

output

per:a scalar containing the period estimation of the quantity of motion
ac:a vector containing autocorrelation functions of quantity of motion
eac:a vector containing enhanced autocorrelation functions of quantity of motion
lag:a vector containing lag values for the autocorrelation functions

examples

```
[per,ac,eac,lag] = mgautocor(mg);  
[per,ac,eac,lag] = mgautocor(mg,2,'first');
```

comments

also see

mcperiod,mcwindow

mgopticalflow

synopsis

Estimates the motion using Horn-Schunck method.

syntax

```
[flow,qom,com] = mgopticalflow(fr2,fr1);
```

input parameters

fr2:the second video frame
fr1:the first video frame

output

flow:optical flow field containing horizontal and vertical velocity component
qom:quantity of motion estimated from optical flow field
com:centroid of motion estimated from optical flow field

examples

```
[flow,qom,com] = mgopticalflow(fr2,fr1);
```

comments

also see

ComputeFlow,Gradient,opticalFlow

ComputeFlow

synopsis

Compute the horizontal and vertical component of the optical flow field using the Horn-Schunck method.

syntax

```
[U,V] = ComputeFlow(dx,dy,dt);
```

input parameters

dx:the horizontal gradient
dy:the vertical gradient
dt:the temporal gradient

output

U:the horizontal component of the optical flow field
V:the vertical component of the optical flow field

examples

comments

also see

Gradient,ComputeFlow

Gradient

synopsis

Compute the dx,dy,dt gradient of two video frames.

syntax

```
[dx,dy,dt] = Gradient(imNew,imPrev);
```

output

imNew:the second video frame

imPrev:the first video frame

input parameters

dx:the horizontal gradient

dy:the vertical gradient

dt:the temporal gradient

examples

comments

also see

ComputeFlow

mgstatistics

synopsis

Calculates the various features of the motion.

syntax

```
features = mgstatistics(mg,'Video','FirstOrder');
```

```
features = mgstatistics(mg,'Video','SecondOrder');
```

input parameters

mg:musical gestures data structure containing the motion data of the video

type:'Video' indicates that compute the features of the video data

type:'FirstOrder' or 'SecondOrder' chooses to compute lower or higher level features

output

features:a matrix containing features. When chooses 'FirstOrder',return a N by 4 dimensions matrix. When chooses 'SecondOrder', return a N by 8 dimensions matrix. N represents the number of the samples.

examples

```
features = mgstatistics(mg,'Video','SecondOrder');
```

```
features = mgstatistics(mg,'Video','FirstOrder');
```

comments

After computing the features, it is necessary to normalize the features such that all features are scaled to range (0,1).

also see

mgpca

synopsis

Computes the principle components and projects the original features space into selected principle components.

syntax

```
[x,p] = mgpca(x,ind);
```

input parameters

x:input feature space

ind:selection componaents

output

out:projected features space

p:p.r,eigenvalue, p.v,eigenvectors, p.ratio, ratios of eigenvalues

examples

```
[out,p] = mgpca(x,1:3);
```

comments

When the selection components variable is not given, mgpca chooses the principle components whose variances are greater than 0.9.

also see

mgvideosample

synopsis

Downsamples the video frame in the space for the sake of memory.

syntax

```
mg = mgvideosample(mg,'Sampling',factor);
```

input parameters

mg:musical gestures data structure containing the video data at least

'Sampling':indicates video sampling

factor:sampling factor of image in space

output

mg:sampled musical gestures data structure, resulting in reduced size of all frames fo the video

examples

```
mg = mgvideosample(mg,'Sampling',2);
```

comments**also see**

mgsimilarity

synopsis

Computes the similarity of the input array or vector based on the pairwise of observations,and returns a similarity matrix.The default method is euclidean method.

syntax

```
sim = mgsimilarity(x,distance);
```

```
sim = mgsimilarity(x);
```

input parameters

x:a vector or a array contains the data input

distance:the type of method, the default is euclidean

output

sim:a matrix contains the measurement of similarity for each pair of observations

examples

```
sim = mgsimilarity(x,'euclidean');
```

```
sim = mgsimilarity(x,'cityblock');
```

comments

also see
pdist

mgsave

synopsis

Saves the musical gestures data in the current workspace into a folder. Meanwhile, mgsave creates a .txt file in this folder.

syntax

```
mgsave(fn);
```

input parameters

fn: musical gestures dataset, variables

output

A folder contains the dataset or variables.

examples

```
mgsave(mg);
```

comments

also see