THE DETECTION OF POTENTIALLY HARMFUL HAND POSTURES IN PIANISTS USING KINECT DEPTH IMAGES

A Thesis

Presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

MENGYUAN LI

Dr. Marjorie Skubic, Thesis Supervisor

MAY 2015

The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled

DETECT POTENTIALLY HARMFUL HAND POSTURES IN PIANISTS USING KINECT

presented by Mengyuan Li,

candidate for the degree of master of science

and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Marjorie Skubic

Dr. Alina Zare

Dr. Paola Savvidou

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my adviser, Dr. Marjorie Skubic, for her valuable guidance and advice and for her vast reserve of patience and knowledge. And, I would like to express my sincere thankfulness to Dr. Paola Savvidou for her professional opinions and her help with data collection and manual labeling work. And, I would like to thank Mr. Bradley Willis for his professional advises and help as a physical therapist. I also would like to thank all the piano professors and students who participated in this study. Finally, I would like to acknowledge my family and friends for their encouragement and devotion.

Table of Contents

ACKNOWLEDGEMENTS ii

LIST OF FIGURES vi

LIST OF TABLES ix

ABSTRACT x

Chapter 1 Introduction 1

1.1 Motivation 1

1.2 Primary Goal 4

Chapter 2 Literature Review 5

2.1 Early Study on Playing-related Injury in Pianists 5

2.2 Related Work on Pianists/Other Musicians Using Kinect 5

Chapter 3 Research Sensing Platform 10

3.1 System Settings Overview 10

3.2 Other Potentially Applicable Devices 13

3.2.1 Leap Motion 13

3.2.2 Intel Creative Senz3D Camera 14

Chapter 4 Methodology of Hand Posture Detection 16

4.1 Foreground Extraction 17

4.1.1 Background Calculation 17

4.1.2 Coordination System Conversion 20

4.1.3 Keyboard Region Detection. 23

4.2 Hand Segmentation 27

4.2.1 Separating the Two Hands 27

4.2.2 Wrist Detection 31

4.2.3 Landmarks Detection 34

4.3 Feature Extraction 34

4.3.1 Side View 35

4.3.2 Top View 38

4.4 Temporal Smoothing 39

4.5 Classifier Design 40

4.5.1 Several Commonly Used Classifiers/ Clustering Methods 40

4.5.2 Multi-label Classification 41

4.5.3 Using FCM Clustering Results for Training a Naïve Bayes Classifier 44

Chapter 5 Experiments 47

5.1 Data Collection 47

5.2 Ground Truth Labeling 49

Chapter 6 Results and Discussion 52

6.1 Neutrals and Extremes of Different Subjects 52

6.2 Classification Results for Selected Pieces 59

6.2.1 Dataset 59

6.2.2 Result 60

6.3 Real-time Capability 62

Chapter 7 Conclusions 64

References 66

Appendix A 73

Appendix B 77

LIST OF FIGURES

Figure 1 Hand postures 2

Figure 2 View from the Kinect 6

Figure 3 Definition of code-numbers of fingers and piano keys 7

Figure 4 Block diagram of fingering learning and recognition 8

Figure 5 Augmented piano performance using Kinect 9

Figure 6 Kinect structure 11

Figure 7 Kinect-piano set-up 12

Figure 8 Examples of Kinect images. 12

Figure 9 Leap Motion device 13

Figure 10 Leap Motion and its hand skeletal model tracking. 14

Figure 11 Intel Creative Senz3D camera 14

Figure 12 A depth sample of Intel Creative Senz3D camera 15

Figure 13 System flow chart 16

Figure 14 Several samples of depth images captured as a pianist plays 17

Figure 15 An example of a background region 18

Figure 16 Standard deviation of a plane fitting residuals at different distances. 19

Figure 17 Segmented foreground in pseudo-color. 20

Figure 18 Coordinate system of the Kinect and the piano keyboard 21

Figure 19 Kinect setup alignment 22

Figure 20 Tilt small area and its ‘height difference’ 23

Figure 21 Flow chart for candidate keyboard pixel detection approach 24

Figure 22 Keyboard area detection 25

Figure 23 An example of height map image and the corresponding 3D point cloud in the piano coordinate space 26

Figure 24 Detecting whether there are two hands on the keyboard 28

Figure 25 Separating the left and right parts of the foreground player 29

Figure 26 Handling small fragments. 30

Figure 27 Separating result of some other circumstance. 31

Figure 28 Preprocess for wrist detection 32

Figure 29 Smoothed right foreground width 33

Figure 30 Hand crop 33

Figure 31 Landmarks detection 34

Figure 32 Side view outline of the different hand postures. 35

Figure 33 Hand side view 36

Figure 34 CAR (Hand Center Height- Hand Arch Height Ratio) feature. 37

Figure 35 A side view example for sample with a negative CAR 37

Figure 36 WAV (Wrist Angle –Vertical) feature 38

Figure 37 WAH (Wrist Angle – Horizontal) feature 39

Figure 38 Classifier structure 44

Figure 39 Ground truth labeling interface over view 49

Figure 40 Interface - side view of left part of the player 50

Figure 41 Interface - top view of foreground (player) 50

Figure 42 Interface - hand and class selection panel 51

Figure 43 Interface - colored labels 51

Figure 44 Continuous movements in feature space 53
Figure 45 Time domain feature waves..... 54
Figure 46 Neutrals and extremes of different subjects – left hand 55
Figure 47 Neutrals and extremes of different subjects – right hand..... 55
Figure 48 ROC of Naïve Bayes classification 61
Figure 49 Sample distribution legend..... 73
Figure 50 Sample distribution for ‘Scale’..... 74
Figure 51 Sample distribution for ‘Arpeggio’ 74
Figure 52 Sample distribution for ‘Chorale’..... 75
Figure 53 Sample distribution for ‘Prelude’ 75
Figure 54 Sample distribution for Schumann ‘Little Study’ 76
Figure 55 Sample distribution for ‘Chord’ 76

LIST OF TABLES

Table 1 Accuracy of the body landmark identification 6

Table 2 Kinect Setup parameters 11

Table 3 Features and parameters of Intel Creative Senz3D camera 15

Table 4 Dominate feature/ features for each class 43

Table 5 Basis for mapping clustering results into binary classification results 46

Table 6 Neutrals and extremes for CAR feature 56

Table 7 Neutrals and extremes for WAV feature 57

Table 8 Neutrals and extremes for WAH feature 58

Table 9 Dataset information 59

Table 10 10-fold cross validation* 61

Table 11 Run time** 62

ABSTRACT

Pianists who practice hours per day may have a risk for developing playing-related musculoskeletal injuries if they do not play with proper hand alignment. In order to detect the harmful, misaligned hand postures (such as wrist flexion and extension, knuckle collapse, and ulnar and radial deviation) and analyze the injury risk, a motion capture system was developed using the Microsoft Kinect depth camera. Data were captured on professional pianists and student pianists from the School of Music, University of Missouri. Hands and forearms were segmented out from the raw depth images and landmarks were then detected. Features were extracted from the 3D positions of the landmarks and used for hand posture evaluation and classification. An interface was developed to review the logged data; ground truth was provided by an expert pianist by manually labeling the hand samples using the interface. The labeled samples were then used to train classifiers using both supervised and unsupervised methods. Results are included for different hand postures of 15 participating pianists.

Chapter 1 Introduction

"The body is capable of fulfilling all pianistic demands without a violation of its nature if the most efficient ways are used; pain, insecurity, and lack of technical control are symptoms of incoordination rather than a lack of practice, intelligence, or talent."[1]

--Dorothy Taubman

1.1 Motivation

Almost every pianist or piano teacher has experienced pain or knows someone who has experienced pain from playing the piano.[2] Prevalence rates for Playing Related Musculoskeletal Disorders (PRMDs) in pianists vary widely from 26% to 93%. [3] As many as 12% of professional classical musicians with PRMD have been reported to give up their profession permanently.[4] Thus, it would be very helpful if an automated assessment was available to detect the potentially harmful hand postures and the corresponding high risk of injury so that corrections could be made early.

In order to achieve this, a hand motion capture system was built up using the Microsoft Kinect to detect the neutral hand posture and 5 typical misaligned/ potentially harmful hand postures including *collapsed knuckles*, *wrist extension*, *wrist flexion*, *ulnar deviation* and *radial deviation*. Examples for each hand posture are illustrated in Figure 1 below.

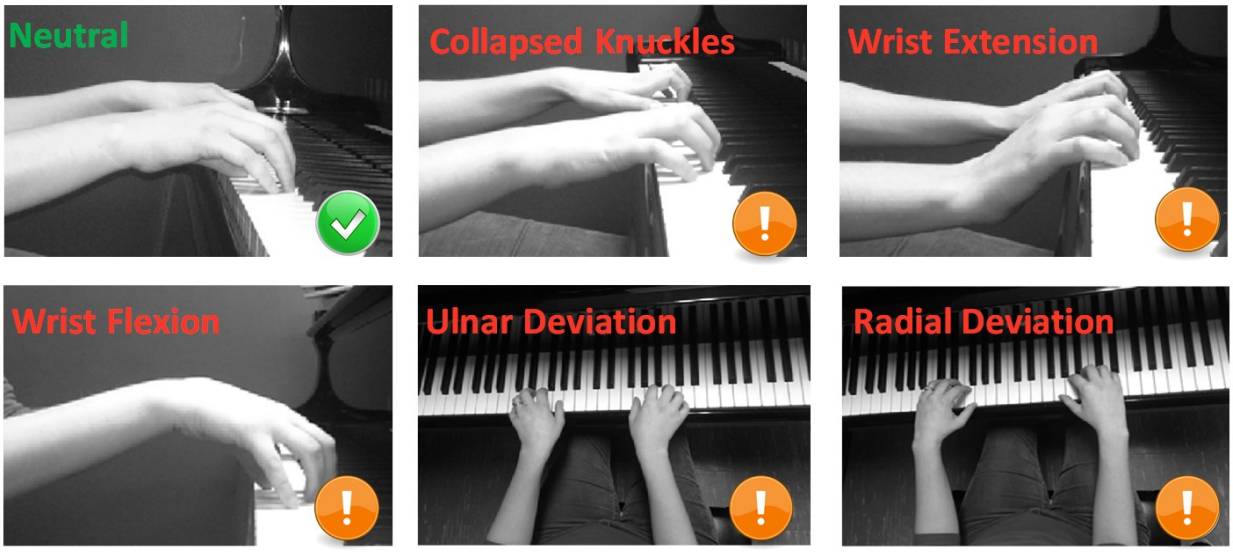

Figure 1 Hand postures

From top-left to bottom-right, the *neutral* hand posture and 5 typical misaligned/ potentially harmful hand postures: *collapsed knuckles*, *wrist flexion*, *wrist extension*, *ulnar deviation* and *radial deviation*.

Here are the descriptions for each type of hand posture we want the system to be able to detect.

1) Neutral hand posture:

- The hand should hold a nice slightly arch, avoiding bending the third knuckle (count from the fingernail) up too much or down.
- Keep wrist and forearm aligned and relaxed, avoiding twisting the hand towards the thumb or pinky; or bending the wrist up and down.

2) Collapsed knuckles:

- The third knuckle (count from the fingernail) bending down while striking the keys.

3) Wrist extension:

   - Bending the wrist down.

4) Wrist flexion:

   - Bending the wrist up.

5) Ulnar deviation:

   - Twisting the hand towards the pinky.

6) Radial deviation:

   - Twisting the hand towards the thumb.

1.2 Primary Goal

The principle objectives of perception in this research include the following:

1) Collecting data from professional and student pianists of designed demonstration hand postures and selected pieces. The collected data gives coverage of variety of hand postures including neutral and the following 5 types of misaligned, potentially harmful hand postures: collapsed knuckles, wrist extension, wrist flexion, ulnar deviation, radial deviation, shown in Figure 1.
2) Extracting proper features from the 3D information of the segmented hand samples and detected land marks. The features will be used for classification.
3) Selecting proper classifiers to classify the hand postures.
4) Evaluate the performance of our classifier by comparing to the manually labeled ground truth.

Chapter 2 Literature Review

This chapter reviews some early study about the play-related injury in pianists and some related work on pianists using the Kinect.

2.1 Early Study on Playing-related Injury in Pianists

There are many causes of playing-related injury, such as not enough warm-up time, practicing too long, not enough rest, awkward postures, and static muscular activity. Some related work has been done. Mohamed et al. found a statistically significant difference in hand temperatures between pianists with and without pain.[5] Other studies are based on captured motion. Rosety-Rodriguez et al. analyzed the active range of motion of wrist joints of young players affected by Repetitive Strain Injury (RSI); their results showed that pianists who could perform both maximal flexion and extension were rarely affected by RSI. [6] Neninger et al. studied the use of wearable devices and markers for motion capture to investigate hand injuries among musicians.[7] One drawback of this approach is that the wearable devices can limit the movement of the player or may cause unnatural movement. Also, marker-based motion capture systems are typically expensive and may limit access.

2.2 Related Work on Pianists/Other Musicians Using Kinect

In related work, Hadjakos used the Kinect for motion capture of pianists [11] and compared the performance of the Kinect with 2D marker tracking. In their paper, a clean setting and Kinect view point was proposed. In their setting, the Kinect is mounted 2~2.5m over the ground. The view of the Kinect as well as body landmarks including head, shoulders, elbows wrists and hands were detected as shown as red dots in Figure 2 below.

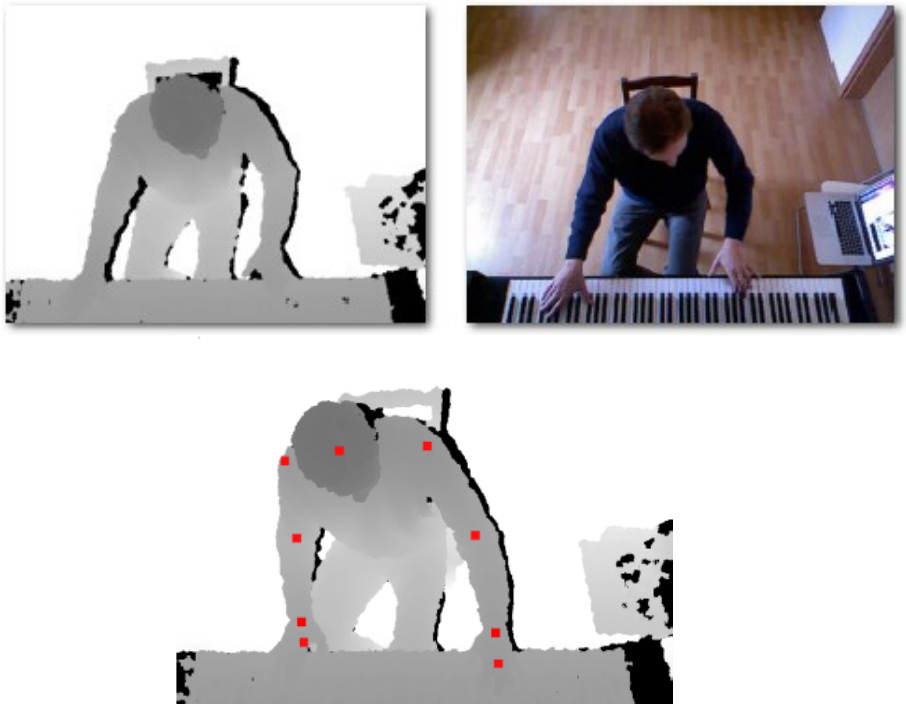

Figure 2 View from the Kinect[11]

depth image (upper-left), RGB image (upper-right) and body land-marks (bottom).

The accuracy of the body landmark identification is shown in Table 1 below in mean distances and the root mean square of the distance between points estimated from the depth image and measured by the markers. Unfortunately, from the table, we can find out that the accuracy is not high enough to be applied for our application. And, the algorithm they used for arm and hand silhouette detection was based on a local height dropping detection which can be sensitive to the threshold value and can lead to segmentation failures.

Point	mean	RMS
Head	11.9 pixel (2.8 cm)	13.3 pixel (3.1 cm)
Shoulder	18.4 pixel (4.3 cm)	19.4 pixel (4.5 cm)
Elbow	20.6 pixel (4.8 cm)	21.0 pixel (4.9 cm)
Wrist	19.0 pixel (4.4 cm)	19.5 pixel (4.5 cm)
Hand	11.8 pixel (2.7 cm)	13.6 pixel (3.2 cm)

Table 1 Accuracy of the body landmark identification[11]

Akiya Oka et al worked on marker-less piano fingering recognition using sequential Kinect depth images.[12] The term “piano fingering” refers to which fingers are used for pressing the keys which is one of the important skills for piano performance, especially for beginners. They proposed a method to recognize piano fingering by building up a dictionary data set in which each data element consists of a depth image sample, the name of the pressed key and the correct information of its fingering and the wrist position of the player in the image. When an unknown sample comes, they reduce the search space size by looking at the wrist position detected from the input image and a note name obtained from a MIDI keyboard. Then the unknown depth image is identified by matching to those in the narrowed dictionary data set. The Nearest Neighbor algorithm is utilized to search for solutions. In their setup, the Kinect was fixed to a height of 70 cm above the keyboard plane. This setup can cover the measurement of approximately four-octaves of a piano keyboard.

Figure 3 below shows the definition of the code-numbers of fingers and keyboard keys while Figure 4 shows block diagram of their system.

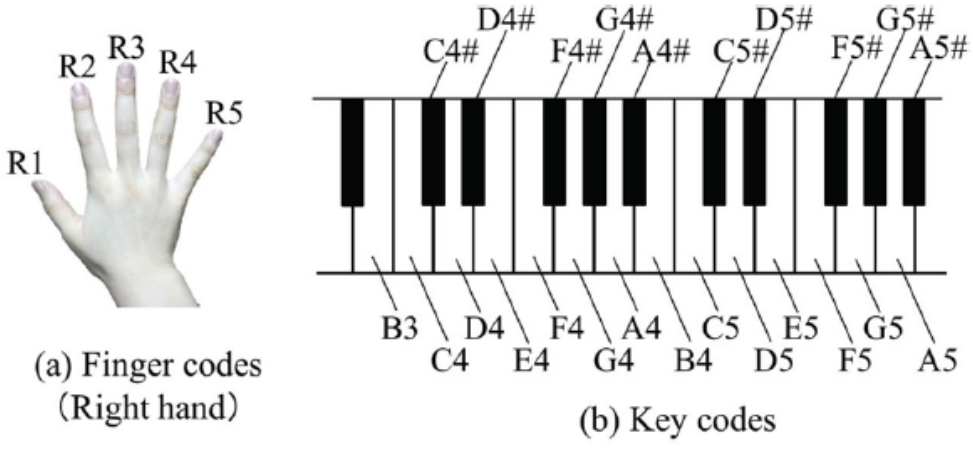

Figure 3 Definition of code-numbers of fingers and piano keys[12]

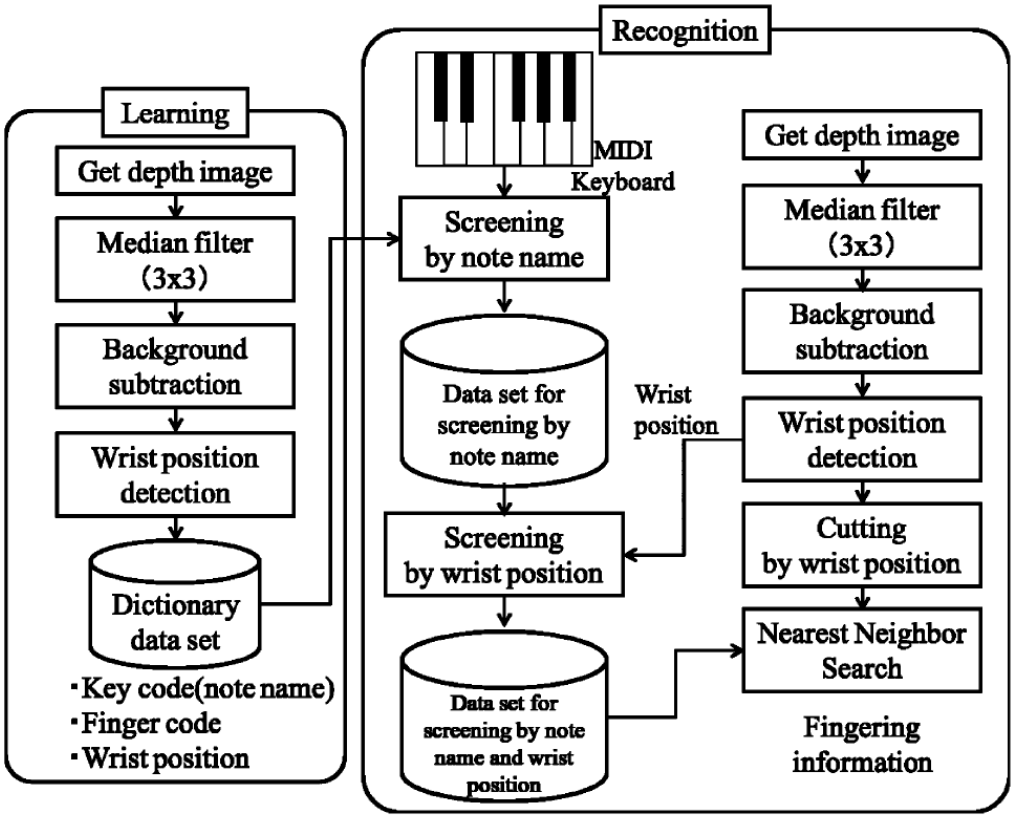

Figure 4 Block diagram of fingering learning and recognition [12]

Qi Yang et al have worked on augmented piano performance using the Kinect depth camera.[13] They think the traditional piano keyboard excels at discrete pitch and note volume controls, but it has little control of the quality of the sound. In contrast, bowed or wind-column instruments have a great range of articulation after sounding each note. In addition, most of the digital or sampler instruments contain more parameters to adjust than acoustic piano. Physical knobs, sliders and switches are used to control those parameters which give more possibility for a performance. But it can be unintuitive and difficult when manipulations need to be done during performance. In order to solve this problem, they proposed an idea of using a Depth camera to capture multi-dimensional hand gestures to control multiple parameters with no need to manipulate multiple physical controls. They

used the Kinect for sensing and a top-down projection for visual feedback as shown in the Figure 5 (a) below. Figure 5 (b) shows a user illustrating performance gestures.

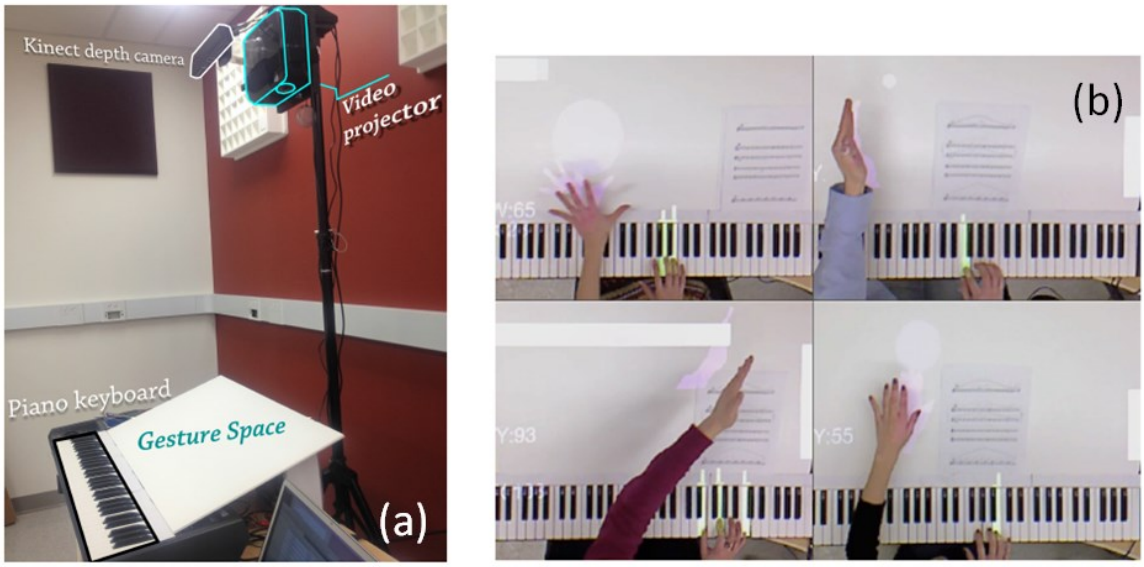

Figure 5 Augmented piano performance using Kinect[13]

(a) Configuration of the augmented piano keyboard; (b) User illustrating performance gestures.

Chapter 3 Research Sensing Platform

This chapter talks about the sensing platform used in this research. It contains two parts. The first part describes the way the system is set up using a single Kinect system to evaluate the performance of the individual pianist, and the second part discusses two potentially applicable devices for the purpose of our application. The superiorities of those devices and reason why they cannot be used in our research for now are explained.

3.1 System Settings Overview

The Microsoft Kinect for Windows (Model 1517) is the core device used in this research. It uses infrared (IR) light to generate depth images which can give a 3D representation after reconstruction. This provides a cost-effective sensing device to replace the expensive, marker-based 3D motion capture equipment. In addition, it offers a significant advantage in being non-wearable which has no effect on the playing movements.

The Kinect provides depth images with resolution up to 480x640 and a frame rate up to 30 fps. When it comes to the working range, there are two settings: the default range is from 800mm to a maximum of 4000mm, while the near mode provides a range of 500mm to 3000mm.

In our research, the near mode is used to get the depth information of the head and shoulder which can be within 800mm to the Kinect depth camera. The setup will be discussed in detail later in this section.

Figure 6 below shows the structure of the Kinect, the pair of IR (infrared) emitter and IR camera in red circles is the 3D depth sensor which produces the depth image.

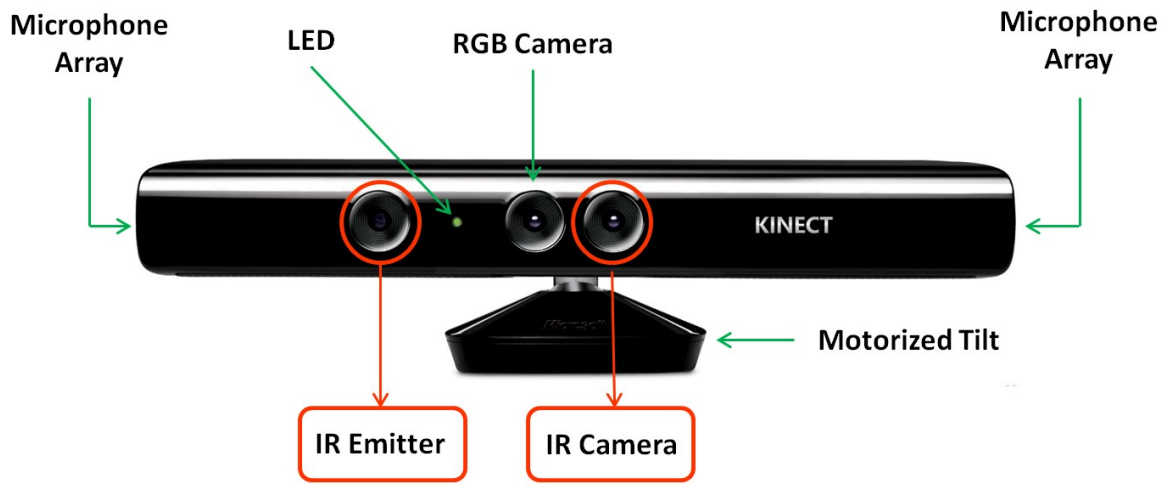

Figure 6 Kinect structure

The Kinect was chosen as our depth sensor to get the 3D information of the hand posture information. In our research, one Microsoft Kinect was set up above the piano using a camera stand, about 40 inches (100 cm) high to the keyboard plane, facing down to the piano keyboard, see Figure 7. The data were collected using the near mode of the Kinect with a working range of 500mm to 3000mm, resolution of 320*240 at a frame rate of 30fps. An example of the RGB color image and our raw depth data is shown in Figure 8. This setting gives coverage of the entire piano keyboard of a standard size with a proper resolution to see the hand posture. Since there is a tradeoff between the resolution and data size and processing cost, a 320*240 resolution was selected instead of the higher possible resolution of 640*480. Table 2 below illustrates the Kinect setup parameters used in this research.

Table 2 Kinect Setup parameters

Distance to Keyboard	**~ 40 inches (1000 mm)**
Frame Rate	30 fps
Working Mode	Near Mode (500 ~ 3000 mm)
Depth Image Resolution	320x240

Figure 7 Kinect-piano set-up

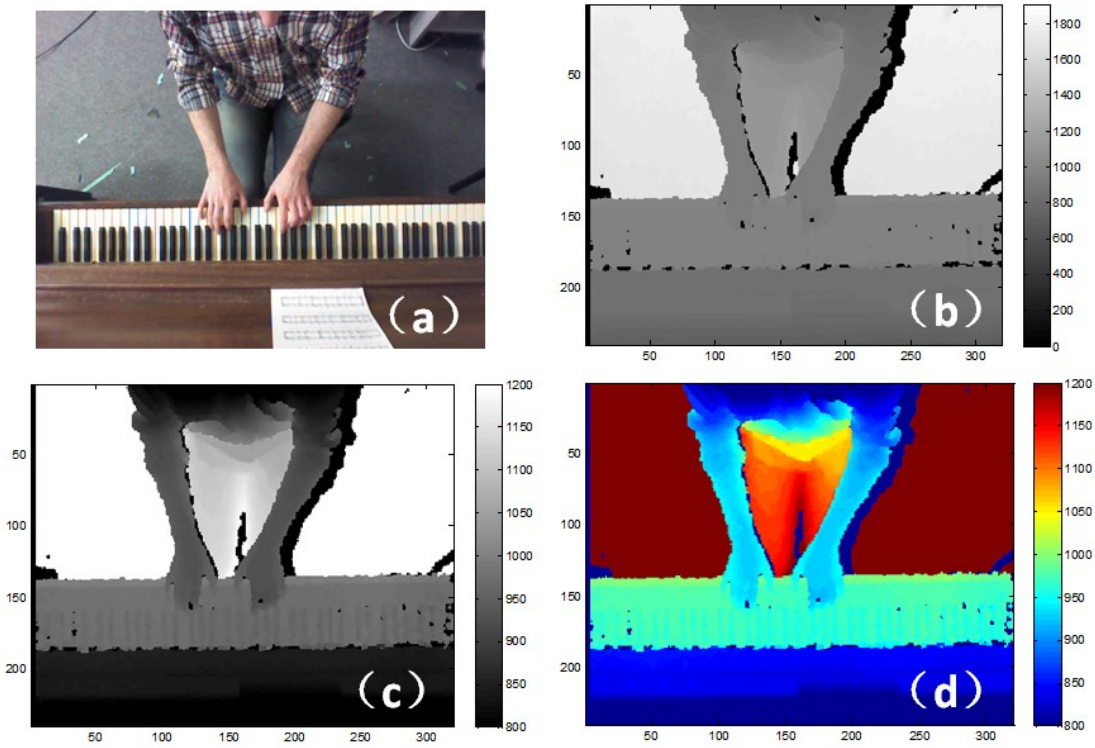

Figure 8 Examples of Kinect images.

(a) RGB image; (b) Raw depth image in gray scale from the nearest point to the furthest to the Kinect in the entire view; (c) Depth image with an enhanced contrast for the view with depth value within 80cm to 120 cm (the color bar shows the gray scale map in millimeter) from the Kinect; (d) The same thing as in (c) but shown in pseudo-color where blue means a smaller depth value while red means larger. *All the color bars in the rest of this thesis are shown in units of mm (millimeter).

3.2 Other Potentially Applicable Devices

There are other new depth sensors recently made available in the market. In this section, two depth sensors: Leap Motion and Creative SENZ3D will be discussed in detail. They have features which can be useful for our application but unfortunately they have some limitations which prevent them from being used for this research for now. This may change in the near future!

3.2.1 Leap Motion

Leap Motion is another IR motion sensor device we tested, shown in Figure 9 below.

Figure 9 Leap Motion device

It focuses on the hand and fingers and gives a hand skeletal model (shown in Figure 10(c) and (d)) which is good for our application. Unfortunately, the Leap Motion has a limited-sized, fixed upside-down pyramid-shaped working space (shown in Figure 10 (a) and (b)), which means the user has to put his or her hands in the air to get the hands detected as seen in Figure 10 (b). However, or our use, we need to detect the hands of the player placed on the piano keyboard and we need coverage of the entire piano. We tried to place the Leap Motion upside down facing to the keyboard, but it cannot get the hand skeletal model properly in this way. This makes Leap Motion not suitable for our application. Hopefully, in the future, it will be able to detect the hands on a plane and give adequate coverage of a larger space.

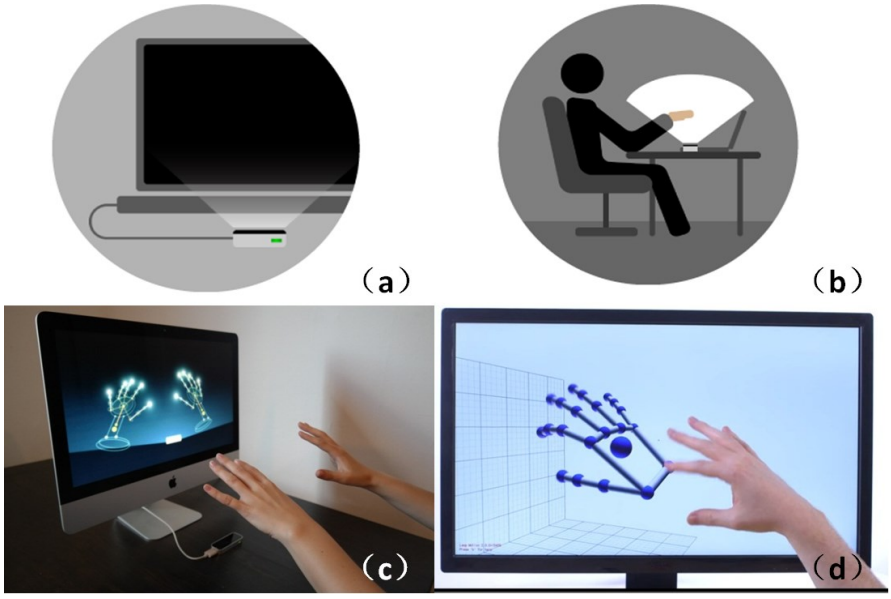

Figure 10 Leap Motion and its hand skeletal model tracking.

(a) and (b) Working setting of Leap Motion[14]; (c) The hand skeletal model of Leap Motion using SDK V1[15]; (d) The hand skeletal model of Leap Motion using newly published SDK V2 with signicicantly improved hand joint tracking capbility[16].

3.2.2 Intel Creative Senz3D Camera

Intel Creative Senz3D camera is also an IR motion sensor with RGB camera, depth camera and microphones integrated (shown in Figure 11).

Figure 11 Intel Creative Senz3D camera

Unlike Kinect, the Intel Creative Senz3D camera was designed for desk usage. It is capable of 3D gesture control and 3D face sensing. It has a 720p HD RGB camera and the working range of the depth camera is 0.5ft ~ 3.25ft (152.4mm ~ 1000mm) comparing to the Kinect's (500mm ~ 3000) mm in its near mode.

There are some superiorities of the Intel Creative Senz3D camera comparing to the Kinect. It is cheaper, much smaller in size and lighter in weight. It doesn't require any external power supply. All it needs is to be connected to the computer using a USB port, which makes it more accessible and portable. But, since its farthest working range is not far enough for our application, it is not applicable for our application. A sample depth image and the features of this device are shown below in Figure 12 and Table 3.

Figure 12 A depth sample of Intel Creative Senz3D camera

Table 3 Features and parameters of Intel Creative Senz3D camera

RGB video resolution	HD 720p (1280x720)
IR depth resolution	QVGA (320x240)
Frame rate	Up to 30 fps
FOV (Field-of-View)	74°
Range	0.5ft ~ 3.25ft
Size	4.27" x 2.03" x 2.11"
Weight	271g

Chapter 4 Methodology of Hand Posture Detection

This chapter describes the algorithm implemented to detect the hand posture using the system setup discussed in Chapter 3. Figure 13 shows the flow chart of the algorithm.

System Flow Chart

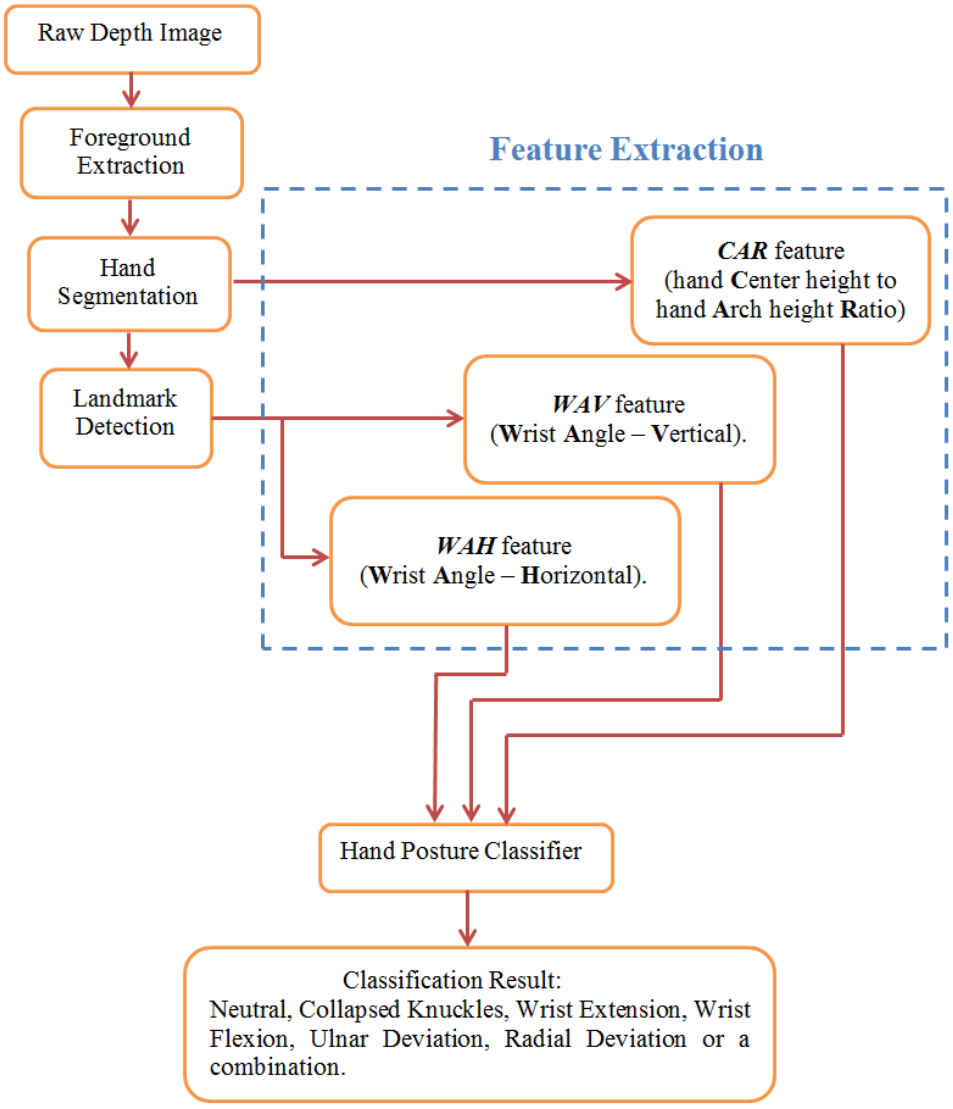

Figure 13 System flow chart

Each of the blocks from Figure 13 will be discussed in the following subsections.

4.1 Foreground Extraction

The foreground (the player) extraction is performed on the depth image frames using a background subtraction algorithm. A background calculation algorithm will be discussed in 4.1.1 and it will be converted to a coordinate system with an origin z plane of the piano keyboard plane in 4.1.2.

4.1.1 Background Calculation

In this research, pianists are asked to play selected pieces, where each piece is within a few minutes in length. For each recorded piece, all the frames are used to calculate the background. When the player is playing, his or her arms and hands will move and keep blocking/ unblocking different part of the background which is farther away from the Kinect, shown in some depth image samples in Figure 14. Thus, the basic idea of our background calculation is: for each pixel of the background, we update the depth value with the larger value (farthest to the Kinect along the optical axis of the lens) throughout all the frames as the background.

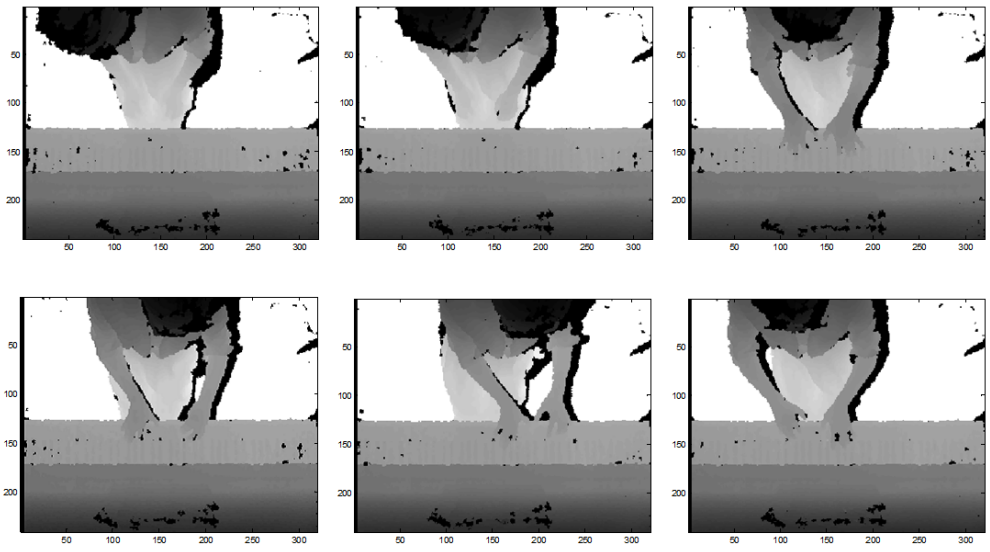

Figure 14 Several samples of depth images captured as a pianist plays

An example of background we get from a selected piece is shown in Figure 15 below. As we can see, the torso and lap of the player are regarded as background. This is because the system doesn't require a clean scene to capture the background. There has to be some area being blocked by the player for the entire recorded time period, which is the exact area shown up as laps and torso in the calculated background. And since those are their furthest position to the Kinect in the calculated background, when the torso leans forward and gets nearer to the Kinect or leans to the left or right blocking part of the background, it becomes foreground. And, we see small portion of the arms and no hand in the calculated background. Because the hands and arms move a lot, we can always get the background being blocked by them when they move away.

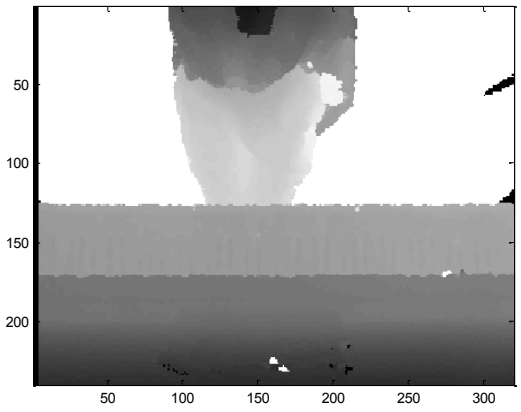

Figure 15 An example of a background region

After we get the background, the pixels in each frame with a distance closer to the Kinect than the background with an amount larger than a certain threshold will be considered as foreground (the player). Here come some problems: How noisy are the Kinect depth values? How do we choose a proper threshold that can segment the foreground from the background?

Khoshelham K. and Elberink SO have investigated the accuracy of Kinect depth data.[17] Their research shows that the accuracy of the Kinect data changes along the distance between the object to the lens. The relationship between the standard deviation of the plane fitting residuals at different distances of the plane to Kinect device is shown in Figure 16. In our case, the objects in the view have a distance to the Kinect in a range from 50 to 200cm, while our targets of interest (the arms and hands) sit in a range from 80 to 120 cm. According to their research results, the standard deviation of the theoretical random error and depth resolution for our application is 0.18~ 0.41cm and 0.36~0.82 cm.

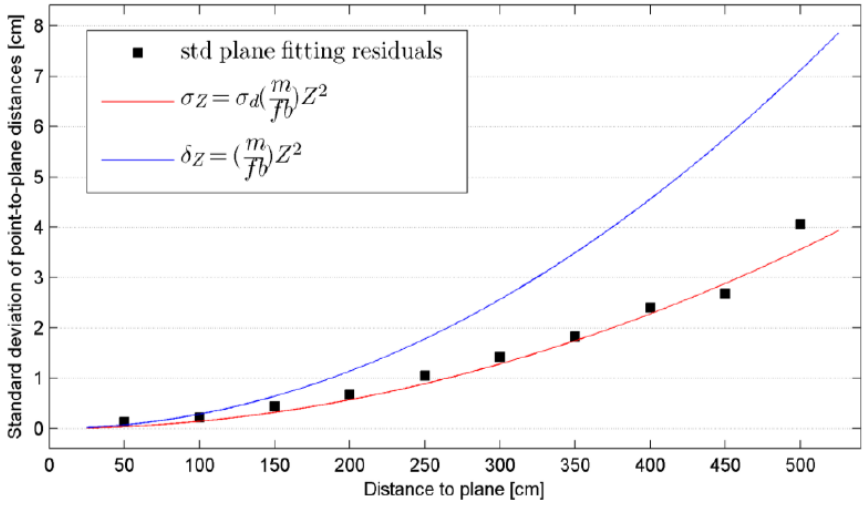

Figure 16 Standard deviation of a plane fitting residuals at different distances. The curves show the theoretical random error (red) and depth resolution (blue). | *m*/*fb* | = 2.85e−5 from the depth calibration result and assuming a disparity measurement error (*σd'*) of ½ pixel. [17]

According to this random error range (up to 0.41cm) and depth resolution (up to 0.82cm), as well as the fact that the finest detail we would like to distinguish is the finger tips, we set the threshold to 1.5cm. Pixels with a depth value more than 1.5cm near the Kinect device

(along the optical axis of the camera lens) than the background will be assigned as foreground.

The foreground regions are further refined by removing small, 4-connected components (objects) that have fewer than 150 pixels in the segmented foreground. Figure 17 shows an example of the segmented foreground. Subfigure (a) shows the segmented foreground with small false positive regions shown in red circles. Subfigure (b) shows the clean foreground segmented out after removing small non-foreground regions which are regarded as foreground by their occasionally small depth value caused by random error, for example.

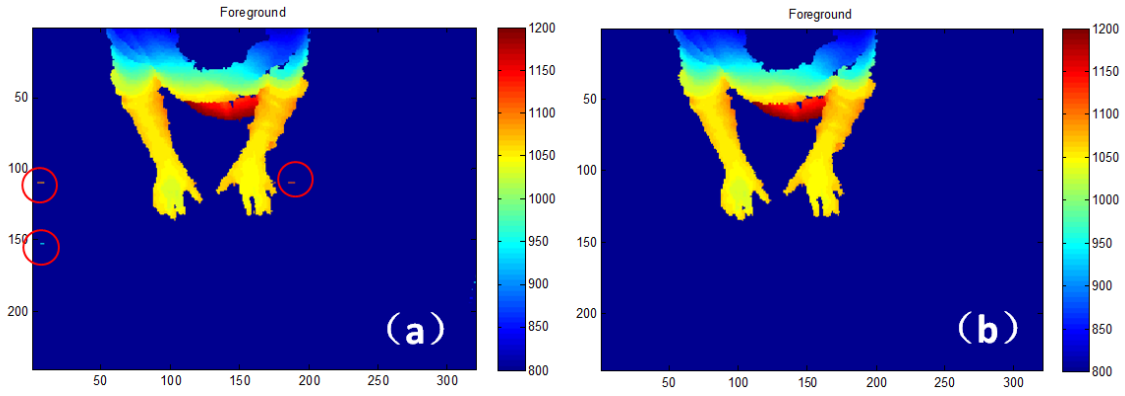

Figure 17 Segmented foreground in pseudo-color.

(a) The foreground before removing small false positive foreground regions; (b) The clean foreground segmentation.

4.1.2 Coordination System Conversion

It was mentioned briefly that the Kinect was set up above the piano keyboard plane in Section 3.1. In this section, more details about the set-up will be discussed, and the method used to convert the depth data in the Kinect coordinate system to the real-world keyboard coordinate system will be discussed.

The depth information we get from the Kinect is in the Kinect coordination system shown on the left hand side of Figure 18 below. Since the goal for this research is to detect the potentially harmful hand postures in pianists, it would make more sense to convert the 3D data into a coordinate system with respect to the piano keyboard shown on the right hand side of Figure 18. The origin of the Kinect coordinate system is projected onto the keyboard plane to be the origin of the piano coordinate system.

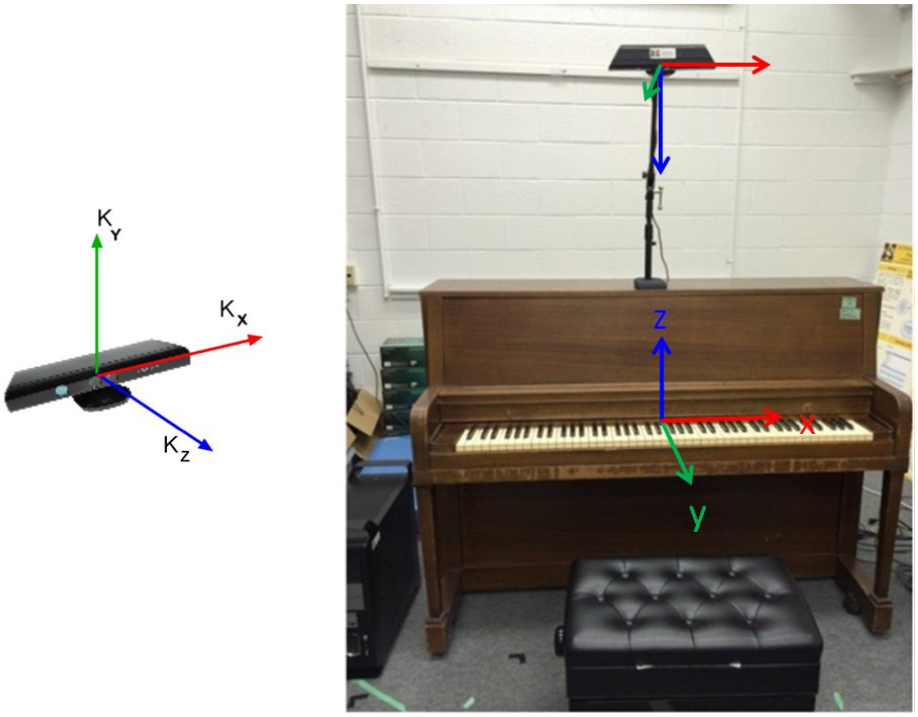

Figure 18 Coordinate system of the Kinect and the piano keyboard

In order to make things simple and intuitive, when the Kinect is set up using a camera stand, the following steps are followed.

1) Adjust the orientation of the Kinect to make the upper and lower edges of the keyboard shown in red solid lines in Figure 19 look parallel to the edge of the depth

image shown as a red dash line. By doing this, we make the x-axis of the Kinect and the Keyboard coordinate systems roughly parallel to each other.

2) Adjust the orientation of the Kinect to make the spacing between the black keys even and the black keys look vertical in the depth image in order to make sure the y-axis of the Kinect and the Keyboard coordinate systems roughly parallel to each other.

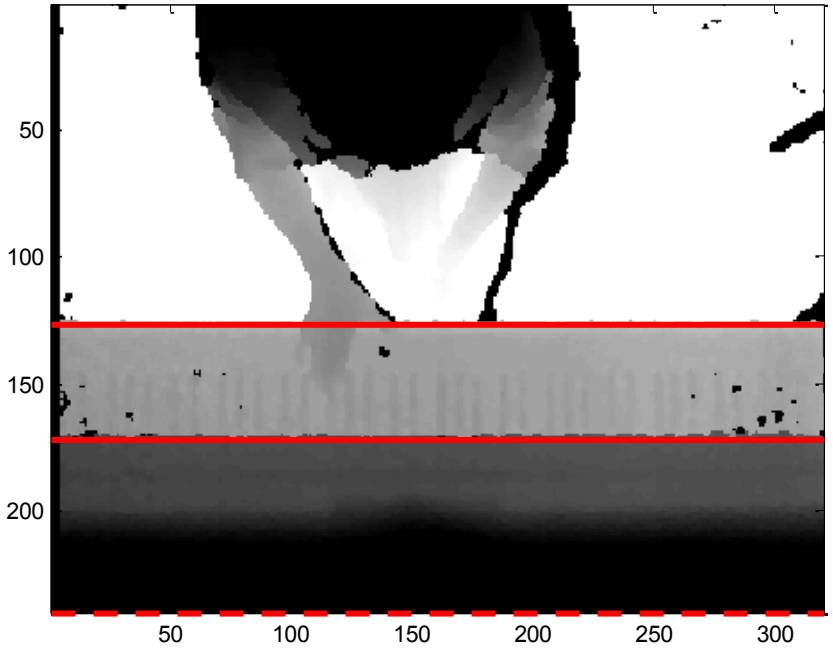

Figure 19 Kinect setup alignment

The two solid red lines should be parallel to each other and also parallel to the red dashed line at the same time. The black keys should be vertical to the red lines and the space between the black keys should be even.

This setting makes the Kinect x-y plane approximately parallel to the new piano coordinate x-y plane. Since the Kinect orientation is adjusted by hand, there likely will be a slight misalignment. Instead of absolutely parallel to the Kinect x-y plane, the keyboard plane can be slightly tilted. In our experiments, we regard a tilt within 20 degrees as

tolerable. The x, y coordinates will not be influenced much but the z coordinates will be affected. Thus, the only correction that needs to be done for the coordinate system conversion is to assign each pixel a new z axis value with the distance to the keyboard plane. Points above the keyboard yield positive values and points below yield negative. The section 4.1.3 will discuss the detection of the keyboard plane / region.

4.1.3 Keyboard Region Detection.

The keyboard plane is a very important reference plane in our research that we would like to segment out in order to capture the hands.

As mentioned in section 4.1.2, the keyboard plane is roughly parallel to the Kinect x-y plane. The logic used to detect the keyboard is the following: (1) take a small square piece of area with a pixel as the upper-left corner and measure the tilt height to the Kinect x-y plane of this area as height difference (shown as the red line in Figure 20); (2) if the height difference is zero, this small area is parallel to the x-y plane; if the height difference is no larger than a certain threshold, we consider it as roughly parallel to the Kinect x-y plane and mark the center pixel as a candidate keyboard pixel.

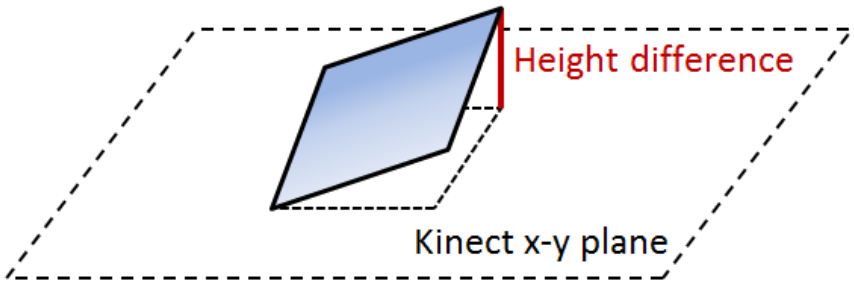

Figure 20 Tilt small area and its 'height difference'.

Considering the random error of the depth data discussed in section 4.1.1 and the fact that we assume the tilt of the keyboard plane should be within 20 degree, the size of the

small area is fixed to be a 6*6 pixel square which is a square area about 20*20 mm^2 large. With this size, it is not too big so we don't lose too much detail about the edge of the keyboard, and it is not too small so that the random depth error of 2~4mm is relatively small.

The threshold selection needs to consider both the largest tolerance of the tilted angle of the keyboard plane with respect to the Kinect x-y plane and the random depth error. Our experiments show the tilt angle is less than 15 degrees by using the manual alignment approach mentioned in section 4.1.2. The threshold calculated for a 20*20 mm^2 region with a tilt angle of 15 degrees and a random depth error of 3mm is 8mm. The flow chart of the candidate keyboard pixel detection approach is shown in Figure 21.

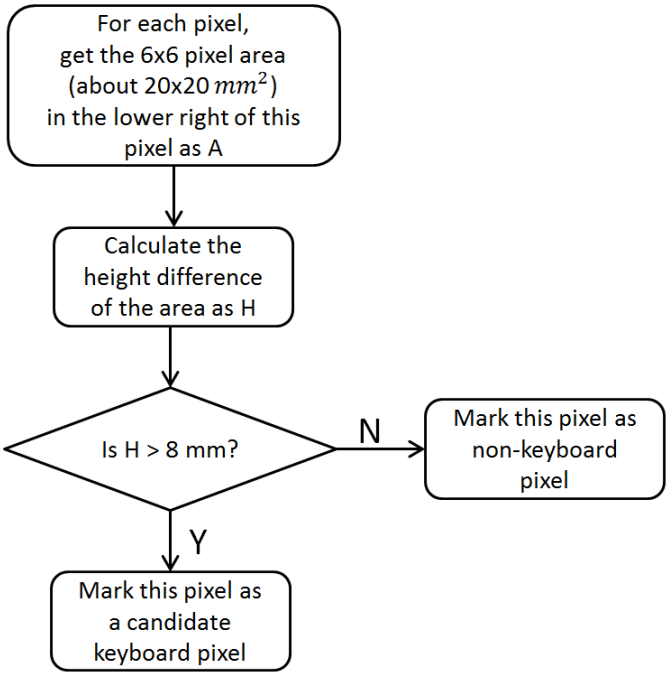

Figure 21 Flow chart for candidate keyboard pixel detection approach

After getting the candidate keyboard pixels, a refinement approach needs to be applied since there are false positive pixels which also belong to a small flat plane roughly parallel

to the Kinect x-y plane which can be part of the shoulder, arm or other flat portion of the piano. In order to get rid of the false positives, the following steps are applied.

1. Remove pixels with a depth value outside the range 900 ~ 1300mm since, in our setting, the keyboard should be in a position around 1 meter away from the Kinect. If a pixel is too near or too far away from the Kinect, it is unlikely to be belong to the keyboard.
2. Remove pixels belonging to the foreground. By doing this, the false positive pixels which actually belong to the flat area of the player will be filtered out.
3. After steps 1 and 2, the only false positive will be some flat portion of the piano other than the keyboard such as the narrow part between the keyboard and the resonance box which sits below the actual keyboard in the depth image. Thus, the upper connected region is selected as the keyboard region.

Figure 22 shows a sample of the keyboard detection. The keyboard region is highlighted in majenta.

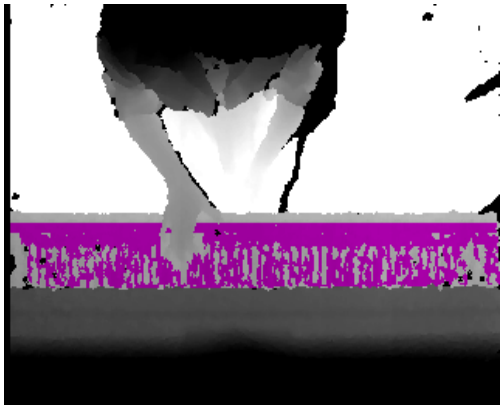

Figure 22 Keyboard area detection

After the keyboard region is detected, a plane fitting approach is applied to get the keyboard plane in the Kinect coordinate system. A height map with respect to the keyboard

plane is then generated from the raw depth images. The range of the height map was clamped within -5 cm to 20 cm which is a reasonable movement range for the hands and forearms while playing. An example of a height map image and the 3D point cloud is shown in Figure 23 below. Subplot (a) shows the segmented foreground depth image; (b) shows the converted height map with respect to the keyboard plane clamped within -5cm to 20cm; (c), (d) and (e) show the 3D point cloud of the foreground in the piano coordinate space with different angles of view.

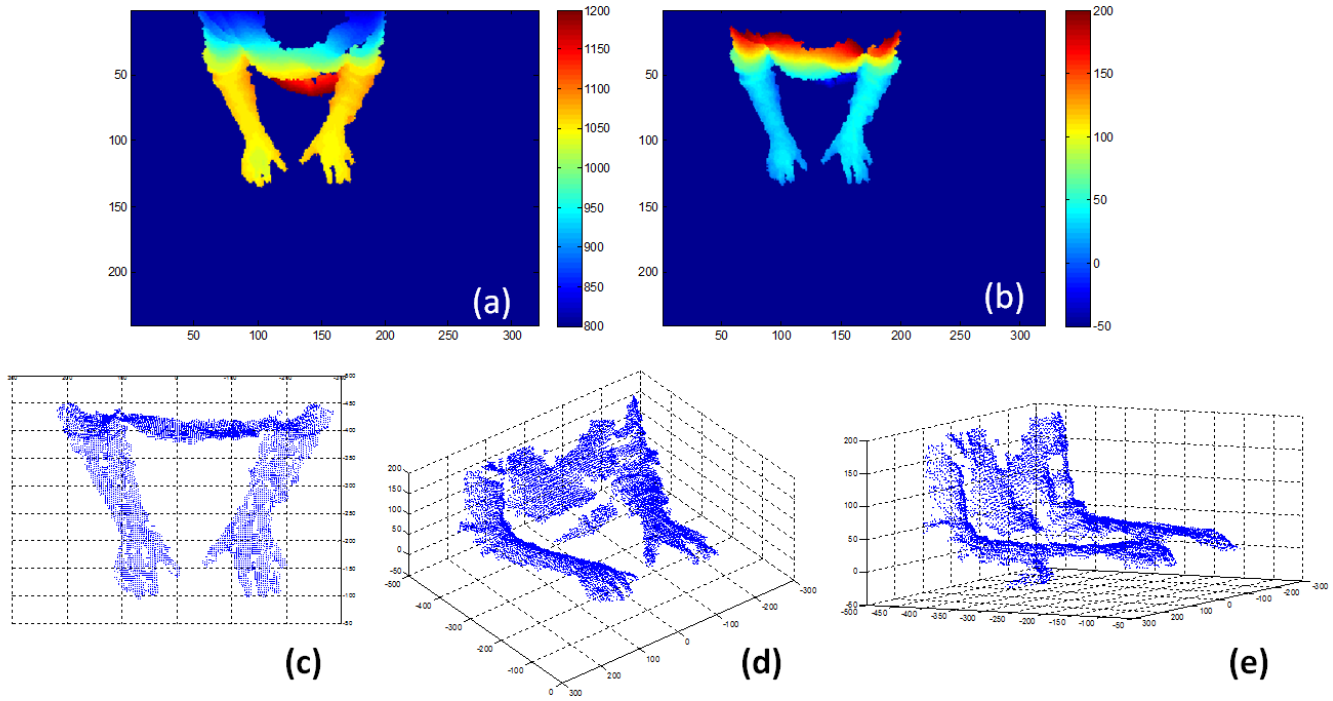

Figure 23 An example of height map image and the corresponding 3D point cloud in the piano coordinate space

(a) the segmented foreground depth image; (b) the converted height map with respect to the keyboard plane clamped within -5cm to 20cm; (c), (d) and (e) show three views of the 3D point cloud of the foreground in the piano coordinate space. This is taken from the chorale "Rejoice, O My Soul", Op. 68, No. 4, (R. Schumann) frame No. 401 of subject 1.

4.2 Hand Segmentation

After getting the foreground region in the keyboard coordinate system, the left and right hand need to be segmented out from the rest of the foreground. The hand segmented out should be cut from the wrist and should be distinguished as left or right. In this section, it will be discussed how this is done.

For related work, there are two broad types of hand detection/ tracking methods. One is appearance-based [25], [26], [27], [28], [29], [30] and one is model-based [18], [19], [20], [21], [22], [23], [24].

In our situation, it is assumed that hands will appear on the keyboard. Since our goal is to detect the potentially harmful hand postures in pianists, we only care about the hand samples of the hands above the keyboard. And, in most of the playing hand postures, hand shapes are usually clear to the depth camera which makes it a fit for appearance based hand detection.

4.2.1 Separating the Two Hands

When segmenting the hands, we would like to know which is which. In this research, the hand segmentation process only triggers when both hands are placed on the keyboard. The logic of deciding whether there are two hands on the keyboard or just one is as follows:

For the row 30 pixels (~ 4 inches) in front of the keyboard edge shown as the green dash line in Figure 24 below,

1. If this line cuts through the foreground twice, then there are two hands on the keyboard.
2. If the number of foreground pixels in this row is less than 20 pixels (2.7 inches), there are fewer than two hands on the keyboard.

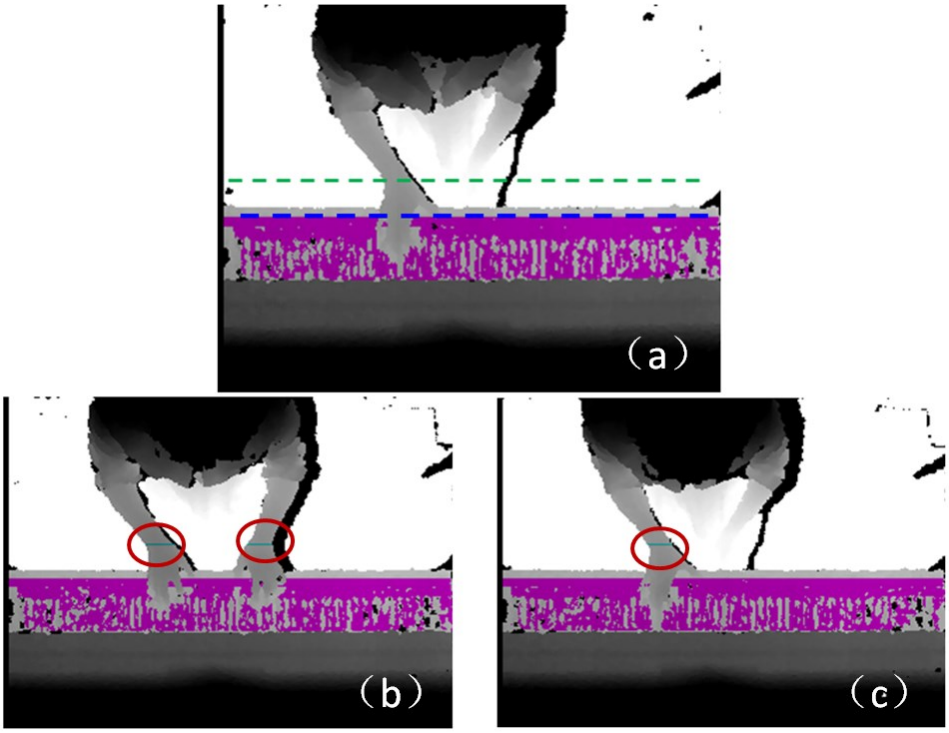

Figure 24 Detecting whether there are two hands on the keyboard

(a) The blue and green dash lines represents the upper keyboard edge and the row 4 inches in front of the keyboard edge; (b, c) the foreground at the row 4 inches in front of the keyboard edge intersects this line, shown as short green lines in the red circles. There are two hands on the keyboard in (b) while there is only one in (c).

When the hand segmentation process is triggered, there are two hands on the keyboard. A midpoint was calculated as the median value of the column index of all the foreground pixels in each row of the height map image, generating a mid-column line.

Figure 25 below shows an example of separating the left and right part of the foreground player using the algorithm mentioned above. Subfigure (b) shows how it works to divide the foreground into left and right parts. The red line in (b) represents the mid-column line. (c) is a

zoomed-in view of the hand part of (b). It is shown that one hand is cut into two parts by the mid-column line. In order to fix this problem, the following approach is applied. After cutting the foreground using the mid-column line, the 4-connected regions smaller than 200 pixels are regarded as small fragments, and the largest 4-connected regions for each side are regarded as big pieces. This yields *'Big_Piece_Left'*, *'Big_Piece_Right'*, *'Small_Fragment_Left'* and *'Small_Fragment_Right'*. In the case of Figure 25, part of the right hand side of the player was cut off and divided into the left hand side, called *'Small_Fragment_Left'*. We combine it with *'Big_Piece_Right'* and perform dilation and erosion to fill the possible gap between them. Then if they become one connected region, the fragment is successfully combined back with the correct side. An illustration for this approach is shown below in Figure 26.

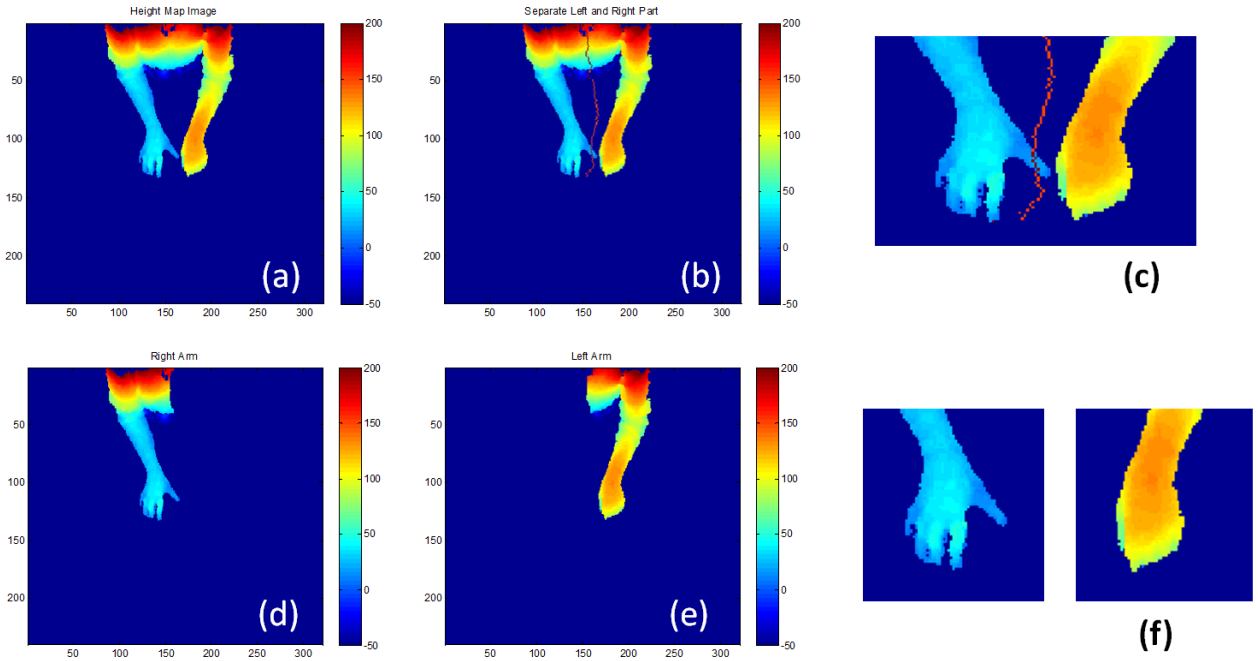

Figure 25 Separating the left and right parts of the foreground player

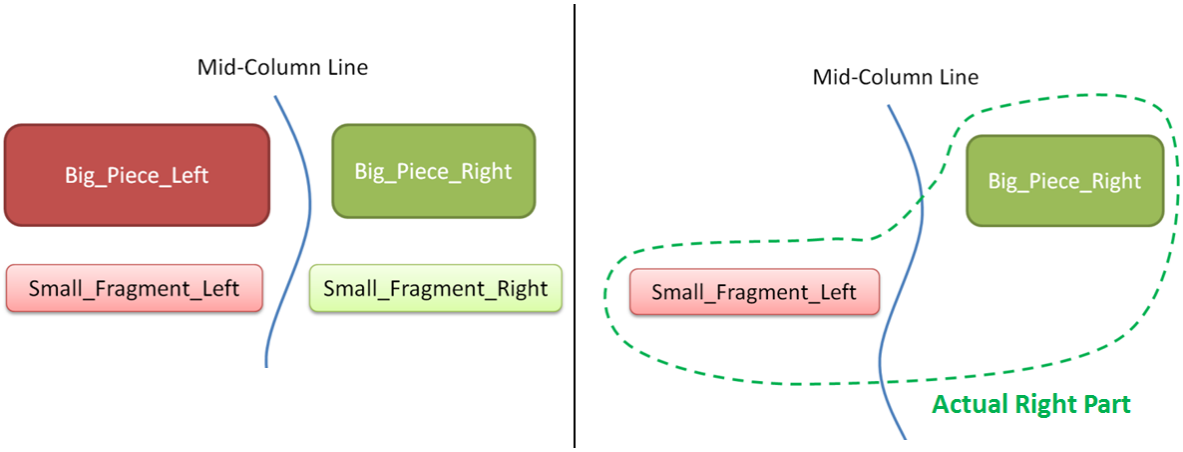

Figure 26 Handling small fragments.

The left hand side shows the big-pieces and small-fragments divided by the mid-column line; the right hand side shows the assignment of the small-fragment on one side to the big-piece of the other side, in the dashed circle. Dilation and erosion operations are performed on the regions in the dashed circle; if they merge into one piece, then the fragment is assigned to the larger component.

After applying this approach, we can get the final separating result. Figure 27 below shows more results of using this approach on some different examples.

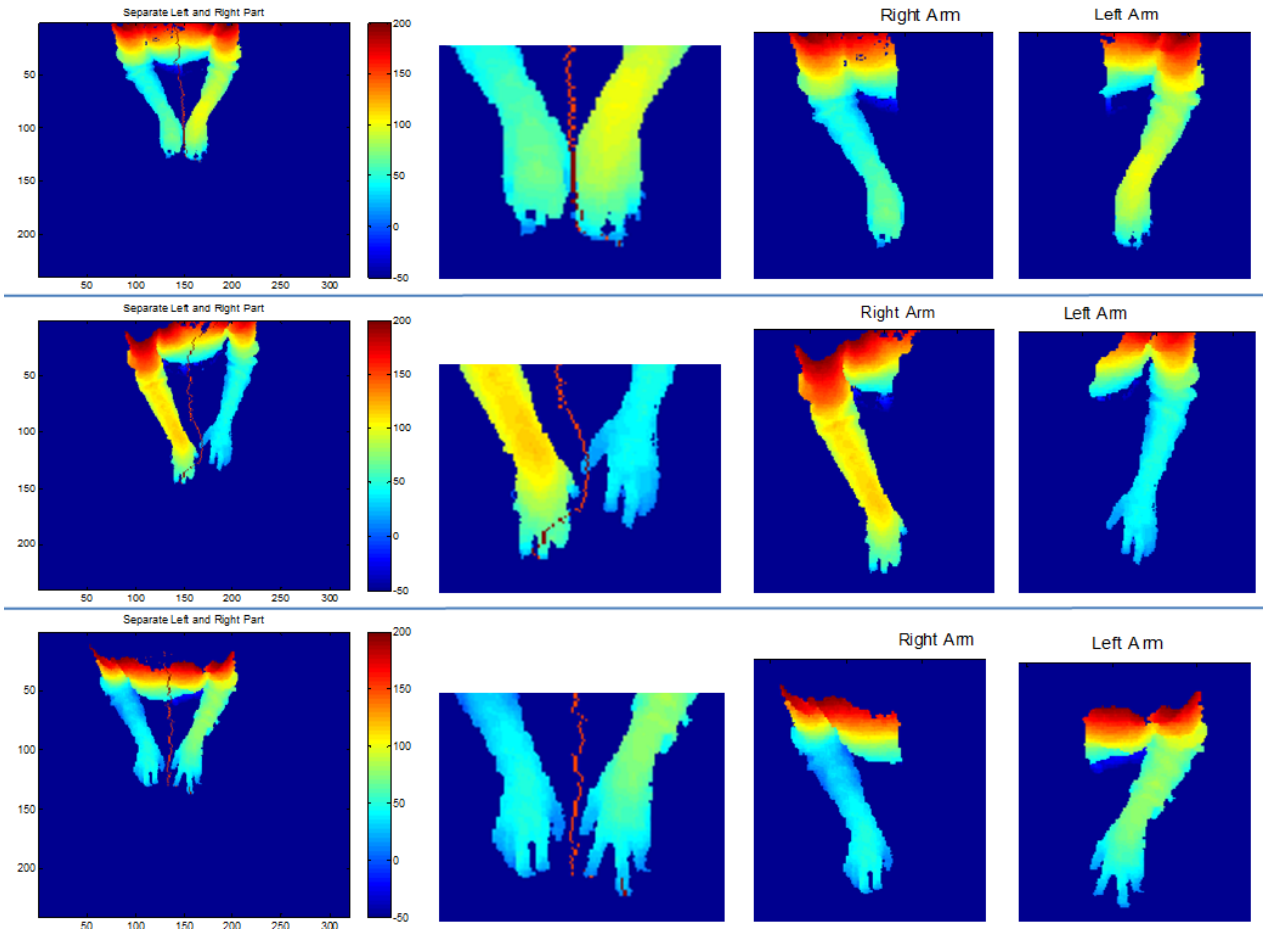

Figure 27 Separating result of some other circumstance.

The first row shows the result of separating two hands that are connected to each other; the second row shows the result when one hand is more forward and is cut into parts by the mid column line; the third row shows the result when the mid column line cuts the finger.

4.2.2 Wrist Detection

The wrist detection is based on appearance since the shape of the arm and the hand narrows in the direction from the arm towards the wrist and markedly widens after the wrist when the hand begins. [11]

In this step, in order to achieve faster processing speed, the image is down sampled to 160*120 pixels from the original 320*240 pixels.

For the preprocessing for wrist detection, a closing operation was done to the mask for the upper torso and arms in order to fill out the space between the fingers to get a more convex outline for the hand part, as shown in Figure 28. For each side (left and right), a candidate wrist position is found by looking for a valley in pixel numbers that belong to the mask in each row, as shown in Figure 29. A bounding box was then generated by the area below the candidate wrist position. A more accurate bounding box was created based on the current one. The diameter, *'D_equal'*, of a circle with the same area as the region was calculated and the new bounding box was updated with left, right and bottom bounds enlarged by 0.1**D_equal* and the height of the bounding box replaced by 1.1* *D_equal*. The wrist position accuracy is significantly improved by this approach. As shown in Figure 30, the thin and bold boxes show the candidate and adjusted bounding box, respectively. Temporal smoothing was then applied to get a more stable and smoother result.

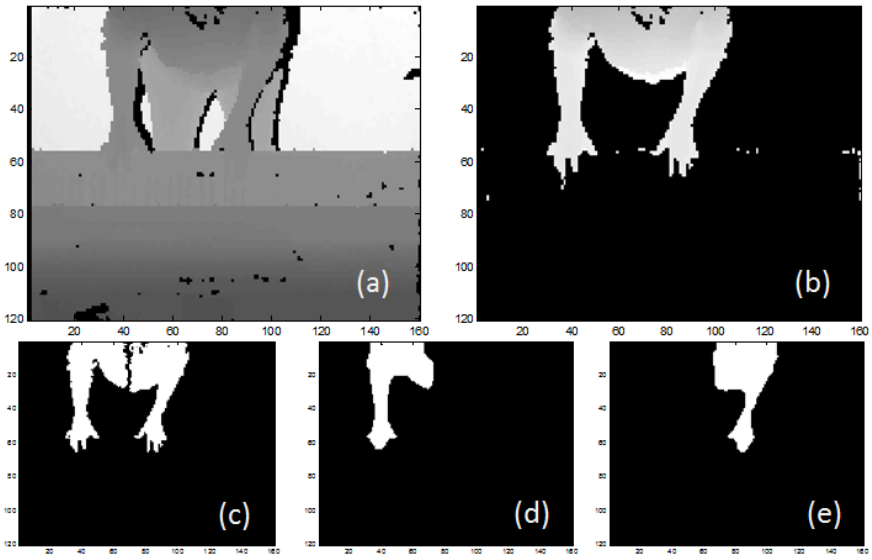

Figure 28 Preprocess for wrist detection

(a) raw depth image, (b) segmented foreground, (c) separated left and right side, (d, e) left and right mask after closing operation.

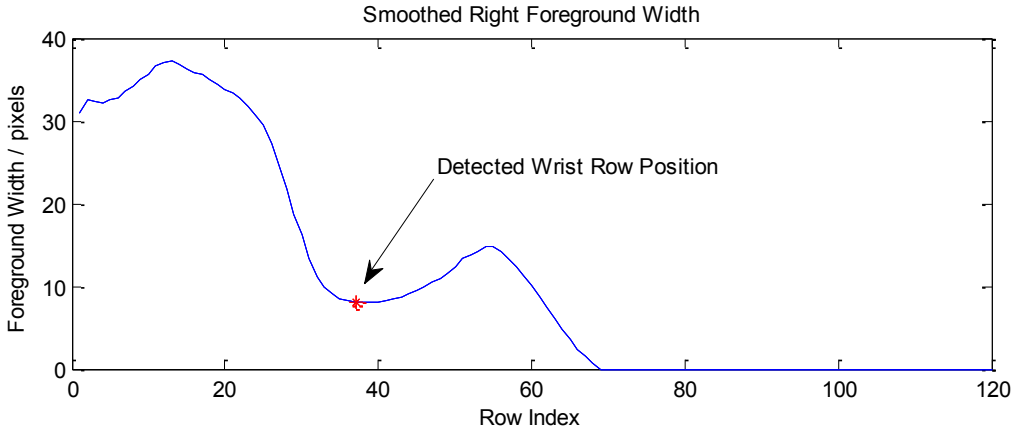

Figure 29 Smoothed right foreground width

The x-axis represents the row index in the depth image (down sampled) from the top row to the bottom; the y-axis represents the number of foreground pixels in each row as the width of foreground. The blue curve shows the distribution of the foreground width. The valley position nearest to the bottom is detected as the wrist position marked as a red star.

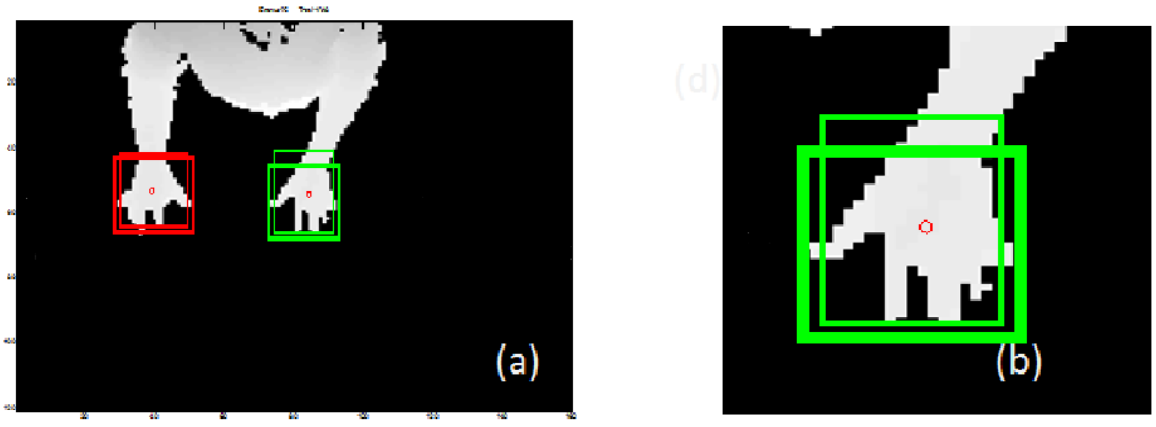

Figure 30 Hand crop

(a) Hand bounding boxes. Red and green boxes represent the left and right hand, respectively. The thin boxes show the candidate bounding boxes, and the bold boxes show the adjusted bounding boxes with a better accuracy. (b) The zoomed in view for (a).

4.2.3 Landmarks Detection

For a hand posture motion capture, there are some landmarks we would like to find: wrist center, hand center, fingertip of the longest finger and a point in the mid forearm.

After we get the hand segmented out, the wrist center position can be found in the center of the wrist row, hand center can be found as the centroid of the hand mask, hand orientation can be detected along that direction, and fingertip of the longest finger can be found. Then from the wrist row up, we can get part of the forearm with a width no larger than 1.5 times the wrist width. Figure 31 shows the result of the landmark detection.

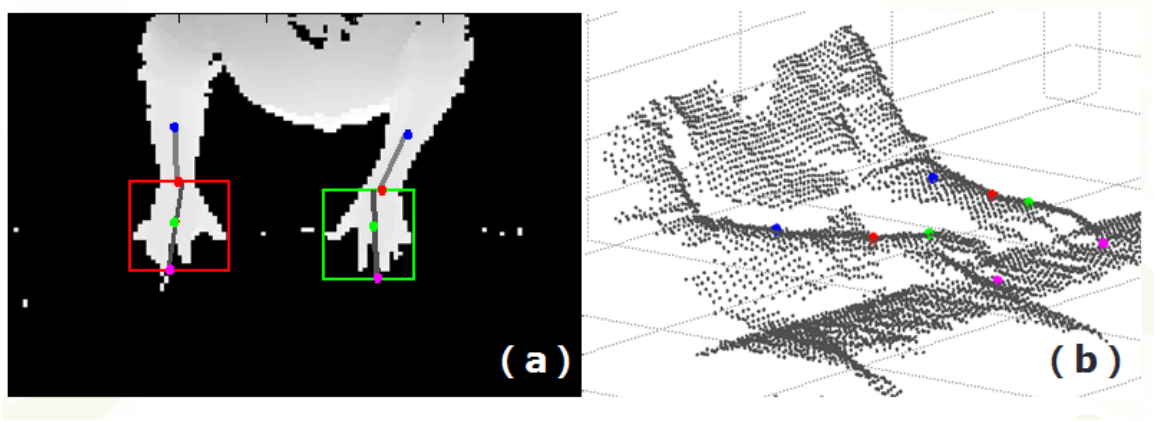

Figure 31 Landmarks detection

(a) Segmented left and right hands in the red and green bounding boxes with detected landmarks. The pink, green, red and blue dots represent fingertips of the longest finger, hand center, wrist center and a point in the mid forearm, respectively. (b) Reconstructed 3D point cloud. The pink, green, red and blue dots represent fingertips of the longest finger, hand center, wrist center and a point in the mid forearm, respectively.

4.3 Feature Extraction

4.3.1 Side View

For hand postures with problems of collapsed knuckles, wrist extension and wrist flexion, the side view outline of the hand gives a nice representation of the problem. Figure 32 below illustrates the side view of those hand postures mentioned above.

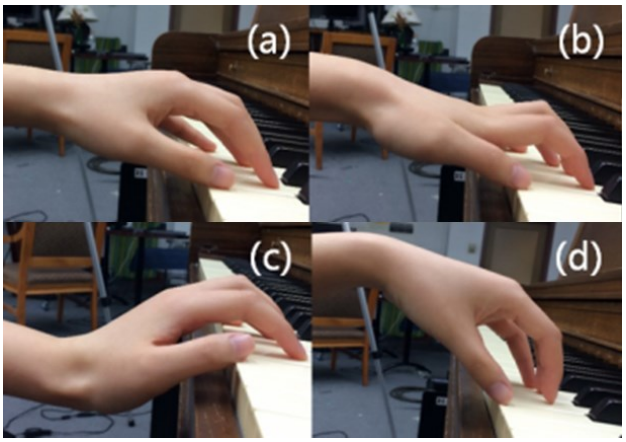

Figure 32 Side view outline of the different hand postures.

(a) neutral; (b) collapsed knuckles; (c) wrist flexion; (d) wrist extension.

The following two features are extracted from the side view:

- *Hand Center Height- Hand Arch Height Ratio (CAR) Feature*

For the collapsed knuckles problem, we propose the CAR (Hand Center Height- Hand Arch Height Ratio) feature.

Following steps are used for calculating the CAR feature:

1) After segmenting the hand, do a 3D reconstruction of the hand pixels using the keyboard coordinate system with the keyboard plane as the zero height plane.
2) Use the highest point in each row of the segmented hand sample as the side view hand outline height as shown in Figure 33. Calculate the height of each point along

the side view outline to the line defined by the wrist and finger to get the side view hand shape, shown in Figure 34 (a, b and c).

3) Calculate the CAR feature by taking the ratio between the hand center height, H(c), and the arch peak height, H(p).

$$CAR = H(c) / H(p)$$

4) Normalize *CAR* into a range of 0 to 1 with a normalized information distribution as shown below.

$$CAR = e^{CAR-1}$$

Here is the reason for applying step 4 to do the normalization after step 3. The CAR value from step 3 can range from a negative value to positive 1. Form our experiments, the negative value can be as small as -5. Figure 35 shows an example for a hand sample with a negative CAR. It is found that the negative CAR carries much less information than the CAR in range of 0 to 1. In other words, there is much less difference between negative CARs than between positive ones. For example, when the CAR drops from 0.8 to 0.2, it can be a change from neutral to strong collapse while then CAR drops from -1 to -4 is just from a strong collapse to an even stronger collapse. To solve this problem, we normalize the CAR into a range of 0 to 1 as introduced in step 4.

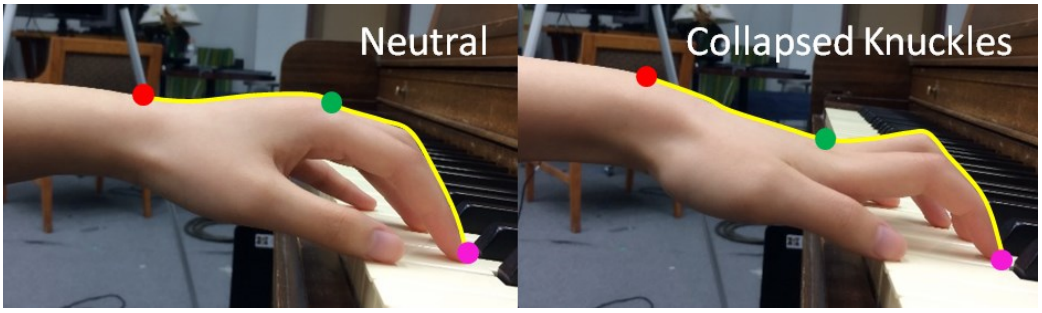

Figure 33 Hand side view

The red, green and pink dots represent wrist, hand center and fingertips respectively.

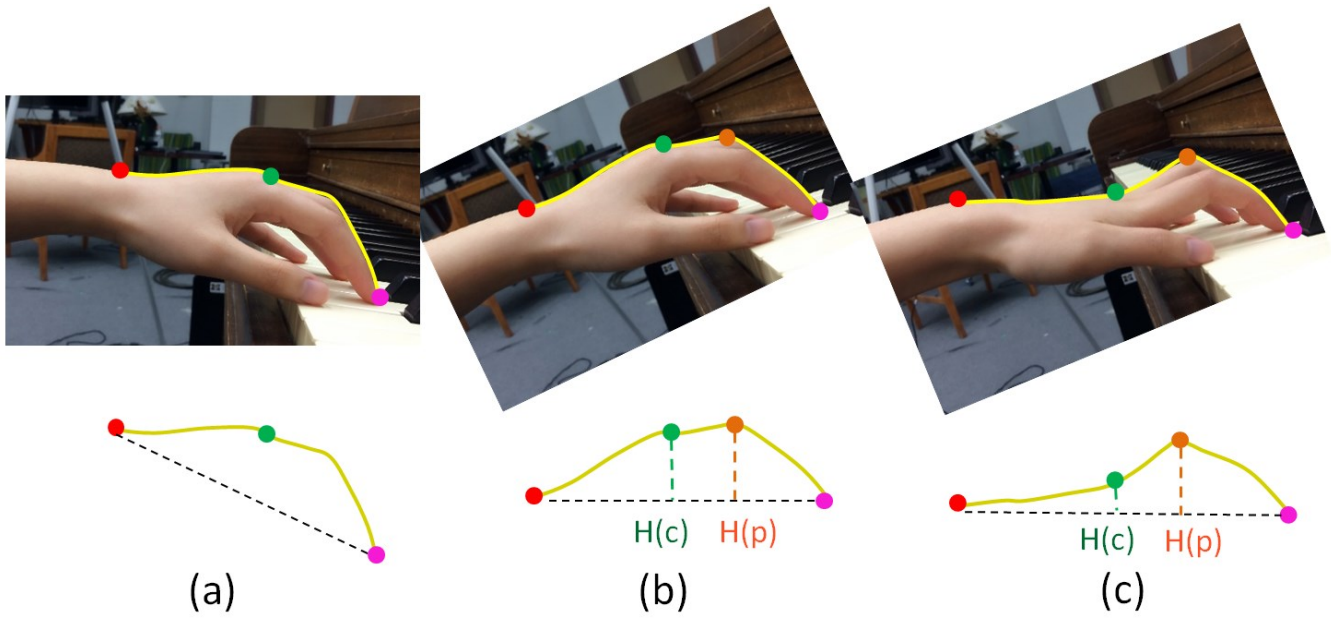

Figure 34 CAR (Hand Center Height- Hand Arch Height Ratio) feature.

(a) Hand side view; (b) Hand arch shape side view of neutral hand posture; (c) Hand arch shape side view of collapsed knuckles hand posture. The red, green and pink dots represent the wrists, the hand center and the fingertips respectively while the orange dots represent the hand arch peak.

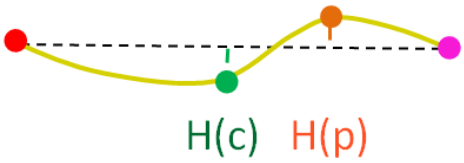

Figure 35 A side view example for sample with a negative CAR

For the neutral hand posture, the CAR is close to 1, while for the collapsed knuckles hand postures, the CAR is smaller. The CAR feature gives a representation of the degree of collapse in the knuckles.

- *Wrist Angle –Vertical (WAV)*

The vertical wrist angle gives a good representation of the wrist extension and flexion degrees. The angle between the forearm extension line and the hand extension line in the vertical plane is calculated as our vertical wrist angle. The flexion yields the positive values while the extension yields the negative values, shown in Figure 36.

The forearm extension line is defined by the landmark on the mid forearm and the wrist center. The hand extension line is defined by the wrist center and the hand center. And our landmarks sit on the surface of the arm and hand.

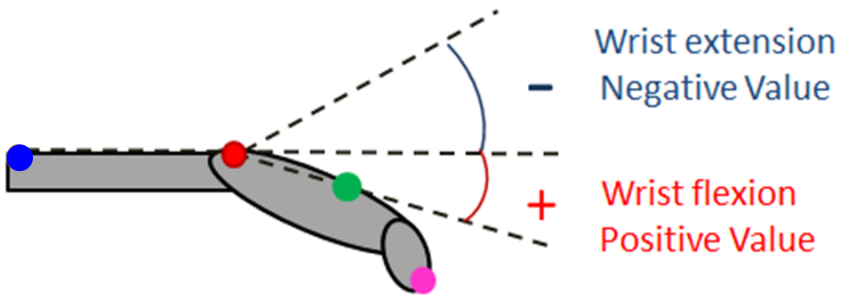

Figure 36 WAV (Wrist Angle –Vertical) feature

Blue, red, green and pink dots represent points in the mid forearm, wrist center, hand center and fingertip.

4.3.2 Top View

- *Wrist Angle – Horizontal (WAH)*

For ulnar and radial deviation we use a similar method as in calculating vertical wrist angle but now in the horizontal plane. The ulnar deviation yields positive values while the radial deviation yields negative values, shown in Figure 37. The 3D coordinates of the mid

forearm center, wrist center and the fingertip of the longest finger are used to calculate the angle.

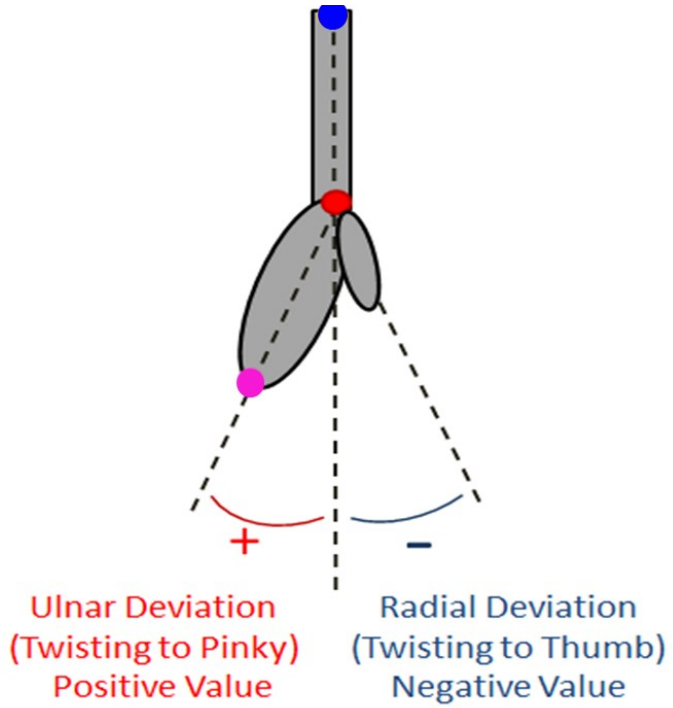

Figure 37 WAH (Wrist Angle – Horizontal) feature

Blue, red and pink dots represent points in the mid forearm, wrist center and fingertip.

4.4 Temporal Smoothing

Tracking 3D objects from 2D image data often leads to jittery tracking results due to ambiguities in the image data. [44] In order to reduce the noise and get a more stable and smooth result, a temporal smoothing is applied for the hand segmentation, landmark detection and feature calculation approaches. A smoothing window of 10 frames (1/3 second) was used.

4.5 Classifier Design

As soon as the feature set is determined, the next step for hand posture recognition is classification. There are various classifiers/ clustering methods with different properties. A proper classifier/ clustering method should be selected for a specified problem.

4.5.1 Several Commonly Used Classifiers/ Clustering Methods

Here are some brief introductions of several classifiers/ clustering methods that are used in this research.

Naïve Bayes Classifier

Naïve Bayes (NB) classifiers are a family of probabilistic classifiers based on applying Bayes' theorem. It assumes that there is a strong independence between the features. [31] The classification label of a sample is based on the probability it belongs to the class. One advantage of NB classifier is that it is highly scalable, requiring a number of parameters linear in the number of features and predictors. Also, depending on the precise nature of the probability model, a Naïve Bayes classifier can be trained very efficiently by a supervised learning process. Maximum-likelihood training can be done by evaluating a closed-form expression, [32] while many other types of classifiers using expensive iterative approximation. A comprehensive comparison with other classification algorithms showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests. [33] Although it is not a new theory, Naïve Bayes classifier is still a widely used machine learning approach.

Fuzzy C-Means

Fuzzy c-means (FCM) is a data clustering technique in which a dataset is grouped into N clusters with every sample in the dataset belonging to every cluster to a certain degree rather

than completely belonging to just one cluster. This means, for points on the edge of a cluster, they may belong to that cluster, but with a lesser degree than points in the center of the cluster. The degree of belonging is affected by the distance from the sample to the center of the cluster. This technique was originally introduced by James Bezdek in 1981[34] as an improvement on earlier clustering methods. The algorithm minimizes intra-cluster variance. There are some limitations of this method such as the minimum is a local minimum, and the results depend on the initial choice of weights, which is very similar to k-means.[34] It is an unsupervised machine learning method and gives not only the clustering result but also the degree. It is widely used in pattern recognition problems where people are interested in partial membership levels of the observed samples to the clusters.

Logistic Regression

Logistic Regression is a type of probabilistic statistical classification model. [35] It is used to predict a binary response from a binary predictor based on one or more features. Logistic regression measures the relationship between the categorical dependent variable and one or more independent variables, which are usually but not necessarily continuous, by using probability scores as the predicted values of the dependent variable.[36] The probabilities describing the possible outcomes of a single trial are modeled as a function of the features, using a logistic function. It is used widely in many fields, including the medical and social sciences. [35] [37] [38]

4.5.2 Multi-label Classification

In some real-world problems, one sample can belong to more than one class. Multi-label classification methods are increasingly required by modern applications such as protein function classification, music categorization and semantic scene classification.[39] In

traditional single-label classification, each example is associated with a single label from a set of disjoint labels L. On the other hand, in multi-label classification, the example can be associated with a set of labels $Y \subseteq L$.[39] Thus, multiple target labels can be assigned to each instance.

There are two main methods for tackling the multi-label classification problem: problem transformation methods and algorithm adaptation methods. Problem transformation methods transform the multi-label problem into a set of binary classification problems, which can then be handled using single-class classifiers. [39][40] Algorithm adaptation methods adapt the algorithms to directly perform multi-label classification. There are some classification algorithms/models that have been adapted to the multi-label task such as boosting, k-nearest neighbors [41], decision trees [42], neural networks [43] etc.

In this research, the problem we want to solve is also a multi-label classification problem. For each sample, it can belong to one or multiple classes. For example, a sample can be neutral or have only one problem while another sample can have a combination of collapsed knuckles and ulnar deviation.

The problem transformation method is applied here because of the characteristics of the features we use. The three features extracted are all physically meaningful. For example, the WAV feature is actually the wrist angle in the vertical plane in degrees. According to its physically meaning, it is naturally the dominate feature for class wrist flexion and extension. This also happens to the other two features. Thus, it is straight forward to build up a multi-label classifier with parallel sub-classifiers using the dominate feature/ features for each class. Table 4 below shows the dominate feature/ features for each class. One thing to note:

for class *'neutral'* all features are important since an actual *'neutral'* sample has to be neutral in all dimensions.

Table 4 Dominate feature/ features for each class

Class Name	Dominate Feature/ Features
Neutral	CAR, WAV and WAH
Collapse	CAR
Extension	WAV
Flexion	WAV
Ulnar	WAH
Radial	WAH

Each sub-classifier is a binary classifier which gives a YES or NO label for that particular class. A block diagram of the classifier structure is shown below in Figure 38. Features are input to the binary sub-classifiers to get labels of different classes.

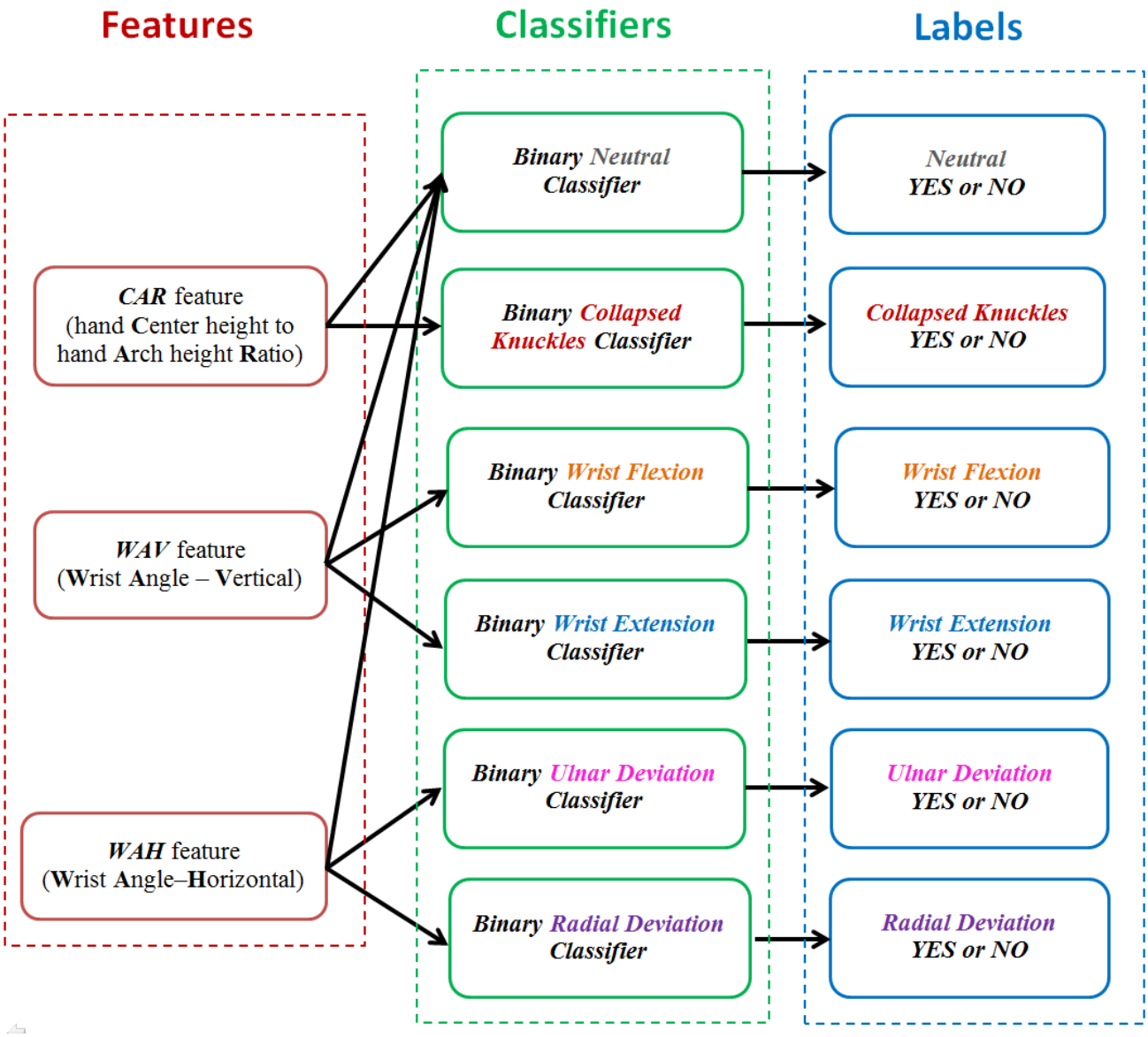

Figure 38 Classifier structure

4.5.3 Using FCM Clustering Results for Training a Naïve Bayes Classifier

The FCM is used to cluster the data into training sets with known labels. Classification is then done using a Naïve Bayes classifier using this dataset; we will refer to this as the unsupervised learning (UL) Naïve Bayes classifier. The labels are assigned to each cluster

by analyzing the cluster center position. As discussed in section 4.4.2 above, there is a sub-classifier for each class, and each sub-classifier is a binary classifier which gives a YES or NO label for the class. For example, for the 'wrist flexion' class, the FCM algorithm will cluster the samples into two clusters. The cluster with the cluster center of a larger WAV value will be assigned as the *'wrist flexion'* class while the cluster with the cluster center of smaller WAV value will be assigned as the *'non wrist flexion'* class. This is because samples of the *'wrist flexion'* class obviously have larger WAV values than those of the *'non-wrist flexion'* class according to the physical meaning of WAV as the wrist angle in the vertical plane introduced in section 4.2.1. A similar method is used for the other classes as summarized in Table 5 below. The *'neutral'* class is more challenging and is handled differently. For the *'neutral'* class, samples are in the center of the feature space with the *'non-neutral'* samples located around the *'neutral'* samples. This makes the clusters less separable and can lead to a misleading cluster center of the *'non-neutral'* samples. To address the second problem, the average absolute feature values of the samples in each cluster (neutral or non-neutral) are used to assign the cluster to the corresponding class. The cluster in which the samples have a larger CAR value and smaller |WAV|* and |WAH |* value will be regarded as the *'neutral'* class.

Table 5 Basis for mapping clustering results into binary classification results

Class Name A	Mapping Basis	A	NOT A
Neutral	average absolute feature value	larger CAR, smaller |WAV|* smaller |WAH|*	smaller CAR, larger |WAV|* larger |WAH|*
Collapse	cluster center position	smaller CAR	larger CAR
Extension	cluster center position	smaller WAV	larger WAV
Flexion	cluster center position	larger WAV	smaller WAV
Ulnar	cluster center position	larger WAH	smaller WAH
Radial	cluster center position	smaller WAH	larger WAH

*** '| |' means taking the absolute value.**

Chapter 5 Experiments

5.1 Data Collection

An IRB-approved study was conducted to collect data from 2 professional pianists (1 male and 1 female) and 10 student pianists (2 males and 8 females) in the School of Music, University of Missouri. Three different types of data were collected. Participants were first asked to hold static hand postures for 10 seconds each. Secondly, participants were asked to slowly move through the different positions for 4 cycles, as outlined below. For example, participants moved continuously from a neutral position to the collapsed knuckles position and back to the neutral position for 4 cycles. Thirdly, participants were asked to play five music selections in their normal hand postures. These pieces were selected to elicit a range of hand postures, including misaligned hand positions, e.g., from students who may use poor hand postures.

- Static samples *(Hold the hands for 10 seconds)*
    - Neutral Position
    - Collapsed Knuckles
    - Wrist Extension
    - Wrist Flexion
    - Ulnar Deviation (Deviation toward Pinky)
    - Radial Deviation (Deviation toward Thumb)
- Continuous samples

  (Move back and forth slowly for at least 4 cycles)
    - Collapsed knuckles to neutral position

- Wrist extension to flexion through neutral position
- Ulnar to radial deviation through neutral position

- Selected pieces

  (Play each three times at a given speed)

  - Chord progression in C major

  - D major Scale (two octaves)
  - D major Arpeggio (two octaves)
  - Chorale “Rejoice, O My Soul”, Op. 68, no. 4 (R. Schumann)
  - Little Prelude No.2 in C major, BWV 939 (J.S. Bach)
  - “Little Study”, Op.68, No. 14 (R. Schumann)

5.2 Ground Truth Labeling

After the data are recorded, a manual labeling approach is applied in order to get the ground truth. An interface was created for a piano professor to review the recorded data and label the classes for each sample. Figure 39 shows a screenshot of the interface.

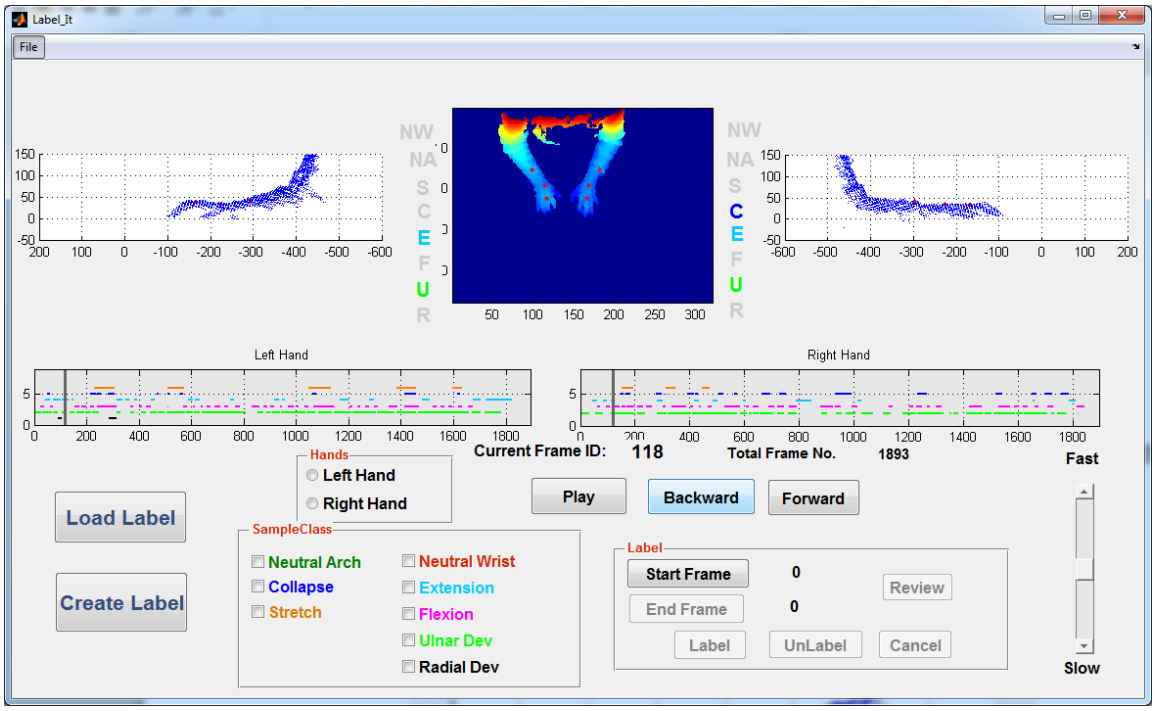

Figure 39 Ground truth labeling interface over view

By using this labeling interface, the reviewer can label the collapse in knuckles, wrist extension and flexion by looking at the 3D side view point cloud shown as a zoomed in figure below in Figure 40. In Figure 40, the blue point cloud shows the left half of the player including the left part of the torso, left arm and left hand. The three red dots represent the landmarks of hand center, wrist center and forearm center.

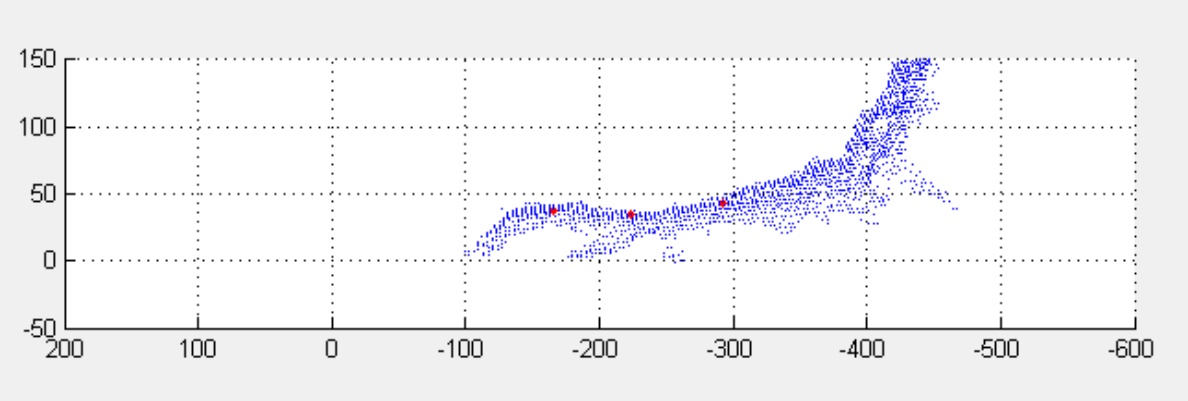

Figure 40 Interface - side view of left part of the player

The interface also shows the height map of the player as a top view from which the reviewer could recognize ulnar and radial deviations. Figure 41 below gives a zoomed in view of the top view. The red dots represent the landmarks the same as in Figure 40.

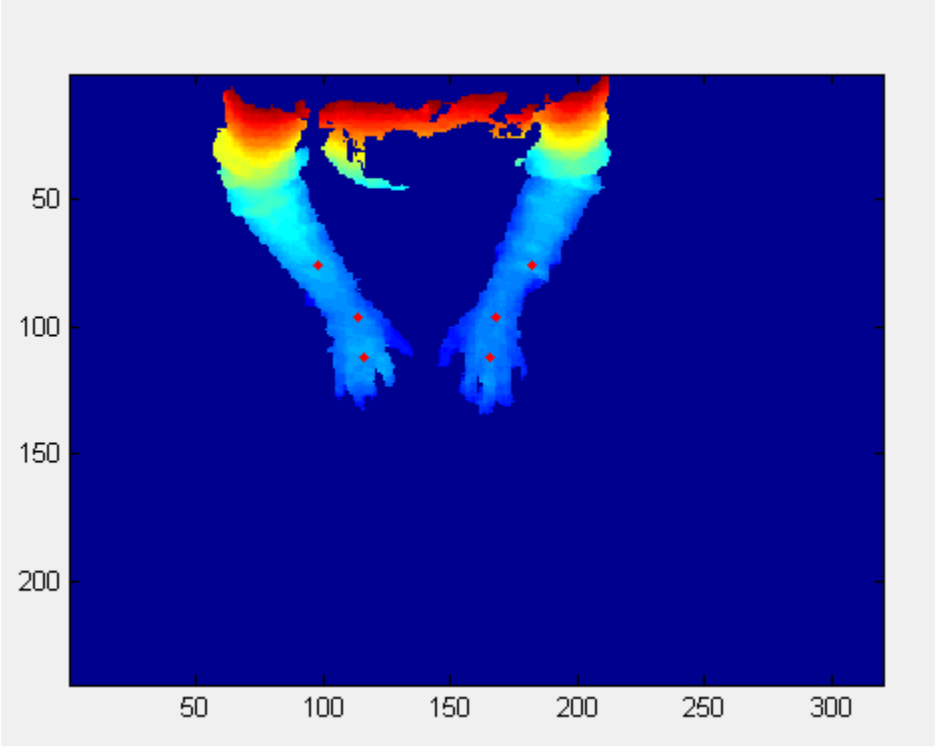

Figure 41 Interface - top view of foreground (player)

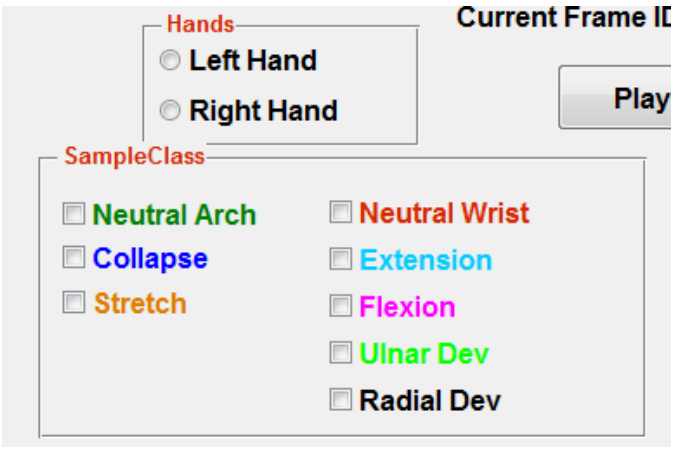

Figure 42 Interface - hand and class selection panel

Figure 42 shows the hand and class selection panel for the reviewer to select a particular class for a particular hand which he/ she wants to label. The different classes are assigned different colors to make it easier for the reviewer to recognize and match up to the colored labels shown in Figure 43 below.

In Figure 43, the x-axis represents the frame index and the vertical gray bar is a slice bar which the reviewer can drag to any frame he/ she wants to see. The colored horizontal bars represent the labels for the frames. The different colors distinguish the type of the class. For example, the green bars represent the Ulnar Deviation which is also colored in green in the hand and class selection panel in Figure 42.

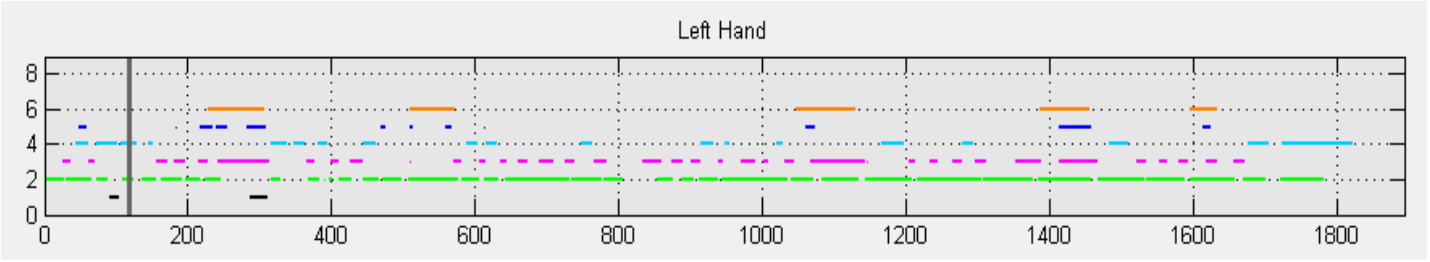

Figure 43 Interface - colored labels

Using this labeling interface, we have the selected piece **Chorale "Rejoice, O My Soul"** of all the 15 subjects labeled frame by frame by the piano professor.

Chapter 6 Results and Discussion

This chapter discusses the results of the experiments described in Chapter 5.

6.1 Neutrals and Extremes of Different Subjects

In this research, 5 different misaligned hand postures in total (collapse in knuckles, wrist extension, wrist flexion, ulnar deviation and radial deviation) are discussed. In this section, results of the proposed features for determining misaligned hand postures from a neutral position will be presented, using the data collection described in Section 5.1.

We first present an example of the continuous, slow movement through different hand postures. Figure 44 Continuous movements in feature space shows the movements in the proposed feature space. The red traces represent neutral position samples. The green traces represent the back and forth movement from neutral to collapsed knuckles; the blue traces show movement from wrist extension to flexion through the neutral position; the orange traces show the movement from ulnar deviation (toward pinky) to radial deviation (toward thumb).

We can clearly see the trace of the movements in the feature space. A control variate method was used here to allow movement along one axis only. For example, in the movement between neutral and collapse, we control the wrist posture such that flexion, extension and twisting are minimized. As shown in Figure 44, the green trace moves along the *CAR* axis, roughly perpendicular to the *WAV*-*WAH* plane. Similarly, the blue trace shows a back and forth movement along the *WAV* axis from extreme wrist extension to extreme flexion passing through the neutral. The orange trace shows a change along the *WAH* axis.

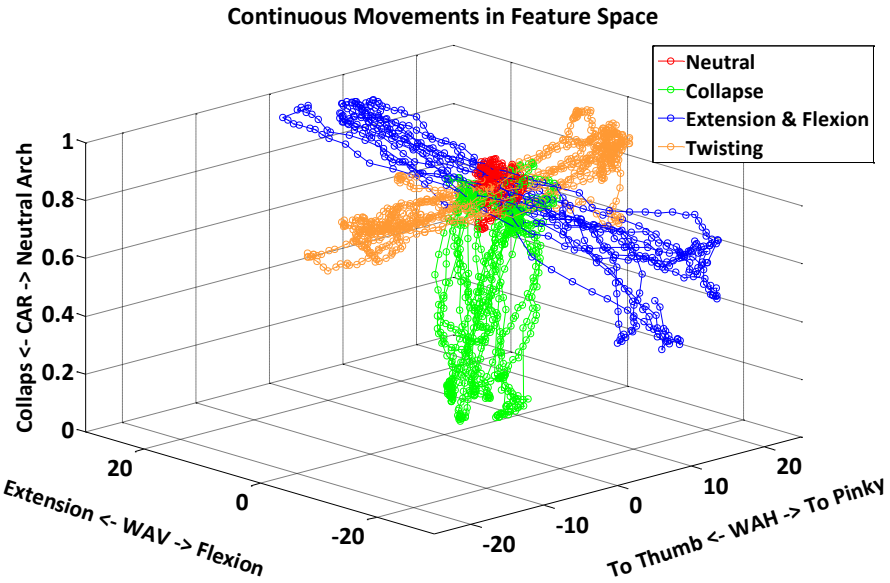

Figure 44 Continuous movements in feature space

The three axes are *WAV* in degrees, *WAH* in degrees and *CAR*. The red shows the neutral hand posture while the green trace represents back and forth movement from neutral to collapse. The blue trace represents back and forth movement from wrist extension to flexion passing through the neutral; flexion has a positive *WAV* value. The orange trace represents back and forth movement between ulnar deviation (toward pinky) and radial deviation (toward thumb) passing through the neutral; ulnar deviation has a positive *WAH* value.

In the time domain, there should be waves of features corresponding to the continuous back and forth movements.

Figure 45 shows the feature waves of one of the subjects. The CAR wave represents the CAR feature of the continuous movement back and forth form the neutral position to collapsed knuckles; the WAV wave represents the WAV feature of continuous movement back and forth form wrist extension to flexion through neutral position; the WAH wave

represents the WAH feature of continuous movement back and forth form ulnar to radial deviation through neutral position.

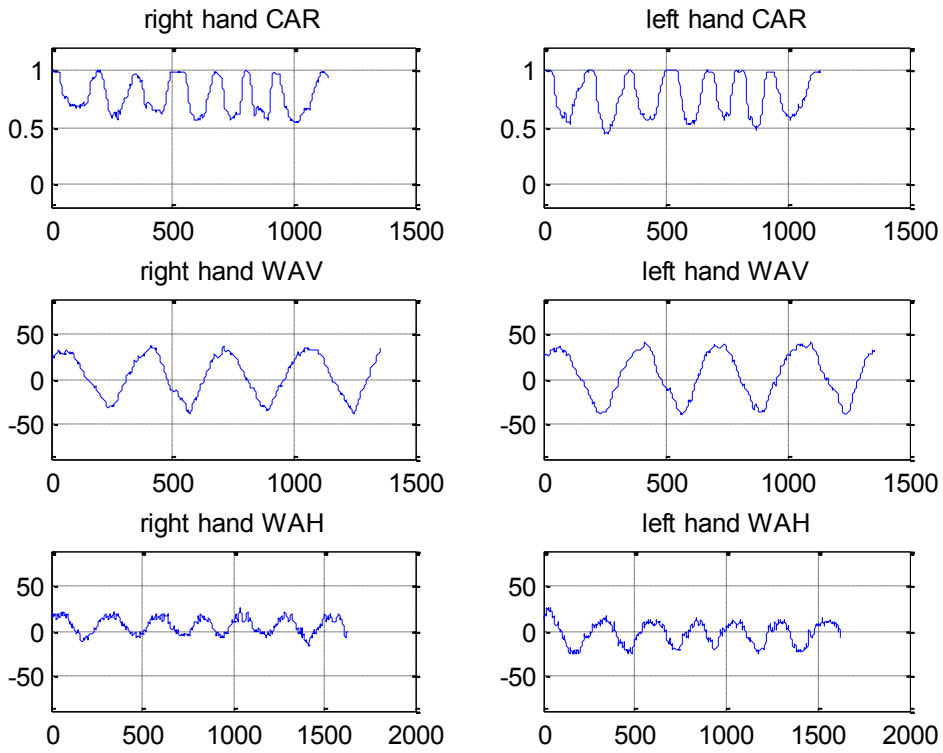

Figure 45 Time domain feature waves

The three rows represent the CAR, WAV and WAH waves respectively while the left and right columns represent the left and right hand. For each subplot, the x-axis represents the frame index along the time with a frame rate of 30fps; the y-axis represents the feature value. The CAR feature value is shown within range between 0 and 1; the WAV and WAH values are shown within range between -90 to 90 degree.

From the peak and valley values of the waves, the extremes of the subject can be detected. By calculating the maximum and minimum value of each feature throughout the continuous movements, we get the extremes. By calculating the mean value of each feature

of the demonstrated neutral positions mentioned in Section 5.1 static samples *(Hold the hands for 10 seconds)*, we get the neutrals for each subject.

Feature values for Neutral and Extremes of 15 subjects are illustrated in Figure 46 and Figure 47. The x-axes show the subject index and the y-axes show the feature values. The red dots represent the neutral feature, while the green crosses and blue crosses show the extreme features.

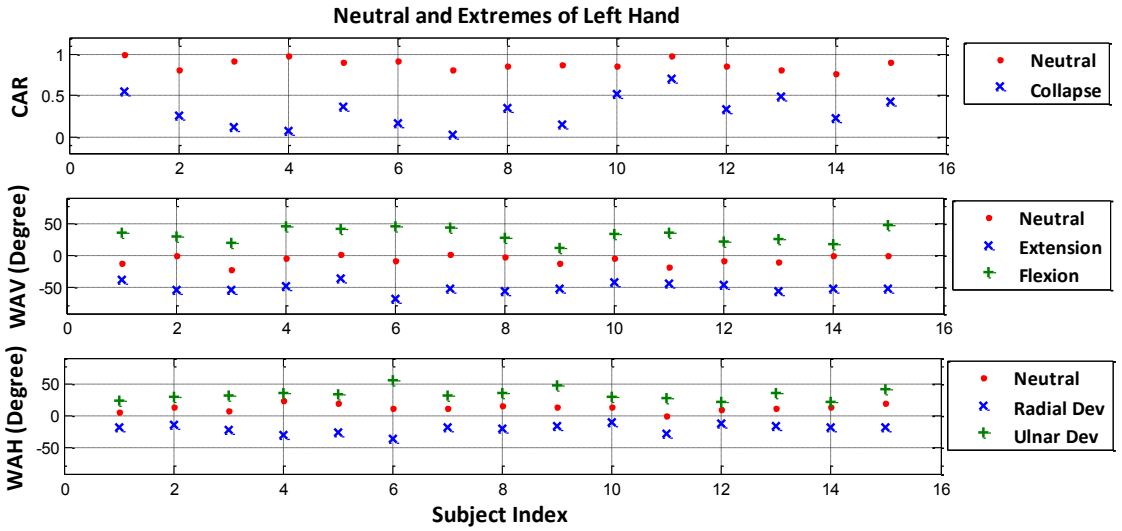

Figure 46 Neutrals and extremes of different subjects – left hand

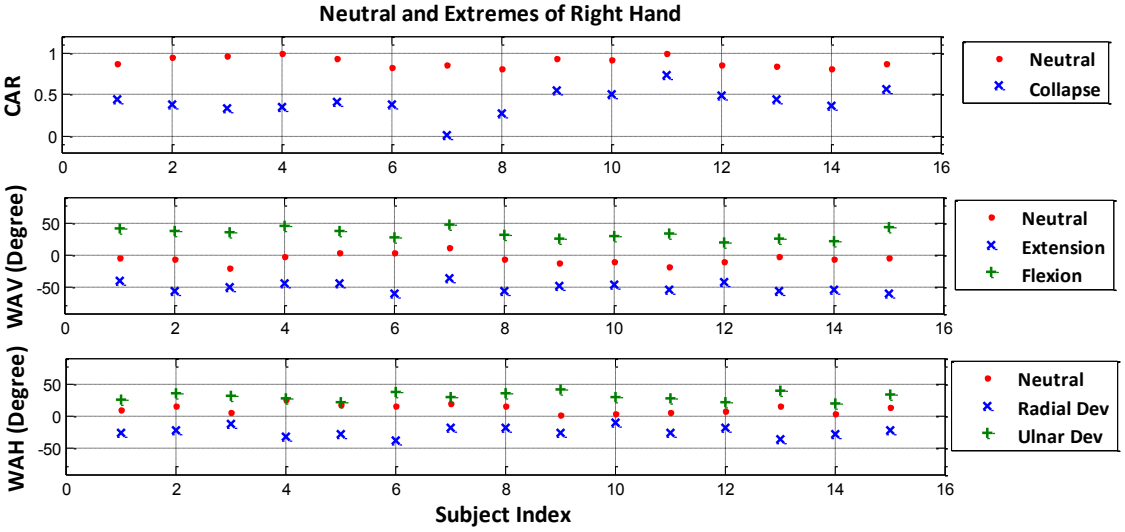

Figure 47 Neutrals and extremes of different subjects – right hand

The following three tables give the numeric result of the neutrals and extremes of all the 15 subjects for each feature.

Table 6 Neutrals and extremes for CAR feature

Subject Index	Left Hand		Right Hand	
	Neutral	Collapse Extreme	Neutral	Collapse Extreme
1	0.99	0.54	0.87	0.44
2	0.81	0.25	0.93	0.38
3	0.91	0.12	0.95	0.32
4	0.97	0.07	0.99	0.35
5	0.9	0.36	0.92	0.41
6	0.92	0.17	0.82	0.37
7	0.8	0.02	0.85	0.01
8	0.86	0.34	0.8	0.27
9	0.87	0.14	0.93	0.55
10	0.85	0.51	0.91	0.51
11	0.97	0.7	0.99	0.73
12	0.85	0.33	0.85	0.49
13	0.81	0.5	0.84	0.44
14	0.76	0.23	0.8	0.37
15	0.9	0.42	0.86	0.56

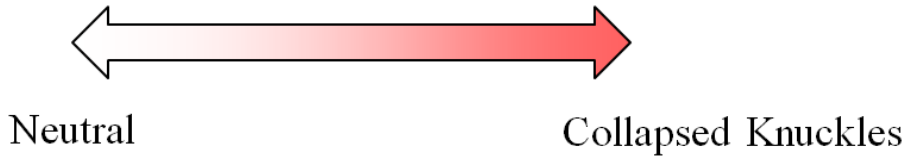

Table 6 shows the neutrals and extremes for the *CAR* feature which has a range of 0 to 1. The closer to 1, the sample is more likely to be neutral; the closer to 0, the sample has more severe collapsed knuckles.

Table 7 Neutrals and extremes for WAV feature

Subject Index	Left Hand			Right Hand		
	Neutral	Extension Extreme	Flexion Extreme	Neutral	Extension Extreme	Flexion Extreme
1	-12.4	-37.9	36.5	-4.2	-39.8	40.9
2	-0.7	-53.7	30.2	-6.0	-55.2	38.0
3	-21.1	-53.0	20.6	-19.2	-49.2	35.3
4	-3.5	-47.9	44.7	-2.4	-43.8	45.1
5	2.2	-36.4	41.5	4.3	-43.6	37.2
6	-7.3	-66.9	46.2	3.9	-59.8	27.8
7	1.3	-50.5	43.3	12.3	-34.6	48.4
8	-1.4	-55.7	27.1	-6.8	-56.0	32.2
9	-12.4	-50.8	11.5	-10.9	-48.4	25.8
10	-4.6	-41.9	33.2	-10.7	-45.9	30.3
11	-17.7	-42.7	35.7	-18.4	-52.5	33.8
12	-7.2	-45.0	22.7	-9.1	-42.0	19.7
13	-10.0	-54.8	26.3	-2.6	-56.1	25.5
14	0.6	-51.6	16.9	-5.1	-53.9	21.3
15	0.7	-51.2	46.5	-3.1	-59.4	43.7

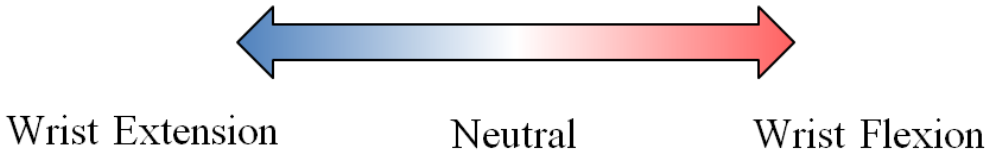

Table 7 shows the neutrals and extremes for *WAV* feature which has a range of -90 to 90 degrees. Flexion yields positive *WAV* values, while extension yields negative *WAV* values. The smaller the *WAV* is, the more severe the wrist extension is; the larger the *WAV* is, the more severe the wrist flexion is.

Table 8 Neutrals and extremes for WAH feature

Subject Index	Left Hand			Right Hand		
	Neutral	Radial Dev Extreme	Ulnar Dev Extreme	Neutral	Radial Dev Extreme	Ulnar Dev Extreme
1	5.5	-17.0	22.8	9.2	-25.6	25.0
2	14.6	-14.8	30.4	15.2	-22.3	34.7
3	7.5	-21.7	32.3	5.0	-12.7	30.7
4	24.3	-28.9	36.5	25.8	-32.6	27.7
5	19.8	-25.7	33.0	17.2	-27.3	22.1
6	12.3	-35.2	55.2	16.3	-37.7	38.3
7	11.5	-18.5	31.9	19.1	-18.7	29.0
8	16.0	-19.2	35.9	16.6	-17.6	35.7
9	13.8	-15.9	47.3	2.2	-25.7	40.7
10	13.3	-8.9	29.5	3.5	-9.3	28.9
11	0.3	-27.6	27.8	5.7	-26.3	27.8
12	10.7	-12.1	21.9	7.6	-17.5	21.7
13	11.1	-16.2	35.0	16.3	-35.0	39.3
14	13.4	-18.3	21.1	3.2	-27.4	19.4
15	19.3	-17.0	41.5	13.2	-22.5	33.0

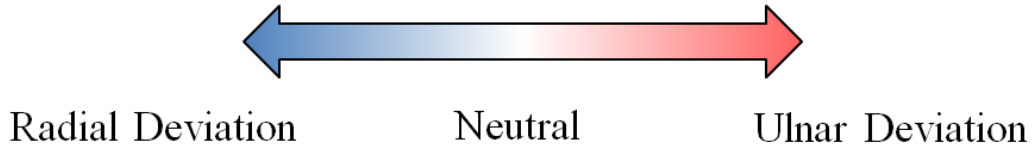

Table 8 shows the neutrals and extremes for *WAH* feature which has a range of -90 to 90 degrees. Ulnar deviation yields positive *WAH* values, while radial deviation yields negative *WAH* values. The smaller the *WAH* is, the more severe the radial deviation is; the larger the *WAH* is, the more severe the ulnar Deviation is.

6.2 Classification Results for Selected Pieces

From the static hand postures and continuous hand movements, we have basic information of the subjects such as their neutrals and extremes. There are more interesting questions such as 'How do they perform in real playing?' and 'How do the features and the classifiers perform in representing the hand postures and detecting the potentially harmful hand postures?'

In this section, we will discuss the performance of the classifier.

6.2.1 Dataset

Because of the fact that the manual labeling is rather time consuming, it is hard to get all the selected pieces labeled by the piano professor. Thus, we picked one of the selected pieces **Chorale "Rejoice, O My Soul"** for the piano professor to do the labeling**.** This particular piece was selected because it gives the best coverage of the different hand motions among the 6 selected pieces.

We put all the labeled samples of **Chorale "Rejoice, O My Soul"** of all the 15 participated subjects together to get the dataset.

Table 9 below shows the information of the dataset including the classes and the number of positive and negative samples.

Table 9 Dataset information

Class Name	Number of Positive Samples	Number of Negative Samples
Neutral	2383	57740
Collapsed Knuckles	14334	45789
Wrist Extension	12912	47211
Wrist Flexion	17766	42357
Ulnar Deviation	51966	8157
Radial Deviation	206	59917

6.2.2 Result

After preparing the dataset, classification as discussed in Section 4.4 is applied. Different classification methods including Naïve Bayes, Fuzzy C-means and Logistic Regression, are used as the binary sub-classifiers for each class.

Table 10 illustrates the performance of each type of binary sub-classifier in both AUC and running time. It shows that all three methods give good classification results with AUC larger than 0.9 for all classes except for the neutral class.

The Naïve Bayes classifier gives both the best AUC and time efficiency. The average processing time including training and testing on a data set introduced in section 6.2.1 Datasetusing a 10-fold validation is less than 0.02s. Its ROC is shown in Figure 48 below.

The Logistic Regression classifier gives competitive performance to Naïve Bayes except slightly less AUC for the neutral class. However, its processing time is much longer (7~9s vs 0.02s).

The UL Naïve Bayes classifier also gives competitive performance to the other two methods for collapse, extension, flexion, ulnar and radial deviation, but the AUC for the neutral class is significantly lower (0.55). The results show that Fuzzy C-means does a poor job in clustering the neutral and non-neutral samples. Although there are limitations for Fuzzy C-means clustering, there are advantages to using an unsupervised learning method, as it does not require manually labeled ground truth. The method of using Fuzzy C-means clustering to get the training data is discussed in detail in section 4.4.3. This makes it a very promising way to get some preliminary labels for an expert to review and correct to reduce the human workload.

Table 10 10-fold cross validation*

	AUC			Time / s		
	NB	UL-NB	LR	NB	UL-NB	LR
Neutral	0.88	0.55	0.86	0.019	0.323	7.306
Collapse	0.92	0.92	0.92	0.010	0.254	6.743
Extension	0.96	0.96	0.96	0.012	0.260	7.875
Flexion	0.94	0.94	0.94	0.012	0.263	6.149
Ulnar	0.91	0.91	0.91	0.014	0.253	7.806
Radial	0.97	0.96	0.96	0.014	0.256	8.829

NB stands for Naïve Bayes classification; UL-NB stands for Unsupervised Learning Naïve Bayes classification and LR stands for Logistic Regression classification. Time is the total time for training and testing.

*The time results are from a Matlab program running on an Intel i7-3770 CPU, 3.4GHz with 32G RAM on 64bit Windows7 OS.

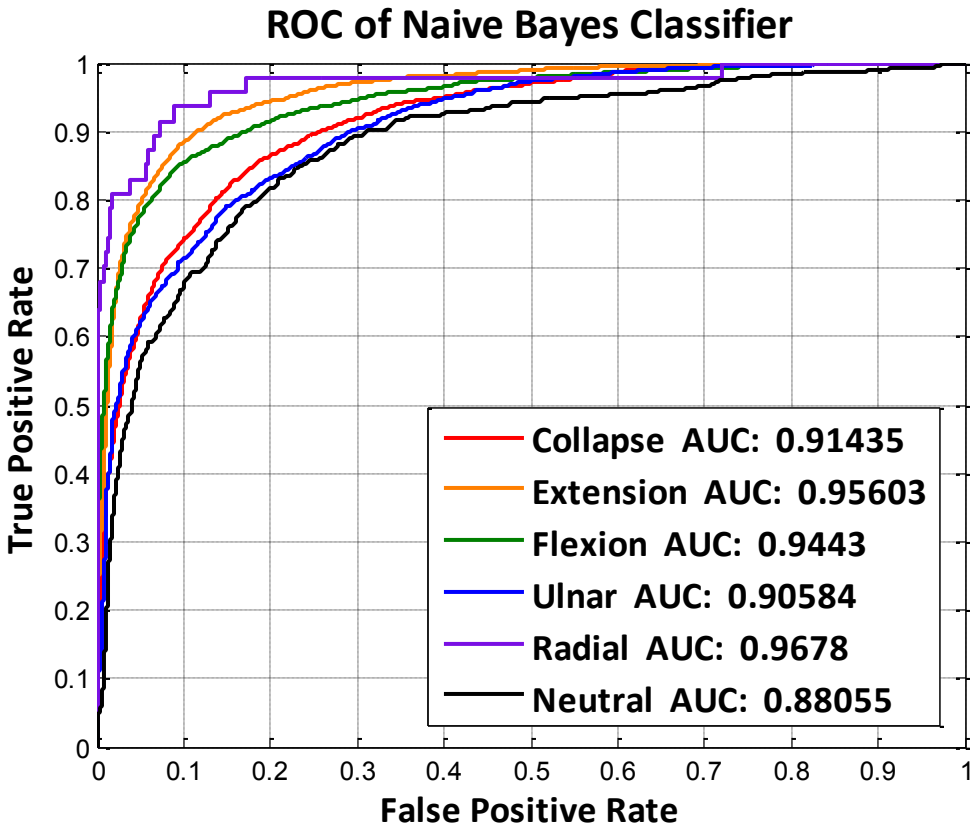

Figure 48 ROC of Naïve Bayes classification

6.3 Real-time Capability

It would be good if the system works in real time so it can be used for augmented piano and piano pedagogy applications.

Since a temporal smoothing approach is used in our system, there will be a time period as long as the temporal smoothing window length before the first smoothed result is generated. Thus, after half the size of the temporal smoothing window (in length of frames) which cannot get smoothed because of lack of prior information, the remaining frames can be processed in near real time [46] with a latency of half the temporal smoothing window length. The structure of our system is eligible for real-time.

The remaining problem is if the system can finish processing one frame before the next frame comes. Table 11 shows the run time of the approaches in our depth image processing. The total time to process one frame is 36.3 ms which is possible to provide real-time performance at a frame rate up to 27 fps. This is less than the actual possible maximum Kinect frame rate of 30 fps, but it still looks promising to achieve a real-time performance by doing code optimization or using a more effective programming language such as C++.

Table 11 Run time**

Approach	Run Time (ms)
Get Keyboard Plane*	8.5
Separate Arms	8.7
Get Bounding Box	10.1
Get Landmarks	8.3
Calculate Feature	9.4
Total except*	36.3

*This approach is not necessary for every frame since the Kinect is not supposed to be moved during the playing.

**The result is got from running the Matlab program on an Intel i7-3770 CPU, 3.4GHz with 32G RAM on 64bit Windows7 OS.

Chapter 7 Conclusions

Several achievements were obtained in this project, which include the following:

1) Designed a hand posture monitoring system using the Microsoft Kinect depth camera with a top-view setting for potentially harmful hand posture detection in pianists.
2) Developed a hand segmentation and hand landmark detection approach that automatically segments the hand sample from the wrist and gives 3D information of hand landmarks including hand center, fingertip of the longest finger, wrist center and forearm.
3) Proposed three features extracted from the depth data for hand posture classification.
4) Tested classifiers using both supervised and unsupervised machine learning methods including Naïve Bayes, Logistics Regression and Fuzzy C-means. Compared their performances and analyzed their limitations.
5) Collected real playing data from 15 subjects including 12 student pianists and 3 professor pianists, 4 males and 11 females on 6 selected pieces with various real playing hand motions.

Our experiments show that

1) The Kinect provides good sensing capabilities for capturing the hand postures of pianists. The setting is non-obtrusive and has no effect on the player's movement. It is portable and inexpensive which makes it accessible to many pianists.
2) The results show that the *WAH*, *WAV* and *CAR* features extracted from the

reconstructed 3D point cloud and landmarks extracted from the raw Kinect depth images can provide a good representation of the hand posture. They are promising for a screening assessment to detect misaligned hand postures, including postures that show a combination of misalignment types.

3) The classifier can achieve AUC around 0.94 by using the Naïve Bayes classifier which takes less than 0.02 second to train more than 50,000 samples. And the system depth image processing speed is 36.3 ms/frame which is promising for real-time use.

References

[1]. Schweitser, Vivien (Apr. 16, 2013, New York Times), *Drorthy Taubman, Therapist for Pianists, Dies at 95*. Retrieved on December 10th 2014 from http://www.nytimes.com/2013/04/17/arts/music/dorothy-taubman-95-dies-helped-pianists-avoid-injuries.html.

[2]. Mark, Thomas Carson. What every pianist needs to know about the body: A manual for players of keyboard instruments: piano, organ, digital keyboard, harpsichord, clavichord. GIA Publications, 2003.

[3]. Bragge, Peter, Andrea Bialocerkowski, and Joan McMeeken. "A systematic review of prevalence and risk factors associated with playing-related musculoskeletal disorders in pianists." Occupational Medicine 56.1 (2006): 28-38.

[4]. Parry, C. B. "Prevention of musicians' hand problems." Hand clinics 19.2 (2003): 317-324.

[5]. Mohamed, Safaa, Monique Frize, and Gilles Comeau. "Assessment of piano-related injuries using infrared imaging." Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE. IEEE, 2011.

[6]. Rosety-Rodriguez, M., Ordóñez, F. J., Farias, J., Rosety, M., Carrasco, et al. "The influence of the active range of movement of pianists' wrists on repetitive strain injury." European Journal of Anatomy 7.2 (2014): 75-77.

[7]. Neninger, C. R., Sun, Y., Lee, SH., Chodil, J. "A Complete Motion and Music Capture System to Study Hand Injuries among Musicians." Emergency Management & Robotics for Hazardous Environments: 1-11.

[8]. Allsop, Lili, and Tim Ackland. "The prevalence of playing-related musculoskeletal disorders in relation to piano players' playing techniques and practising strategies." Music Performance Research 3.1 (2010): 61-78.

[9]. Foxman, Irina, and Barbara J. Burgel. "Musician health and safety: Preventing playing-related musculoskeletal disorders." AAOHN journal: official journal of the American Association of Occupational Health Nurses 54.7 (2006): 309-316.

[10]. Sakai, Naotaka, and Satoshi Shimawaki. "Measurement of a number of indices of hand and movement angles in pianists with overuse disorders." Journal of Hand Surgery (European Volume) 35.6 (2010): 494-498.

[11]. Hadjakos, Aristotelis. "Pianist Motion Capture with the Kinect Depth Camera."Proceedings of the International Conference on Sound and Music Computing, Copenhagen, Denmark. 2012.

[12]. Oka, Akiya, and Manabu Hashimoto. "Marker-less piano fingering recognition using sequential depth images." In Frontiers of Computer Vision, (FCV), 2013 19th Korea-Japan Joint Workshop on, pp. 1-4. IEEE, 2013.

[13]. Yang, Qi, and Georg Essl. "Augmented piano performance using a depth camera." Ann Arbor 1001, no. 2012 (2012): 48109-2121.

[14]. https://www.leapmotion.com/setup; Accessed on December 10 2014.

[15]. http://www.redicals.com/wp-content/uploads/2014/04/leap-motion.jpg; Accessed on December 10th 2014.

[16]. https://www.youtube.com/watch?v=zXghYjh6Gro; Accessed on December 10th 2014.

[17]. Khoshelham, Kourosh, and Sander Oude Elberink. "Accuracy and resolution of kinect depth data for indoor mapping applications." Sensors 12.2 (2012): 1437-1454.

[18]. Ballan, Luca, et al. "Motion capture of hands in action using discriminative salient points." Computer Vision–ECCV 2012. Springer Berlin Heidelberg, 2012. 640-653.

[19]. Sudderth, Erik B., et al. "Visual hand tracking using nonparametric belief propagation." Computer Vision and Pattern Recognition Workshop, 2004. CVPRW'04. Conference on. IEEE, 2004.

[20]. De La Gorce, Martin, David J. Fleet, and Nikos Paragios. "Model-based 3d hand pose estimation from monocular video." Pattern Analysis and Machine Intelligence, IEEE Transactions on 33.9 (2011): 1793-1805.

[21]. Hamer, Henning, et al. "Tracking a hand manipulating an object." Computer Vision, 2009 IEEE 12th International Conference On. IEEE, 2009.

[22]. Oikonomidis, Iasonas, Nikolaos Kyriazis, and Antonis A. Argyros. "Tracking the articulated motion of two strongly interacting hands." Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012.

[23]. Rehg, James M., and Takeo Kanade. "Model-based tracking of self-occluding articulated objects." Computer Vision, 1995. Proceedings., Fifth International Conference on. IEEE, 1995.

[24]. Oikonomidis, Iason, Nikolaos Kyriazis, and Antonis A. Argyros. "Efficient model-based 3D tracking of hand articulations using Kinect." BMVC. Vol. 1. No. 2. 2011.

[25]. V. Athitsos and S. Sclaroff. Estimating 3D Hand Pose From a Cluttered Image. In CVPR, pages II–432–9. IEEE, 2003.

[26]. R. Rosales, V. Athitsos, L. Sigal, and S. Sclaroff. 3D hand pose reconstruction using specialized mappings. ICCV, pages 378–385, 2001.

[27]. Y. Wu and T. Huang. View-independent recognition of hand postures. In CVPR, volume 2, pages 88–94. IEEE, 2000.

[28]. Romero, Javier, H. Kjellstrom, and Danica Kragic. "Monocular real-time 3D articulated hand pose estimation." Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on. IEEE, 2009.

[29]. C. Keskin, F. Kirac, Y. E. Kara, and L. Akarun. Real time hand pose estimation using depth sensors. ICCV Workshop, pages 1228–1234, 2011.

[30]. Raheja, Jagdish L., Ankit Chaudhary, and Kunal Singal. "Tracking of fingertips and centers of palm using kinect." Computational Intelligence, Modelling and Simulation (CIMSiM), 2011 Third International Conference on. IEEE, 2011.

[31]. Friedman, Nir, Dan Geiger, and Moises Goldszmidt. "Bayesian network classifiers." Machine learning 29.2 (1997): 131-163.

[32]. Russell, Stuart, Peter Norvig, and Artificial Intelligence. "A modern approach." Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs 25 (1995).

[33]. Caruana, Rich, and Alexandru Niculescu-Mizil. "An empirical comparison of supervised learning algorithms." Proceedings of the 23rd international conference on Machine learning. ACM, 2006.

[34]. Bezdek, James C. Pattern recognition with fuzzy objective function algorithms. Kluwer Academic Publishers, 1981.

[35]. McCullagh, Peter. "Regression models for ordinal data." Journal of the royal statistical society. Series B (Methodological) (1980): 109-142.

[36]. Bhandari, Mohit, and Anders Joensson. Clinical research for surgeons. Thieme, 2011.

[37]. Boyd, Carl R., Mary Ann Tolson, and Wayne S. Copes. "Evaluating trauma care: the TRISS method." *Journal of Trauma-Injury, Infection, and Critical Care*27.4 (1987): 370-378.

[38]. Truett, Jeanne, Jerome Cornfield, and William Kannel. "A multivariate analysis of the risk of coronary heart disease in Framingham." *Journal of chronic diseases* 20.7 (1967): 511-524.

[39]. Tsoumakas, Grigorios, and Ioannis Katakis. "Multi-label classification: An overview." *International Journal of Data Warehousing and Mining (IJDWM)* 3.3 (2007): 1-13.

[40]. Read, Jesse, et al. "Classifier chains for multi-label classification." *Machine learning* 85.3 (2011): 333-359.

[41]. Zhang, Min-Ling, and Zhi-Hua Zhou. "ML-KNN: A lazy learning approach to multi-label learning." *Pattern recognition* 40.7 (2007): 2038-2048.

[42]. Madjarov, Gjorgji, et al. "An extensive experimental comparison of methods for multi-label learning." *Pattern Recognition* 45.9 (2012): 3084-3104.

[43]. Zhang, Min-Ling, and Zhi-Hua Zhou. "Multilabel neural networks with applications to functional genomics and text categorization." *Knowledge and Data Engineering, IEEE Transactions on* 18.10 (2006): 1338-1351.

[44]. Rosenhahn, Bodo, et al. "Online smoothing for markerless motion capture."*Pattern Recognition*. Springer Berlin Heidelberg, 2007. 163-172.

[45]. Bulas-Cruz, J., A. T. Ali, and Erik L. Dagless. "A temporal smoothing technique for real-time motion detection." *Computer Analysis of Images and Patterns*. Springer Berlin Heidelberg, 1993.

[46]. Federal Telecommunications Standards Committee. "Federal Standard 1037C: Glossary of Telecommunications Terms (FED-STD-1037C)." *National Communications System Technology Program Office, Arlington, Virginia*(1996).

Appendix A

Here are some figures illustrating the sample distribution (shown as distribution mean position and standard deviation) of different pieces for the 15 participating subjects mentioned in this thesis.

The distribution mean position shows the feature value in which the subject spends most of the time. The standard deviation shows the range of movements in each feature direction.

The red, blue, green lines shows the neutral and extremes of the subjects as shown in the legend figure below.

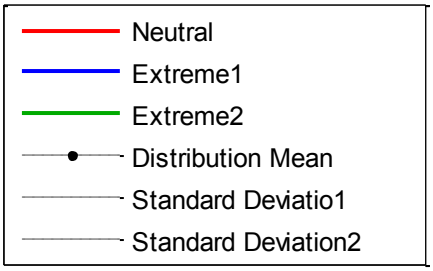

Figure 49 Sample distribution legend

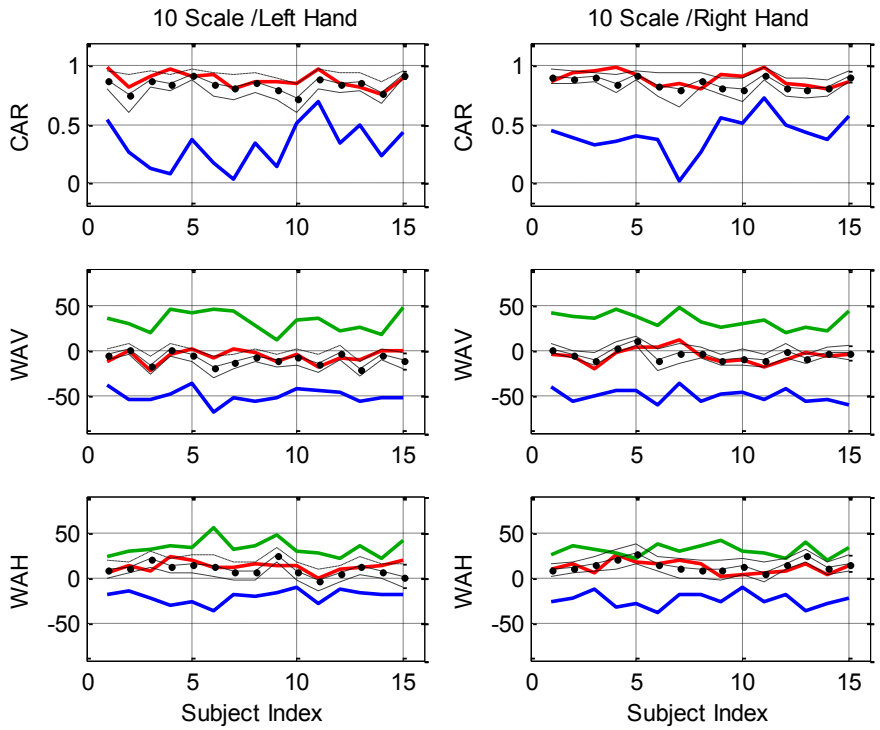

Figure 50 Sample distribution for 'Scale'

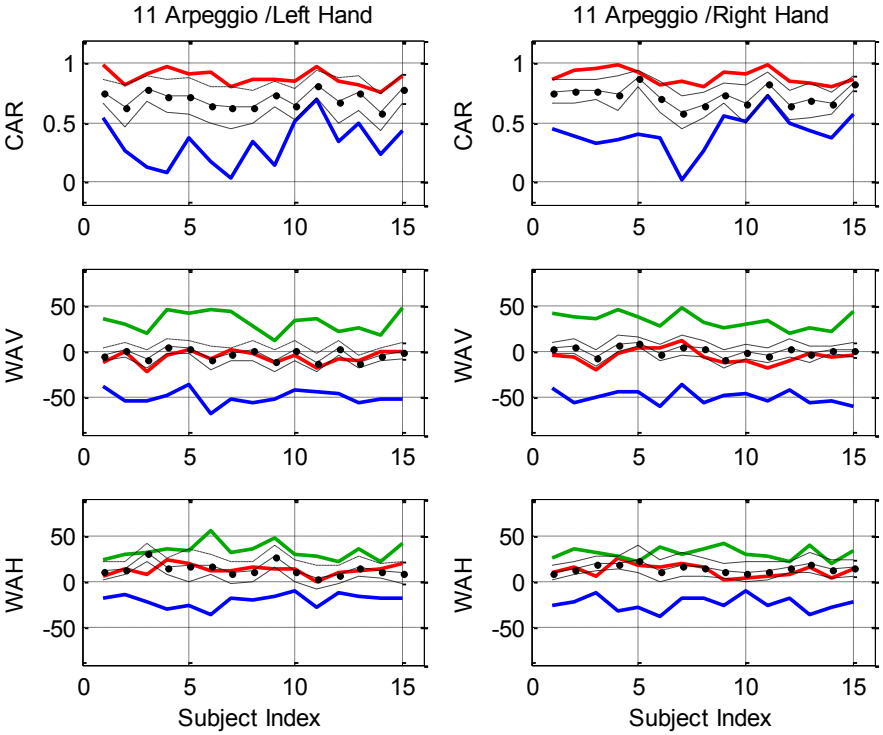

Figure 51 Sample distribution for 'Arpeggio'

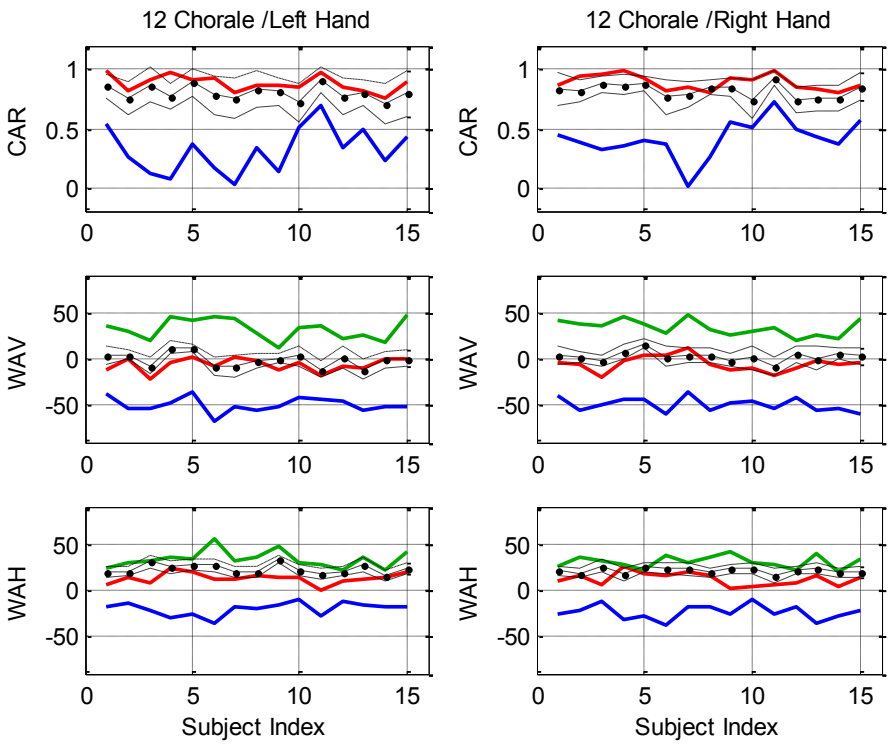

Figure 52 Sample distribution for 'Chorale'

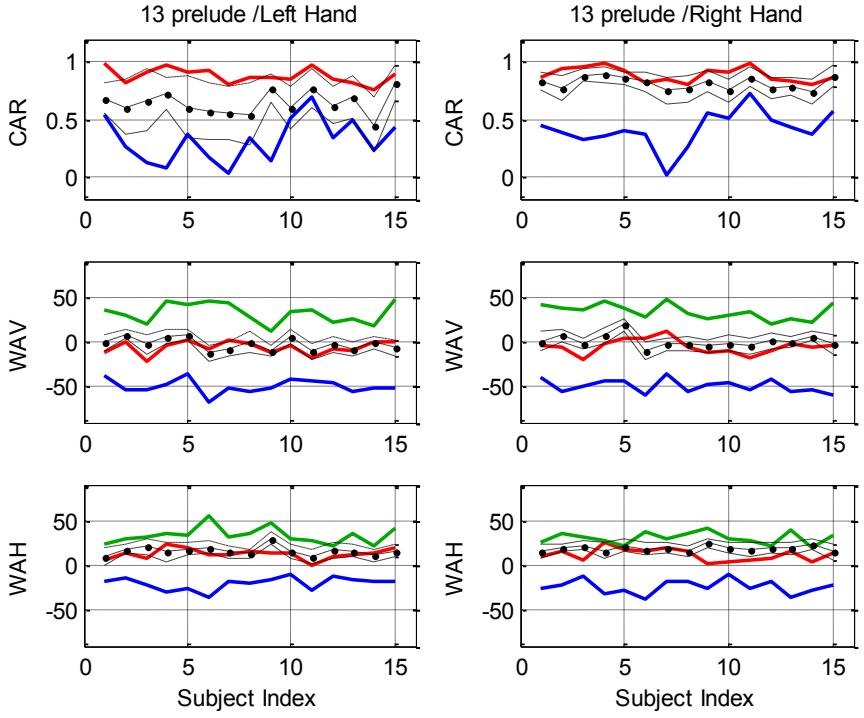

Figure 53 Sample distribution for 'Prelude'

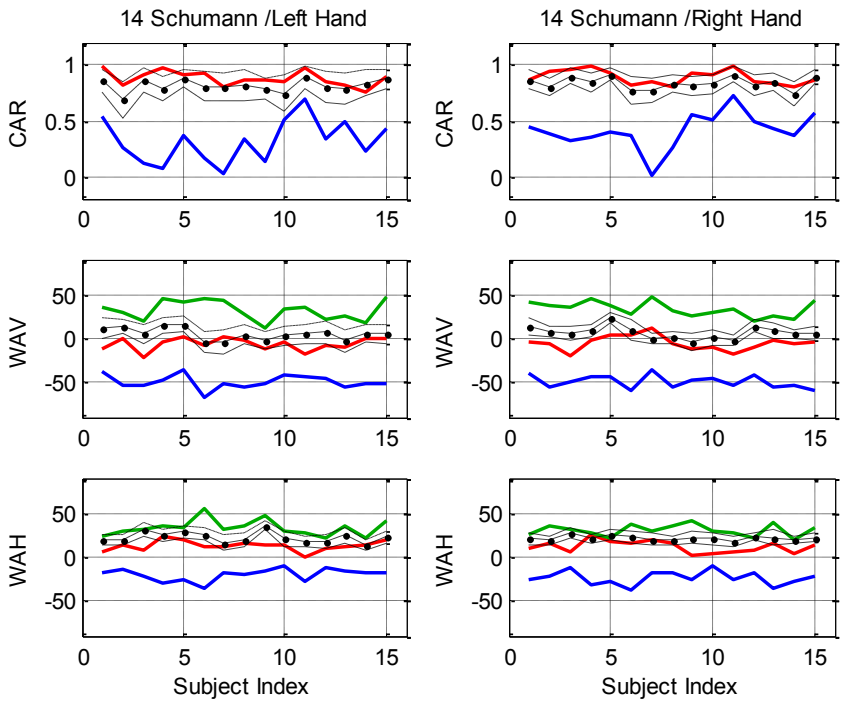

Figure 54 Sample distribution for Schumann 'Little Study'

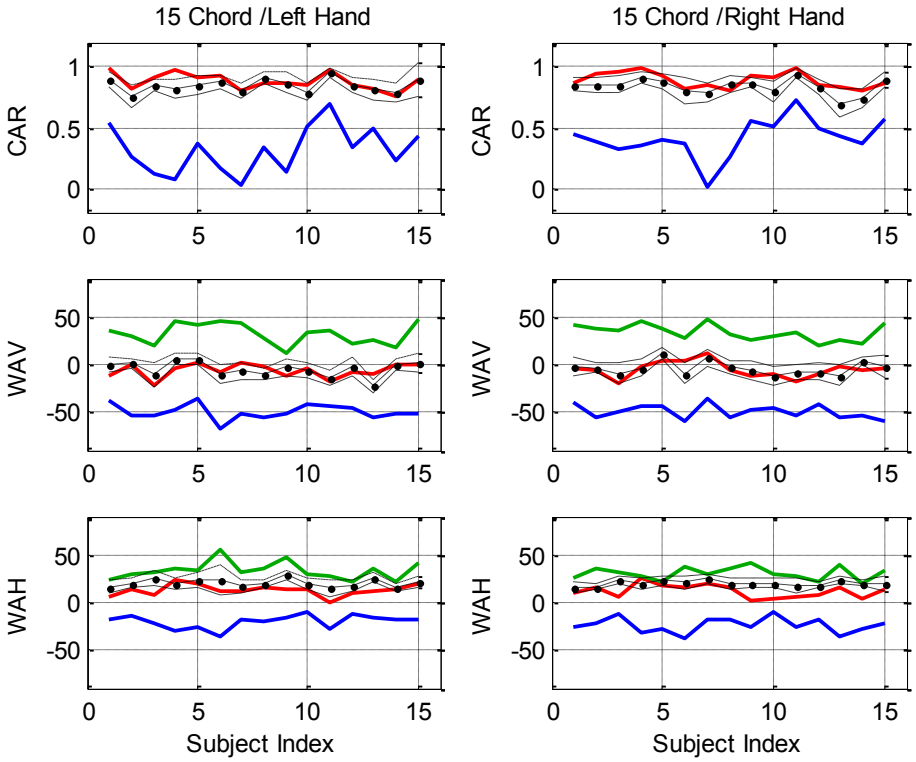

Figure 55 Sample distribution for 'Chord'

Appendix B

- The data collected for this research work is backed up on our 100 TB Network File System (NFS) storage at this address:

  nsf.isilon.rnet.missouri.edu/ifs/piano/Data_of_15_subjects

- The code is archived in a Git repository here:

  kronos.cirl.missouri.edu:/home/shared/Git/PianoProject_v2.git