Design of an Interactive Smart Board Using Kinect Sensor

Supervisor: Dr. Jia Uddin

Nasrul Karim Sarker - 13201025

Muhammad Touhidul Islam - 13201021

Md. Shahidul Islam Majumder - 13201022

Department of Computer Science and Engineering,

BRAC University

Submitted on: 22nd August 2017

DECLARATION

We, hereby declare that this thesis is based on the results found by ourselves. Materials of work found by other researcher are mentioned by reference. This Thesis, neither in whole or in part, has been previously submitted for any degree.

Signature of Supervisor	Signature of Author
Dr. Jia Uddin	Nasrul Karim Sarker
	Muhammad Touhidul Islam
	Md. Shahidul Islam Majumder

ACKNOWLEDGEMENTS

All thanks to Almighty ALLAH, the creator and the owner of this universe, the most merciful, beneficent and the most gracious, who provided us guidance, strength and abilities to complete this research.

We are especially thankful to Dr. Jia Uddin, our thesis supervisor, for his help, guidance and support in completion of our project. We also thankful to the BRAC University Faculty Staffs of the Computer Science and Engineering, who have been a light of guidance for us in the whole study period at BRAC University, particularly in building our base in education and enhancing our knowledge.

Finally, we would like to express our sincere gratefulness to our beloved parents, brothers and sisters for their love and care. We are grateful to all of our friends who helped us directly or indirectly to complete our thesis.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Interactive smart board is emerging as a reality all around the globe with the advancement in technology. Our goal is to make a cost efficient interactive smart board with the aid of XBox 360 Kinect Sensor which will recognize and process gesture as well as voice to implement the Human-Computer-Interaction (HCI) methodology. The system will allow a user to fully control the mouse of a computer using only their hands. Furthermore, different predefined gestures can be used to trigger key press of a keyboard. This functionality may be useful in a wide variety of applications like changing slides using gesture while conducting a PowerPoint presentation or such. Moreover, certain voice commands will be available to make the system more effective and convenient to use.

CHAPTER 01: INTRODUCTION

1.1 Motivation

Human Computer Interaction (HCI) is booming in all fields of work in our modern world. We aspire to implement the concepts of HCI for eliminating one of the most common problems faced in classrooms, offices or conferences. During presentations, a speaker has to casually walk up to the hardware of a computer system each time he/she wants to change a slide or gain control of the computer. This causes unnecessary disruption to the meeting as well as wastes valuable time. Therefore, we came up with a system to minimize hardware and peripheral devices dependency. The system will allow the speaker to take control of the computer from any part of the room using gesture and voice in an easy way without causing any unnecessary disruption. Moreover, the solution will be low cost as it only needs Kinect device along with SDK toolkit.

1.2 Problem Statement

Dependency on peripheral devices forces us to be physically in contact with a device in order to control it. This becomes bothersome when someone has to go back and forth to their desk just to change slides or zoom onto content during a presentation. However, gesture based system can be implemented to solve this problem. When a lecturer moves back and forth during giving a lecture just to control the presentation slide, it causes disruption in class. Time management becomes a hassle due to this disruption. Students find it hard to focus and get carried away.

1.3 Contribution Summary

The summary of the main contributions is as follows:

- Minimize hardware and peripheral devices dependency
- Hand gesture and voice command based easy control system
- Implementation of low cost HCI
- Creating a disruption free environment

1.4 Thesis Orientation

The rest of the thesis is organized as following chapters:

- Chapter 02 consists of the review of previous works.
- Chapter 03 introduces with proposed model and the summary of the working process.
- Chapter 04 describes the complete working procedure and in depth analysis.
- Chapter 05 concludes with results and future working plan.

CHAPTER 02: LITERATURE REVIEW

Kinect [11] has a depth sensor, video sensor, and multi-array microphone. Kinect uses these sensors for tracking and recognition voice, gesture, and motion. In paper [12, 13], authors gave introductions of using depth information from the depth sensor to create skeleton images, to track the coordinate of both user‟ hand. Our team had to go through numerous articles in order to gain knowledge about the implementation of interactive smart board using Kinect Sensor. Another [1] article educated us about the usage of Kinect‟s depth sensor, an infrared camera which will be used to identify pre-defined hand gestures. Hand gesture recognition is an important research issue in the field of Human-Computer-Interaction, because of its extensive applications [7]. Despite lots of previous work, building a robust hand gesture recognition system that is applicable for real-life applications remains a challenging problem. Existing vision-based approaches [8, 9, 10] are greatly limited by the quality of the input image from optical cameras. In addition, gesture recognition is a complex task which involves many aspects and while there may be variations on the methodology used and recognition phases depending on the application areas, a typical gesture recognition system involves such process as data acquisition, gesture modeling, feature extraction and gesture recognition. The detail description of each process is given in [4] and [5]. The main goal of gesture recognition involves creating a system capable of interpreting specific human gestures via mathematical algorithms and using them to convey meaningful information or for device control [4]. Kinect detects multiple joints of hand movement, explore the impact of different joints on the overall hand movement and validate the system in a noisy environment [2]. Moreover, we learned about various methods of performance evaluation by collecting data set from user experiences and depicting it in a graphical manner to determine accuracy of gesture and voice recognition [3]. In addition, we became familiar with voice command control by speech recognition technology of Kinect and learned about its strength and limitations [6].

CHAPTER 03: PROPOSED MODEL

3.1 Introduction

Figure 1 is a demonstration of complete workflow of our proposed model. It indicates the base algorithm of our system. Firstly, we detect the hand using Microsoft Kinect‟s depth sensor which uses an infrared camera. Secondly, we determine different hand gestures based on hand movements and different hand positions. Next, we analyze the data we got from the last step and compare it with our predefined dataset. Finally, we determine the intended action by the user and perform it.

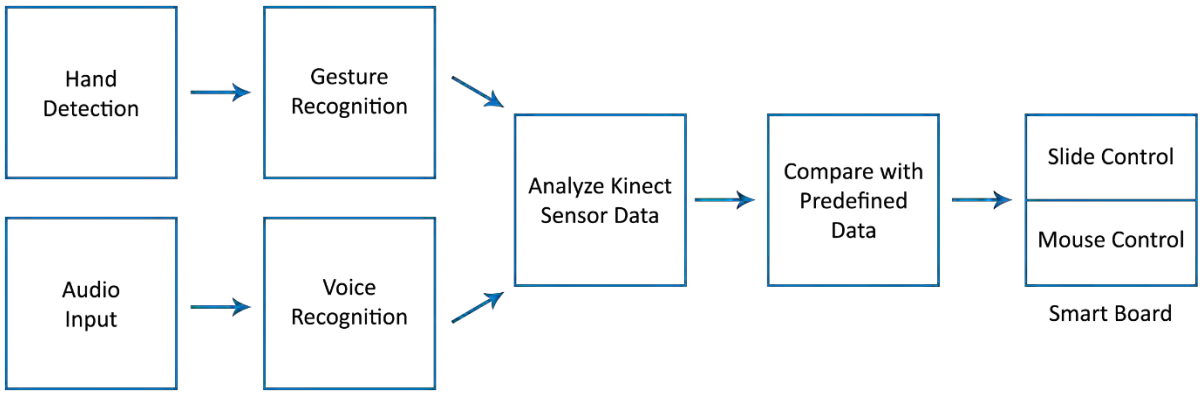


Figure 1 Workflow of Our Proposed System

3.2 Working Environment Setup

There are various types of frameworks for HCI to communicate between hardware like sensors or audio devices and User Interfaces. Microsoft‟s Kinect for Windows Software Development Kit is one of them. It provides the drivers to use a Kinect Sensor on a computer running Windows operating system. It also provides APIs and device interfaces to work with Kinect sensor.

The hardware for our research is a first generation Microsoft Kinect which is designed for the XBox 360 gaming console.

In Figure 2 we have shown how the Kinect and software library interact with application.

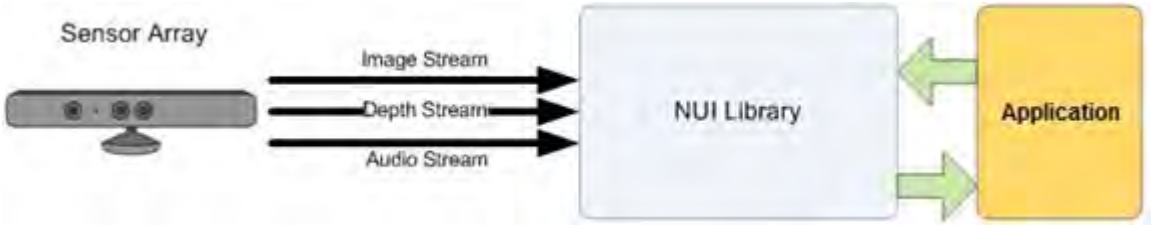


Figure 2 Kinect Sensor and Software Interaction with Application

3.3 Hand Detection

Kinect is a motion detection and recognition smart sensor which allows human/computer interaction without the need of any physical controllers [14]. Microsoft Kinect captures skeleton frame of a person in front of it. It can recognize up to six persons but only two persons can be tracked simultaneously [15]. Minimum distance between the Kinect sensor and a person should be around six feet. If two persons stand in front of the Kinect sensor it tracks the nearer person depending on the Z-axis value.

Kinect detects about twenty joints of the body of a person. Among these joints we only used wrist and hand joints from the skeleton frame. Joint points are mapped to color points as showed in Figure 3(a). Then the lengths between the points are calculated and both the points are connected with same color dots like Figure 3(b). We assumed that the length of the hand would be double of the distance between the joints. So, we doubled the length and completed the line with same color dots [Figure 3(c)].

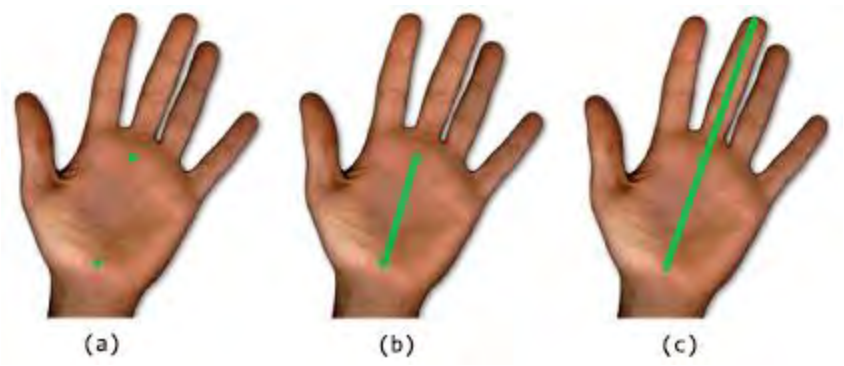


Figure 3 Hand Joints Detection and Length Calculation

After calculating the distance, we detected the center by using the midpoint formula. Using the center, we formed a circle around the hand where radius is the half of the length of the hand [Figure 4(a)]. We selected four points on the line of the circle shown in Figure 4(b). Finally, we drew a rectangle along the points to trace the hand and locked it.

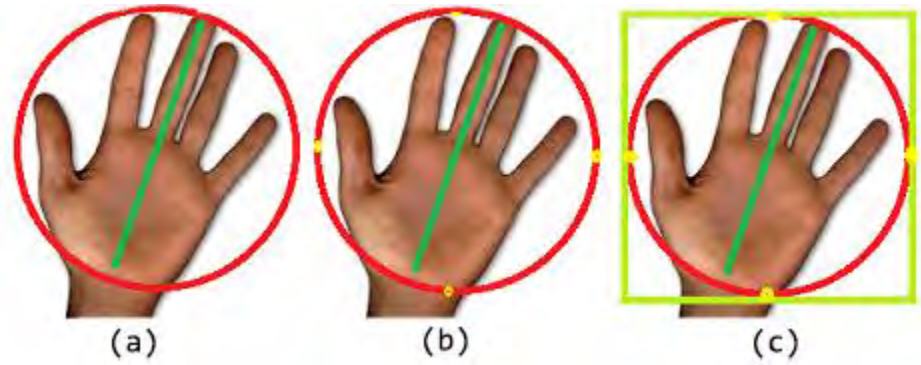


Figure 4 Tracking The Hand

The same procedure is used to detect the both hands.

3.4 Gesture Recognition

Human activity is a sequence of gesture [17]. Kinect sensor will be used to measure multiple joint position and movement to determine whether a gesture is made. The challenge is to ensure that gesture can be separated from random movements of the user. Margin detection, noise

threshold handling and classification of objective characteristics are used to separate the body from background [16]. The application will use Kinect SDK Skeleton tracking to constantly track head and hand joints [Figure 5(a)]. For the slide control application, a gesture is made when a certain threshold distance is exceeded between the head and hand joint. We have set the threshold to be 0.45 meter or 45 centimeters. Therefore, a gesture is recognized only when the left hand or the right hand of the user exceeds the distance of 45 centimeters in comparison to their head joint. Figure 5(b) below shows a gesture being recognized. However, the mouse control process recognizes gesture in a different way [19, 20]. Once the user's body and controlling hand (right hand by default but can be changed) is detected, any movement made by the controlling hand is detected as a gesture to move the mouse cursor while a threshold distance is set for the other hand [Figure 5(c)]. When the other hand (left hand by default but can be changed) exceeds the threshold distance, it is also recognized as a gesture to perform a right click [Figure 5(d)]. After a gesture is recognized the relevant data will be sent for processing.

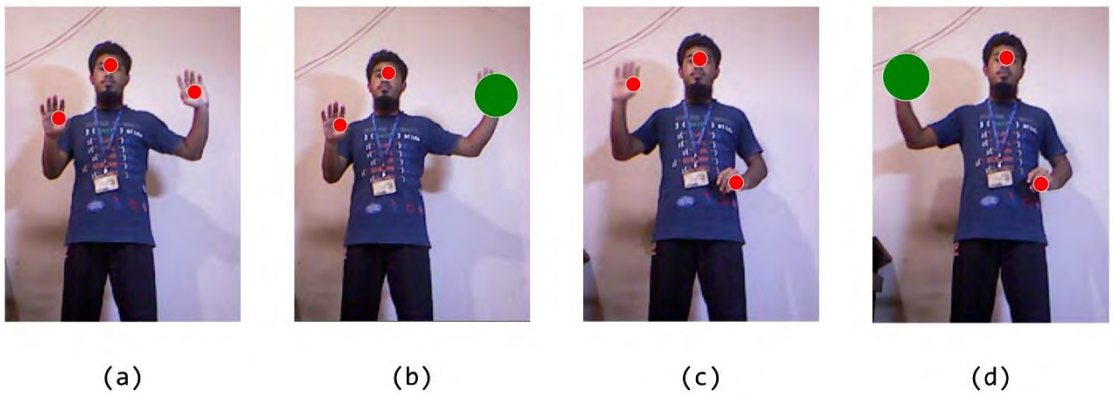


Figure 5 Gesture Recognition

3.5 Analyze Kinect Sensor Data

Efficient algorithm is used to ignore multiple skeletons that might be present in the background and focus on the nearest one to the sensor. The nearest skeleton is considered to be the user and all other skeleton data are considered redundant. Skeleton tracking is used to map the position of

the user at real time. Tracking of head and hand joints are done simultaneously at all times for processing.

For mouse control system hand joint position and primary window size was collected simultaneously. Joint position was returned as X, Y, Z values where:

X = Horizontal position measured as the distance in meters from the Sensor along the X-axis

Y = Vertical position measured as the distance in meters from the Sensor along the Y-axis

Z = Distance from Kinect measured in meters

We only need the horizontal and vertical values to control the mouse. So we scaled the X and Y position values to the primary window size of the monitor to calculate the mouse cursor position. We used Kinect for Windows SDK‟s *ScaleTo()* method to scale the values. Following is a

Audio data was collected from Microsoft Kinect‟s multi-array mic [18] and we analyzed it to convert it to text. We used Kinect for Windows‟s Speech API to analyze and recognize the audio input. This SDK includes a custom acoustical model that is optimized for the Kinect's microphone array.

3.6 Compare with Predefined Data Set and Perform Action

Each time a gesture is recognized it is compared with a predefined data set. This comparison is made to recognize the gesture indicated by the user. If a user extends their right hand while using PowerPoint as their foreground application, it will act as a right arrow key press of keyboard and the presentation will go to the next slide. Similarly, if the user extends their left hand while using PowerPoint as their foreground application, it will act as a left arrow key press of keyboard and the presentation will go to the previous slide. In Figure 6 we demonstrated the algorithm of slideshow control task:

```
Set jointRight to righthandJoint
Set jointLeft to leftHandJoint
Set jointHead to headJoint

If jointRight, jointLeft and jointHead all are in tracking state then
 processController(jointRight, jointLeft, jointHead)
endIf

processController(rightHand, leftHand, head)

 if right hand X position is greater than head X position + 0.35 then
  press RIGHT arrow key
 endIf

 if left hand X position is less than head X position - 0.35 then
  press LEFT arrow key
 endIf

 if right hand X position is greater than head X position + 0.35 &&
 left hand X position is less than head X position - 0.35 then
  press CTRL and + key
 endIf

 if right hand X position is less than head X position + 0.25 && left
 hand X position is greater than head X position - 0.25 then
  press CTRL and - key
 endIf

end processController function
```

Figure 6 Slide Control Pseudo Code

During mouse control, the right hand is set as the default hand to control the cursor but it can be reversed from window. Once skeleton and the hands are tracked, any movement made by the controlling hand (right hand) is recognized as a gesture and the cursor is moved accordingly. If the user lifts the right hand up, the cursor moves up. If the user moves the right down, the cursor goes down. Movement of any fashion made by the right hand is depicted by the cursor in the

similar fashion. Left click of a mouse is represented by the other hand. Whenever the threshold of left hand is crossed, a click is represented by the gesture. Any icon or menu being pointed by the cursor at that time will be left clicked once. Figure 7 demonstrates our mouse control algorithm:

```
set clickThreshold to 0.25f

for each sk in all SkletonFrame

If sk is in tracking state then
   If sk.Joints[RightHand] && sk.Joints[LeftHand] is in tracking
   state then

   Set jointRight to righthandJoint
   Set jointLeft to leftHandJoint

   Scale jointRight and jointLeft to primary window size

   If user is left handed then
    set cursorX to scaledLeft X coordinate
    set cursorY to scaledLeft Y coordinate
   else
    set cursorX to scaledRight X coordinate
    set cursorY to scaledRight Y coordinate
   endIf

   if user is left handed and right hand Y position is greater than
   clickThreshold
    set leftClick to true
   else if user is right handed and left hand Y position is greater
   than clickThreshold
    set leftClick to true
   else
    set leftClick to false
   endIf

 endIf

endIf
```

Figure 7 Mouse Control Pseudo Code

Furthermore, we have initialized four distinct voice commands to make the control even easier. The voice command “computer show window” can be used to see the skeleton and hand tracking of a user. In contrast, the command “computer hide window” can be used to hide the window showing the skeleton tracking. In addition, the command “computer show circles” is used to show the head and hand joints with circles whereas the command “computer hide circles” can be used to make the circles disappear.

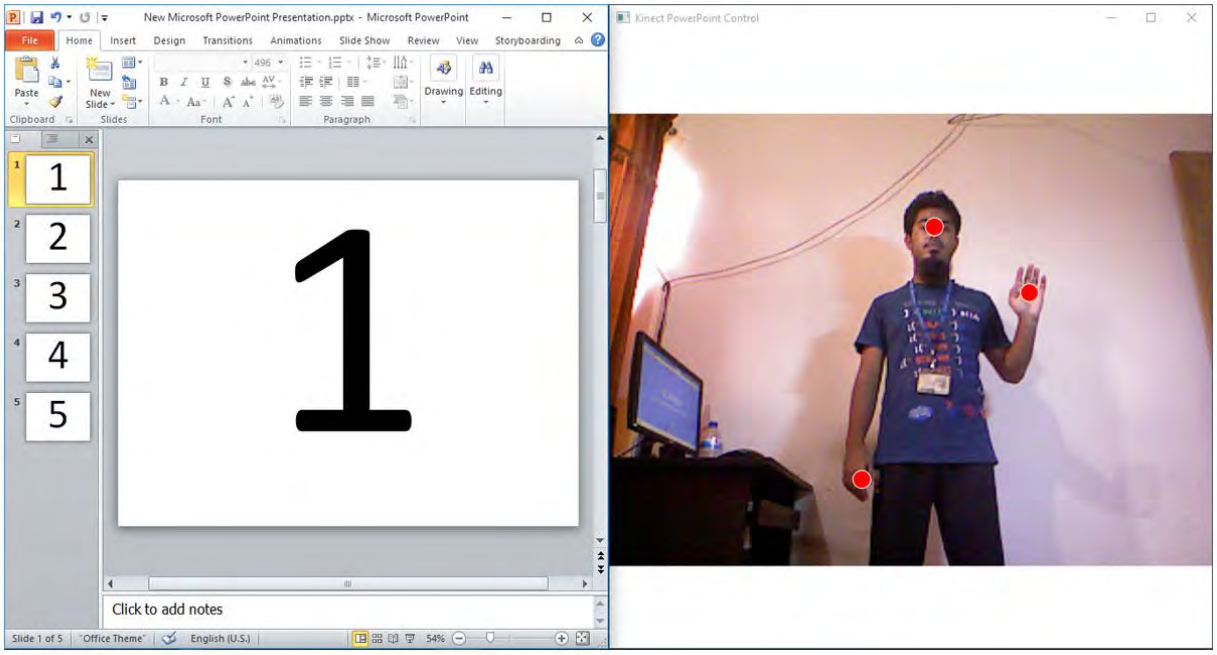
CHAPTER 04: EXPERIMENTAL ANALYSIS AND RESULTS

Kinect can detect six persons at a time, but it can track two persons simultaneously. In our smart board system, we need only one person‟s hands to be tracked. We tracked the nearest person to the Kinect Sensor by calculating the z-axis value. Figure 8 shows how only the nearest person is racked in our system.

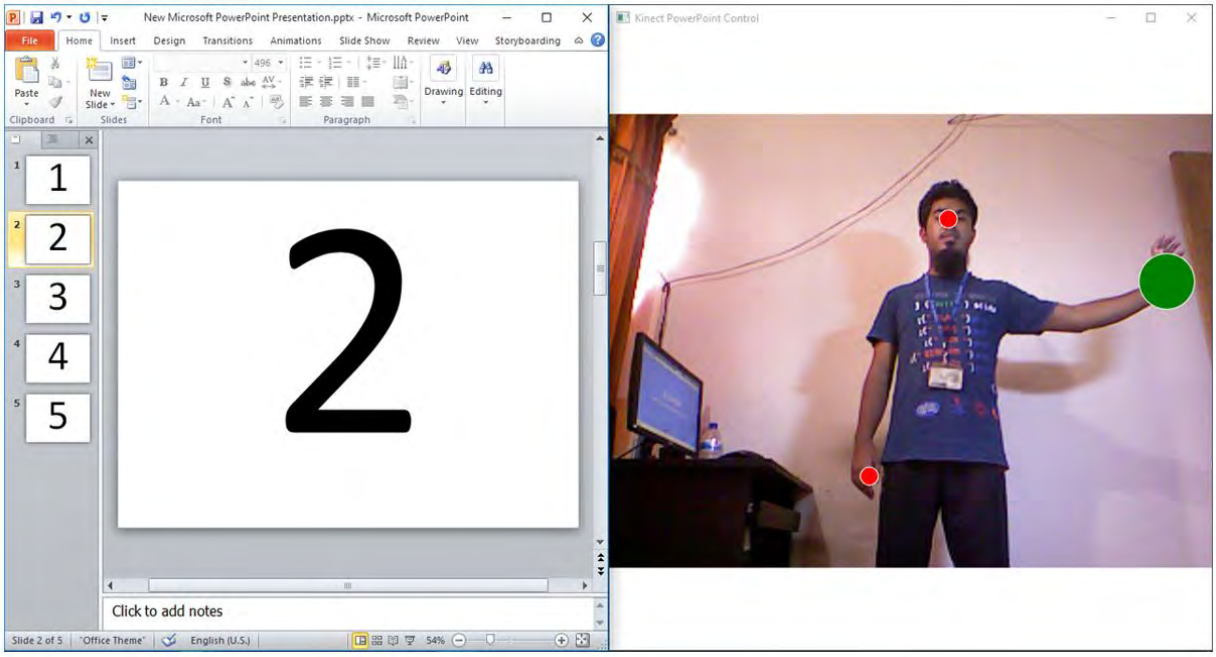
Figure 8 Kinect Sensor Detects Only The Nearest Person

After tracking both hands and head, all the joint positions of hands and head are calculated simultaneously. If right hand moves horizontally 0.45 meter from head to right (along the X-

axis) „Right" arrow key press happens to go to the next slide. Figure 9(a) and 9(b) demonstrates the result:



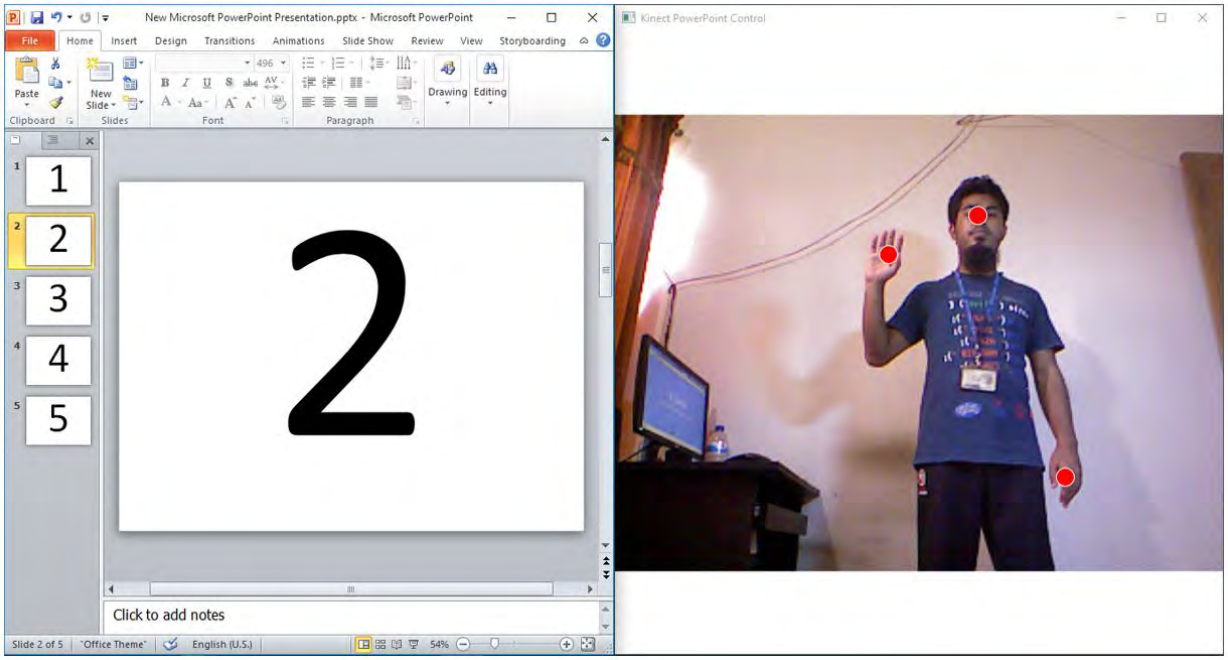
(a)



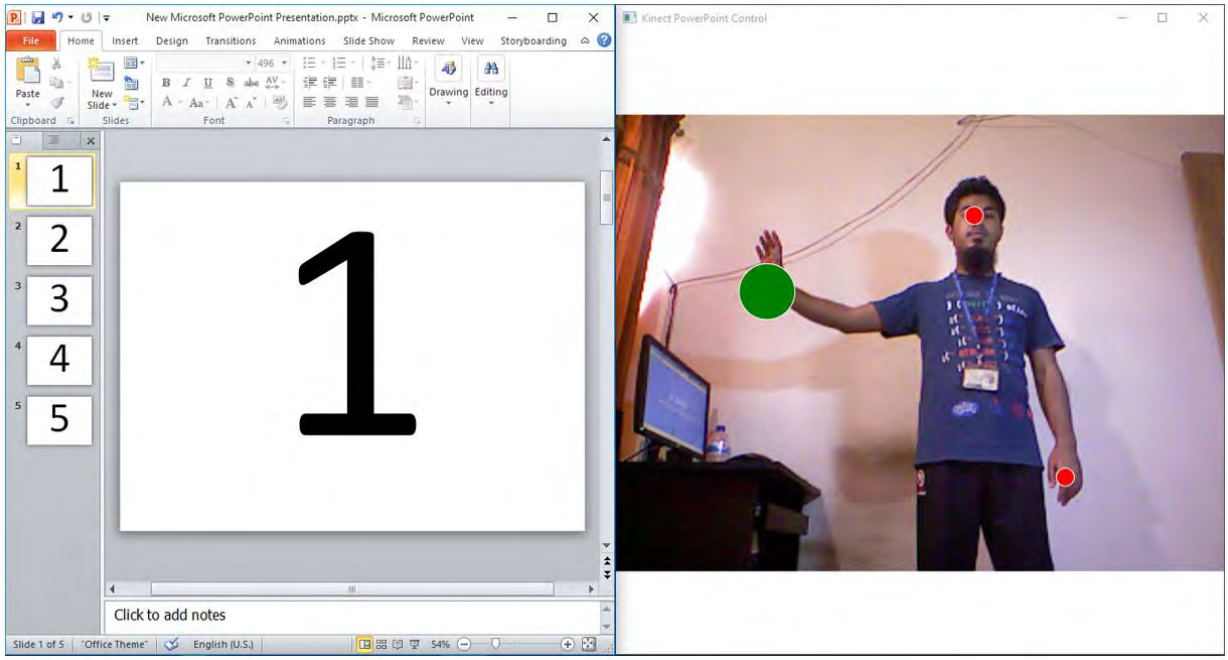
(b)

Figure 9 Changing Slide Forward Using Right Hand Gesture

Similarly, if right left hand moves horizontally 0.45 meter from head to left (along the X-axis) „Left" arrow key press happens to go to the previous slide. Figure 10(a) and 10(b) shows the result:



(a)



(b)

Figure 10 Changing Slide Backward Using Left Hand Gesture

In mouse control, we collected the X-axis and Y-axis values of both hands simultaneously. At the same time, we also collected the primary window height and primary window width. Then we scaled the joint position to the primary window size. Figure 11(a) shows the initial mouse position where X = 969, Y = 299 and mouse click = false. Figure 11(b) shows rightward shift where X increased from 969 to 1120. Figure 11(c) indicates further shift where X = 1251, Y = 317. All these values are generated based on hand position.

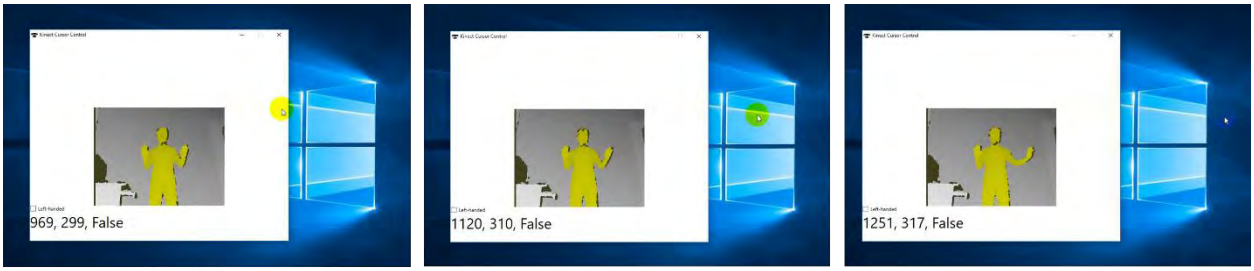


Figure 11 Mouse Control With Hand Gesture

For mouse click we used 0.25 as threshold value. As right hand is used to control the mouse movement left hand gesture is used to make the click. We collected both hands‟ Y-axis values. If left hand Y value is greater than right hand Y value + 0.25, left mouse click occurs. Figure 12(a) shows that initially left hand Y value is less than the threshold value, so the mouse click is false. Figure 12(b) indicates left hand Y value is greater than the threshold value, so the mouse click is set to true. The red circle under the mouse pointer indicates the mouse click.

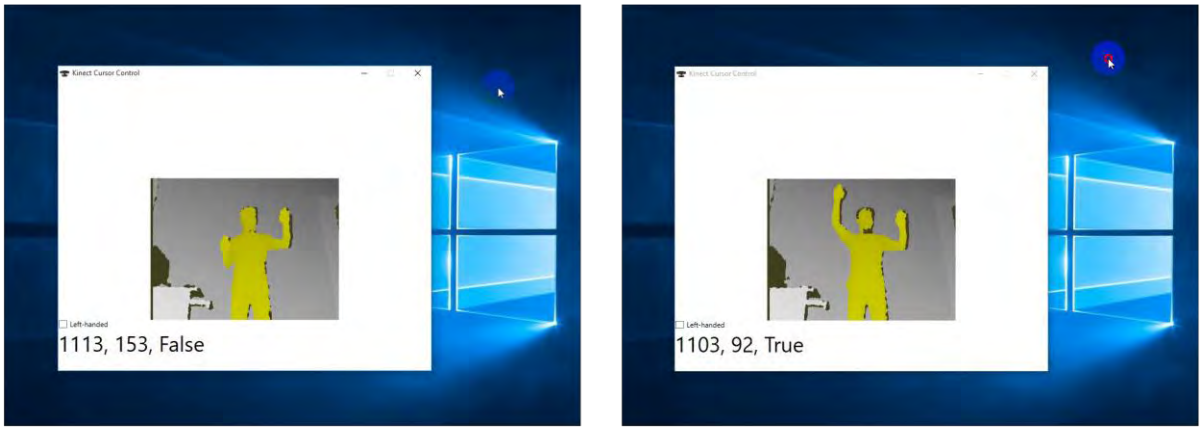


Figure 12 Mouse Click By Hand Gesture

CHAPTER 05: CONCLUSION AND FUTURE WORK

With the advancement of technology interactive smart board is going to take an important position all around the world. Nowadays, to control a peripheral device is not very much easy because it requires physical contact. As researcher said that in the whole world, people don‟t want complex anymore, rather they eager to get something with easier to control. In this new era, people are ready to pay high just for easy and comfortable life. We focused on those points and designed a cost efficient interactive smart board using Kinect sensor. We are initially motivated to design our system to make the most work easier for classrooms, offices or conferences. While presenting, a presenter can control the system using voice command and hand gestures. Our remotely hand control system using voice command and hand gestures will create a good image to the modern society. Interactive smart board is our starting step on this field. One day this type of technology will take place all the remote controlled system. We are determined to step ahead on working on this field and bring the solutions for different other problems.

REFERENCES

[1] Li, Y. (2012). Hand gesture recognition using Kinect.

[2] ElgendiM, Picon F, Magenant-Thalmann N. (2012). Real-Time Speed Detection of Hand Gesture Using Kinect.

[3] Ren, Z., Yuan, J., Meng, J., & Zhang, Z. (2013). Robust Part-Based Hand Gesture Recognition Using Kinect Sensor.

[4] Kawade Sonam P & V.S. Ubale. Gesture Recognition - A Review. In OSR Journal of Electronics and Communication Engineering (IOSR-JECE), pages 19- 26.

[5] Rafiqul Zaman Khan & Noor Adnan Ibraheem. Hand Gesture Recognition – A Literature Review. In International Journal of Artificial Intelligence & Applications (IJAIA), Vol.3, No.4, July 2012.

[6] Sawai P.S., Shandilya V. K. (2016). Gesture & Speech Recognition using Kinect Device - A Review. International Conference On Science and Technology for Sustainable Development, Kuala Lmpur, MALAYSIA, May 24-26, 2016.

[7] J. P. Wachs, M. K¨olsch, H. Stern, and Y. Edan. Vision-based hand-gesture applications. Communications of the ACM, 54:60–71, 2011.

[8] N. Shimada, Y. Shirai, Y. Kuno, and J. Miura. Hand gesture estimation and model refinement using monocular camera-ambiguity limitation by inequality constraints. In Proc. of Third IEEE International Conf. on Face and Gesture Recognition, 1998.

[9] B. Stenger, A. Thayananthan, P. Torr, and R. Cipolla. Filtering using a tree-based estimator. In Proc. of IEEE ICCV, 2003

[10] C. Chua, H. Guan, and Y. Ho. Model-based 3d hand posture estimation from a single 2d image. Image and Vision Computing, 20:191 – 202, 2002.

[11] Microsoft.com. (2017). Kinect - Windows app development. [online] Available at: http://www.microsoft.com/en-us/kinectforwindows/ [Accessed 19 Aug. 2017].

[12] Arici, T.: Introduction to programming with Kinect: Understanding hand / arm / head motion and spoken commands. In: Signal Processing and Communications Applications Conference (SIU), pp. 18-20 (2012).

[13] Tam, V., Ling-Shan Li: Integrating the Kinect camera, gesture recognition and mobile devices for interactive discussion. In: Teaching, Assessment and Learning for Engineering (TALE), IEEE International Conference, pp.H4C-11, H4C-13 (2012).

[14] G. A. M. Vasiljevic, L. C. de Miranda, and E. E. C. de Miranda, "A case study of mastermind chess: comparing mouse/keyboard interaction with kinect-based gestural interface," Advances in Human-Computer Interaction, vol. 2016, Article ID 4602471, 10 pages, 2016.

[15] Msdn.microsoft.com. (2017). *Skeletal Tracking*. [online] Available at: https://msdn.microsoft.com/en-us/library/hh973074.aspx [Accessed 19 Aug. 2017].

[16] Liu, Y. Dong*, M. Bi, S. Gao, D. Jing, Y. Li, L. (2016). Gesture recognition based on Kinect.

[17] Paul, S. Basu, S. Nasipuri, M. (2015). Microsoft Kinect in Gesture Recognition: A Short Review.

[18] Microsoft.com. (2017). Kinect - Windows app development. [online] Available at: http://www.microsoft.com/en-us/kinectforwindows/ [Accessed 19 Aug. 2017].

[19] Hojoon Park. (2012). A Method for Controlling Mouse Movement using a Real Time Camera.

[20] Grif, H. and Farcas, C. (2016). Mouse Cursor Control System Based on Hand Gesture. Procedia Technology, 22, pp.657-661.