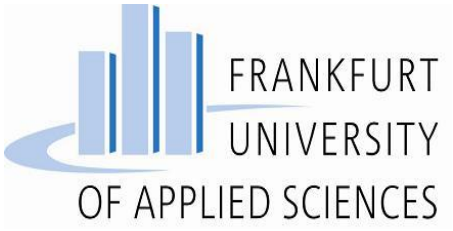


# Using Gesture Recognition to Control PowerPoint Using the Microsoft Kinect

Winter Semester 2015/2016

**Course**

Advanced Real Time Systems

**Supervisor**

Prof. Dr.Ing. Matthias Deegener
School of Computer Science & Engineering
FH Frankfurt am Main University of Applied Sciences

**Participants (Group 12)**

Arnob Mahmud (1079386)

**Date of Submission: 02.03.2016**

# Abstract

This report describes the design and implementation of a speech and gesture recognition system used to control a PowerPoint presentation using the Microsoft Kinect. This system focuses on the identification of natural gestures that occur during a PowerPoint presentation, making the user experience as fluid as possible. The system uses C# the performed gestures to perform real-time segmentation of gestures. The incorporation of speech commands gives the user an additional level of precision and control over the system which we are working on. This system can navigate through a PowerPoint presentation and has a limited control over slide animations.

# Acknowledgments

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them. We are highly indebted to **Prof. Dr.Ing. Matthias Deegener** for his guidance and constant supervision as well as providing necessary informations regarding the project and also for his support in completing the project. We would like to express our gratitude towards our parent‘s encouragement which helped us in completion of this project. Our thanks and appreciations also go to our colleagues in developing this project and people who have willingly helped us out with their abilities.

Finally, thanks to everyone for everything whoever supported us during the implementation of the project.

# Contents

# List of Figures



# List of Tables

# 1. Introduction

## 1.1 Motivation

In today's world, technology pervades nearly every aspect of the average person's daily life. People interact with computers and other technology as frequently as they do with other people and they should have the ability to communicate with computers as naturally as they do with other humans. Speech is perhaps the most comfortable form of communication between humans. It is quick, efficient, and allows people to express themselves with great degrees of freedom, limited only by their own vocabulary. Since the dawn of computers half a century ago, people have dreamed of being able to have conversations with robots and other artificial intelligences as easily as they do with other humans. Unfortunately, keyboard and mouse have been the primary means of interfacing with computers even to this day. While effective in many situations, they are limiting and not a particularly natural means of interaction. Gesture recognition is a current area of research that is trying to address this problem. Everyone is familiar with gestural interaction with other humans. It occurs naturally during speech as a way people for people to express themselves. Gestures are a form of body language that is essential to effectively communicate ideas in addition to spoken language. People already gesture when communicating with other humans, so why not use this mode of communication for natural interaction with computers.

# 2. Objective of Real Time

Formulate requirements for embedded systems with strict constraints on computational delay and periodicity. Categorize and describe the different layers in system architecture for embedded real-time systems. Construct concurrently-executing tasks for real-time applications that interface to hardware devices (sensors/actuators). Describe the principles and mechanisms used for designing run-time systems and networks for real-time applications. Apply the basic analysis methods used for verifying the temporal correctness of a set of executing tasks.

# 3. Goal

This report aims to build a gesture recognition system that uses natural gestures to control a PowerPoint presentation. When people give PowerPoint presentations, they usually have a clicker object that can control the slides remotely. However, holding onto the clicker during the presentation occupies the hand. When a person's hand is already occupied, the range of motions and gestures that can be performed is limited. This limitation is not necessarily a physical limitation; the presenter may simply be (potentially unconsciously) unwilling to perform certain gestures while their hand is occupied. The primary goal of this system is to free the user from these restraints and automatically react to the naturally spoken words and gestures throughout the presentation. In many gesture recognition systems, the vocabulary of recognizable gestures is contrived and unnatural. While they are usually not arbitrary as they have some logical connection between the gesture and its response, they are not gestures that a user would perform naturally on their own. This system focuses on recognizing natural gestures and phrases that users would be likely to perform or say during a PowerPoint presentation even without the system. By focusing the system's attention on natural phrases and gestures, the user should not have to think about performing artificial and awkward gestures to control the PowerPoint, which could potentially be even more distracting to the user than holding onto a small clicker. The gesture recognition system should not hinder the user, allowing the human-computer interaction to be as seamless and intuitive as possible.

# 4. Real Time Systems

This means: correctness and execution time of the results are guaranteed. On the other hand a real-time-system will also guaranty that a certain deadline is met. In computer science, real-time computing (RTC), or reactive computing, is the study of hardware and software systems that are subject to a "real-time constraint"— e.g. operational deadlines from event to system response. Real-time programs must guarantee response within strict time constraints, often referred to as —deadlines .

## 4.1 Types of Real Time Systems

There are three types' real time systems as -

1. Hard RT-System: Missing a deadline is a total system failure.

2. Soft RT-System: The usefulness of a result degrades after its deadline, thereby degrading the system's quality of service.

3. Firm RT-System: Infrequent deadline misses are tolerable, but may degrade the system's quality of service. The usefulness of a result is zero after its deadline.

## 4.2 What is a gesture?

In this report, a gesture is defined as a meaningful sequence of hand and arm poses over time. A pose is simply a configuration of the arm and hand joint positions at a single point in time. In this work, the body is divided into three sections: left arm, right arm, and torso. For the types of gestures in this system, the two arms are the most significant parts of the body that are tracked. The torso is used for identifying the general location of the body with respect to the Kinect. The two arms are analyzed separately allowing the gestures to be performed with either arm independently of the other.

| Gesture | Functionality | Possible Speech |
|---|---|---|
| Forward | next slide | "moving on" |
| Backward | previous slide | "going back" |
| forward scroll | skip ahead x slides | "skipping the next x slides" |
| backward scroll | skip back x slides | "going back x slides" |
| Pointing | trigger animation | none |

Table 1: Possible gesture and speech combinations

## 4.3 Target Gestures

This work focuses on two functions that are a key part of any presentation: navigation through slides and triggering of onscreen animations. Desired navigation functionalities are "next slide", "previous slide", and jumping forward or back any number of slides or to a specific slide. These are general navigation commands that can be applied to any PowerPoint presentation. The onscreen animation triggering, on the other hand, are less consistent between each slide. The exact effect that the triggering command has on a slide is dependent on the design of the slide. Slides may have any number of different animations or none at all. Common animations are on build slides where there is a list of bullet points that are initially hidden and sequentially revealed. The animation triggering function can be used to reveal each of these bullet points. In general, the triggering command will cause the next (if any) animation to occur.

Based on these desired functionalities there are five basic gestures that can be performed with either hand: forward, backward, forward scroll, backward scroll, and pointing. These gestures are diagrammed in Figure 1 and Table 1 shows which gestures control which functionalities as well as possible corresponding phrases.

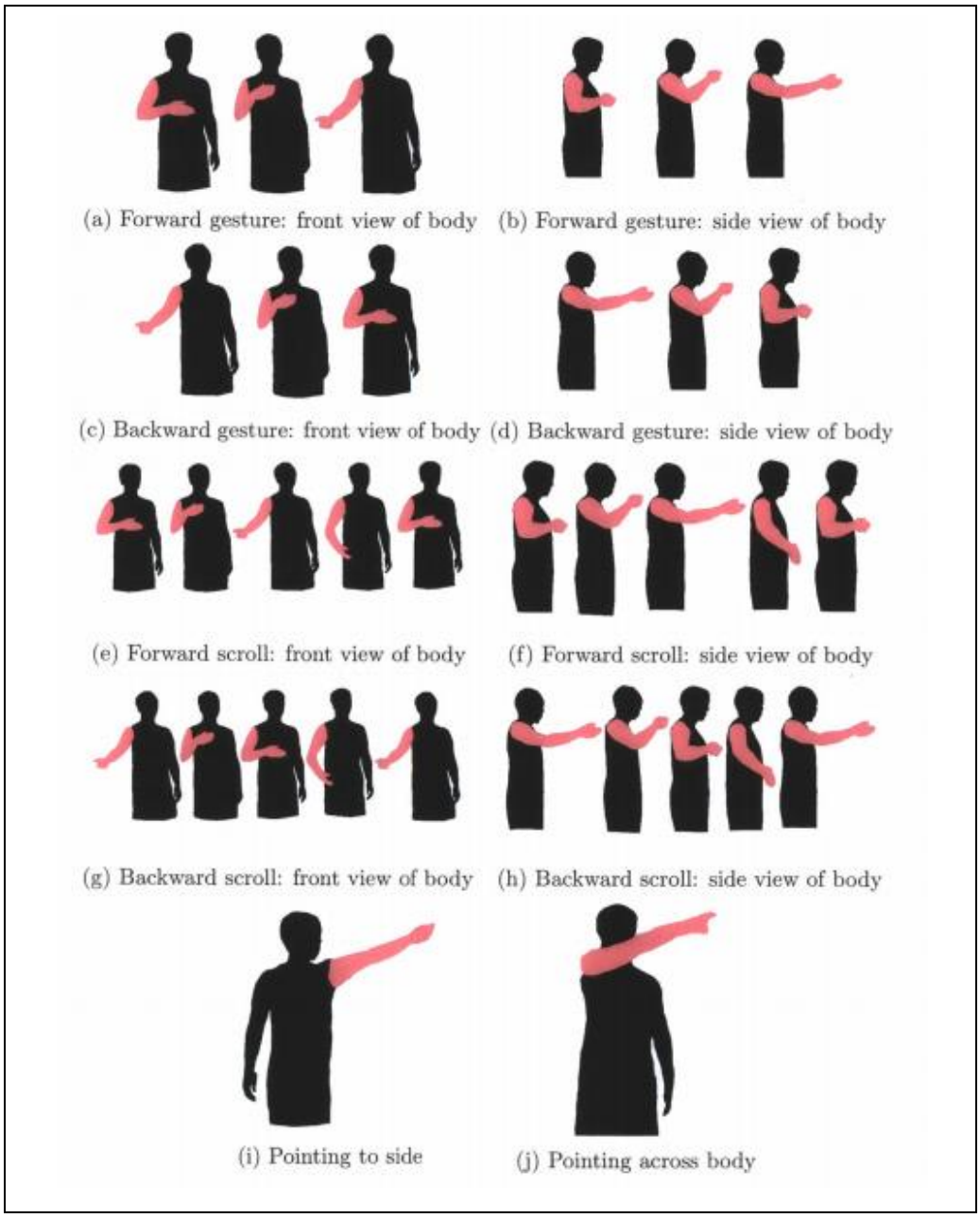


Figure 1: This figure shows some basic body poses of the five different gestures with the right hand from the front and side points of view. 1i and 1j illustrate the two different ways a user can perform a pointing gesture to the left: either by extending the left hand out to the side or crossing the right hand across the body. Each of these gestures can be performed with the left hand in the same manner.

Figures 1a and 1b show the how the forward gesture is performed: an outward directed extension of the arm that starts at the mid-torso level close to the body and ends with an outstretched arm and an upward facing palm in front of the body. The backward gesture, Figures 1c and 1d, is the opposite, performed as an inward directed movement with the arm starting with an outstretched arm palm-up and ending close to the body, again at mid-torso level. The forward and backward scroll gestures can be visualized as continuous and smooth repetitions of the forward and backward gestures. Figures 1e through 1h each shows a single cycle of the forward and backward scroll gestures. The pointing gesture can be performed in one of two ways. Given a pointing direction, to the left of the body for example, the user can use either hand to perform the pointing gesture: either with the left hand extended out on the left side of the body (Figure 1i), or with the right hand crossing in front of the body resulting in a leftward point (Figure 1j).

## 4.4 Communicative vs. Manipulative Gestures

In this work, we consider two types of gestures: communicative and manipulative. Communicative gestures convey a complete idea with the gesture. A system can only begin to react appropriately to a communicative gesture after it has been performed to completion and has been disambiguated from other gestures. An example of a communicative gesture on a touch screen is a "flick" to navigate between pages of an application like a web browser or photo viewer. The system will not know to interpret the gesture as a "flick" instead of a drag until the entire gesture has been performed. Manipulative gestures by contrast give the user direct control of the system which reacts in real-time to the gesture as it is being performed. An example of a manipulative gesture is a two finger pinch-to-resize an image on a touch screen. As soon as two fingers touch the image, the image starts to resize itself depending on the size of the pinch. In this system, the forward, backward, and pointing gestures are communicative gestures and the scroll gestures are manipulative gestures. The system cannot interpret a forward, backward, or pointing gesture appropriately until it has been performed to completion. On the other hand, the system should be able to identify when the user is performing a scroll gesture and react accordingly. In this case, the system should navigate forwards or backwards an additional slide each time another cycle segment has been performed.

## 5. System Development Model

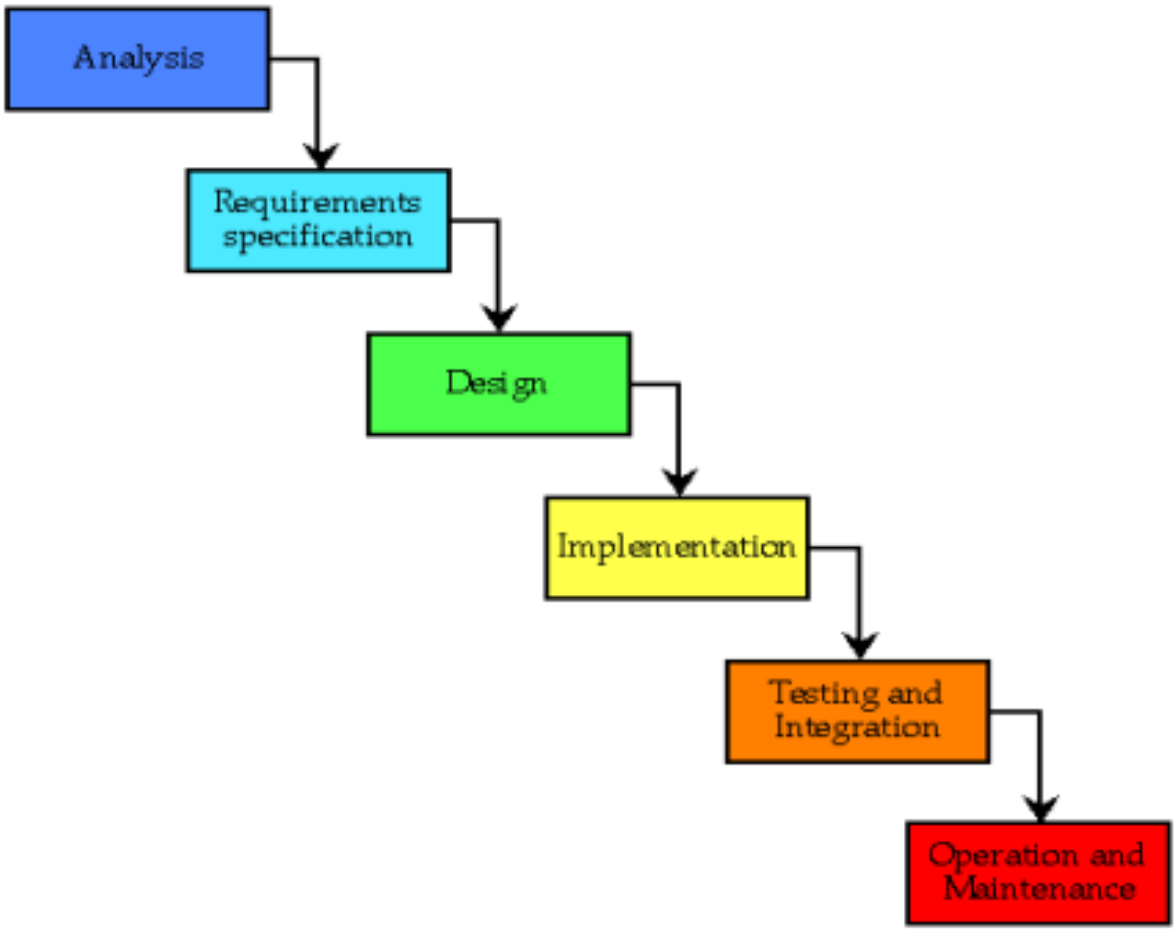


Figure 2: System Development Model.

## 6. Background

This gesture recognition and PowerPoint controlling system uses a number of different technologies and techniques to achieve its goal. The major technologies used are the Microsoft Kinect sensor for body input data using C# for gesture recognition. These are described in this section.

## 6.1 Microsoft Kinect

The Microsoft Kinect (Figure 3) is a motion sensing device with a variety of inputs that performs real-time skeleton tracking. The Kinect was originally released in November 2010 as an accessory to the Xbox gaming system for playing video games without using a physical controller. In 2011, Microsoft released the Kinect SDK to the public. The Kinect has a standard RGB camera, a 3D infrared depth sensor, and a multi-array microphone allowing a variety of different inputs signals to be used. Body tracking had previously been its own area of research. With the release of the Kinect SDK the general public was able to start using body tracking in their own applications and research with little overhead and cost.

Figure 3: Kinect sensor

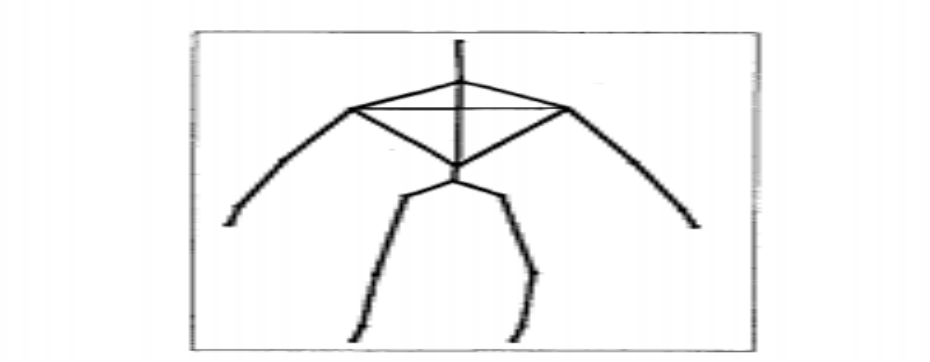

Figure 4: Sample output skeleton from the Kinect, drawn in Processing.

## 6.2 How It Works

The Kinect performs 3D depth sensing by emitting a structured point cloud pattern of infrared (IR) light and calculating the depth from the images taken with its IR sensor. Because this IR point cloud originates at a single point in the Kinect sensor, as the distance from the Kinect increases, the point cloud pattern disperses proportionally to the distance the light has traveled. By measuring the offset between the expected location of the IR grid pattern at a calibrated distance and its actual location, the Kinect can calculate the depth at each point in the projected point cloud. Using this depth image, the Kinect can identify foreground objects and determine people and their poses by comparing the detected body to millions of stored examples of body poses. The Kinect then uses a randomized decision forest technique to map the body depth image to body parts from which the skeleton representation can be built (Figure 4).

# 7. How the system works

## 7.1 Gesture Recognition Approach

This system achieves gesture recognition by using C# programming language. While it appears that C# is perfectly suited for gesture recognition.

Gestures are sequences of body positions, and a body cannot be gesturing if it is motionless. This approach is a good starting point; however there are a number of limitations. A motion silence based system would only work if the user froze after the completion of each gesture. This is not well suited our focus of giving a PowerPoint presentation. The system's goal is to identify the relevant gestures that are performed naturally and react accordingly. The assumption of natural gestures is that people perform gestures continuously hence motion silence is not always applicable: gestures are typically not immediately preceded and followed by freezing in place. While there are often abrupt velocity changes before or after a gesture, they are not particularly reliable measures of gesture beginnings and endings.

## 7.2 Skeleton Preprocessing

The skeleton data provided by the Kinect are the joint positions in a coordinate system where the Kinect is at the origin. Before these features are sent to the SVM, they are converted to a body-centered coordinate system and scaled. The body origin is defined as the 3D centroid of the left, right, and center hip positions. The conversion from the Kinect's world coordinates to the body coordinates makes future classification of the skeletons robust to the body's relative position to the Kinect sensor.

# 8. Technologies Used

1. Microsoft Visual Studio 2013
2. Kinect SDK
3. Microsoft Kinect Sensor
4. Programming language: C#

# 9. Controlling a PowerPoint Presentation

## 9.1 Identifying Gestures

With the gesture abstraction described above, there are essentially only 3 identifiable gestures: forward, backward, and pointing. Despite, there only being 3 identifiable gestures, the system still needs to differentiate between the 5 original gestures. This is achieved by keeping track of the recently classified gestures within a time frame. The system keeps track of the average length of the singular forward and backward gestures. In its default state, the system is idling and waiting for a gesture. During this idling state, once a forward or backward gesture is identified, the system begins to wait for another gesture.

If the specified time threshold elapses without identifying another gesture, the gesture is classified as a singular forward or backward gesture. On the other hand, if another gesture is identified within the allotted time, the system classifies the gesture sequences as a scroll gesture. The sequence of recently identified gestures is recorded. In this way, the system keeps a tally of the direction of each identified scroll cycles. As each new scroll cycle is identified, the running tally is updated. If the identified cycle direction matches the majority of the recent cycle directions, it is counted towards the overall scroll gesture. If the direction contradicts overall direction of the scroll so far, it does not send a command to the PowerPoint presentation but is still counted towards the tally in case the overall direction of the scroll reverses. This procedure provides a method for real-time error correction for each of the scroll segments as well as count the number of scroll cycles. This allows the system to reliably identify the direction of a scroll gesture without having to wait for the entire gesture to be performed to completion.

| **Gesture** | **Action** |
|---|---|
| forward | forward one slide |
| backward | backward one slide |
| forward scroll | forward n slides, determined by number of scrolls |
| backward scroll | backward *n* slides, determined by number of scrolls |
| pointing | trigger animation, not next slide |

Table 2: Default gesture commands.

## 9.2 System Architecture

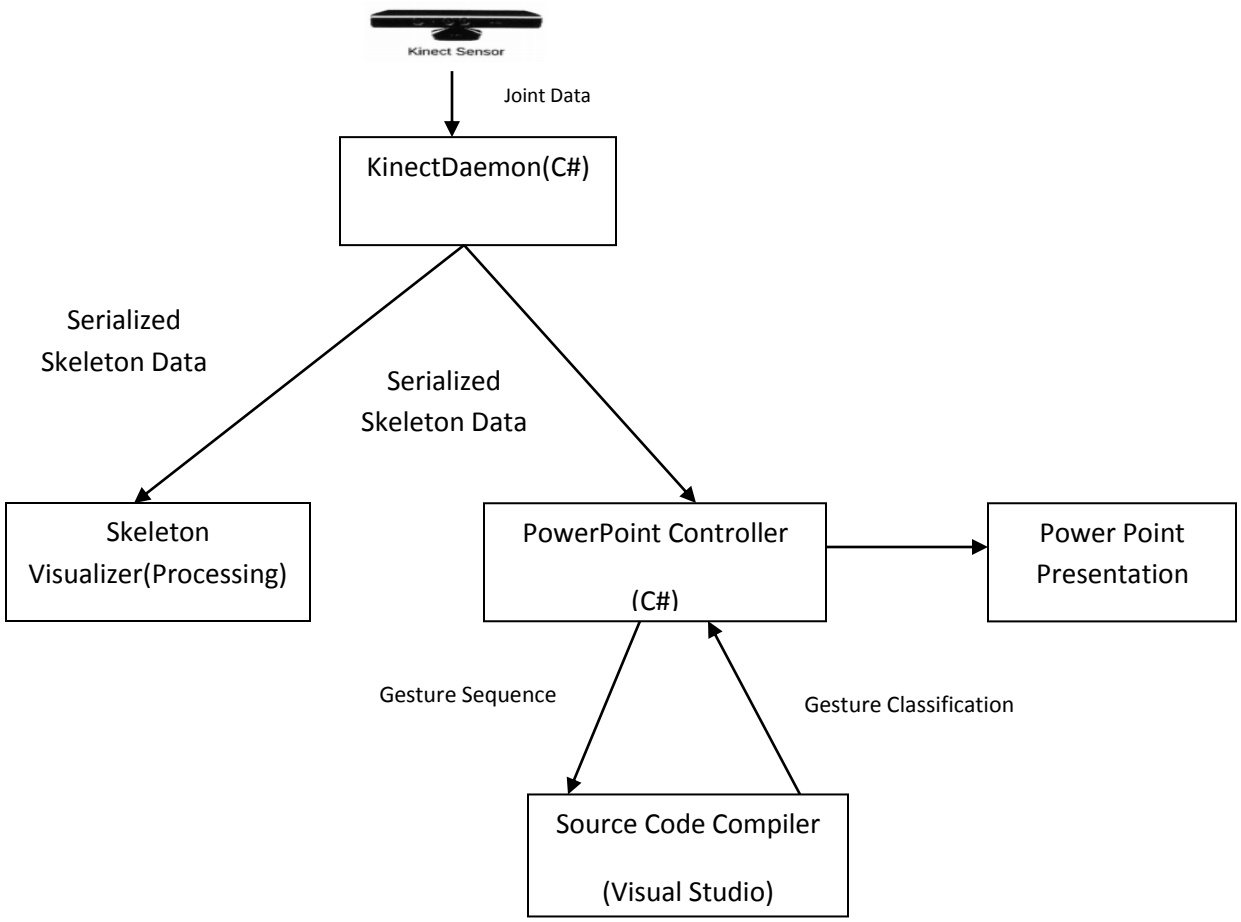


Figure 5: A diagram of the overall system architecture showing the data flow.

# 10. How to run the Project

1. Compile and run the application.

2. Point the Kinect at you and stand at least five feet away.

3. You can see yourself in the application window and the three circles will track your head and hands.

4. Extend your right arm to activate the "right" or "forward" gesture. Extend your left arm to active the "left" or "back" gesture. These gestures will send a right or left arrow key to the foreground application, respectively.

5. Run your PowerPoint show so PowerPoint is the foreground application, and the right and left gestures will go forward and back in your deck.

The ellipses grow and change color when your hand exceeds the threshold of 45 centimeters. The gestures will only activate once as your hand exceeds the threshold, and only one of the gestures can be active at once. You must bring your hand back closer to your body to activate the gesture a second time.

The gestures will also work for any other application. For example, open Notepad and type some text then use the gestures to move the cursor left or right one character at a time.

# 11. Conclusion

## 11.1 Evaluation of Results

With final classification accuracies ranging between 90-100% for the forward-backward gestures distinctions and 100% for the pointing gestures, the system works fairly reliably. It is less disruptive for no action to occur than for the wrong actions to be performed.

The gestures that can be identified by gesture recognition systems are often contrived and unnatural. While the range of identifiable gestures is relatively limited, they are performed naturally in the context of giving a PowerPoint presentation.

## 11.2 Limitations

There is currently no way to activate embedded videos using gesture, we have to activate it using mouse or keyboard.

The gesture is triggered based upon the distance between the head and the hands, so we might accidentally trigger the gesture if we put our arms out, or bend over to pick something up perhaps.

**Speech Integration** One of the limitations in this system is in the speech integration. The speech commands must be spoken to completion before the gestures are performed. This creates an unnatural disconnect between the speech and the gestures. Another possible solution would be to prevent the system from reacting to a gesture until the speech has been fully processed. However, there are a number of reasons that would make this approach undesirable. The primary reason is the trade-off between the importances of speech versus the speed of the system's reaction to performed gestures. For this particular system where speech is not required and gestures are often performed independently, it seemed more important to have a reactive system than one that waits for speech to finish. This leads into the second problem, which shows why the system cannot begin to react to the performed gestures as it finishes processing speech. Some of the relevant commands, such as "move ahead 3 slides", are dependent on the current presentation. Beginning to navigate slides before that particular speech command is understood could result in confusing effects. One case is if there are more than 3 scroll cycles, the slides would progress past the destination slide then suddenly skip back to it. Even though the end result is the same, this behavior would be confusing to an audience that is trying to follow the slides.

**Generalizing the System** More general improvements could be to further generalize the system with more gestures and actions. With the expansion of different gestures and controllable actions, it would be useful incorporate an API that offers programmatic control over PowerPoint presentations. There is also room for training the system on different users. As every person will perform the gestures differently, it would be useful to collect more data from other people to generalize the models. User tests would also be useful to identify further potential deficiencies in the overall recognition system.

## 11.3 Future Expansion

Now on our testing and experiments are still going on about proper using of speech recognition system. In future, we can add speech recognition system to navigate the slide and we can add another gesture for clicking the embedded videos if any in the slide.

# Bibliography

[1] C.M. Bishop. *PatternRecognition and Machine Learning.* 2007, 2007.

[2] B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. *Proceedings Fifth Annual Workshop on ComputationalLearning Theory (COLT),* 1992.

[3] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology,* 2:27:1- 27:27, 2011.

[4] C. Cortes and V.N. Vapnik. Support vector networks. *Machine Learning,*1995.

[5] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal StatisticalSociety,* 1977.

[6] T. Huang, R.C. Weng, and Chih-Jen Lin. Generalized bradley-terry models and multi-class probability estimates. *Journalof Machine Learning Research* 7, 2006.

[7] F. Jelinek. *StatisticalMethods for Speech Recognition.* MIT Press, 1997.

[8] Jerdak, 2011. http://www.seethroughskin.com/blog/?p=1159.

[9] C.D. Manning and H. Schiitze. *Foundations of Statistical Natural Language Processing.*MIT Press, 1999.

[10] G.J. McLachlan and T. Krishnan. *The EM Algorithm and its Extensions.* Wiley, 1997.

[11] J.R Rabiner and B.H. Juang. *Fundamentals of Speech Recognition.* Prentice Hall, 1993.

[12] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceddings of the IEEE,* 1989.

[13] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images. CVPR, IEEE, 2011.