

Remote Exercise Monitor Using Kinect

CS 2310 – Multimedia Software Engineering

Final Project Report

Submitted by – Tazin Afrin

Date – 12/8/2016

1. Introduction

Microsoft Kinect (XBOX 360) is widely used for gaming and exercise monitor. There are popular apps for dancing and activity monitoring too. Now-a-days Kinect based remote physical therapy is also becoming very popular. Some of the Kinect based popular fitness apps are 'Your Shape', 'Nike + Kinect training' etc. However, remote Kinect exercise monitors are not that available. Remote exercise/fitness monitor could be specifically useful for remote physical therapy in rural areas where a physical therapist is not readily available.

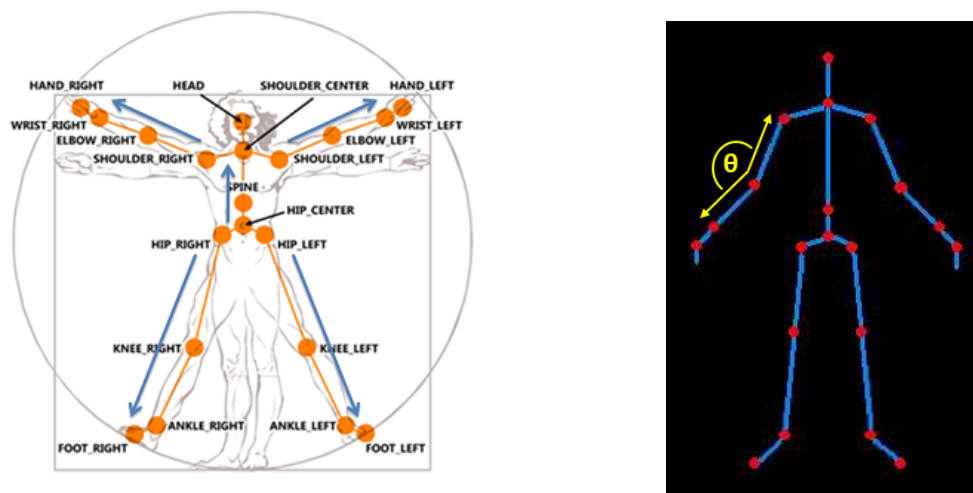


Figure 1: Kinect Skeleton Structure and Tracker

In the figure above, Kinect skeleton tracker is shown. Kinect can track 18 skeleton joints. In this experiment, I have used the SHOULDER, ELBOW and WRIST joints and calculated the ELBOW angle. The figure on the right shows the joints as in Kinect sensor and the angle calculated. More details about the method is described in section 4.

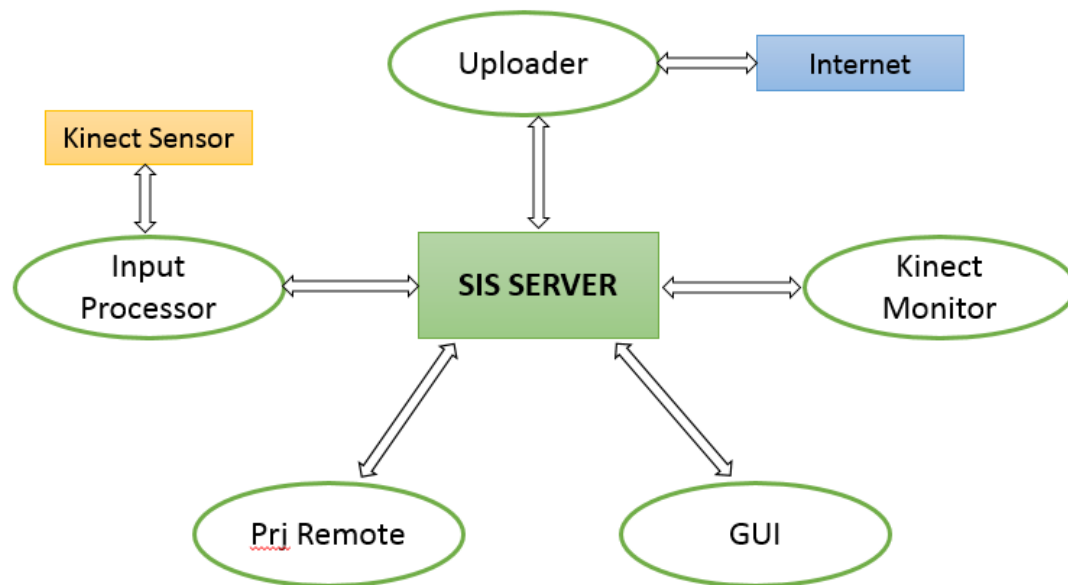
In this project, I have implemented a simple Kinect based remote exercise monitor. This focus on real-time arm motion monitor and feedback. In this exercise monitor, a user can

move his left arm in a motion of taking weight. The Kinect monitor tells the user if the elbow angle is within specified angle. Kinect skeletal diagram is used for assessment of arm joint angle and motion. Connection has been established between Kinect sensor and SIS testbed server. An input processor collects the data from the Kinect sensor and a Kinect monitor gives the feedback to the user. At the end of exercise Kinect Monitor shows a graph of the angles to the user and sends email to the chronobot@ksiresearch.org and uploads the data to the database.

Extra Deeds:

- a. Setting timer for exercise at the beginning by getting input from the user.
- b. Plot the exercise motion graph at the end of exercise.
- c. A dummy calorie calculator shows how many calories are burned using the arm motion data.

2. System Description



In the diagram above the components of the whole system is shown. This system has the following main components –

- **SIS Server:** The SIS server is the central of all the system components. It is technically not a SIS component, rather it is the forwarding center of all the message flow. It processes the messages and dispatches them based on the

routing table. The routing table is built at the starting of each component through registration. Each active component has to register to SIS server.

- **GUI:** The GUI is a predesigned monitor component. It monitors all the different types of components with the current scope in the system. It also works as a panel to display and control component parameter.
- **Kinect Sensor:** This is the Kinect XBOX 360. It can be thought of as a sensor input to the SIS components. This sensor is responsible for sensing the angle of elbow and sending the data to the Input Processor.
- **Input Processor:** Input Processor is a predesigned basic component. It takes in the stream of angle data value from the Kinect sensor and sends the reading message to the Kinect Monitor. At the beginning, it waits at a port to receive data from the Kinect sensor. After the exercise is over it receives a message from Kinect monitor and stops sending data to Kinect Monitor.
- **Kinect Monitor:** This is a basic component to give feedback to the user. At the very beginning of the exercise Kinect Monitor asks the user how long the user wants to exercise. Then it sets a timer for exercise. Kinect Monitor receives the movement data from Input Processor. Then it checks if the angle is within the specified threshold and gives feedback if the angle is too high, too low or if the user is moving it right. After exercise is done, Kinect Monitor sends message to the uploader to update the database and also to the Input Processor to stop sending messages. Then it shows a plot of arm motion to the User.
- **Uploader:** Uploader waits to get message from the Kinect Monitor. Once it gets message that the exerciser is done, it sends an email to chronobot@ksiresearch.org and updates the database.

3. Running the server and components

At the beginning the SYS Server boots up and waits for registration message from the components. Each component has to send a registration message to the server. Then the components can start communicating between each other through the server. In the Figure 2 below the registration of InputProcessor and KinectMonitor components is shown as an example –

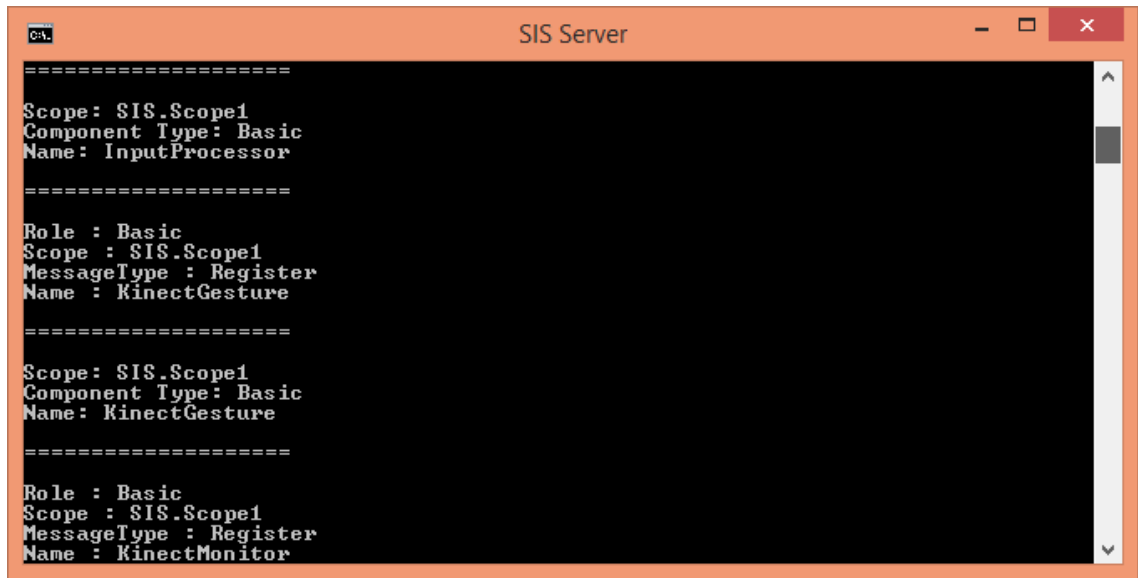


Figure 2: Components InputProcessor and KinectMonitor registers themselves with the SYS server

After successfully registration the components boots up and sends a Connect message to the server. After receiving the connect message the server sends a confirm message to each component. In this system, we have mainly three components – GUI, InputProcessor and KinectMonitor. So, upon connecting to the server, the components waits to send or receive signals from the server. Also when InputMonitor starts it waits to listen from the Kinect sensor. Also the uploader needs to be initialized and run. In the Figure 3 below the connection and booting up of the components are shown –

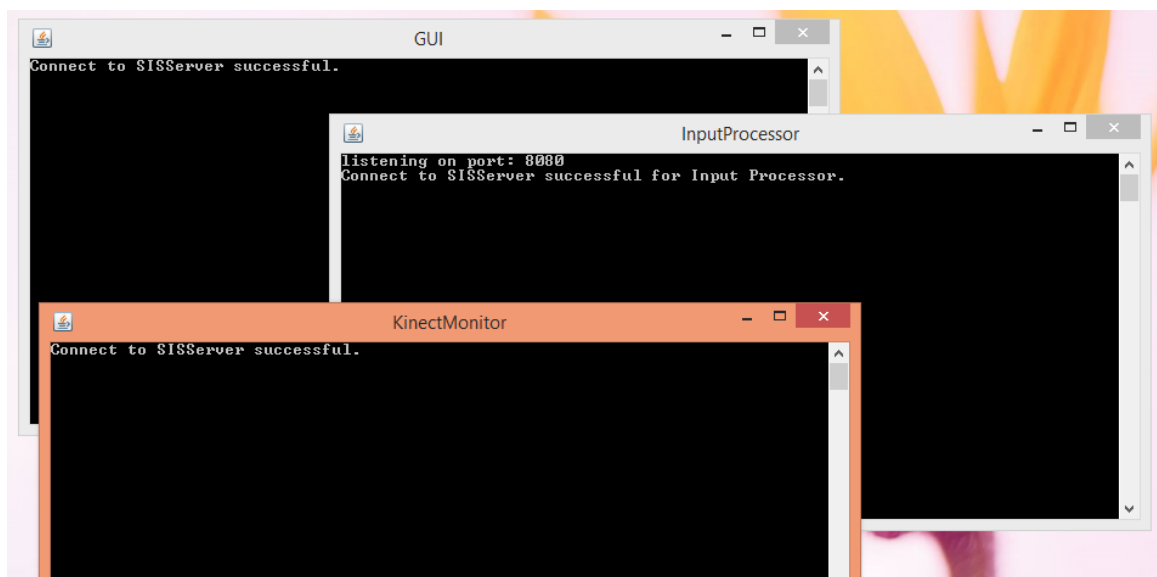
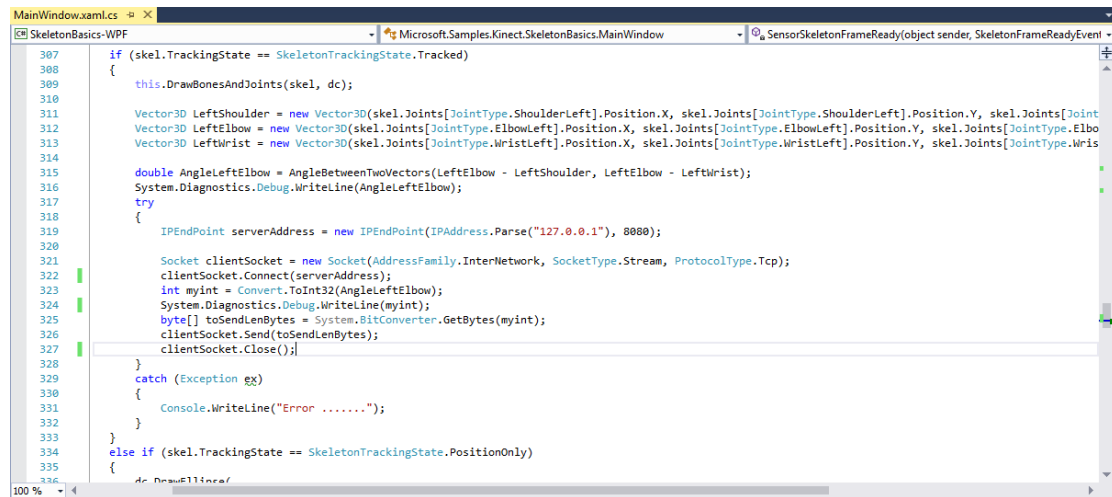


Figure 3: Components GUI, InputProcessor and KinectMonitor connects with the SYS server

4. Communication with Kinect Sensor

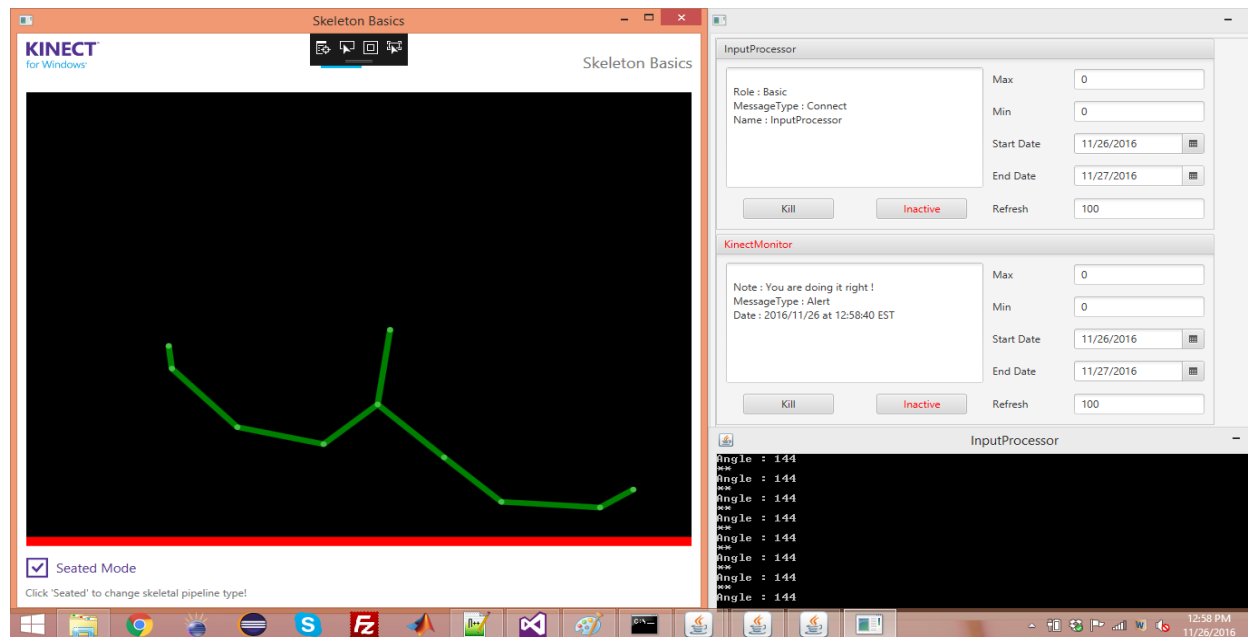
The input processor component communicates with Kinect Sensor. The Input Processor works as a server and waits a server port to listen from the Kinect sensor client. When the Kinect sensor boots up, it detects the user at the window and start calculating the angle of arm motion. Figure 4 shows the calculation Kinect sensor side –



```
307 if (skel.TrackingState == SkeletonTrackingState.Tracked)
308 {
309     this.DrawBonesAndJoints(skel, dc);
310
311     Vector3D LeftShoulder = new Vector3D(skel.Joints[JointType.ShoulderLeft].Position.X, skel.Joints[JointType.ShoulderLeft].Position.Y, skel.Joints[JointType.ShoulderLeft].Position.Z);
312     Vector3D LeftElbow = new Vector3D(skel.Joints[JointType.ElbowLeft].Position.X, skel.Joints[JointType.ElbowLeft].Position.Y, skel.Joints[JointType.ElbowLeft].Position.Z);
313     Vector3D LeftWrist = new Vector3D(skel.Joints[JointType.WristLeft].Position.X, skel.Joints[JointType.WristLeft].Position.Y, skel.Joints[JointType.WristLeft].Position.Z);
314
315     double AngleLeftElbow = AngleBetweenTwoVectors(LeftElbow - LeftShoulder, LeftElbow - LeftWrist);
316     System.Diagnostics.Debug.WriteLine(AngleLeftElbow);
317     try
318     {
319         IPEndPoint serverAddress = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 8080);
320
321         Socket clientSocket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
322         clientSocket.Connect(serverAddress);
323         int myint = Convert.ToInt32(AngleLeftElbow);
324         System.Diagnostics.Debug.WriteLine(myint);
325         byte[] toSendLenBytes = System.BitConverter.GetBytes(myint);
326         clientSocket.Send(toSendLenBytes);
327         clientSocket.Close();
328     }
329     catch (Exception ex)
330     {
331         Console.WriteLine("Error .....");
332     }
333 }
334 else if (skel.TrackingState == SkeletonTrackingState.PositionOnly)
335 {
336     // do something
337 }
```

Figure 4: Arm angle calculation

After the Kinect sensor knows the angle it sends the raw data to the Input Processor. Then Input Processor processes the data and sends a message to the Kinect Monitor. In the figure below the components along with Kinect sensor is shown –

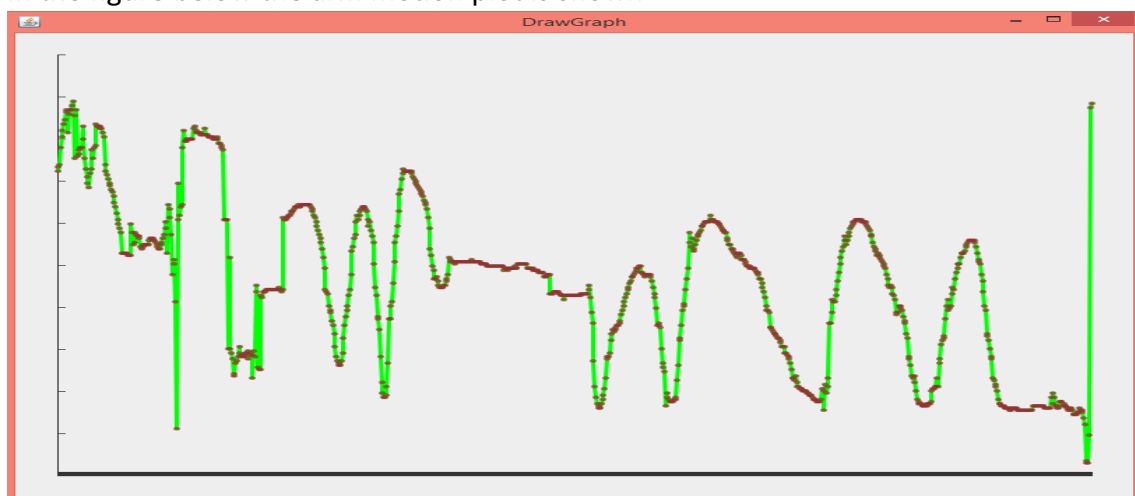


```
KinectMonitor
Angle too low !
You are doing it right !
Angle too low !
You are doing it right !
Angle too low !
You are doing it right !
Angle too low !
You are doing it right !
Angle too low !
You are doing it right !
Angle too low !
You are doing it right !
Angle too low !
You are doing it right !
Angle too low !
You are doing it right !
Angle too low !
You are doing it right !
Angle too low !
You are doing it right !
Angle too high !
Sending Email to Uploader
Calories Burned: 182.7433078393881
```

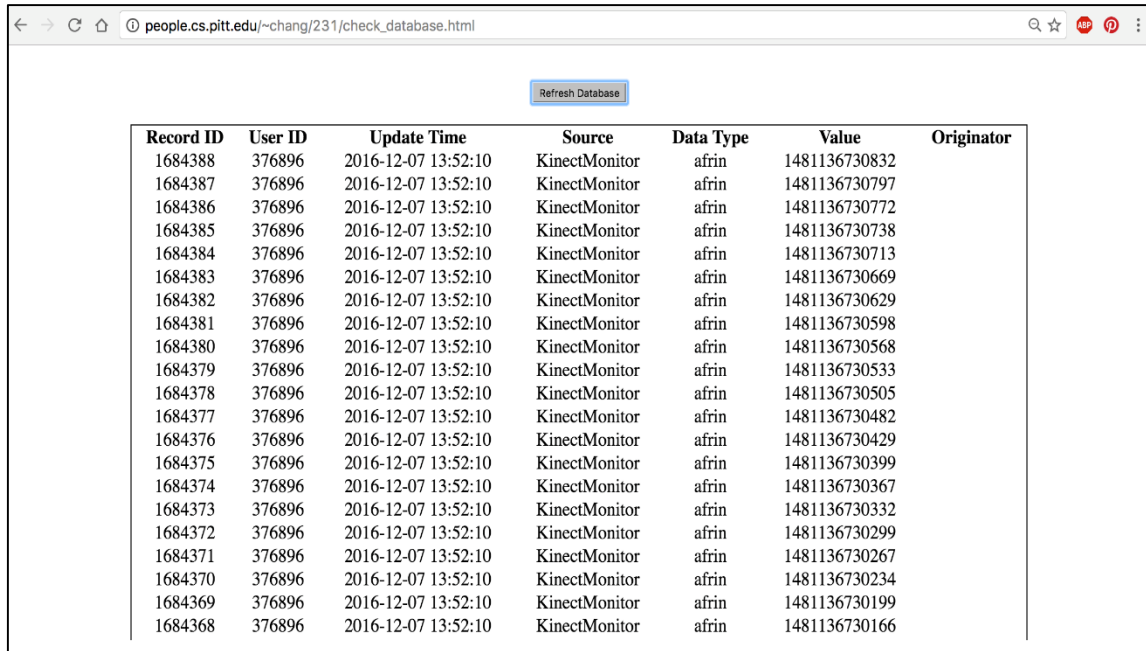
5. Exercise Steps

- Input Processor boots up and waits to listen from Kinect sensor
- Kinect Monitor boots up and asks the user how long the user wants to exercise.
- Taking the time to exercise from the user, Kinect Monitor sets the timer.
- When user start moving the arm, Kinect sensor gets the angle data and sends data to Input Processor.
- Input Processor sends the message to Kinect Monitor.
- Based on the angle, the Kinect Monitor shows the user if he/she is doing it right.
- After the exercise is done, or the selected time is over the Kinect Monitor sends data to Uploader to upload to the server.
- Kinect Monitor then shows the arm motion plot to the user.
- Kinect Monitor also shows the user how many calories are burned.

In the figure below the arm motion plot is shown –



A screenshot of the uploaded data to the data base is given below –



The screenshot shows a web browser window with the address bar displaying "people.cs.pitt.edu/~chang/231/check_database.html". Below the address bar is a "Refresh Database" button. The main content area displays a table with the following data:

Record ID	User ID	Update Time	Source	Data Type	Value	Originator
1684388	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730832	
1684387	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730797	
1684386	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730772	
1684385	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730738	
1684384	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730713	
1684383	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730669	
1684382	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730629	
1684381	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730598	
1684380	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730568	
1684379	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730533	
1684378	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730505	
1684377	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730482	
1684376	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730429	
1684375	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730399	
1684374	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730367	
1684373	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730332	
1684372	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730299	
1684371	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730267	
1684370	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730234	
1684369	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730199	
1684368	376896	2016-12-07 13:52:10	KinectMonitor	aftrin	1481136730166	

6. Challenges and Future Work

The first challenge while implementing the exercise monitor was to set the rules of exercise. In this project a user just moves his left arm and the components get the arm angle and works on it. But there are lots of opportunity here to improve. There are different kinds of exercise motion that can be implemented through this process. Also, if the user can select what exercise they want to do it can be a great improvement too. Also at the ends of the current implementation the Kinect Monitor returns a calculate burned calorie, which use the average arm motion angle to calculate calorie. This may not reflect the actual calorie burned all the time. If another sensor that calculates the heart rate can be incorporated with the Kinect sensor, the calories burned might be more accurate.

7. Demo Video:

Here is the link for the demo video :

<https://www.youtube.com/watch?v=pwJV5Vn0tnk&feature=youtu.be>