Zuha Agha zua2@pitt.edu

CS 2310 Software Engineering Final Project Report

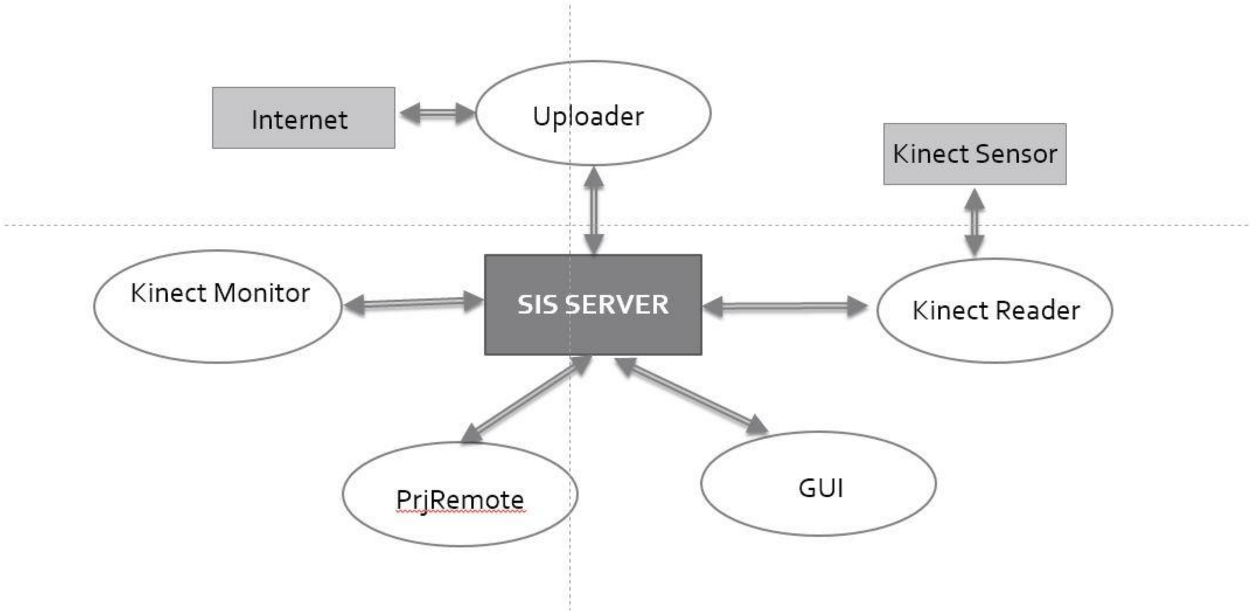
Arm Motion Assessment for Remote Physical Therapy Using Kinect

I. Introduction & Motivation

Kinect based remote physical therapy systems are becoming increasingly popular and a number of start-ups have emerged in recent years to tackle the problem of effective remote physical therapy using Kinect. Remote physical therapy aims to alleviate the inconvenience of in-person physical therapy as it is inconvenient for elderly people to commute to clinics for physical therapy. Not only this but rural areas or small towns may not have access to experienced physical therapists. In addition, remote physical therapy systems can greatly improve the efficiency of therapists as it enables them to attend multiple patients at the same time.

In this project, I implement a use case of remote physical therapy systems by focusing on real-time arm motion assessment for therapy using Kinect. Arm motion is assessed by using the left arm and right arm joint information obtained from Kinect tracking of the skeletal stream. Details of algorithm and implementation are given in Section 4 of the report. Section 2 discusses the system description and components, followed by message exchange between the components of my system in Section 3 of the report.

2. System description & Components



The system has the following main components,

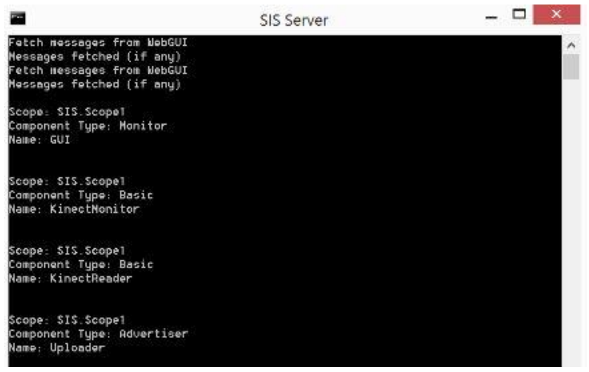
- GUI : a control panel to display and adjust component parameters
- Kinect Reader: The Kinect sensor will take three dimensional camera input to capture movement of skeletal points on the patient while the patient performs the exercise and computes arm motion angles (rightElbow angle and leftElbow angle) sent out as reading messages to Kinect monitor.

- Kinect Monitor: This component is meant to evaluate the patient arm motion of the patient and see if the movements lie within the threshold parameters specified by the instructor. If the movement exceeds the thresholds, count of the number of mistakes will be updated and an alert message with mistake count will be sent out.
- Uploader : Sends out an email to the therapist when alert message is received
- SIS Server: The SIS server is the center of all interactions and all components will interact with each other through communication via the SIS server. All components are registered to the SIS server and all messages are sent through the server.

Message Communication

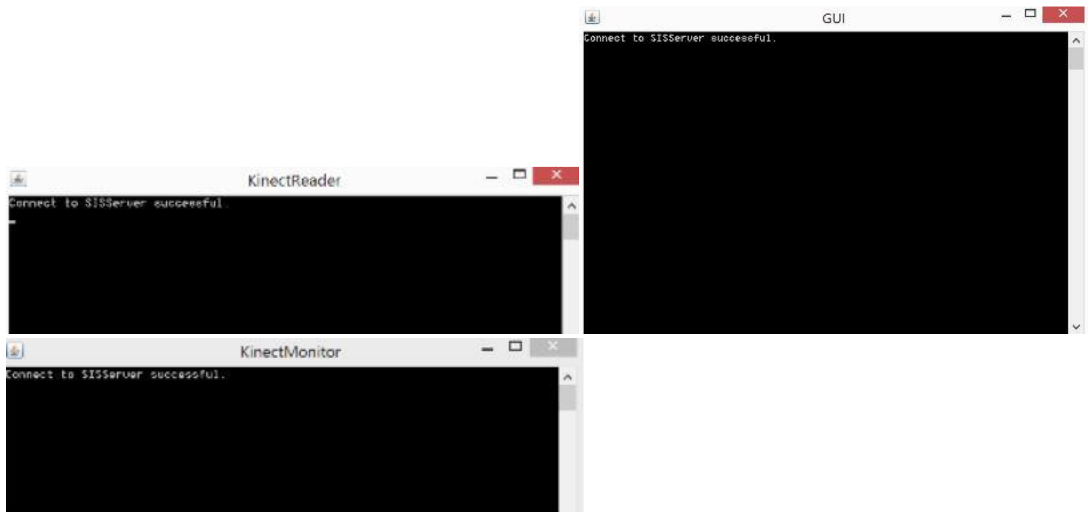
Given below is the sequence of snapshots showing message exchange between the components of the system.

1.All components are first initialized and registered to the SIS server.



```
Fetch messages from WebGUI
Messages fetched (if any)
Fetch messages from WebGUI
Messages fetched (if any)

Scope: SIS.Scope1
Component Type: Monitor
Name: GUI


Scope: SIS.Scope1
Component Type: Basic
Name: KinectMonitor


Scope: SIS.Scope1
Component Type: Basic
Name: KinectReader


Scope: SIS.Scope1
Component Type: Advertiser
Name: Uploader
```

2.Each component connects to the SIS server

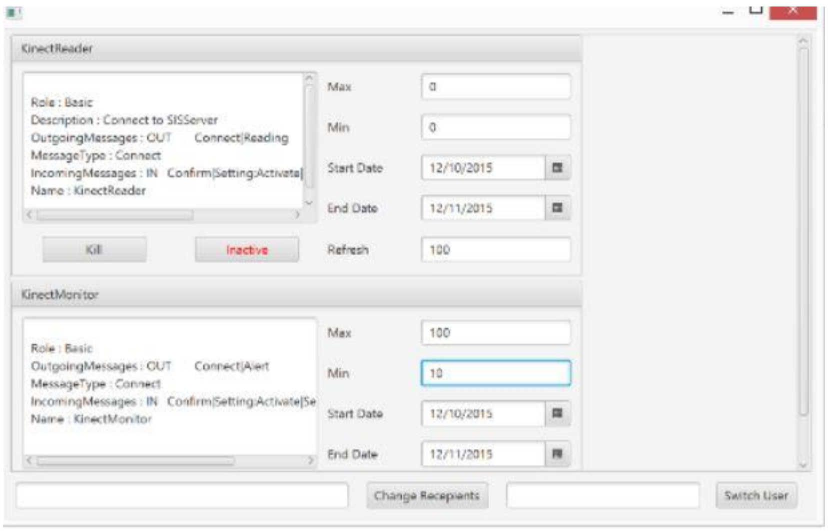


```
Connect to SISServer successful.
```

```
Connect to SISServer successful.
```

```
Connect to SISServer successful.
```

GUI launches the components after they are connected. The minimum and maximum thresholds are set on the GUI



KinectReader

Role : Basic
Description : Connect to SISServer
OutgoingMessages : OUT Connect|Reading
MessageType : Connect
IncomingMessages : IN Confirm|Setting:Activate|
Name : KinectReader

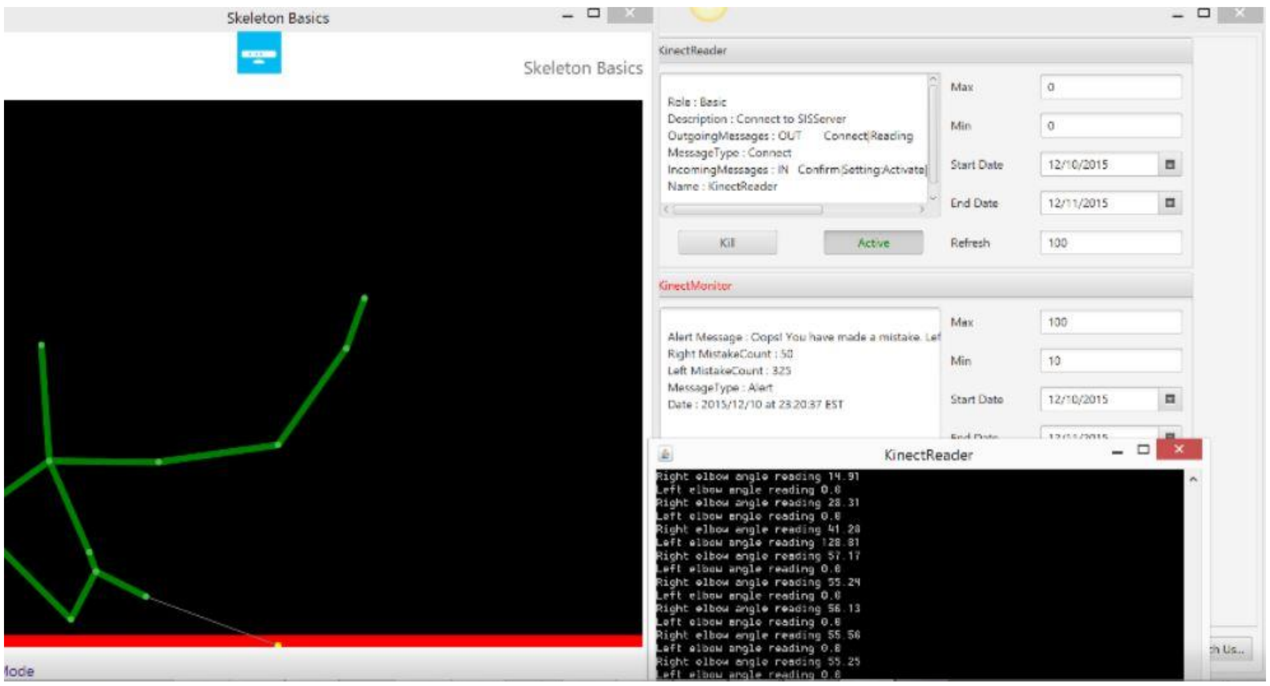
Max	0
Min	0
Start Date	12/10/2015
End Date	12/11/2015
Refresh	100

KinectMonitor

Role : Basic
OutgoingMessages : OUT Connect|Alert
MessageType : Connect
IncomingMessages : IN Confirm|Setting:Activate|Se
Name : KinectMonitor

Max	100
Min	10
Start Date	12/10/2015
End Date	12/11/2015

3. After activation, Kinect interface shows up and real-time left and right angle messages are sent from KinectReader to KinectComponent



KinectReader

Role : Basic
Description : Connect to SISServer
OutgoingMessages : OUT Connect|Reading
MessageType : Connect
IncomingMessages : IN Confirm|Setting:Activate|
Name : KinectReader

Max	0
Min	0
Start Date	12/10/2015
End Date	12/11/2015
Refresh	100

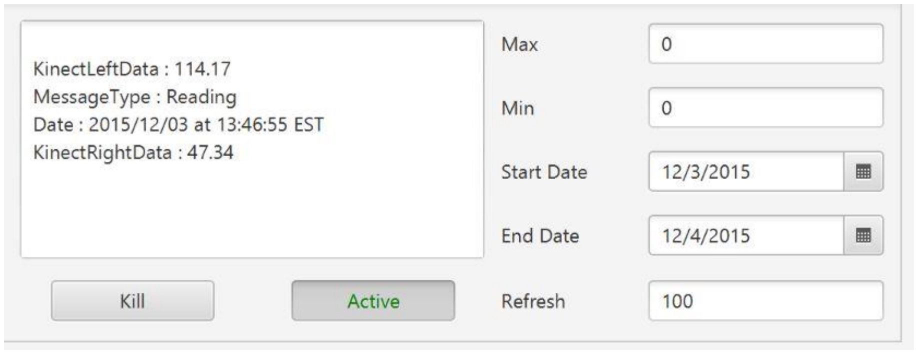
KinectMonitor

Alert Message : Oops! You have made a mistake. Lef
Right MistakeCount : 50
Left MistakeCount : 325
MessageType : Alert
Date : 2015/12/10 at 23:20:37 EST

Max	100
Min	10
Start Date	12/10/2015

```
Right elbow angle reading 14.91
Left elbow angle reading 0.0
Right elbow angle reading 28.31
Left elbow angle reading 0.0
Right elbow angle reading 41.28
Left elbow angle reading 128.81
Right elbow angle reading 57.17
Left elbow angle reading 0.0
Right elbow angle reading 55.24
Left elbow angle reading 0.0
Right elbow angle reading 56.13
Left elbow angle reading 0.0
Right elbow angle reading 55.56
Left elbow angle reading 0.0
Right elbow angle reading 55.25
Left elbow angle reading 0.0
```

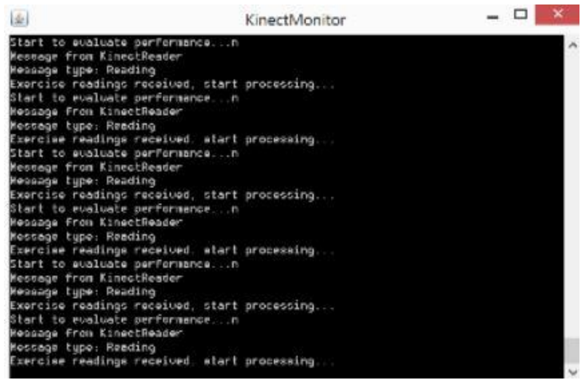
Given below is the format of the reading message sent by the KinectReader.



```
KinectLeftData : 114.17
MessageType : Reading
Date : 2015/12/03 at 13:46:55 EST
KinectRightData : 47.34
```

Max	0
Min	0
Start Date	12/3/2015
End Date	12/4/2015
Refresh	100

The Kinect monitor receives the reading message, checks if it is outside the provided thresholds, updates the mistake counts and sends an alert message shown in the KinectMonitor component on GUI above in (3)

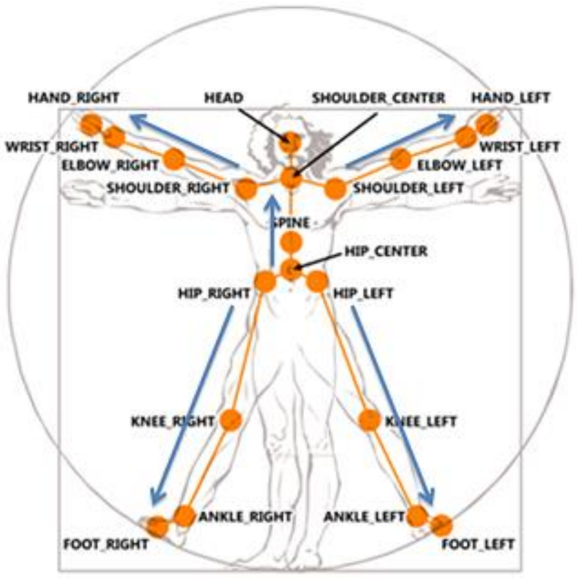


```
Start to evaluate performance...n
Message from KinectReader
Message type: Reading
Exercise readings received, start processing...
Start to evaluate performance...n
Message from KinectReader
Message type: Reading
Exercise readings received, start processing...
Start to evaluate performance...n
Message from KinectReader
Message type: Reading
Exercise readings received, start processing...
Start to evaluate performance...n
Message from KinectReader
Message type: Reading
Exercise readings received, start processing...
Start to evaluate performance...n
Message from KinectReader
Message type: Reading
Exercise readings received, start processing...
Start to evaluate performance...n
Message from KinectReader
Message type: Reading
Exercise readings received, start processing...
```

Kinect Sensor Angle Implementation

The Kinect Sensor component runs in C# and the KinectReader component in Java communicates with the sensor via sockets. As the KinectReader component needs angles only, I do not transfer all 20 skeletal joints information from the sensor but instead compute the angles in the C# program and send them as messages to the Java KinectReader component. Below is the snippet of the code that computes angles. Other than this input processing has been done in the C# program to obtain the skeleton figure and draw it on the interface from SkeletonStream of Kinect data.

```
    public double GetBodySegmentAngle(Skeleton skeleton, JointType jointType1, JointType jointType2, JointType jointType3)
    {
        Joint joint1 = skeleton.Joints[jointType1];
        Joint joint2 = skeleton.Joints[jointType2];
        Joint joint3 = skeleton.Joints[jointType3];

        Vector3D vectorJoint1ToJoint2 = new Vector3D(joint1.Position.X - joint2.Position.X, joint1.Position.Y - joint2.Position.Y, 0);
        Vector3D vectorJoint2ToJoint3 = new Vector3D(joint2.Position.X - joint3.Position.X, joint2.Position.Y - joint3.Position.Y, 0);
        vectorJoint1ToJoint2.Normalize();
        vectorJoint2ToJoint3.Normalize();

        Vector3D crossProduct = Vector3D.CrossProduct(vectorJoint1ToJoint2, vectorJoint2ToJoint3);
        double crossProductLength = crossProduct.Z;
        double dotProduct = Vector3D.DotProduct(vectorJoint1ToJoint2, vectorJoint2ToJoint3);
        double segmentAngle = Math.Atan2(crossProductLength, dotProduct);

        // Convert the result to degrees.
        double degrees = segmentAngle * (180 / Math.PI);

        return degrees;
    }
}
```



Demo

The demo of the project can be viewed at the link provided below,

http://screencast.com/t/YkjAYnZ6nJJ

Gems

The gems implemented are,

1.) Using Kinect Sensor
2.) Getting realtime reading from KinectSensor program rather than just an alert message
3.) Demo video