Project Milestone 2

Yue Dai
yud42@pitt.edu

April 6, 2020

1 Introduction

In milestone 2 I will briefly present my current progress on project. According to MS 1, I planned to have a Kinect gesture based light controller. Based on the design mentioned before, I will describe the C# based implementation, which includes the Kinect Sensor processing, and corresponding lightning scenarios.

2 Implementation Scenarios

The Current implementation mainly focus on Kinect sensor processing, and decision tree based lightning. In the following part I will present demo scenarios by screen shot and coding designs. And I will add socket/SIServer based feature to distribute and linking sensor/actuators in the future work. Additionally, better decision model would be implemented in the future if possible.

The entire project is coded in C#, with dependency on Visual Studio 2017, Microsoft Kinect SDK v1, and Microsoft Kinect Toolkit.

a Decision Tree based manager

The manager is written in C#. Due to lack of data for training, I used decision tree based manager currently. The manager will take input of two joint information from Kinect: Head and Activehand. Then it will make decision based on relative position between these two components. The Active hand is pre-processed by gesture recognizer such as the hand closer to the sensor (Z-axis based). And The manager will give signal to actuator for lightning case. The decision tree as figure 1. Though I used hard hop decision tree to make decision, the framework is flexible to any other decision models which take same input and return same signals. The reason that I choose these hand and heads is that usually these two components

is distinguishing among gestures. For instance, in case of signal 0, the hand is above head for a certain distance, would shows a gesture like raising or calling for help.

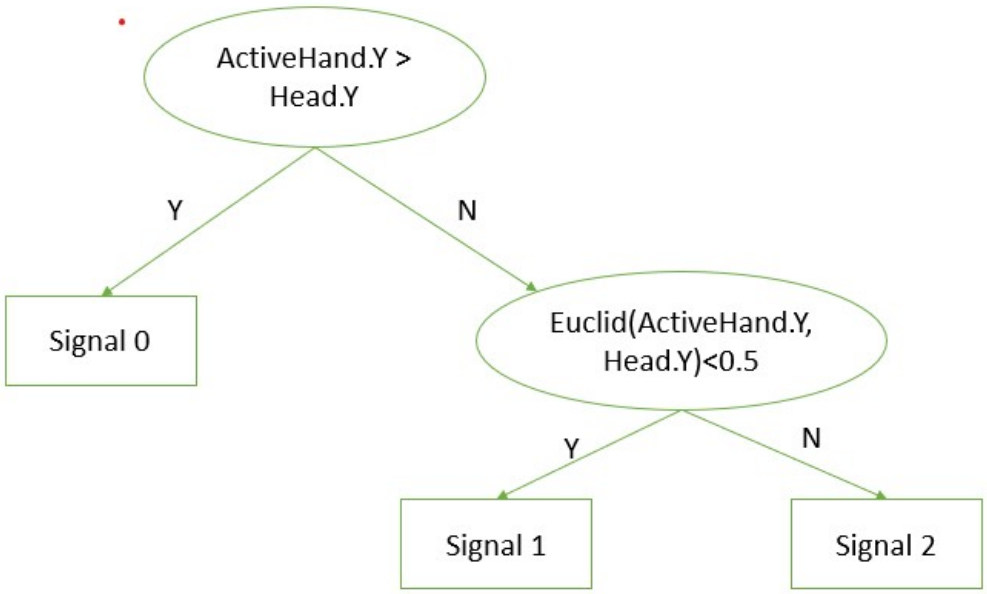

Figure 1: The manager decision tree

b Kinect Processor

The Kinect processor is implemented with Microsoft Kinect SDK v1, and built by Visual Studio 2017. It will triggered per frame, and shows abstract feature information such as joint position. Also to show the hand position I referred a KinectCursor open source project to show hand position on screen (https://github.com/Vangos/kinect-controls). The gesture processor will take joint information includes active hand position and head position as following.

- **Active hand.** The joint object represent effective hand, which contains spatial coordinates of the hand. The active hand is chosen by comparing Lefthand.Z and Righthand.Z to get the hand closer to the Kinect sensor.
- **Heads.** The joint object represent head spatial coordinates.

The detailed situation information shown in examples in following session.

c Lights

I planned to use programmable LED strip in MS1 yet the order is not delivered yet. Therefore, I used canvas drawing on GUI to represent the lights actuator in different scenarios. By taking signal 0 it will light Red, signal 1 it will light pink, signal 0 it will light lightblue. The demo scenarios are shown in figure 2.

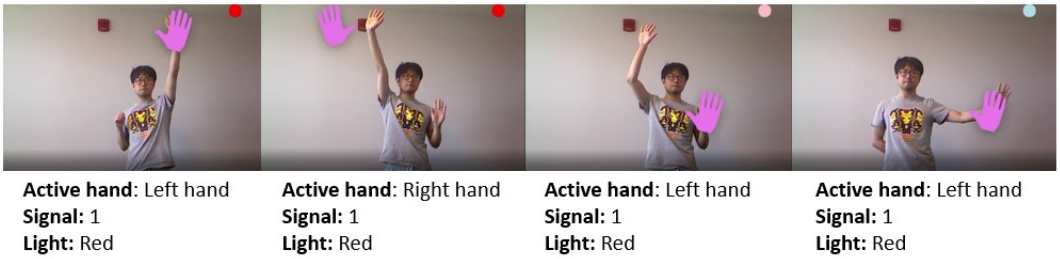

Figure 2: The Situation Demo

3 Conclusion and Future works

An video demo will be attached as well as current project codes. The Kinect processor and manager worked as expected so far. In the following weeks I will tried to link to an actual light client if possible, also, I will look for some pre-trained model to diverse the methods/approaches within manager component.