SHAPE Voice Interaction Report

Patrick Bowling
Andrew Lewis
Nick Strazis
Jonah Simon
Aaron Kwan

Concept And Research

To allow the user to incorporate gestures into the creation of the window, we wanted a method that could easily be expanded on so that there could be the possibility of combining gestures with eye tracking. Because of this, we chose to use the Kinect since the software development kit (SDK) allows for the tracking of many joints and body parts as well as the ability to develop our own gestures. To accomplish this, we used C# since the microsoft SDK can only be used with this language. We also added extensions to the project adding kinect libraries to the code. This was done via nuget package manager.

Development

Development of the Kinect started off on shaky ground. Much of the documentation online were for the beta SDK that microsoft released on launch day. Since then, Microsoft updated the libraries removing many of the vital functions and attributes that would have made the development smoother. One instance of this was the removal of a preset skeleton that could be attached to the user. Because of this, the team had to recreate these functions and attributes ourselves. Afterwards, development went smoothly. All of us had worked extensively with C# in the past. This made it easy to move forward and rapidly develop the application. Through the kinect, we track the hands and the thumb to indicate which symbol the hand is forming. We then compare the hand with a list of pre-built gestures to determine which gesture is being made. Once we get a match, we change the box based on the gesture made. Then, we translate the box into coordinates on the wall to form the window. Using the C# HTTP library, we then send the new window coordinates to the server and inform the user once it has been sent. This app functions very similarly to the Android App. Both take user input and then send the new window to the wall.

Current State

The app in its current state supports-

- Move it (left, right, up, down) (a lot, a little, normal)
    - Simply moves the window in the desired direction for the desired distance
- Make it (bigger, smaller, wider, thinner, taller, shorter) (a lot, a little, normal)
    - Simply resizes the window in the desired way for the desired amount
- Submit
    - Submits the wall to grasshopper from the preview

Future Plans

At this time the Kinect is fully functional and the backend is well integrated with our server, thus allowing all inputs to be directly sent to GrassHopper. We plan to improve the user experience by improving gestures, the user interface, and the way in which the Kinect interacts with the actual wall.

To improve the gestures, we will be working closely with Mona and Hamid and create gestures that follow their specifications. Currently, the gestures that we have developed are at times challenging to complete, and have high risk of user error. What we plan to do to resolve these issues is to create gestures for resizing the wall that allow the user to grab the top and bottom sides of the wall and drag them further open or drag them together, effectively opening or closing the wall. To complement this, we can create a similar gesture for grabbing the left and right side of the wall and making it wider or skinnier. This is a more intuitive approach that we think will allow users to use the full functionality of the Kinect with relative ease.

At this stage in the Kinect development process, we have created a simple User Interface that shows the Kinect's camera image, as well as a rough outline for how the user is

transforming the wall, and a rough position of the wall. We plan to change this UI so that it offers the user a more descriptive view of the wall and the changes that they are making to both the window's position on the wall, as well as the width and height of the window. We plan to create more of a visualization of what is happening that can be simplified for the user's convenience.

Lastly, the change the user makes to the current Kinect's representation of the window is only roughly translated to changes to the window on the actual wall. This is due to the information received from the Kinect being very different than the sizes and dimensions of the actual wall. For the most part, this will require fine tuning of the dimensions, and the translation to the actual wall. This is an issue that we are confident we will be able to fully resolve.

Requirements/Environment and Instructions to Run

To run the Kinect application, the user must have a Kinect and a computer running our Kinect Application. Additionally, the user must have the basic requirements for the wall (Server, Computer with Rhino, Internet Connection). To run the app, download the Kinect SDK and run our program in Visual Studio.