Shape Wall

Patrick Bowling (Team Lead), Aaron Kwan, Andrew Lewis, Jonah Simon, Nick Strazis

Problem Statement

As technology advances, we are able to interact more and more with the world around us. Our project aims to tackle the question of how we can have more control of our environment in a home setting based on our wants, needs, and mood, at any given time.

Solution - 3 Components

1. Grasshopper
    a. Rhino extension to generate a model based on given data
2. Server
    a. Hosts XML files containing the data for grasshopper
    b. Hosts a preview canvas
    c. Serves the data to Grasshopper
3. Front End (Interact with the wall)
    a. Android App
    b. Alexa Skill
    c. Kinect App
    d. Tobii Eye Tracking App

Server

- Consists Of
  - Server to host data that is stored as XML
  - PHP Scripts to receive data
  - Javascript/HTML5 Preview Canvas to view the wall before moving the actuators

Preview Canvas

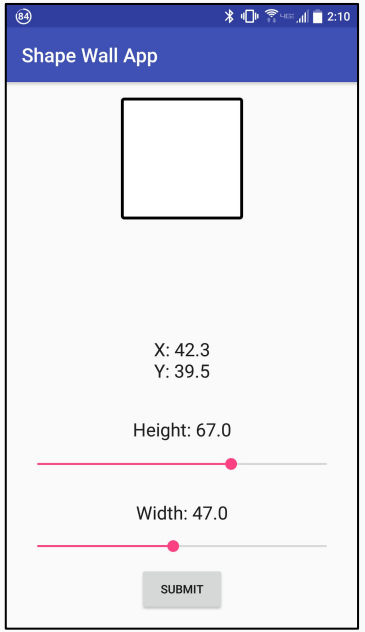
- A combination of Javascript and html which displays updates to the wall in real time before they are committed and sent to be processed by grasshopper
- Receives updates from all peripherals, stores them in an XML file, and displays them to a user's web browser. These represent changes made which have not been committed yet.

Grasshopper

- A attachment software to Rhino that allows data to be inputted via code
- Takes in an XML file from the server and converts the data into coordinates for the Rhino software

Android App: Overview

- Features
  - Place the window with the tap of a finger
  - Resize the window with height and width sliders
  - Submit the new window with the submit button



Android App: Technical Details

- Created in Android Studio with Java and XML
- Record the X and Y position of the user’s tap to place the center of the window
- Scale the X and Y coordinates to fit into the wall
- Send the X,Y coordinates and width/height slider values to server via Http Request

Android App: Demo

Live Demo

Android App: Future Plans

- Simplify the User Interface
- Add the preview canvas so that users can preview the changes within the app
- Add support for tablets
- Port to iOS
- Add support for multiple windows

Alexa Skill: Overview

- Features
  - Move the window
    - Up, down, left, right
  - Resize the window
    - Wider, taller, skinnier, shorter, smaller, bigger
  - Specify the amount for movement and resizing
    - A lot, a little, (normal)
  - Submit the wall from the preview canvas to the actual wall
    - Submit

Alexa Skill: Technical Details

- Written in Node.js
- Hosted on AWS Lambda
- Program Flow
  - Alexa processes speech
  - Speech text sent to AWS Lambda
  - Node.js App parses speech text, retrieves the current window specifications, then modifies the specs
  - Next the app makes a http request and returns the result (success/fail) to the user

Alexa Skill: Demo

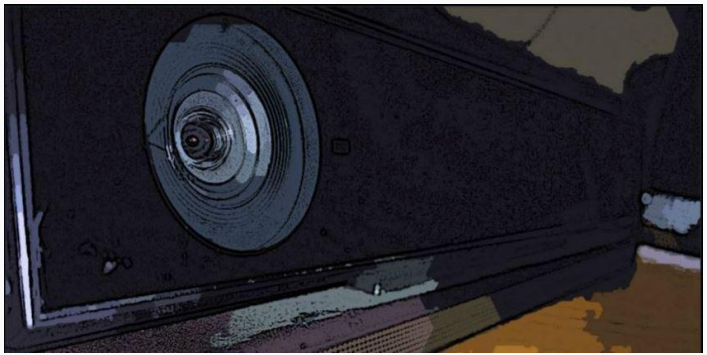
Live Demo

Alexa Skill: Future Plans

- Add support for multiple windows
- Add support for moving diagonally
- Add a larger variety of phrases to make interaction more natural

Kinect Application: Overview

- Features
  - Move the window by grabbing and dragging to new location
    - Up, down, left, right
  - Resize the window
    - Wider, Narrower, Taller, Shorter
  - Submit wall dimensions and location to preview after 5 seconds of user inactivity



Kinect 2: Tech Specs

- 1080p Color Camera
  - 30 Hz (15 Hz in low light)
- Body Tracking
  - Up to 6 bodies simultaneously
  - Support for 25 joints per body ex. Right Hand, Left Hand, etc.
- 4 Microphones
  - Record audio input
  - Determine location of audio source
- Depth Sensor
  - 512 x 424 resolution
  - 30 Hz

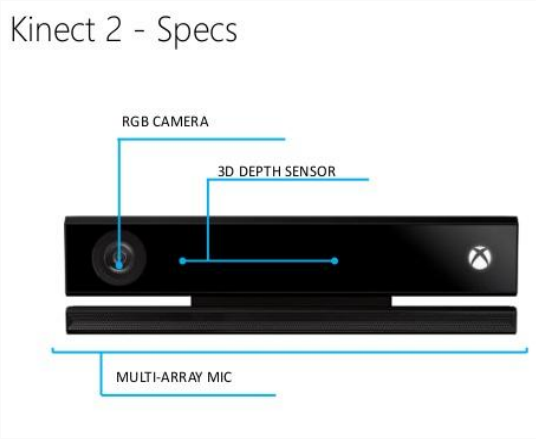


Photo credit: Matteo Valoriani

Kinect Application: Gesture Demo

- Right-Hand Closed Drag
  - Drag to change x, y location
- Right/Left Hand Swipe
  - Left-Hand Swipe Right
    - Increase Width
  - Right-Hand Swipe Left
    - Decrease Width
- Right-Hand Lasso (two fingers)
  - Drag Down
    - Decrease Height
  - Drag Up
    - Increase Height
- Left-Hand Closed
  - Clear all current commands

Kinect Application: Submit Demo

- Remove Hands from Kinect View
  - Application submits to preview after 150 frames of inactivity (approximately 5 seconds), the box will turn yellow.
  - After 10 seconds of inactivity, preview will be submitted to GrassHopper.
  - User will be notified of submission via the box turning green.

Kinect Application: Future Plans

- Improve gestures in accordance with mentor requests.
- Improve Kinect Application interface, including making it more user-friendly and displaying more useful information.
- Incorporate with Tobii Eye to allow users to use both vision and gestures in unison.

Tobii Eye Tracking App

- Hardware: Tobii EyeX
- SDK provided for Microsoft .NET
- Compatible with WPF

Tobii Eye: Future Plans

- Create GUI
- Develop methods for data interpretation
- Integration with Kinect

Plans For Continuation

- Develop eye-tracking support
- Polish app UI, make it more intuitive
- Create iOS app
- Provide support for multiple walls
- Update data collection methods as physical wall is developed
- Combine eye-tracking and gestures into one interface