3D Manipulation Using Leap Motion
Project Design Report

Daniel Haberstock
Maciej Ogrocki
Marcin Pietrasik
Randy Wong
Yufei Zhang

Project Description:

Team members: Daniel Haberstock, Maciej Ogrocki, Marcin Pietrasik, Randy Wong, Yufei Zhang

Client: Kumaradevan Punithakumar

Teaching Assistant: Michael Feist

The goal of our project is to implement a system that allows a user to interact with an object using hand motions and gestures. We will use a high level 3D graphics toolkit to create an object and then interface it using the Leap Motion.

Project Use Cases:

Use Case 1.1: As a user, I want to rotate a 3D object

Category	Description
Participating Actors	User
Goal	Rotate a 3D object
Trigger	User has an object that they would like to rotate
PreCondition	The application is running and the user knows gesture
PostCondition	Object is rotated to the user's desire
Flow	1. User has both hands open and facing the leap 2. User moves one hand in both the x and y direction (depends on mode) 3. System responds with corresponding movement
Exceptions	1. User not in correct rest position 2. User moves both hands

Use Case 1.2: As a user, I want to move a 3D object

Category	Description
Participating Actors	User
Goal	Move a 3D object
Trigger	User has an object that they would like to move
PreCondition	The application is running and the user knows gesture
PostCondition	Object is moved to the user's desire
Flow	1. User has both hands open and facing the leap 2. User clenches one of their hands 3. User moves open hand in both the x and y direction 4. System responds with corresponding movement
Exceptions	1. User not in correct rest position 2. User clenches both hands 3. User moves fist instead of hand

Use Case 1.3: As a user, I want to zoom in on a 3D object

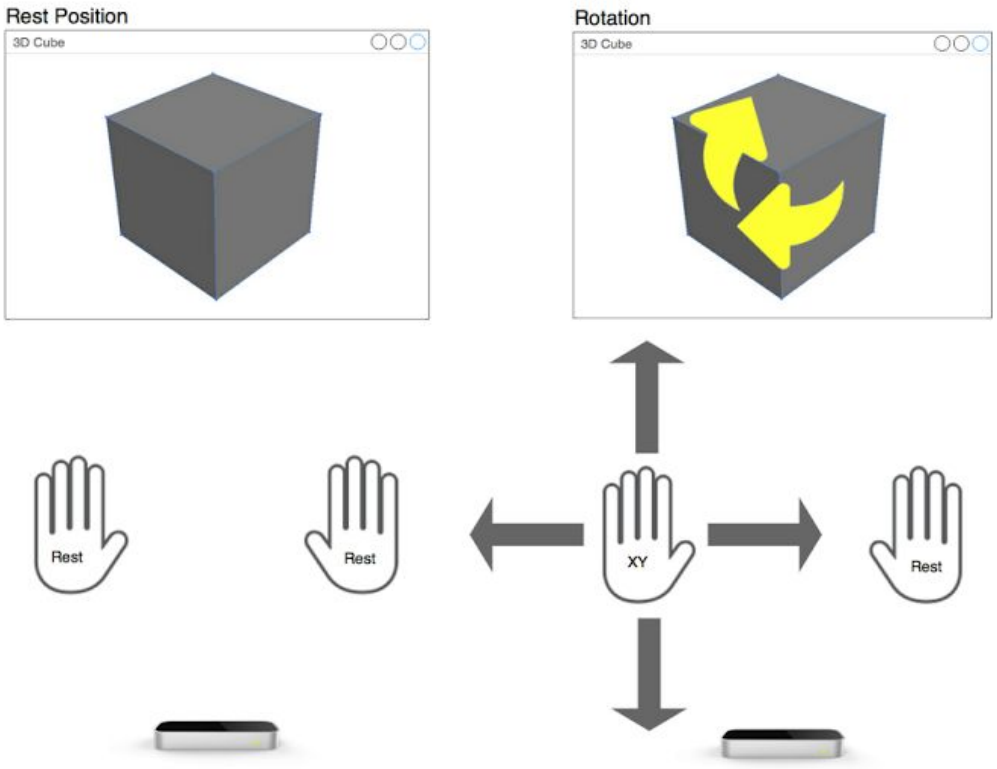
Category	Description
Participating Actors	User
Goal	Zoom in on a 3D object
Trigger	User has an object that they would like to zoom in on
PreCondition	The application is running and the user knows gesture
PostCondition	Object is zoomed to the user's desire
Flow	1. User has both hands open and facing the leap 2. User clenches one of their hands and leaves their thumb out 3. User moves open hand in the y direction 4. System responds with corresponding movement
Exceptions	1. User not in correct rest position 2. User has other fingers out 3. User moves fist instead of hand

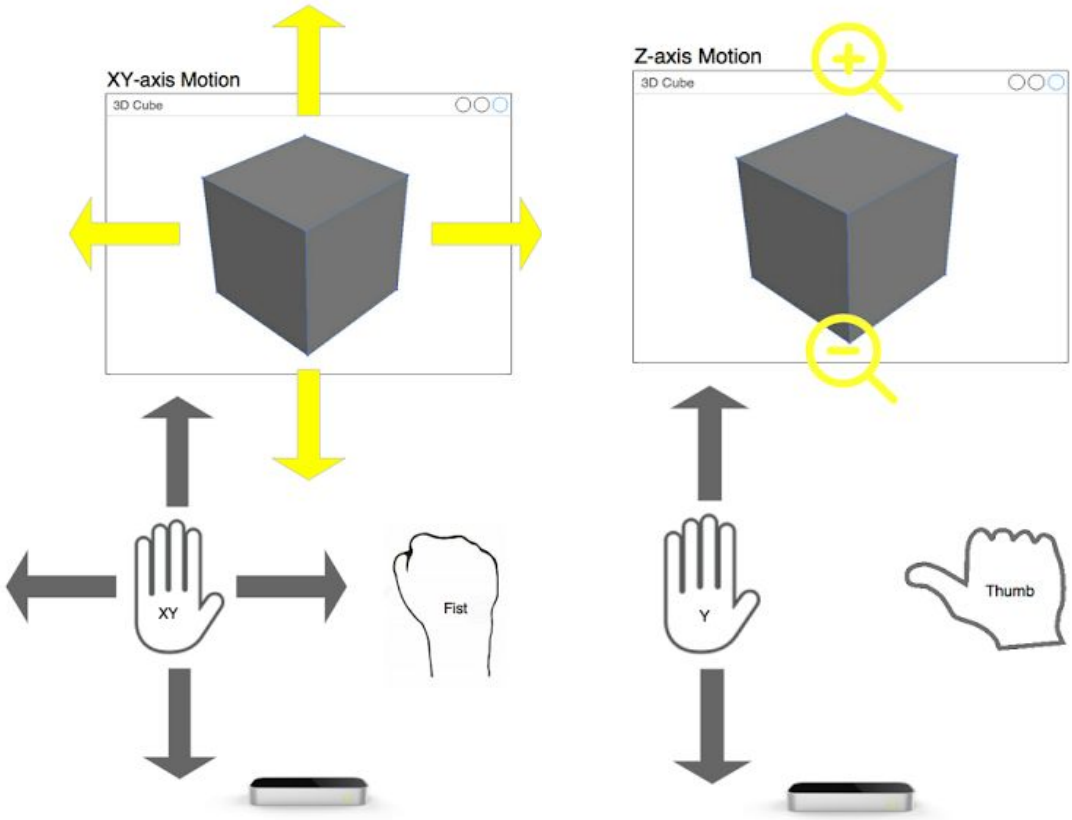
Motivation and goals:

Currently the client has a system that allows a user to wear glasses to see a projected object in 3D, and can manipulate it using a stylus. While this kind of development is a start in the right direction, being forced to use a stylus to interact with a 3D image can be detracting from the task. The Leap Motion, is a motion sensor that allows users to interact with software using hand movements and gestures. Our client is asking for an initial prototype application that creates an interface using the Leap to manipulate objects. We will be creating a testing ground within Unity to allow users to practice and try out designed hand gestures when interacting with simple objects, say a plain black cube, which will be a first step towards being able to replace the clients stylus interface.

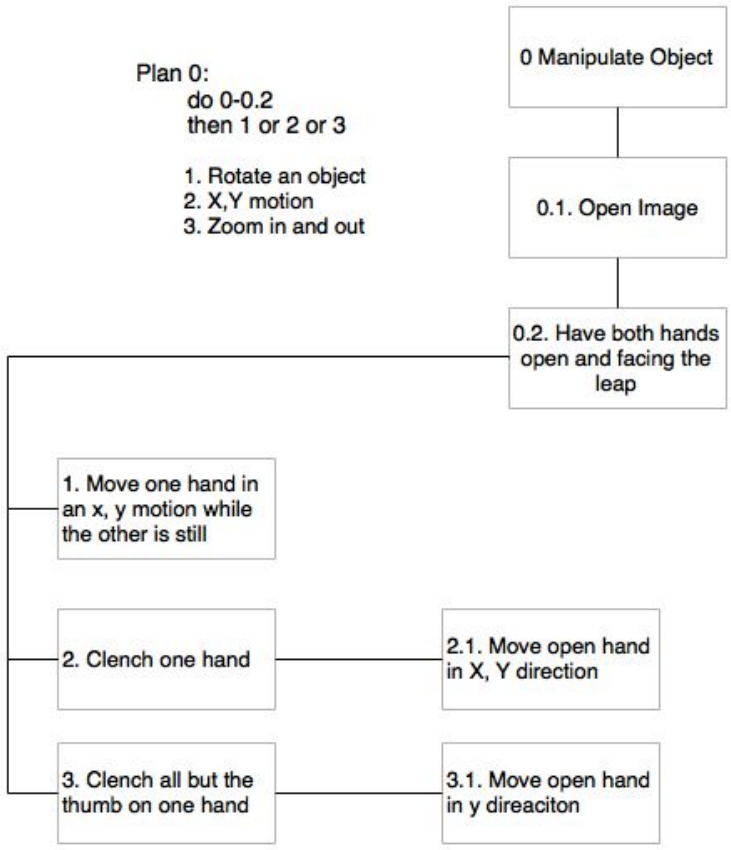
Our goals are to create a set of hand gestures that allow a user to quickly and easily manipulate viewing of a 3D object, though projected to the user from a normal 2D screen, that are easy to extrapolate into 3D projection and later fully onto the clients predefined system. Ultimately we will create prototype code so design can be modified or improved after testing, before importing it as a replacement for the stylus.

Interaction Design:





System Design:



Software and Implementation:

Hardware:

Leap Motion Controller

The leap motion controller is a computer hardware sensor that supports hand and finger motions as input, analogous to a mouse but requires no hand contact or touching. Connected to a host computer via USB port The Leap Motion system recognizes and tracks hands, fingers and finger-like tools. The device operates in an intimate proximity with high precision and tracking frame rate and reports discrete positions, gestures, and motion.

Specifications:

- 200 Frames per Second using infrared cameras
- 150° Field of View
- 8 Feet³ Total Input Area

Software:

Unity 5

Unity is a cross-platform game engine developed by Unity Technologies and used to develop video games for PC, consoles, mobile devices and websites. This project will be using the current latest stable version is Unity 5.3.2. It is Written in C, C++ and C# this project will be using C# as it is also what the Leap SDK uses.

Specifications:

- PhysX 3.3
- WebGL preview
- HDR Reflection Probes
- 64-bit and 32-bit Editor
- Pre-built AI and Animation support

Leap Motion SDK

Leap Motion API is a SDK by Leap Motion, Inc for Windows, Mac OS, and Linux that allows for simple connection of the data recorded from the Leap Motion Detector Hardware and the

Software of Unity 5. The Leap Motion API has a C# SDK for Unity 5 which allows for simple integration. As well as direct high level commands within Unity 5 for ease of use.

API Overview:

The leap motion API measures physical quantities with the following units:

- Distance : Millimeters
- Time : Microseconds
- Speed: Millimeters/Second
- Angle: Radians

It tracks motion tracking data per Frame and detects objects as well as information about their position and orientation. It classifies objects as : Hands, Arm, Fingers and Tools. Following this these tools are able to have motions which is basic type of change in 3D space. For example “a Hand can have the motion of Translation to reflect the change in position. Finally the leap motion software also recognizes certain movement patterns as gestures to indicate commands such as swiping or circling. Along with the computed tracking data, it is also possible to use the raw sensor images from the Leap Motion cameras.

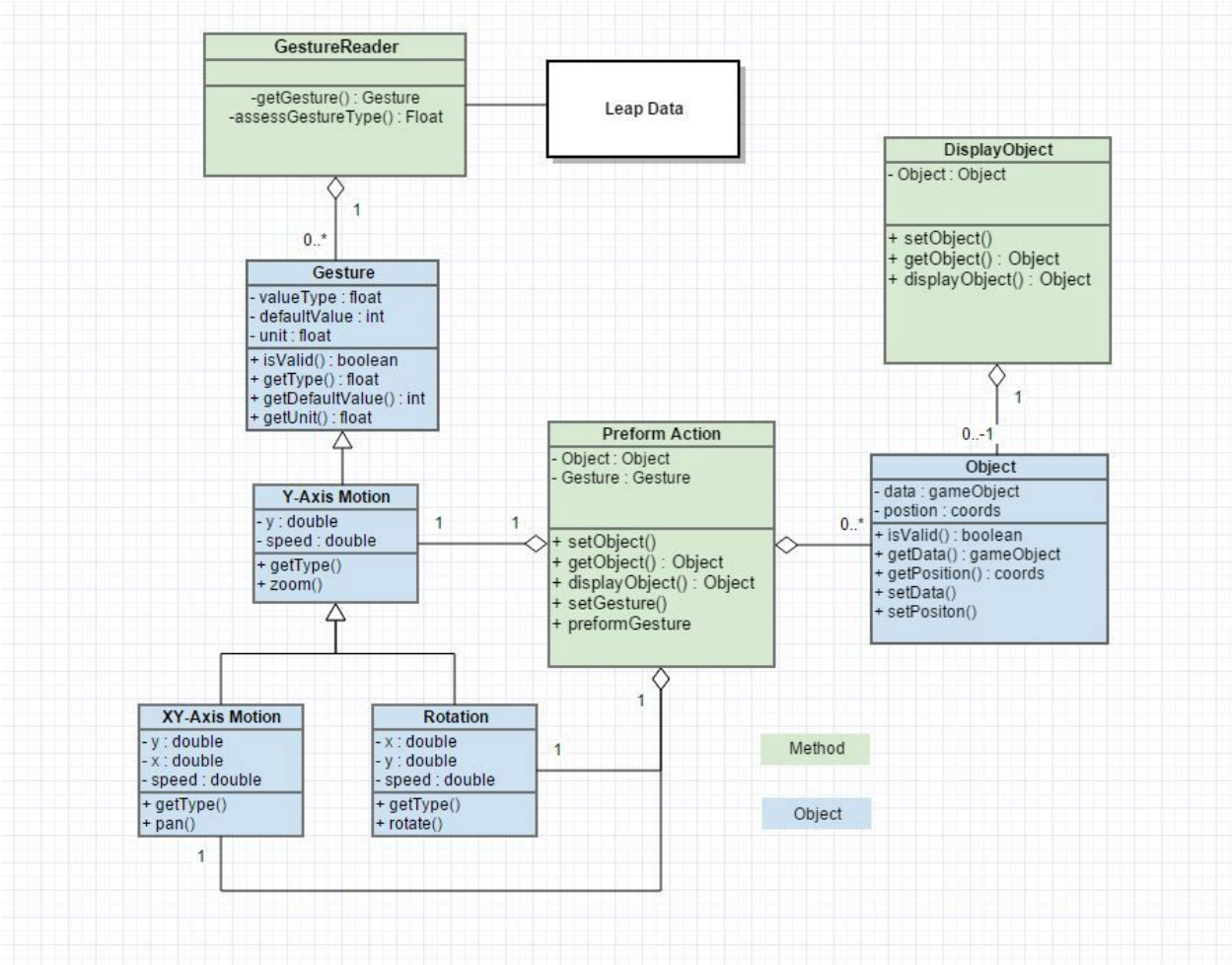
Implementation:

This project will use the Leap Motion Detector to record sensor data that will be then read by the Leap Motion SDK and sent to the Unity 5 environment in which the modification of the object will actually occur. Using the Gestures provided by the Leap Motion SDK users should be able to rotate the object in the 3D environment. Any Gestures that are not implemented by default in the SDK will be added using the tracking information provided from the Leap Motion Detector.

A example of a basic action done by a user:

The user swipes their hand from left to right above the Leap Motion Detector which records the change in sensors which is sent to the Leap Motion SDK. The Leap Motion SDK classifies the data as a swipe from left to right of a specific speed and of a specific distance. Finally Unity 5 receives the command that a swipe command has been performed which causes Unity 5 to rotate the 3D object counter clockwise on the X axis at a specific speed based on the speed of the swipe.

UML DIAGRAM:


GestureReader
-getGesture() : Gesture
-assessGestureType() : Float
Leap Data
DisplayObject
- Object : Object
+ setObject()
+ getObject() : Object
+ displayObject() : Object
1
0..*
Gesture
- valueType : float
- defaultValue : int
- unit : float
+ isValid() : boolean
+ getType() : float
+ getDefaultValue() : int
+ getUnit() : float
1
0..-1
Preform Action
- Object : Object
- Gesture : Gesture
+ setObject()
+ getObject() : Object
+ displayObject() : Object
+ setGesture()
+ preformGesture
Object
- data : gameObject
- postion : coords
+ isValid() : boolean
+ getData() : gameObject
+ getPosition() : coords
+ setData()
+ setPositon()
Y-Axis Motion
- y : double
- speed : double
+ getType()
+ zoom()
1
1
0..*
XY-Axis Motion
- y : double
- x : double
- speed : double
+ getType()
+ pan()
Rotation
- x : double
- y : double
- speed : double
+ getType()
+ rotate()
1
1
1
Method
Object

Empirical Evaluation:

Purpose: The purpose of this experiment is to evaluate if the speed of the leap interface is different to the traditional stylus.

Materials: 10 Subjects, stopwatch, computer, leap motion, stylus, interface softwares.

Methods:

1. There will be two groups of 5 subjects (randomly assigned):
    a. Testing the time with the stylus interface
    b. Testing the time with the leap motion interface
2. Each group will perform 3 tasks each starting from rest position:
    a. Rotate the object 360 degrees
    b. Move the object all the way left, then right
    c. Zoom in all the way
3. The time for each task will be recorded

Controls: The group that tests the time with the stylus interface is our control group. The independent variable for our experiment is the interface software that we use. The dependent variable is the time it takes to execute each task.

Data Interpretation: We will gather the data and perform a t test with $\alpha = 0.05$ to examine whether the two populations means are different. Time permitting we will test an additional model of interaction (model 3) using the leap; an anova test will be used to test for differences.

We will define our null hypothesis for the t-test as:

H_0: $\mu_1 = \mu_2$ (where μ_1 is model 1, μ_2 is model 2)

For our anova test we define the null hypothesis as:

H_0: $\mu_1 = \mu_2 = \mu_3$ (where μ_1 is model 1, μ_2 is model 2, μ_3 is model 3)

Project Schedule: