3D Manipulation Using Leap Motion
Final Report

Daniel Haberstock
Maciej Ogrocki
Marcin Pietrasik
Randy Wong
Yufei Zhang

Abstract:

This project evaluates the use of touchless input methods applied to the task of manipulating a virtual three dimensional object. Through the use of a holographic projection we were able to recreate an intuitive user interaction for object manipulation. In order to test our method against more traditional ones, we tested subjects on their speed of finding a specified swide of a coloured cube. The results indicate that our method is slower than traditional methods.

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 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.

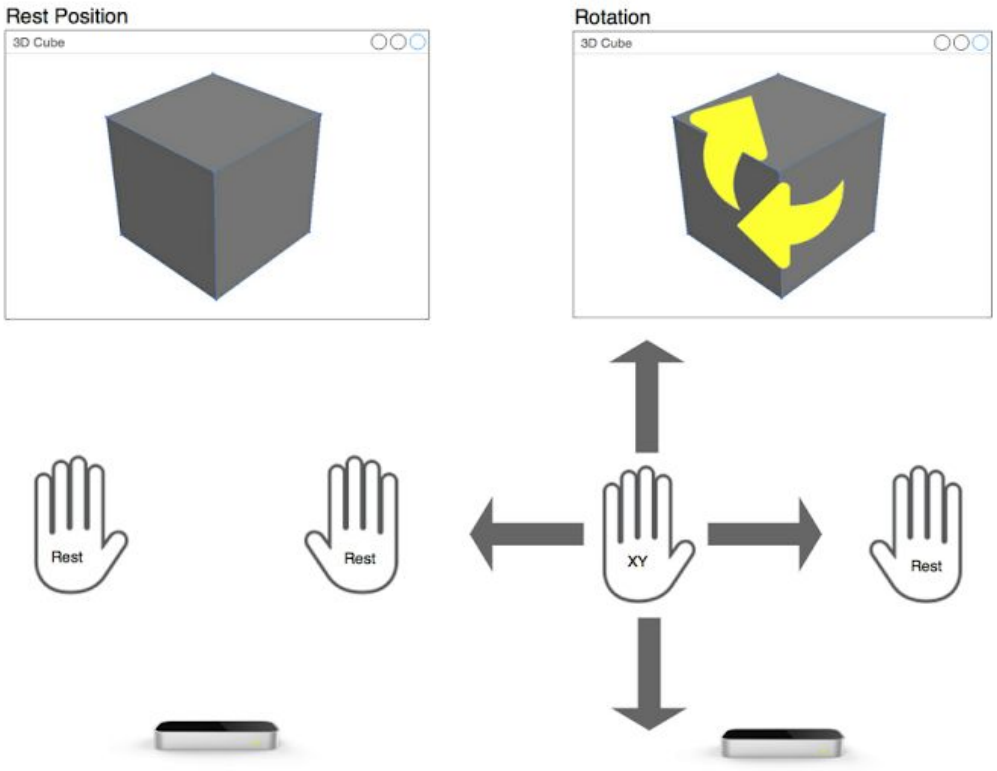
Our goals were originally to create a set of hand gestures that allow a user to quickly and easily manipulate viewing of a 3D object, though projected to the user from a normal 2D screen, that are easy to extrapolate into 3D projection and later fully onto the clients predefined system.

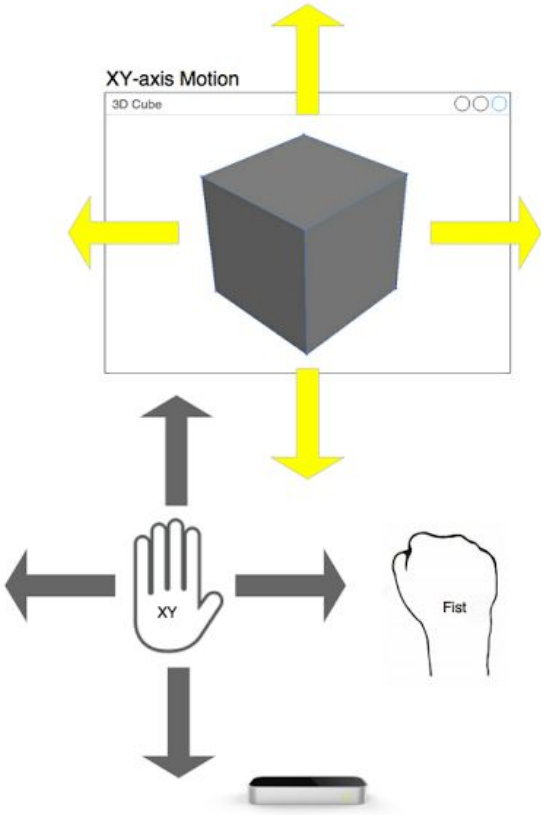
However, through iteration we found that gestures felt clunky to use, and found smoother more real-world-like ways to interact with a 3D object through pinching, and “hand

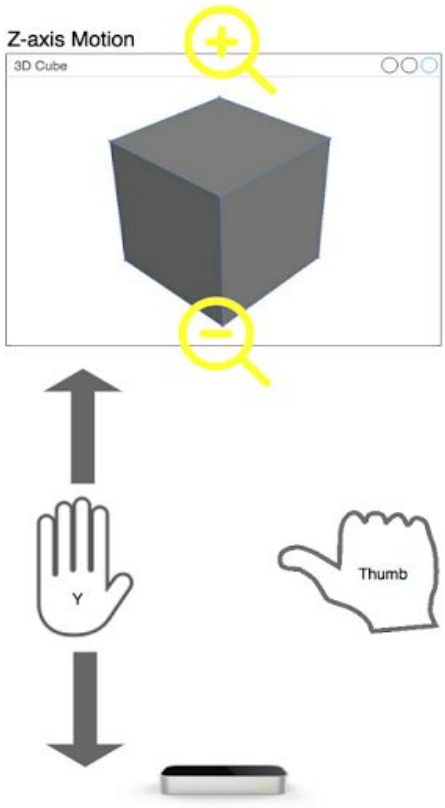
colliding". Ultimately we disabled panning and zooming for our testing because while they added functionality, they greatly detracted from a user's ability to control with finesse the object itself and we wished to measure control. As we're very confident from the field of human-computer interaction, that physical manipulation beats all other kinds of interaction, we wish to achieve as physical-like of an implementation as possible and this will hopefully be an improvement over a detached keyboard style of virtual object manipulation.

Our project will use hand-tracking, and a hologram-like projection to give the user a feel of interacting with the virtual object but in real-world space.

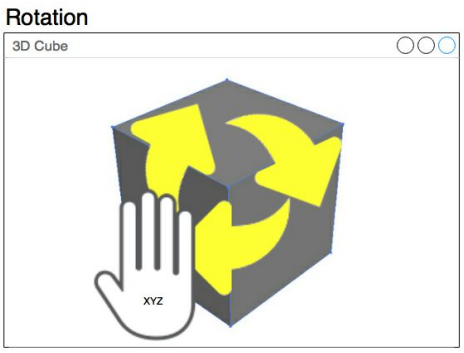
Original Interaction Design:







Current Interaction Design:



In the current iteration, the tested version, of our design we have eliminated gestures as they seem qualitatively worse to use in comparison to interacting with an object as if it was a collidable, physical entity. We now instead track hand position and create a hand-object that simulates our real-world movement within the virtual space. In addition, we have used a screen design that gives us hologram-like projection which means that a user can see the object projected in open air in front of them so that if calibrated for the person, their hand in physical space is directly mapped visually to the project. In essence we are attempting to create a hologram that can be interacted with physically.

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.. 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 across the “hologram” cube causing the virtual hand to collide with the virtual object, and it reacts as if it were physical. Swiping from right to left will make the right hand side of the cube come to the forefront as it spins to the user’s left.

Empirical Evaluation:

Purpose: The purpose of this experiment is to evaluate if the speed of our design using the leap interface is improved over using a classical keyboard set up that allows rotation, and compared to a person's ability to manipulate a physical object.

Materials: Subjects, stopwatch, computer, leap motion, interface softwares, rubik's cube.

Methods: In our experiment we gave subjects 3 ways to attempt to perform the same task. Our task was to have the user bring to the forefront the correct side (determined by colour) of a cube repeatedly.

Our three interfaces:

1. Physical (Rubik's Cube)
2. Virtual (Keyboard)
3. Virtual (Leap Motion)

The task: For each of the above interfaces we generated 5 trials where each trial consisted of 10 consecutive colours randomly selected (from the 6 colours of a rubik's cube, with the exception being that a colour cannot be the same as the one directly before it). A user started from the green side at the beginning of each trial and must navigate to each of the colours in the trial, in order. The next colour the user must navigate to is read out of the examiner when they are satisfied with the subjects correct execution of the previous task. Time is taken until the completion of the 10th task.

Controls: 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.

Statistical Method: We will use ANOVA to test the difference between the physical, keyboard, and Leap Motion results. The ANOVA test will be carried out with $\alpha = 0.05$.

Our null hypothesis is that the three groups (physical, keyboard, Leap Motion) are the same:

H_0: $\mu_1 = \mu_2 = \mu_3$ (where μ_1 is physical, μ_2 is keyboard, μ_3 is Leap Motion)

H_1: Not all μ are the same

After running ANOVA, we will run two t-tests that compare our new Leap Motion interface against the traditional methods. We will continue to use $\alpha = 0.05$ and our hypothesis will be defined as follows:

Test 1:

H_0: $\mu_1 = \mu_2$ (where μ_1 is physical, μ_2 is Leap Motion)

H_1: $\mu_1 \neq \mu_2$

Test 2:

H_0: $\mu_1 = \mu_2$ (where μ_1 is keyboard, μ_2 is Leap Motion)

H_1: $\mu_1 \neq \mu_2$

Furthermore we would like to see whether or not there is a learning process involved in the Leap Motion test. It would be reasonable that as the subjects get more familiar with the user interaction their times would decrease. To test this, we will use a simple linear regression.

Results:

The results of the ANOVA analysis:

Source of Variation	Sum of Squares	Degrees of freedom	Mean Squares	F value
Between	2784	2	1392	94.70
Within	352.8	24	14.70	
Total	3137	26		

Since the F value is 94.70, we can reject the null hypothesis.

The results of the t-test comparing the physical data with the Leap Motion data:

Rubik's cube	Leap Motion
N = 9 Df = 8 Mean = 13.73 SS = 13.73 S^2 = 1.72	N = 9 Df = 8 Mean = 39.88 SS = 277.52 S^2 = 34.69

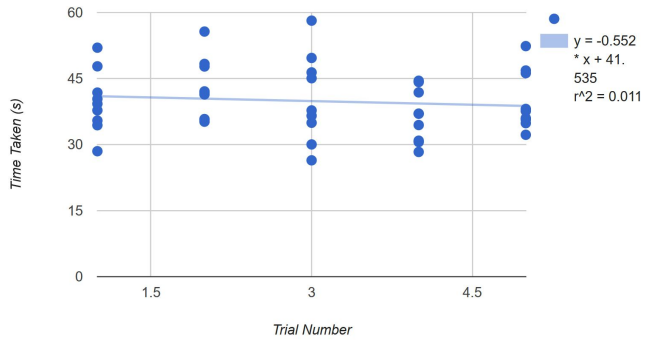
Thus, the *t*-value is -12.3107. The *p*-value is < .00001. We can reject the null hypothesis.

The results of the t-test comparing the keyboard data with the Leap Motion data:

Keyboard	Leap Motion
N = 9 Df = 8 Mean = 25.45 SS = 61.53 S^2 = 7.69	N = 9 Df = 8 Mean = 39.88 SS = 277.52 S^2 = 34.69

The *t*-value is -6.648. The *p*-value is < .00001. We can reject the null hypothesis.

The results of the simple linear regression.



As we can see, there is a slight negative correlation, indicating that times are decreasing. That being said, however, with $R^2 = 0.011$ the model only explains 1.1% of the variability. Based on this data, it is not reasonable to assume that the times are going down due to practice from the subjects.

Discussion: We found that there is a statistically significant difference between all three of our test models with $\alpha = 0.05$. The physical model, as expected, performs the best. Unfortunately for us, there is also a statistically significant difference between the keyboard version and our Leap motion design with our design coming out as the worst model for object manipulation. We believe that the poor performance on our Leap design is due to inexperience, in comparison to users who have much higher pre-learned keyboard usage capabilities, as well as calibration issues that we had with our Leap Motion model. The first error could be handled by using subjects that are heavily inexperienced, or heavily experienced with both systems, and is thus more of an experimental error on our part as our access to subjects is limited. The second issue we had with calibration caused user's hands to be in incorrect position relative to the hologram

cube, which we believe could have caused complications where the discrepancy tricks the mind and affects subjects scores. In addition to calibration of the Leap's detection position being off, we believe that better calibration of cube weight, and collision detection would improve usability. While we did find our current Leap implementation to perform worse than a keyboard controlled rotation model, we believe that improving on experimental error and calibration error would greatly improve our system.

Future Work: To follow up on this project we would firstly approach the above mentioned calibration error, and attempt to either find parameters that work for all users, or perhaps develop a way for users to calibrate the system to their own height, arm-length, etc. Since we believe this would have a significant impact on improving our system we would also then have to re-run an experiment, and control for experience with manipulation interfaces so that we do not compare the highly refined experience modern young students have with keyboards with a fresh technology we are developing that has not been used before. The next step after that of course is to incorporate real 3D technology like our clients current system with the leap and run more tests on this setup. After this, if results are promising we can move forward to completely replace the stylus model with a hands-free Leap powered solution to 3D model viewing and manipulation, ultimately fulfilling the client's end goal.

Appendix

People	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	Subject 7	Subject 8	Subject 9
Physical									
1	15.92	14.2	16.55	13.64	16.61	13.33	15.81	13.87	17.24
2	14.96	13.77	17.6	14.62	17.6	16.13	14.45	13.39	14.29
3	15.54	16.99	19.96	14.95	17.84	16.35	14.79	13.35	15.6
4	14.94	11.57	16.63	13.89	15.67	13.28	11.68	14.8	13.52
5	15.04	13.97	15.64	14.37	18.36	15.65	14.55	12.84	14.67
Avg	15.28	14.1	17.276	14.294	17.216	14.948	14.256	13.65	15.064
Keyboard									
1	27.3	26.14	22.32	26.52	22.43	29.46	24.79	28.46	28.32
2	33.04	24.37	23.89	35.38	27.15	20.61	23.37	26.93	26.11
3	28	23.93	22.48	26.83	30	25.86	21.82	22.48	26.84
4	27.23	20.49	23.58	32.31	26.22	24.5	19.05	20.85	24.07
5	32.6	25.4	22.19	29.87	25.44	18.6	24.43	21.11	22.66
Avg	29.634	24.066	22.892	30.182	26.248	23.806	22.692	23.966	25.6
Leap									
1	52	40.36	35.46	47.77	37.78	34.4	28.5	41.81	39.26
2	35.61	47.73	35.2	48.32	41.76	42.08	41.39	35.8	55.66
3	46.39	45.07	30.04	49.67	37.76	34.96	26.43	36.55	58.14
4	30.89	37.01	37.03	44.5	34.44	28.33	30.58	41.85	44.2
5	34.87	38.1	36	46.82	35.67	37.61	32.23	46.22	52.36
Avg	39.952	41.654	34.746	47.416	37.482	35.476	31.826	40.446	49.924