CARNEGIE MELLON UNIVERSITY

School of Architecture

College of Fine Arts

Thesis

Submitted in Partial Fulfillment of the requirements for the degree of

Master of Science in Computational Design

TITLE:

A Mixed Modeling Environment Exploring Tangible and Digital Interactions for Iterative Design Modeling

AUTHOR:

Hongtao Ma

ACCEPTED BY ADVISORY COMMITTEE:

[redacted] May 19, 2020

Daniel Cardoso Llach Principal Advisor DATE

[redacted] May 17, 2020

Eddy Man Kim Advisor DATE

Abstract

This thesis explores the concept of embodied virtuality in design modeling and proposes a way of creating 3D models through the concurrent manipulation of both physical and digital representations. Inspired by researchers' efforts to explore the interaction patterns for 3D modeling, the idea of taking advantage of both tangible and graphical interfaces is emerging: the physical space supports embodied interaction, while the digital space offers parametric editing operations as well as some post-processing functions. Based on an analysis of previous researches, this thesis argues that by combining the half-edge mesh data structure with a physical encoding of geometries in voxels, we can leverage some of the strengths of both digital and physical modeling environments, opening new ways for professional and non-professional users to create geometric models.

In this thesis, a mixed modeling system is prototyped as a modeling tool. The system supports concurrent interaction with both the tangible cubes and the graphical interface, making it convenient to iterate the design. The project aims to set up a mixed modeling framework that is extendable, so future research can extend either the tangible or the graphical interface.

Acknowledgements

First, I want to express my sincere gratitude to my primary advisor, Associate Professor, Daniel Cardoso Llach for his guidance. He helped me improve my thesis with his professional knowledge of the literature as well as academic writing, and guided me when I got stuck in the research prototype.

I also want to thank my advisor, Assistant Teaching Professor, Eddy Man Kim, for his suggestions on framing my thesis and improving my research prototype.

Besides, I also want to thank everyone in CodeLab for supporting each other during such a special period and also for their advice and hardware which helped me solve a lot of technical problems.

Thank you all for making my experience in CodeLab invaluable.

Table of Contents

Chapter 1. Introduction……..1

Chapter 2. Background……..4

2.1 Literature Review……..……4

2.2 Case Studies……6

2.2.1 Research experiment of Salim……..7

2.2.2 ChainFORM……8

2.2.3 Tangible Landscape……..9

2.2.4 A Concurrent Physical and Digital Modeling Environment……..…10

2.2.5 The Reactable*.....12

2.3 Reflection on Literature and Case Studies.....14

Chapter 3. Hypothesis & Research Question.....16

3.1 Hypothesis.....16

3.2 Research Question……..…16

3.3 Research Goal.....16

Chapter 4. Method.....18

4.1 Design Idea.....18

4.2 System Set-up.....20

4.2.1 System Architecture.....20

4.2.2 System Environment And Functionality.....21

4.3 The Tangible System.....23

4.3.1 Tangible Cubes As Voxels.....23

4.3.2 Technical Solution.....24

4.3.2.1 QR Code.....24

4.3.2.2 Marker.....27

4.3.3 Final Prototype Of The Tangible System.....30

4.4 Communication Between Physical And Digital Worlds.....33

4.5 The Digital System.....34

4.5.1 Halfedge Data Structure.....34

4.5.2 Code Structure Of The Digital System.....39

4.5.3 Final Prototype Of The Digital System.....40

4.6 Modeling Interactions Of The Final Prototype.....41
4.6.1 One Manifold Mesh Example.....42
4.6.2 Failed Examples.....46
Chapter 5. Discussion.....47
5.1 Benefits.....47
5.2 Contribution.....48
5.3 Limitations.....49
5.4 Next Steps…….....50
Chapter 6. Conclusion.....52
References……..……..53
Appendix.....56

Chapter 1. Introduction

Nowadays, designers are provided with various design tools, namely sketch up, Rhino(grasshopper), Maya, Cinema4D, and so on. But considering the diversified data structure and modeling logic inherent to different software to continuously satisfy the needs of designers, it's often time-consuming to learn to use all functions and make a model. Traditional tools, while still very necessary in the industrial field, are not perfect for scenarios where people want to make a prototype very fast and iterate their design frequently, or they are just non-professional users who want to make a digital model without learning that much. Therefore, what is needed in such conditions are: 1) a tool to build models with low fidelity, making it easy to iterate the design; 2) a tool with less learning expense and less time cost and has a short learning curve. Some potential applications of such a tool are:

1) For product/Industrial design: This tool can be used in the prototype design stage. Designers can build a prototype and iterate the designs frequently. It could also act as an agency between professional designers and other non-professional designers (e.g. clients or participants) therefore communication and collaboration between professional designers and non-professional participants are more effective.

2) For art-making: The tool can be used by artists to create the digital model they need. For example, digital artists not professional in 3D modeling can also create the geometries they need for their animation. Furthermore, if the modeling system is included as part of digital art, the audiences can use it to interact with digital art.

3) For education: The tool can be used for educational purposes. It can be used for kids to enhance their perception of complex geometries and train them with basic modeling operations. In this process, their perception of geometries as well as logical

thinking can be practiced. Besides, having several kids working together to make a model provides an opportunity for teamwork.

Therefore, the questions are: 1) Can we make it easy to create and iterate a design? 2) Can we achieve design experience similar to handcraft so the learning curve won't be a problem?

When thinking of the way to interact with a digital model, it's interesting that although the designers often complain about it, the medium that users can use to interact with the digital model is still limited to the mouse and keyboard. According to anthropologist Lucy Suchman's discussion of Heidegger's "ready-to-hand", when using a mouse, one acts through the mouse as an extension of one's hand. The mouse thus fits into a meaningful network of actions, purposes, and functions. It "has a tendency to disappear" (Suchman, 2007). However, it can still condition the kind of output that users produce strongly and thus influence the modeling logic of 3D modeling software.

In fact, some researchers argue that the manipulation of digital models is limited by the use of graphical display constraints such as the screen, keyboard, and mouse (Kim et al., 2005). To address this problem, researchers has explored new interaction patterns to prove that the physical embodiment of GUI contributes to utilizing "human senses and skills people have developed through a lifetime of interaction with the physical world" (Ishii & Ullmer, 1997). They argue that the benefits of these new tools have certain benefits compared with the traditional ones: 1) They provide more immersive scene and more real-time feedback; 2) The user may have more freedom regarding the ways that they can interact with their design work; 3) They can facilitate collaboration in a teamwork scenario.

Could these new patterns offer some inspirations on replacing the traditional mouse interactions, or at least part of them, and improve peoples' design experience is the question

that I would like to explore. The improvements could be: 1) reduced time cost for certain operations; 2) shorter learning curve; 3) less time spent iterating a design. Besides, it would be more valuable to figure out what kind of modeling interaction is better with tangible interfaces, and what is better with graphical user interfaces, then we can leverage the strength of both.

Chapter 2. Background

This chapter catalogs some inspiring researches and case studies that try to address the problem mentioned in the last chapter. Researches are classified according to their position in the spectrum of MR (Mixed Reality), which is proposed by Milgram and Kishino (Milgram & Kishino, 1994). In Milgram and Kishino's view, MR refers to a particular subclass of VR (Virtual Reality) related technologies which involve the coexistence of physical and virtual worlds, and there are various ways that the physical and virtual aspects of MR environments can be realized.

2.1 Literature Review

A review of the literature shows that a lot of researchers in the tangible interaction and mixed reality field started their research by doubting the role of graphical user interfaces (Ishii & Ullmer, 1997; Kim et al., 2005; Fukuda et al., 2016; Al-Qattan et al., 2016). These researchers argue that the manipulation and visualization of digital models are limited by the use of graphical display constraints such as the visual screen, keyboard, and mouse (Kim et al., 2005), which is restrictive. Furthermore, they explore new interaction patterns to prove that the physical embodiment of GUI(Graphical User Interface), as well as the coupling of physical and digital spaces, contribute to utilizing "human senses and skills people have developed through a lifetime of interaction with the physical world" (Ishii & Ullmer, 1997).

Just as Milgram and Kishino defined (Milgram & Kishino, 1994), there're various patterns for MR (Mixed Reality) that connects completely real environments to completely virtual ones, which are measured in two opposite directions, AR (Augmented Reality) and AV

(Augmented Virtuality). The two different directions are discussed by researches as design choices. Generally, in AR, users are interacting with physical objects in the physical world, while in AV, users are interacting with the virtual world, yet both the virtual and physical objects are displayed seamlessly in the respective worlds (Hughes et al., 2005).

In the design field, AV (Augmented Virtuality) is widely explored in the consideration that bringing tangibility to digital information can "bridging the gaps between both cyberspace and the physical environment" and "break free" from "the image of the computer as a monitor, keyboard, and pointer-endowed terminal (Ullmer et al., 1997). In this approach, studies create TUIs (tangible user interfaces) that are coupled with digital interfaces and collaboratively present useful information, where the digital interface is an output of the rendered model(Ullmer et al., 1997; Underkoffler & Ishii, 1999; Al-Qattan et al., 2016). Furthermore, some studies argue that the digital component improves the design process by providing real-time feedback by visualizing in itself or projecting on the tangible component (Petrasova et al., 2014; Salim et al., 2011; Underkoffler & Ishii, 1999).

Different from the augmented virtuality approach, though not many are found to the best of my knowledge, some researches provided another choice (Han, 2017; Fukuda et al., 2016; Abdelmohsen & Do, 2007). The researches point out that aside from AR and AV, there's embodied virtuality as another form of mixed reality, where there is no longer a clear separation between interaction with the physical and virtual world (Salim et al., 2011). Han gives an example of a modeling environment that combines design affordances specific to each media: the physical space supports embodied interaction, while the digital interface offers different views and parametric editing operations as well as the basic analyses (Han, 2017).

Although the latter approach justifies itself to be a system that makes the best of both tangible and virtual interfaces and enables fast iteration in the prototype design phase, the projects in the studies are relatively simple. For example, Abdelmohsen mentioned that their TangiCAD was just capable of generating, selecting and editing 3D objects, and hopes future improvement could have embedding design knowledge, multi-user collaboration, and design flexibility (Abdelmohsen & Do, 2007). Lu Han's study was also restricted to very simple cubic geometries and a few simple parametric manipulations, which means the potential to use both interfaces for some real design task is not explored effectively. Another limitation is that these researches didn't have a discussion about what kind of functions should be assigned to the physical part.

Therefore, my research aims to dive deeper into this approach and explore to what extent it can facilitate the prototype design process with more functions, flexibilities, real-time feedback, and collaboration. And this research will also explore what kind of functions should be migrated from the digital part to the tangible part to improve design experience.

2.2 Case Studies

This section documents some related works with various interaction patterns. In the process of exploring different interaction patterns in tangible modeling, two questions seem to become important. The first question is how to use tangible interfaces. Tangible interfaces can be used as controllers or modeling entities. The second question is where do interactions happen. Do all interactions happen with the TUI (tangible user interface) or the GUI (graphical user interface) or both? These related works are analyzed regarding the 2 questions.

2.2.1 Research experiment of Salim

This is a research experiment done by Salim's research group where a Nintendo Vii remote is used for 3 tasks: select points, draw 3D curves, and do view change for geometry (Salim et al., 2011). This is an attempt to explore how people can interact with the digital world and finish some tasks through physical objects (see figure 1).

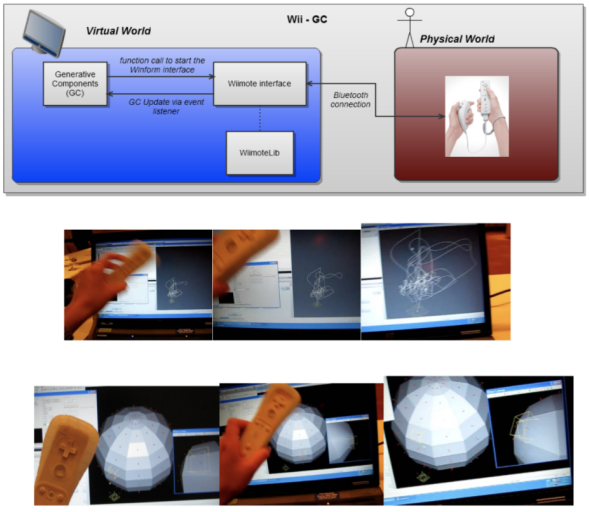


Figure 1. Interacting with digital model (Salim et al., 2011)

In this research, the tangible interface is obviously a controller and all interactions happen with the tangible interface (see figure 2).

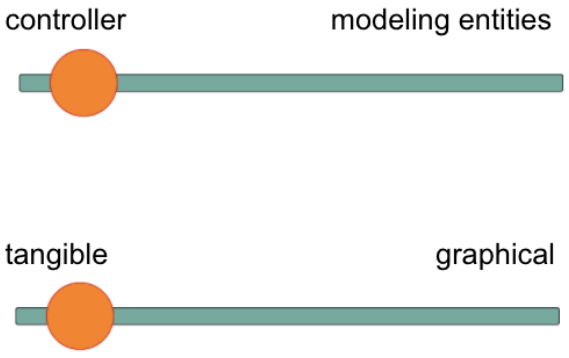


Figure 2. Evaluation of Salim's research experiment

2.2.2 ChainFORM

ChainFORM is a project from MIT tangible media group, where a linear, modular, actuated hardware system is created as a novel type of shape-changing interface. With rich sensing and actuating technologies, the modular system enables users to build various interactive applications. The prototype consists of many identical modules that are connected and each module has rich input and output capabilities like touch/angular detection, visual output, and motor actuation (Nakagaki et al. 2016).

With the modules, researchers were able to introduce novel interactions with shape-changing interfaces. As figure 3 shows, the tangible units can join together to encode a curve that controls the shape of the mesh.

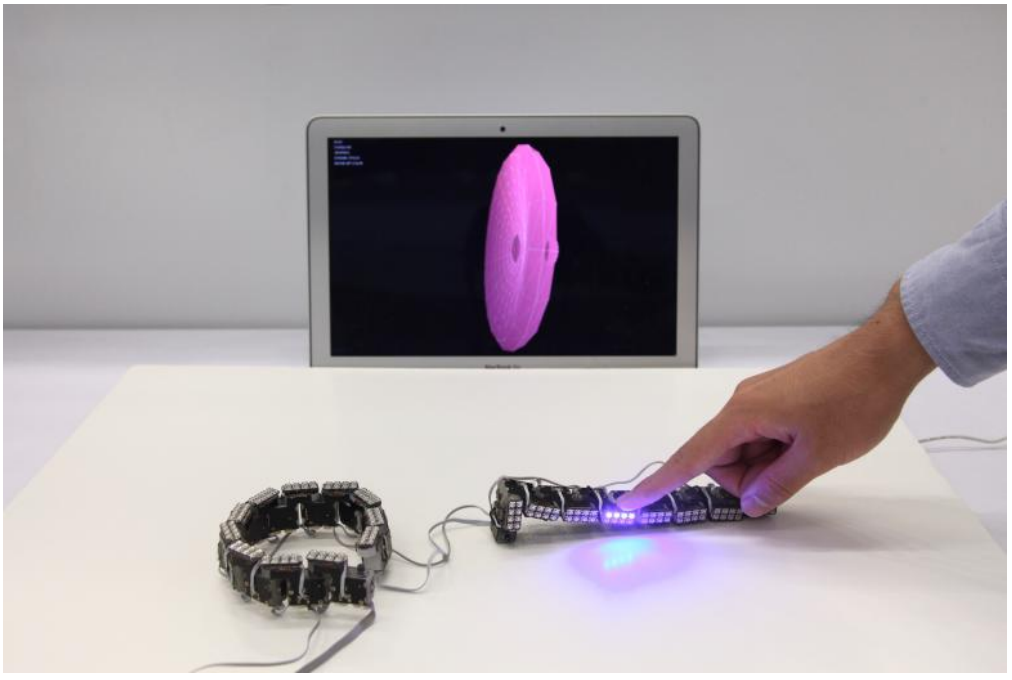


Figure 3. Use the modules to control a mesh. source:https://tangible.media.mit.edu/project/chainform/

The chain formed by modules, as a shape-changing interface, plays the role of a modeling entity. And it also provides some control functions, so it lies between a controller and a modeling entity. Again, all the interaction happens in the tangible interface.

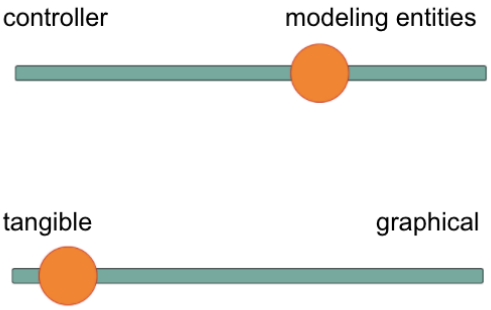


Figure 4. Evaluation of ChainForm

2.2.3 Tangible Landscape

Tangible Landscape is a tangible interface for landscape design and is powered by GRASS GIS and Blender. The method to build this modeling environment is to couple a physical landscape model with a digital model that is rendered in a computer so that users can naturally feel, sculpt, and interact with the landscape. This makes landscape design more intuitive and accessible, especially for non professional users. It also acts as an agency for people like geospatial experts to communicate with designers effectively (Tangible Modeling with Open Source Gis 2018).

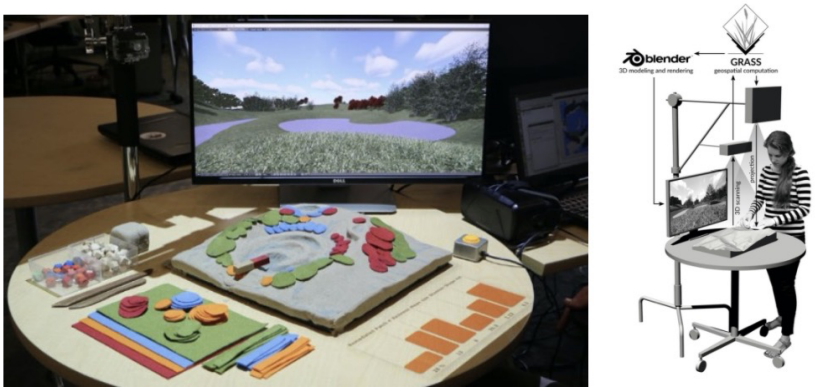


Figure 5. Tangible Landscape, source: https://tangible-landscape.github.io/applications.html

Key features of this project include a real-time feedback cycle of interaction, 3D scanning, point cloud processing, geospatial computation and projection, 3D modeling, and 3D rendering. One example of using tangible landscape is shown in figure 6.

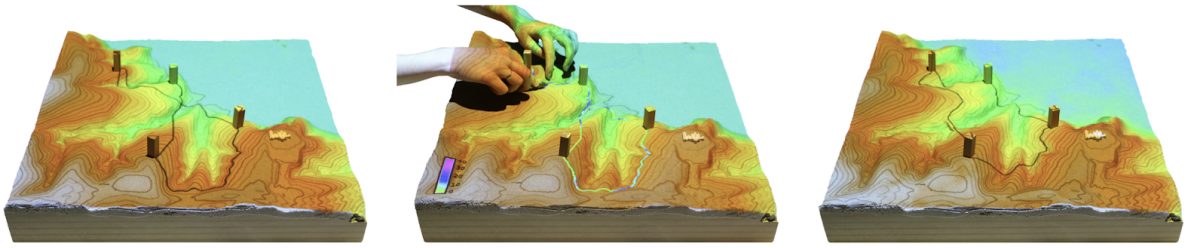


Figure 6. Trail planning using Tangible Landscape, source:

https://tangible-landscape.github.io/applications.html

Here the tangible face itself is the modeling entity, which makes it very easy to understand and use. And again the digital interface is just used to render the design and doesn't provide any control functions.

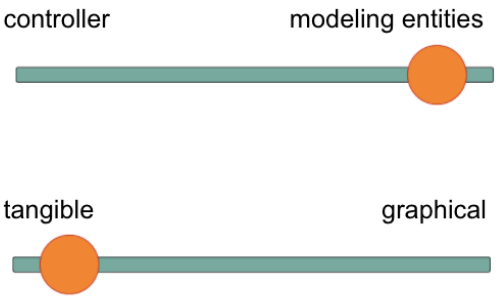


Figure 7. Evaluation of Tangible Landscape

2.2.4 A Concurrent Physical and Digital Modeling Environment

In her master thesis, Lu Han developed a concurrent system that she claims to leverage both the strength of the tangible interface and graphical interface (Han, 2017). She uses a Kinect device to keep track of wood cubes in the physical world and map them to voxels in the

virtual world. Then Lu provides some operations like scale, rotate, and change shape in the graphical interface (see figure 8). Therefore, the tangible interface is used to change configurations of the units and the digital interface is responsible for editing the digital representation. The researcher also explored some other possibilities like representing the abstraction of time through color, saving configurations, and physical objects as global manipulators.

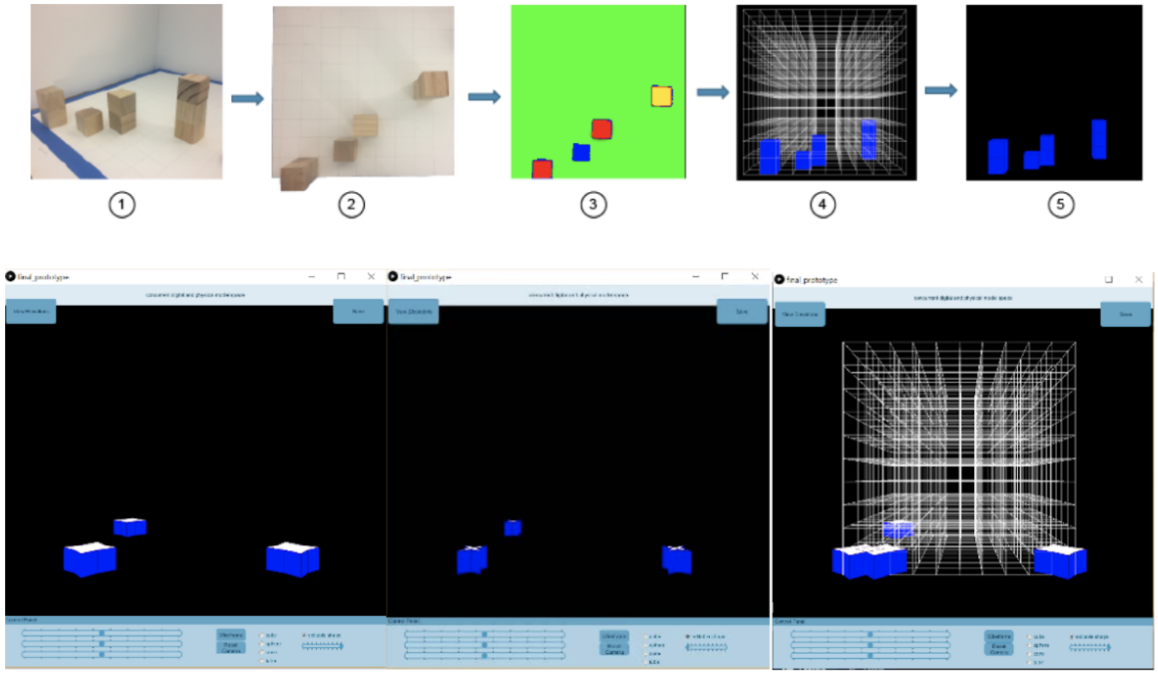


Figure 8. The concurrent system, source: Han, 2017

She used a Kinect to detect the center of each cell in the top view and get the number of cubes through depth data, then the cubes are rebuilt as voxels in the digital space. This solution makes it easier to set up the physical environment, but bring certain limitations:

-All the cubes are the same, which means they can't represent different primitives. Furthermore, since they don't have a unique id, any parametric operation from the graphical interface will be applied to all voxels.

-The system can't recognize any rotation of the cubes. So the cubes can only represent symmetrical geometries.

In this research project, the tangible interface is used as a modeling entity, and this project lies between the tangible and digital worlds. (But Lu also mentioned that since the digital interface is relatively simple, users didn't really pay much attention to it).

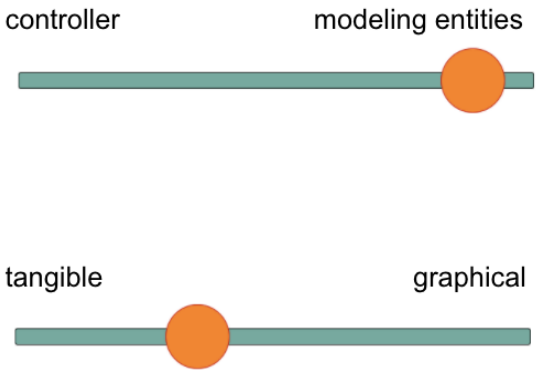


Figure 9. Evaluation of Lu's research project

2.2.5 The Reactable*

Developed at the MTG (Music Technology Group) in Barcelona, the reacTable* is a novel tabletop tangible user interface created as a multi-user electro-acoustic music instrument (Jordà et al. 2007). The interface is based on a round table. A camera is placed beneath and constantly analyzes the condition of the table surface. The nature, position, and orientation of the modules placed by users can be tracked. Users can interact with them by moving them or changing the orientation or their faces (see figure 10). They can also interact with the graphical interface projected around the modules to modify the parameter of the synthesizers represented by the modules (see figure 11).

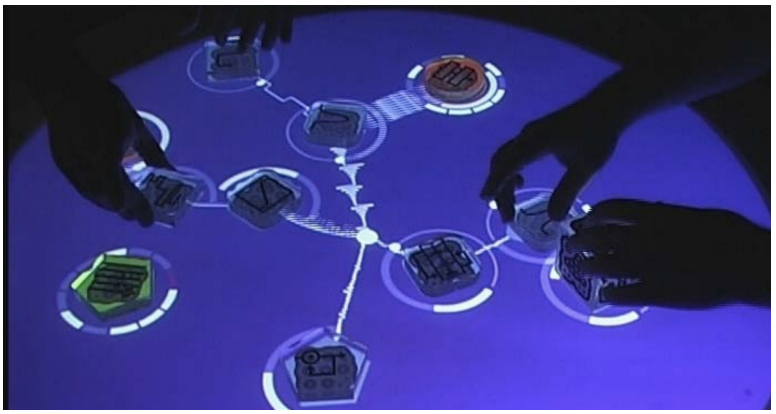


Figure 10. Moving tangible units, source: (Jordà et al. 2007)

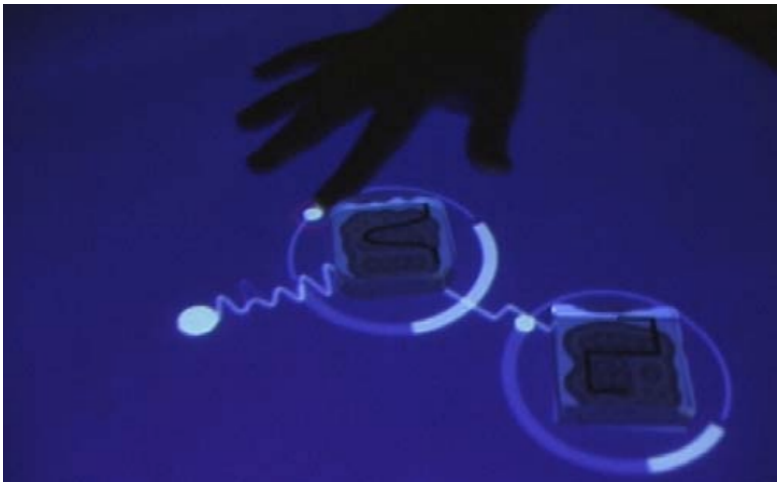


Figure 11.Modifying a parameter with the finger, source: (Jordà et al. 2007)

Although this is not a research that explores modeling interactions, it does show that the tangible interface and the graphical interface can be integrated for some tasks. It also shows that the affordance of the tangible modules is that users can use their body movement to move the cubes, and the affordance of the graphical interface is that users can provide parametric inputs to adjust the instrument. Therefore, the strength of both can be taken advantage of.

2.3 Reflection on Literature and Case Studies

The above researches make strong arguments that design experience can be made more interactive and easier for nonexperts. And I also have 2 observations:

First, if the tangible interface is used more as controllers, the modeling logic is more similar to traditional modeling software. It's more like an improvement of input technologies or a traditional modeling environment. In comparison, if the tangible interface is used as a modeling entity, it's more like handcraft and has the potential to break away from that learning curve caused by modeling logic of traditional software (see figure 12).

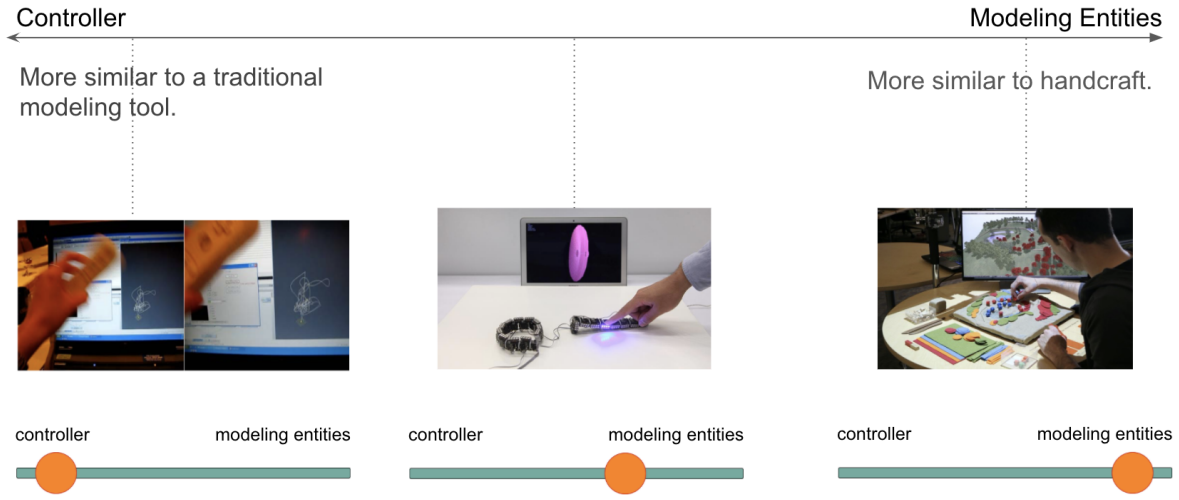


Figure 12. Controller vs modeling entities

However, among research projects that treat the tangible interface as modeling entities, many pieces of research like Tangible Landscape are fully dependent on the tangible interface as input, the digital world is only used to render the output. And when putting it together with Lu's project and Rhino, we can see the more interactions we have in the digital world, the more control we have over the details of the model (see figure 13).

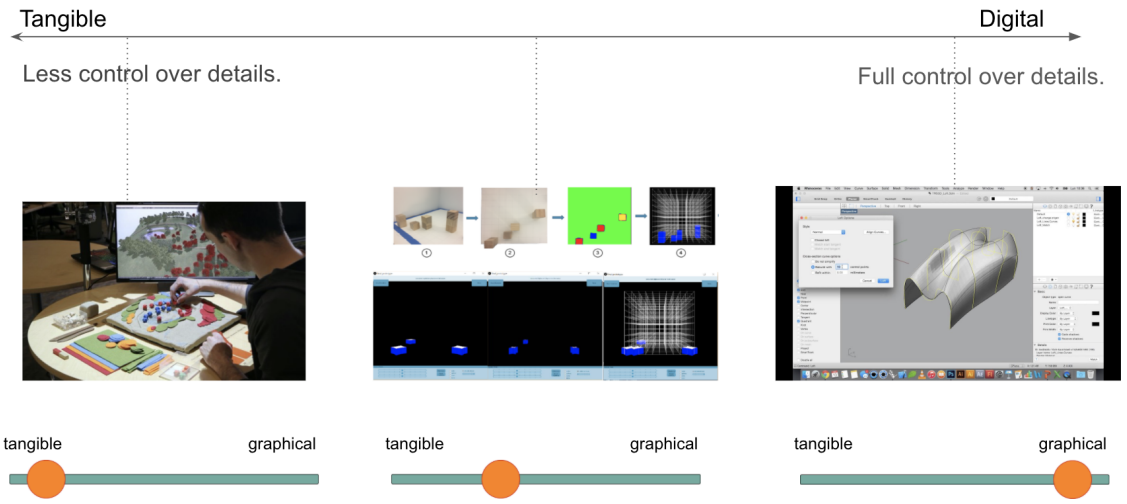


Figure 13. Tangible vs digital entities

Since my thesis aims to make a modeling environment that doesn't require a long learning curve, using the tangible interface as modeling entities would be more reasonable. At the same time, to give users more control over the details of the model, digital interactions are also important. This design choice falls in the embodied virtuality approach. And according to Lu Han, this approach can take advantage of both worlds (Han, 2017):

1) The physical world provides situated interactions easy to understand. Users can create a model without concerns about details, and there can be collaboration.
2) The digital world promises accuracy, realistic renderings, and encapsulation of data in data structures.

Therefore, I want to follow the embodied virtuality approach for my thesis and contribute by exploring a way to combine tangible embodiment of geometries with a specific digital modeling logic to open new ways for professional and non-professional users to create geometric models.

Chapter 3. Hypothesis & Research Question

3.1 Hypothesis

If tangible objects can serve as the physical representation of geometries, using the half-edge data structure (which is designed for effective local edit operations) to manage digital models will enable a shared way of encoding geometries in both the physical and digital space. This will allow digital models to respond effectively to both the tangible and graphical interfaces, which helps leverage both tangible and digital interfaces regarding the affordances of both.

3.2 Research Question

How tangible and digital interactions can be integrated in a mixed modeling environment to leverage the affordances of both and facilitate the design process with efficient prototype iteration?

The target scenario is the prototype design stage in art-making or industrial design. The requirements are:

1)There's no need for high fidelity, designers may iterate the design frequently.

2)The model should be easy to modify and less time-consuming.

3.3 Research Goal

To test my hopothesis, I have 2 goals for my project:

1) To prototype a mixed modeling environment using a half-edge meshes as a shared way of encoding geometries in both the physical and digital worlds.

2) To experiment with the combination of tangible interaction (assembling physical cubes) and digital modeling operations (local and global operations for the half-edge data structure) and make it possible to switch between the two worlds effectively.

In this way, the tangible interface is not just an input device but a physical existence of the model, and users can assemble the geometries in the physical world while editing details in the digital world.

Chapter 4. Method

This chapter documents my prototyping process, including design idea, system set-up, selection of hardware and software environment, and design choices. I experimented with different solutions for hardware and software and provided some advice on building a mixed modeling environment. The chapter starts with an overall system setup and explores solutions for each part.

4.1 Design Idea

The design of the tangible interface is inspired by the Lego product (see figure 14), which is legible and playful. Users can assemble the pieces to make creative physical models, which is practiced through their life experience and doesn't need an extra professional training.



Figure 14. Lego Mindstorms product line. source: https://education.lego.com/en-us

A review in the literature of the tangible interaction field also shows that some tangible interfaces also use a similar approach. For example, as figure 15 shows, the Materiable project in MIT tangible media group uses a voxel approach to imitate the surface of objects or some forces to interact with human hands (Nakagaki et al. 2016).

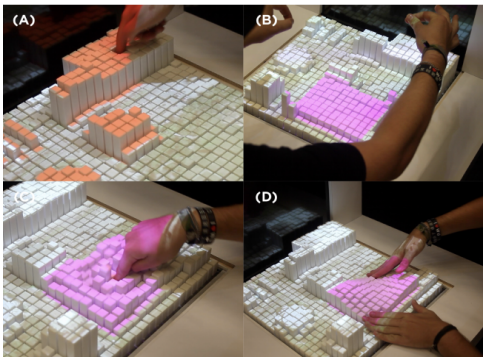


Figure 15. Materiable: Rendering Dynamic Material Properties in Response to Direct Physical Touch with Shape Changing Interfaces, source: Nakagaki et al. 2016

Therefore, assembling modules to create some voxel representation of a model can be a way to use the tangible interface as modeling entities, which is easy to understand. To add a little more complexity, I want each voxel in the tangible interface to be able to encode different geometries, and the geometries can be easily translated into some data structure in the digital world. My exploration in the computer graphics field shows the half-edge data structure could be a good solution in that it's a mesh data structure for manifold polygon meshes and supports various modeling operations, the logical of which is close to sculpting an art piece (see figure 16). More features of this data structure will be introduced in section 4.5.

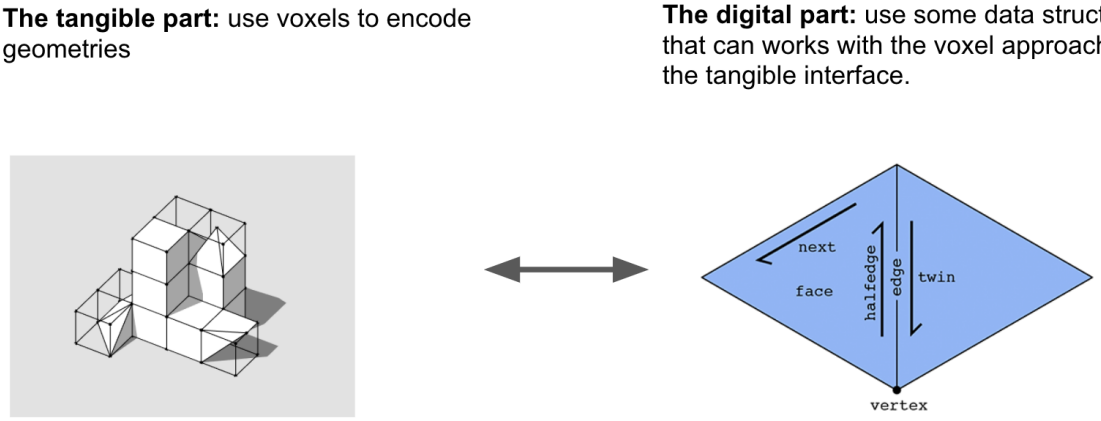


Figure 16. Design idea

As is shown in figure 17, users can build a low-resolution voxel model in the physical space, add more details and operations in the digital world. Furthermore, users can also improve the quality of the mesh, for example, smoothing it. Users can switch between the physical and virtual worlds to change the design and any change in either of them will change the model in real-time.

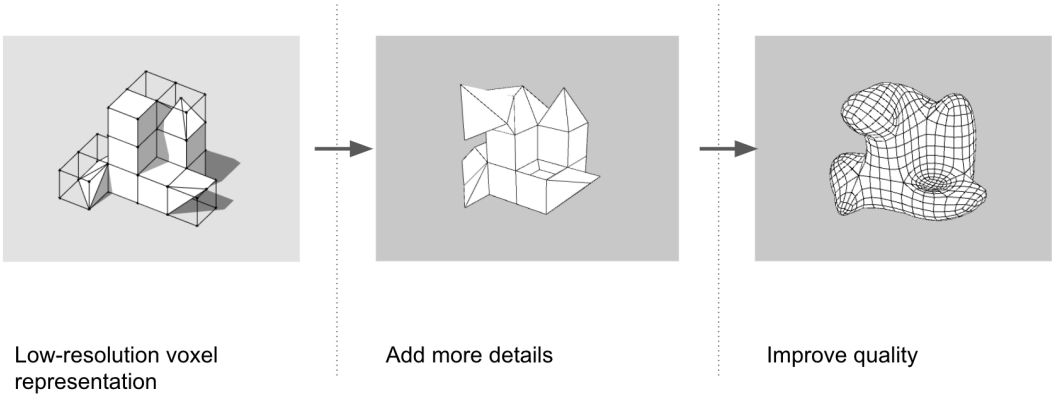


Figure 17. Representation of geometries in different worlds

4.2 System Set-up

4.2.1 System Architecture

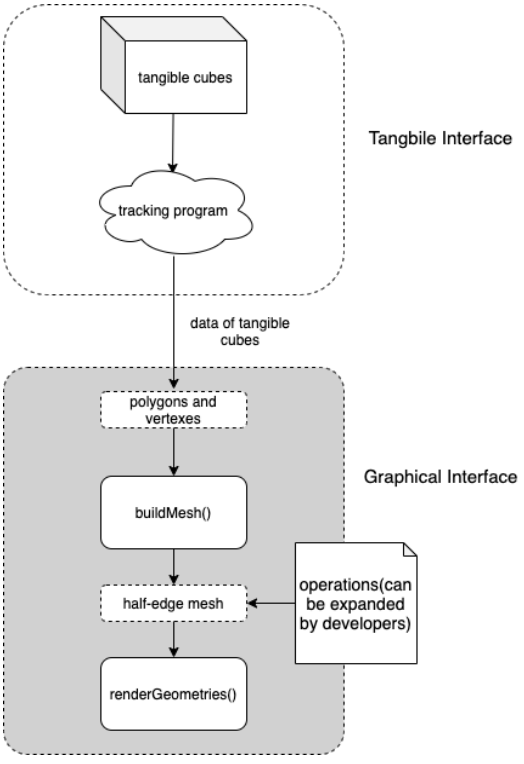


Figure 18. System architecture

Figure 18 shows the architecture of the mixed modeling system:

The Tangible System

The tangible system consists of tangible cubes and the program to track the cubes and rebuild them as voxels. The data is sent to the digital part for visualization and digital interaction.

The Digital System

The digital system is a computer program that launches a graphical user interface that has the following functions:

-interpreting the voxel representation of geometries into polygons and vertices

-rebuilding the model as a half-edge mesh

-rendering the model

-editing the model

Generally, the part that is responsible for rebuilding and rendering half-edge meshes is a black box since it doesn't necessarily need to be understood by others. And the collection of operations enabled for the digital model is a white box and can be expanded by developers as long as it works for this data structure.

4.2.2 System Environment And Functionality

The virtual environment and technologies I use to develop the system are shown in Figure 19:

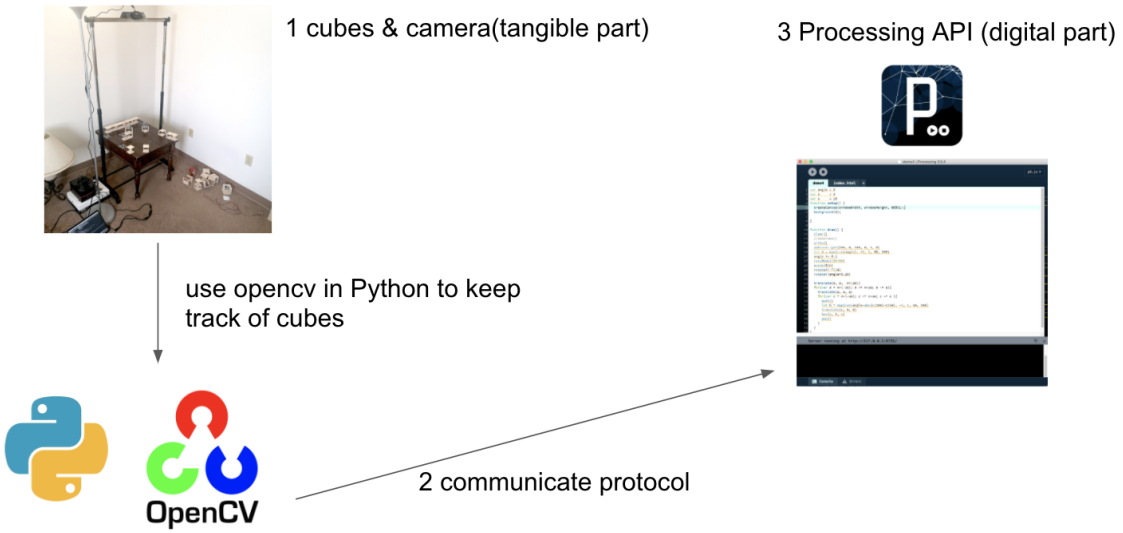


Figure 19. System environment

The tangible system contains the tangible cubes and a camera in the physical space as well as the python program using OpenCV to track the cubes and rebuild all cubes as data. Besides, the data should be easy to be serialized to strings which can be easily deserialized by the digital system to get information about all the voxels.

The communication between the tangible and digital system is enabled by by using web socket and self-defined format for the data being transmitted.

The digital system is developed in Processing and relies on the P3D library to visualize the geometries. It launches a graphical user interface that has the following functions:

1) Provide encapsulated half-edge data structure with basic methods for building, editing and rendering the mesh in the Processing environment.
2) Keep listening to the tangible system and change the digital model when informed of any change in the physical space.

4.3 The Tangible System

For this part, the most important choice is about the computer vision library to be used to realized the design idea.

To be able to rebuild all the cubes and manage their movement, I need to recognize the cubes and get their 3d location. The latter is feasible if a depth camera is used. Note that although there are libraries that computes the 3d position of a marker, it doesn't work well for this project since there are multiple cubes and some of them can obscure views. Therefore, my solution is combining rgb and depth cameras that both provides a top view of the physical space.

After looking at technologies in the computer vision field, two choices, QR code and Marker, emerge to be possible solutions. So I experimented with the two approaches and chose to use Marker based on the experiments.

4.3.1 Tangible Cubes As Voxels

The 3d position, ID and orientation information for each cube is stored, making it possible to encode different geometries in the voxels:

Unique id: encode different geometries

Each module can have a unique id so cubes can encode different geometries (see figure 20). Since the digital representation is half-edge data structure, both planar and non-planar goemetries are supported.

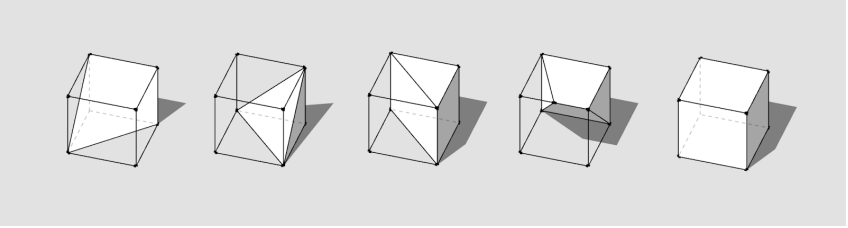


Figure 20. Different geometries encoded in voxels

Orientation information: more freedom in interaction

The system can recognize the rotation of cubes so users have more freedom to play with the tangible part just like toy bricks .

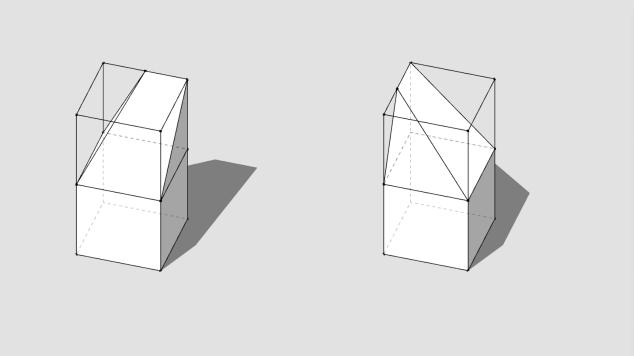


Figure 21. Ability to detect rotation of the voxels

Through the design of the voxel representation, the system enables users to make a low-resolution model in the physical world, which will be rebuilt as a non-planar mesh, and the affordance of the digital system enables more local edit operations on the mesh as well as some global mesh operations like smoothing or improving the quality of the geometries. This system, therefore, has the potential to break away from the low-resolution voxel representation and planar geometries and offer more possibilities for users who use it to create models as well as developers that want to add more features to it.

4.3.2 Technical Solution

4.3.2.1 QR Code

A QR Code is a two-dimensional pattern for encoding data, which is popular in the industrial world for all-direction high-speed reading, resistant to distortion, and data restoration functionality(Soon 2008). The encoded data is therefore extracted from the image patterns. An example of QR code is shown in figure 22.



Figure 22. QR code, source: https://www.learnopencv.com/barcode-and-qr-code-scanner-using-zbar-and-opencv/

ZBar

ZBar is one of the most popular cross-platform QR code detection tools and is integrated in OpenCV, a widely used computer vision library. A QR code object returned by ZBar has three fields("Barcode and QR Code Scanner Using ZBar and OpenCV | Learn OpenCV" n.d.):

-Type: For QR codes, the returned type is QR-Code.

-Data: The data encoded in the black and white squares inside the QR code is returned. Usually, the data is a string, but other types like numeric types are also supported.

-Location: The location information is returned as an array of points that locate the 4 corners of the QR code.

An example of the information in QR codes being decoded is shown in figure 23.

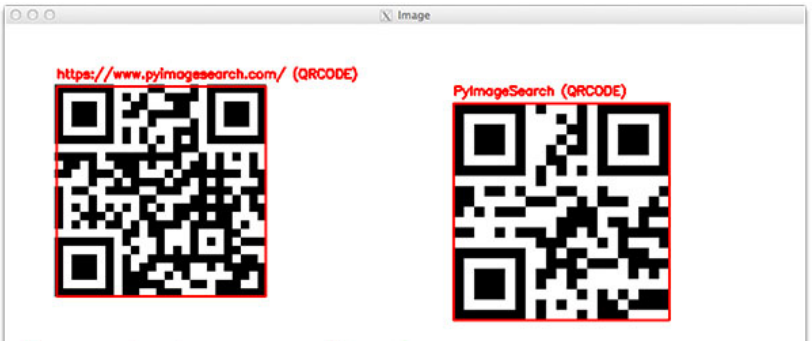


Figure 23. Detection, source:

https://www.learnopencv.com/barcode-and-qr-code-scanner-using-zbar-and-opencv/

Experiment with QR codes

I tested the QR code detection algorithm in ZBar library and it turns out that if I want to use a camera to detect QR codes from 1-meter away, the camera must have a resolution of more than 1960 X 1280, and the QR code should be at least larger than 75mm X 75mm to get accurate results (see figure 24).

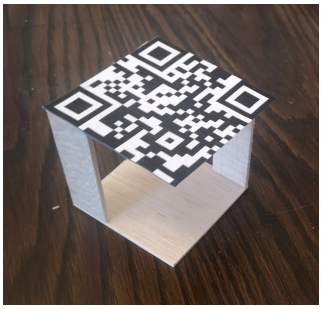


Figure 24. A cube with QR code

This means that I have to use a high-resolution WebCam to get a 2d position of the QR codes. At the same time, I also need a Kinect to detect the depth of the QR code since I need to know the 3d position of this unit.

Another problem found in the experiment is that the python library doesn't act as documented. One approach to get the orientation of a QR code is to compute the rotation matrix by looking at the position of corners. The document says that the returned corners will always be in counter-clockwise order and the original top-left corner will always be the first one no matter how the QR code is rotated. However, in my experiment, the python library acts randomly and this order doesn't exist.

Advantages and Disadvantages of QR code

QR code is very powerful in that I can encode a lot of information, which could add a lot of complexity to the tangible system considering what kind of geometries the voxels can encode.

The limitation is that using QR code will require using a high-resolution WebCam and a Kinect together, which lays more burden on the computer vision algorithm and requires synchronizing 3d coordinates between the two cameras.

4.3.2.2 Marker

For markers, a very successful and mature library is ArUco. An ArUco marker is a synthetic square marker composed of a black border and a binary matrix that only encodes its id (Garrido-Jurado et al. 2014). The black border facilitates detection speed when the marker is placed with other things around. The binary codification not only encodes id but also allows for error detection which makes detection results more robust. The marker size is defined by the size of the internal matrix. For instance, a marker size of 5x5 is composed of 25 bits (as figure 25 shows).

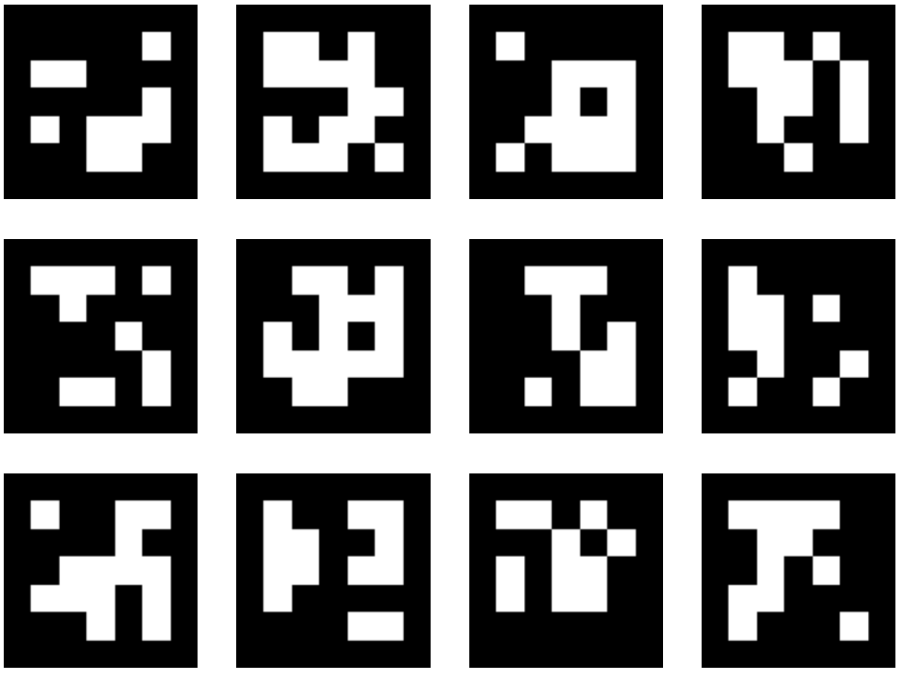


Figure 25. Marker examples

The aruco module in Python provides the Dictionary class to represent a collection of markers. Users can pick a dictionary according to the size of the matrix they expect as well as how many distinct IDs they will use.

In general, the detection of markers with larger sizes is more complex because more bits need to be extracted from the input image. A recommended idea is that users pick dictionaries with smaller size and lower number of markers.

For instance, if one needs 30 distinct markers in the application, it is better to use a dictionary of 50 markers rather than a dictionary of 500 markers. The former, with a smaller size, will have a higher inter-marker distance and will be more robust to errors.

Given an image containing ArUco markers, the detection algorithm can return a list of detected markers. Each detected marker consists of 2 fields:

- The position of its four corners in the image (in their original order, starting from top-left corner).
- The marker's ID.

Figure 26 shows an example of marker detection:

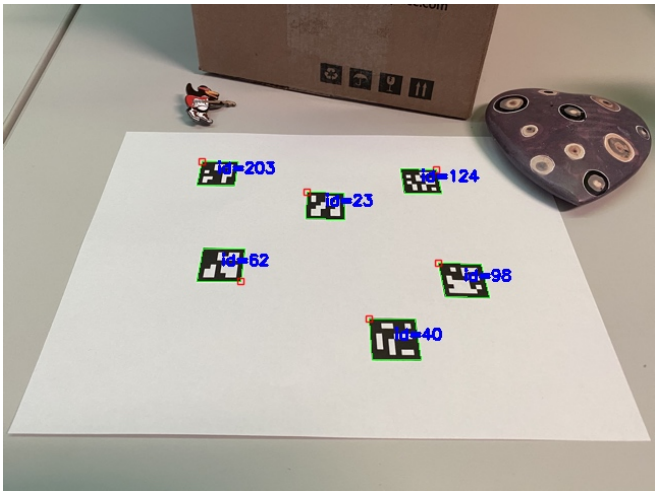


Figure 26. Marker detection, source: https://docs.opencv.org/trunk/d5/dae/tutorial_aruco_detection.html

Experiment with ArUco Markers

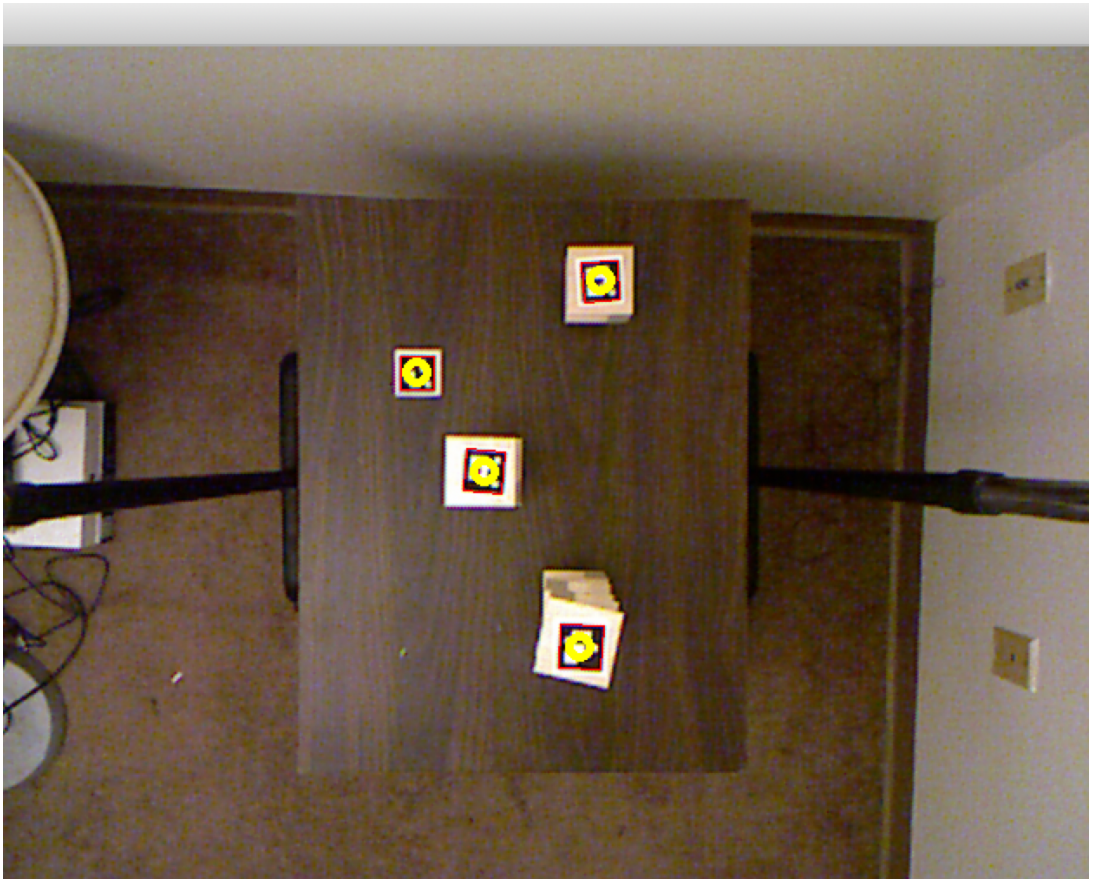


Figure 27. Experiment with ArUco markers and cubes of different sizes

Experiment with ArUco library shows that ArUco Markers are easy to detect, even the resolution of 640 X 480, which is the RGB image from a Kinect, is enough for the detection task. This brings a strong advantage: it's possible to use only one Kinect for marker detection and depth sensing. And we can have a marker as small as 5.5 X 5.5cm for the detection task. In addition, the depth data extracted from Kinect is very accurate at the millimeter scale, therefore depth data won't be a problem.

In addition, the corners detected are returned in a specific order as the official document promises, which will make it convenient to calculate the orientation of markers.

Advantages and Disadvantages

The advantage of using markers is that users don't need to synchronize the 3d coordinates between a WebCam and a Kinect, which will make it easier to build a tangible system.

The disadvantage is that markers are relatively simple. In fact, they can only encode maker ids, which is apparently more simple than QR codes. So the only way to use it is to map ids to hardcoded geometries.

4.3.3 Final Prototype Of The Tangible System

After comparing different solutions, a design choice can be made. Since using QR codes will bring many workload burdens considering image size, synchronizing coordinates, and a much larger size for voxels, ArUco marker is a better choice. And hard coding the geometries for each id is tolerable.

I chose ArUco markers and a Kinect as the solution for the tangible system and illustrates my design in this part.

Hardware And Physical Objects

The tangible system consists of a Kinect on top of the table and markers that define the corners of the workspace (see figure 28). Figure 29 shows that the 3 markers define the corners of the tangible space, and cubes in the space can be recognized.

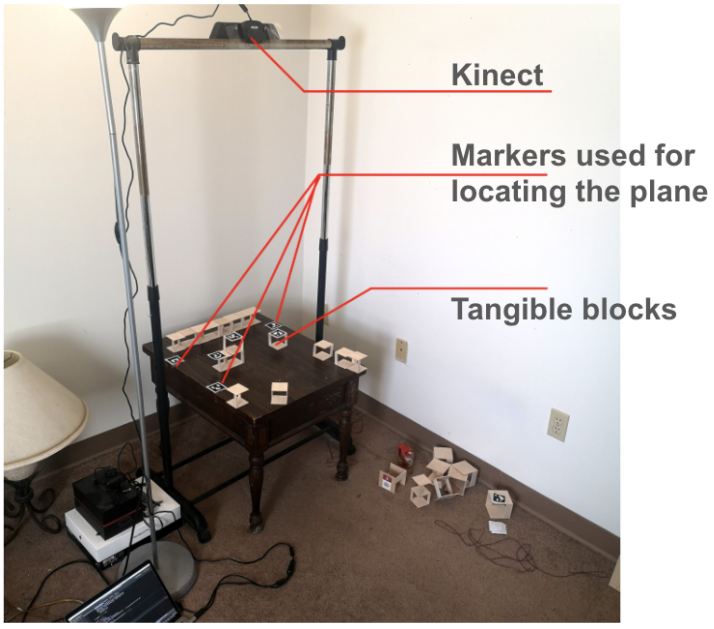


Figure 28. Tangible system

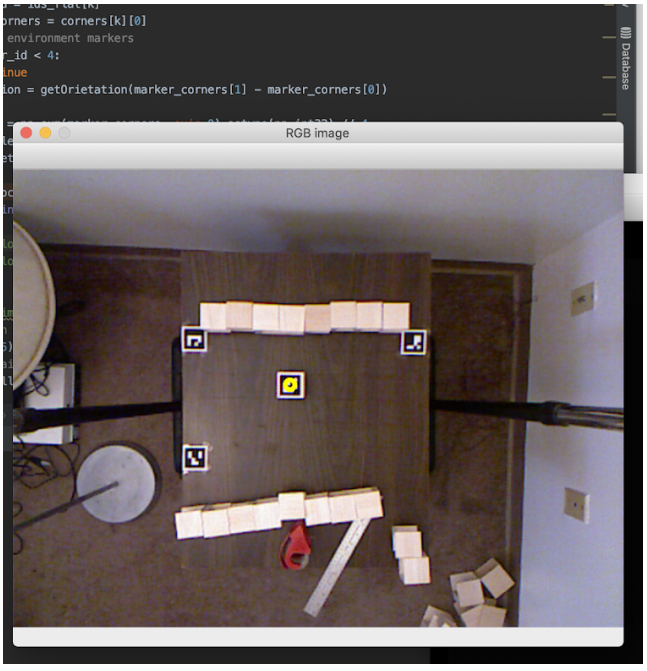


Figure 29. Top view from Kinect

Examples of tangible cubes as well as the digital geometries they encode are shown in figure 30:

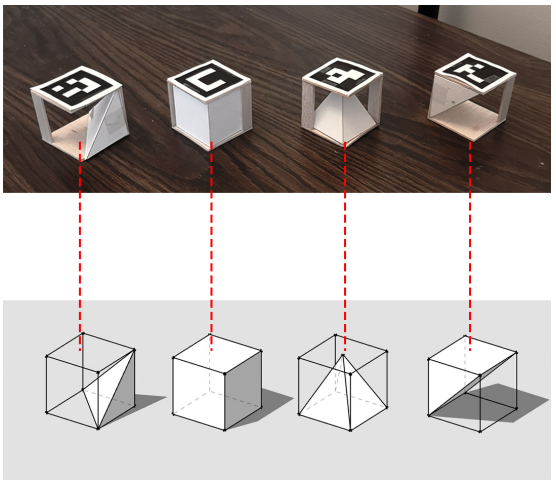


Figure 30. Tangible cubes and the geometries encoded

The size of the physical space available is a 8 x 4 x 4 grid (see figure 31), and each cell can hold one tangible cube (5.5cm X 5.5cm X 5.5cm). The physical space is limited by Kinect since perspective distortion is a big problem for cubes that are far away from the center.

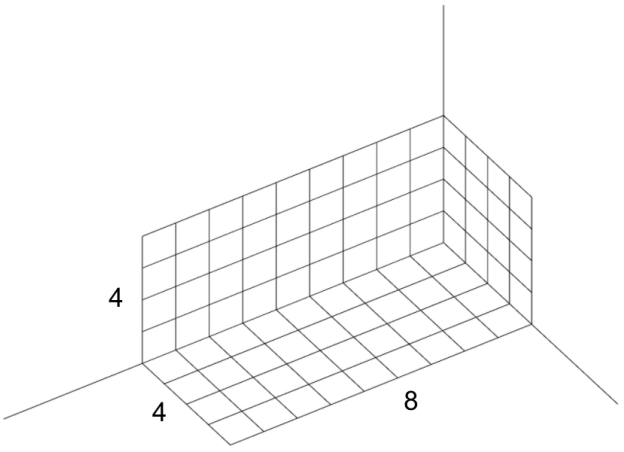


Figure 31. Physical space

Detection Algorithm

After the physical components are prepared, I implemented the tracking algorithm to keep track of all cubes with markers on top of them. The algorithm first initializes the workspace by detecting the 3 corners of the workspace (top-left, top-right, and bottom-left) and build the

grid according to it. Then all the cubes placed in the grid can be tracked and their position in the grid will also be computed and stored. The algorithm keeps the location and orientation information of all markers even if they are occluded by others or the users' hands. A demo of the program is shown in figure 32. In this demo cubes with different IDs are visualized as colored squares in different layers. It is used to debug the algorithm and promise that no matter how I move the cubes or replace them with others, the data for the voxel representation is always correct.

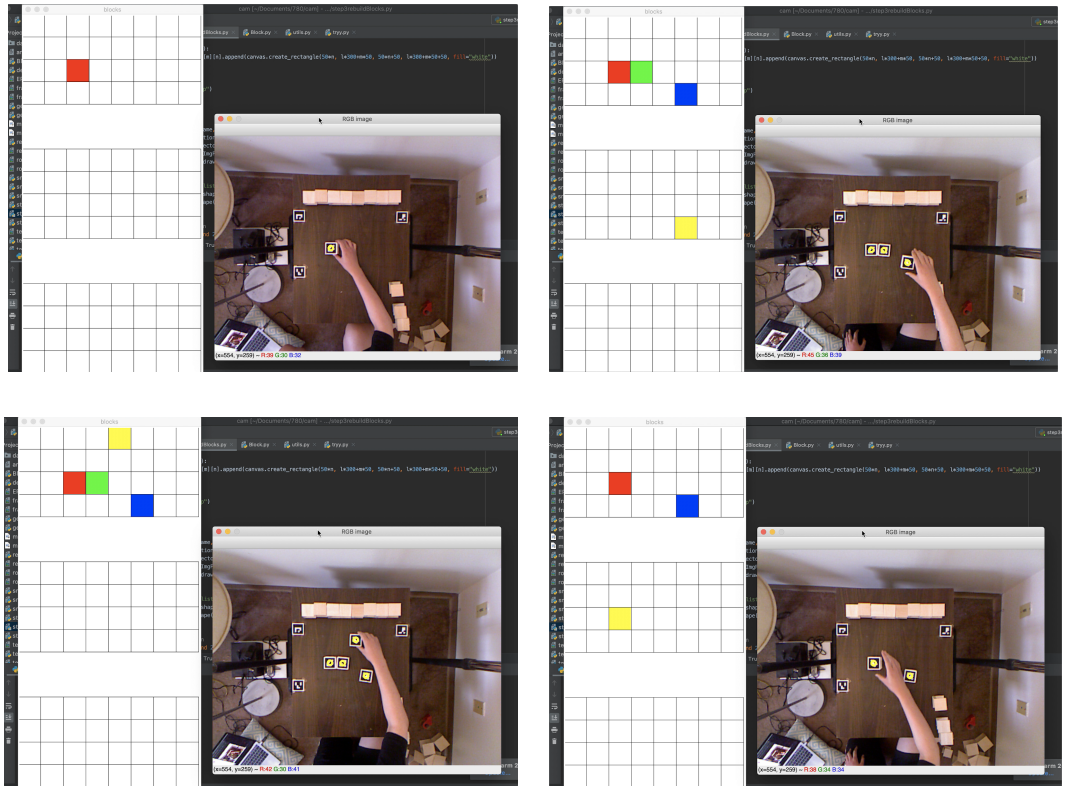


Figure 32. Test of the tracking algorithm

4.4 Communication Between Physical And Digital Worlds

Communication between the tangible part and the digital part is supported by using WebSocket in both Python and Processing. A port in the localhost is used by the digital part

as a server, and the tracking algorithm performs as a client that constantly informs the server of any change in the tangible system.

4.5 The Digital System

This part introduces halfedge data structure and documents my design of the digital system.

4.5.1 Halfedge Data Structure

Introduction to halfedge data structure

Halfedge data structure is a manifold mesh data structure designed to make local operations lightweight.

For example, a common way of storing a polygon mesh is using a list of vertices and another list of faces having pointers to all of their vertices. Whenever some local operations are needed, like collapsing an edge into a vertex, it's required to discover adjacency relationships between faces and vertices, which will result in a search through the entire list of all elements. Likewise, many other adjacency queries(e.g. face-edge, face-vertex, edge-vertex, face-face relationships) on a polygon mesh have the same problem. Therefore, the half-edge data structure is designed to implement these types of adjacency queries efficiently to save computation overhead (McGuire 2000).

The half-edge data structure is shown in figure 33. A half-edge is a half of an edge that has a direction and is constructed by splitting an edge into halves to form a pair of half-edges in opposite directions. All the half-edges in one face are in the counterclockwise direction. The

key idea is that two half-edges act as “glue” between mesh elements. All the edge, face, and vertex instances only store one half-edge that is related to itself:

- Face: stores one of the half-edges inside it.

- Edge: stores one of the 2 half-edges belonging to this edge.

- Vertex: stores one of all half-edges starting from this vertex and going outward.

- Halfedge: stores the face, edge and vertex this half-edge belongs to. In addition, it stores its twin on the other side of its edge and its next half-edge in the same face.

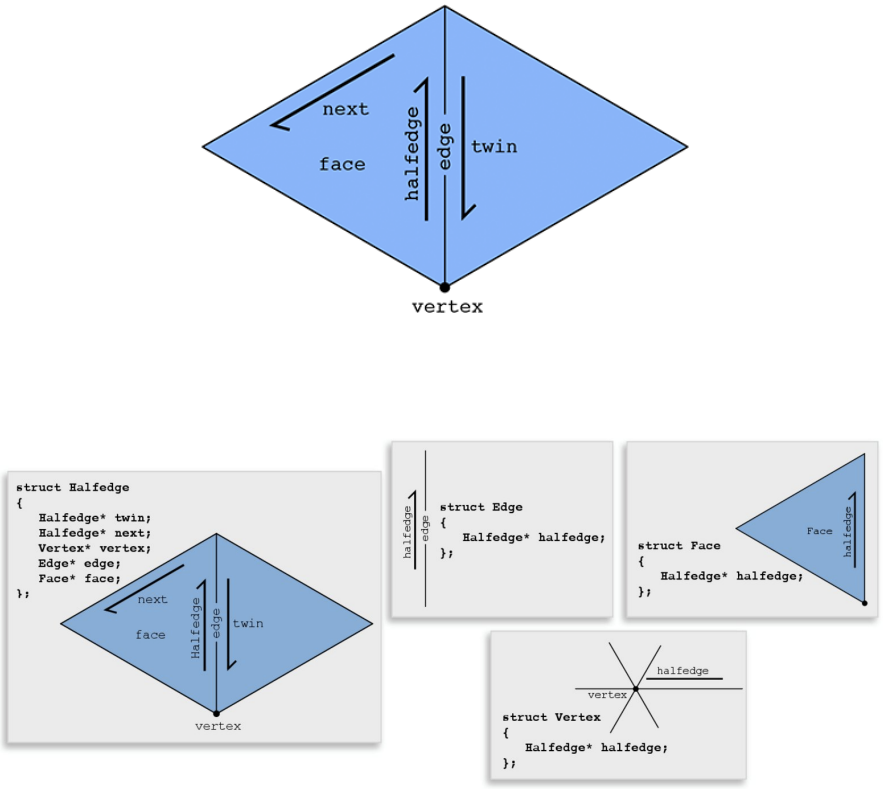


Figure 33. Halfedge data structure, source: https://github.com/cmu462/Scotty3D/wiki/Halfedge-Mesh

One important point is that half-edge meshes must be manifold. The reason is that processing mesh elements in this data structure requires some “common sense”. For example, one half-edge -> twin -> twin = the half-edge itself, which is not promised in non-manifold meshes (see figure 34).

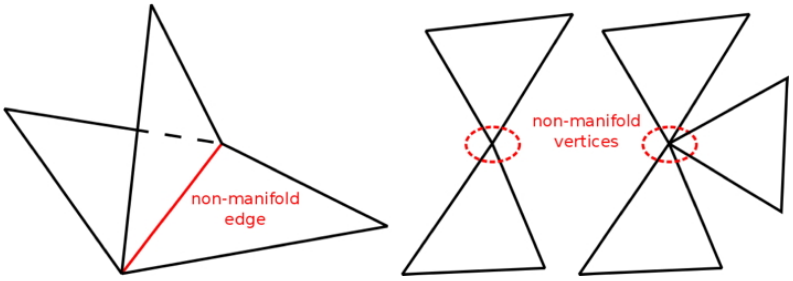


Figure 34. Examples of non-manifold meshes, source: https://www.shapeways.com/blog/archives/29453-tutorial-tuesday-5-quick-fixes-with-meshlab.html

A brief example of manifold mesh vs non-manifold mesh encoded by voxels can be seen in figure 35:



Figure 35. Example of correct and incorrect behavior in the tangible system

Editing Halfedge Meshes

Editing half-edge meshes is computationally fast because all the relationships between faces, edges, and vertices are glued by half-edges. Basically, anything can be done by traversing elements, adding or deleting elements, and reassigning pointers. At the same time, it's important to preserve manifoldness otherwise subsequent operations could fail.

The operations that can be applied to half-edge meshes can be divided into 2 parts:

1) Local operations: move, scale, rotate, EdgeFlip, EdgeSplit, VertexBevel, EdgeBevel, FaceBevel, EraseVertex, EraseEdge, EdgeCollapse, and FaceCollapse. These local operations

prove to be very efficient modeling techniques to allow users to modify the mesh freely without experiencing the learning curve of traditional design software (see figure 36 and 37). These operations enable users to sculpt the mesh extracted from the tangible system.

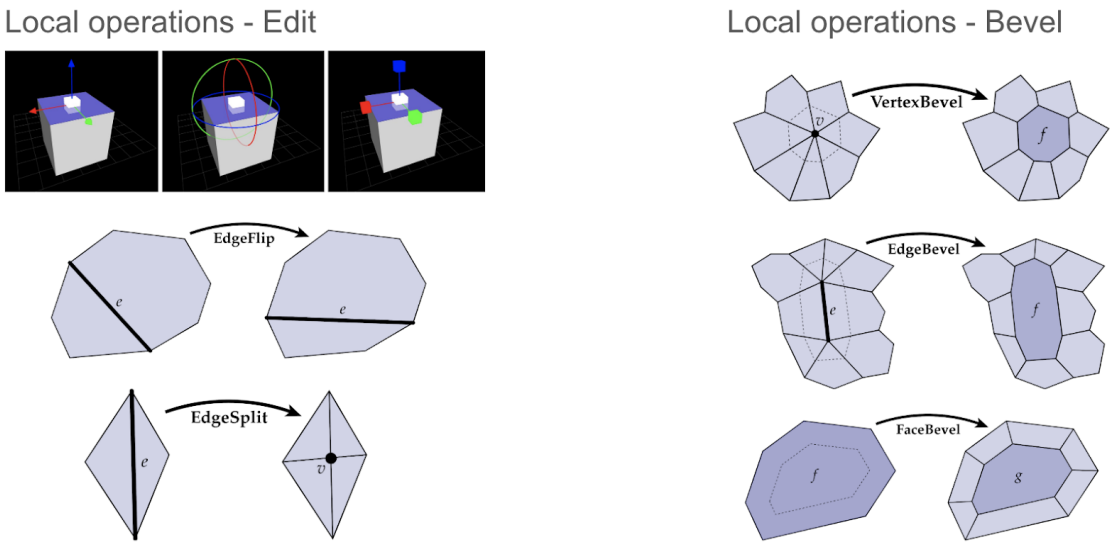


Figure 36. Local operations set 1, source: https://github.com/cmu462/Scotty3D/wiki/User-Guide-(MeshEdit)

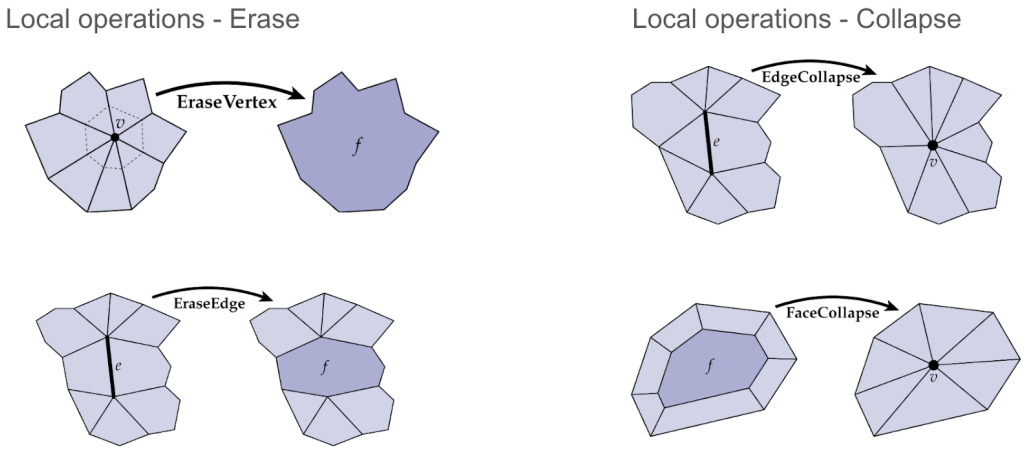


Figure 37. Local operations set 2, source: https://github.com/cmu462/Scotty3D/wiki/User-Guide-(MeshEdit)

2) Global operations: operations that are applied to the entire mesh. Examples are isotropic remeshing, simplification, Catmull-Clark subdivision, loop subdivision, etc. Global

operations are usually designed to subdivide(smooth)/simplify meshes or improve the quality of meshes (see figure 38).

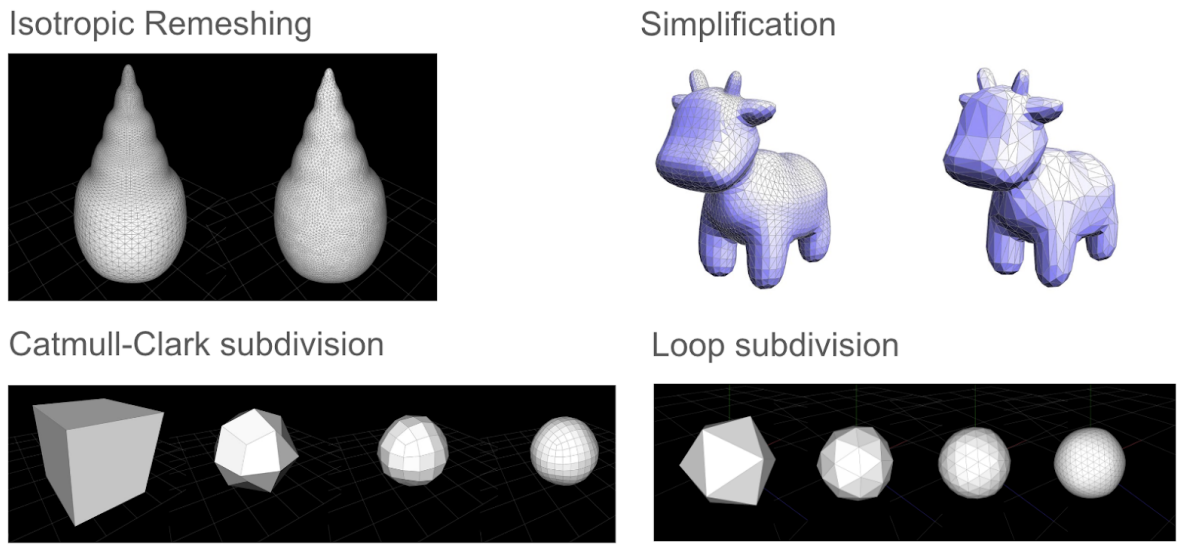


Figure 38. Global operations, source: https://github.com/cmu462/Scotty3D/wiki/User-Guide-(MeshEdit)

Here's an example using local and global operations for 3d modeling (see figure 39). Starting from a cube, users can create a bug in a few steps, and each step is easy to understand. Users can also use subdivision functions to generate a smooth shape out of the control cage.

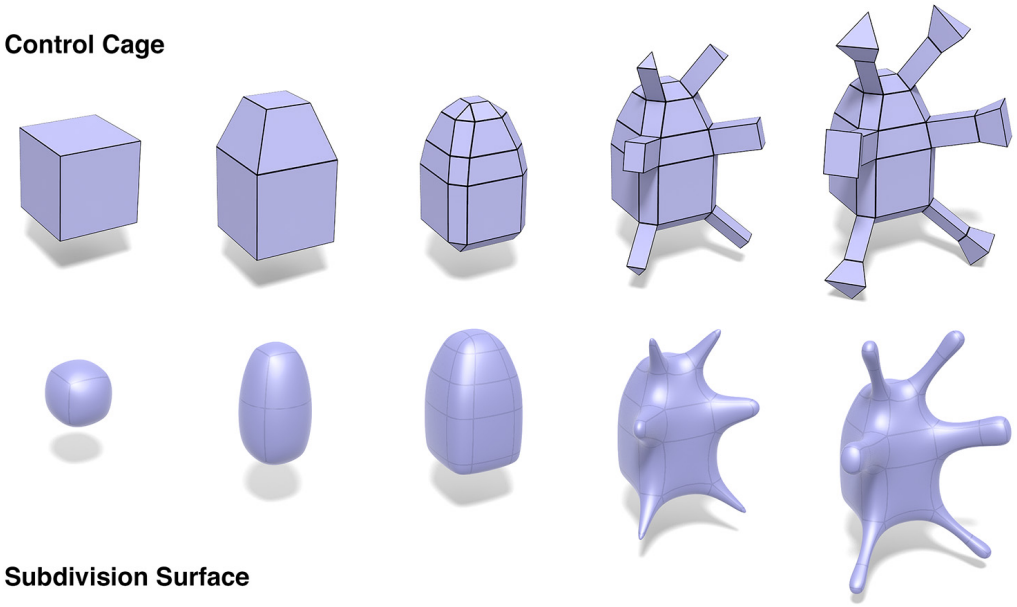


Figure 39. An example of modeling with half-edge data structure, source: http://15462.courses.cs.cmu.edu/fall2017/article/8

In conclusion, half-edge data structure provides a way to encode planar and non-planar polygon meshes and is friendly to various local and global operations which enable users to manipulate the digital model as they wish. These characteristics could enable the translation of tangible cubes in the digital environment and further modifications.

4.5.2 Code Structure Of The Digital System

As is already shown in system architecture, the code of the digital system consists of:

- buildMesh(): accepts a list of polygons (marked by the index of all the vertices in this polygon, in counter-clockwise order) and a list of all vertices and builds a half-edge mesh.

-renderGeometries(): process all the vertices of all faces and use P3D library to render the mesh.

-gui: a graphical user interface that supports digital interaction.

-operations: local and global operations.

Here's an example of the renderGeometries() function, which is very simple as you can see in figure 40. However, the algorithm to build the Mesh from polygons is very complicated, therefore others can ignore its implementation details and just view it as a built-in function. However, the gui and operations can be extended by others to add more digital interactions.



Figure 40. Code for rendering half-edge meshes

4.5.3 Final Prototype Of The Digital System

The graphical user interface is created by using G4P library in Processing. It has two components:

-The main window provides a perspective view of the model.

-The panel on the left provides supported operations. For global operations, there's no need to do anything else since it's applied to the entire model. Some local operations only need the user to select one mesh element (e.g. EraseVertex, EraseEdge, EdgeCollapse, and etc.), while others will also require the user to provide numeric inputs using sliders (e.g. EdgeBevel, FaceBevel, and etc.).

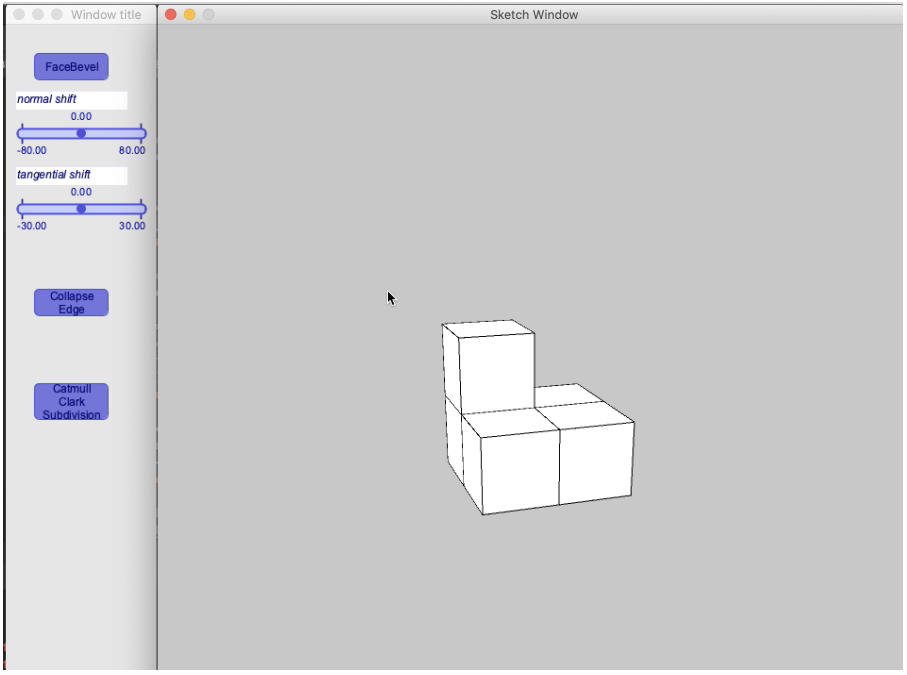


Figure 41. Graphical user interface

In my prototype, I picked 3 operations for the digital system (see figure 42):

-FaceBevel: need to select one face and use sliders to provide the magnitude of normal shift and tangential shift.

-EdgeCollapse: need to select the edge to be collapsed.

-Catmull-Clark subdivision: subdivide the entire mesh to make it smooth.

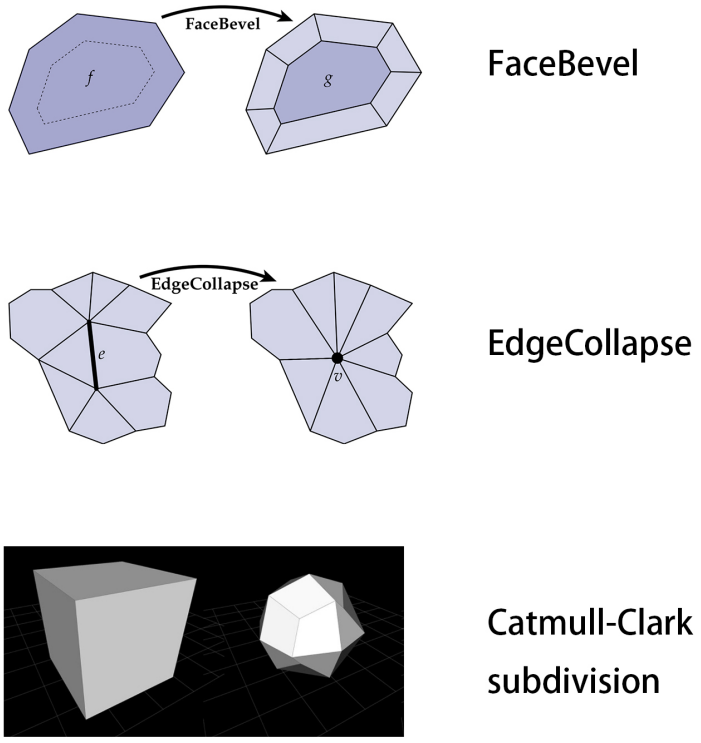


Figure 42. Operations of the final prototype

With these 3 different operations, I'll be able to explore how different kinds of operations work in the digital world and contribute to the mixed modeling environment.

4.6 Modeling Interactions Of The Final Prototype

This part explores the modeling interactions of the mixed modeling environment when combining the tangible interface with the digital interface. The sequence of interactions is

shown together with the output in each step. Some unsuccessful experiments are also listed and explained.

4.6.1 One Manifold Mesh Example

Tangible Interaction: place tangible cubes

The way to interact with the tangible interface is simply to place cubes in the physical space and the geometries encoded by them will be reconstructed in the digital space in real-time (see figure 43).

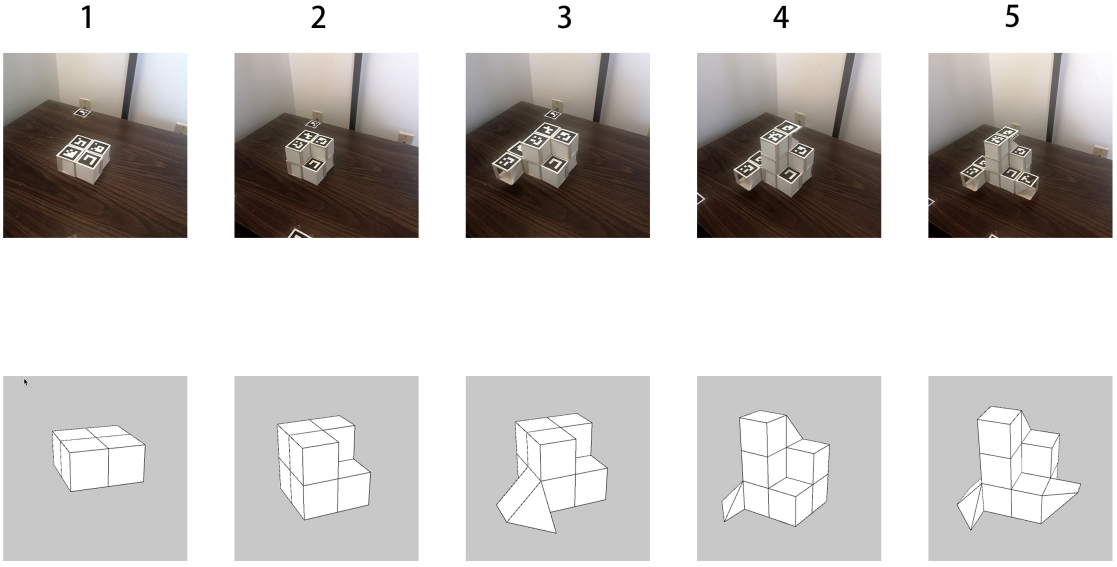


Figure 43. Tangible interaction: place cubes

Digital Interaction: adjust details of the model

The affordance of the graphical interface allows users to add details or trim the model to add details to the control cage. The effect of the 3 operations are shown in figure 44:

Face Bevel:

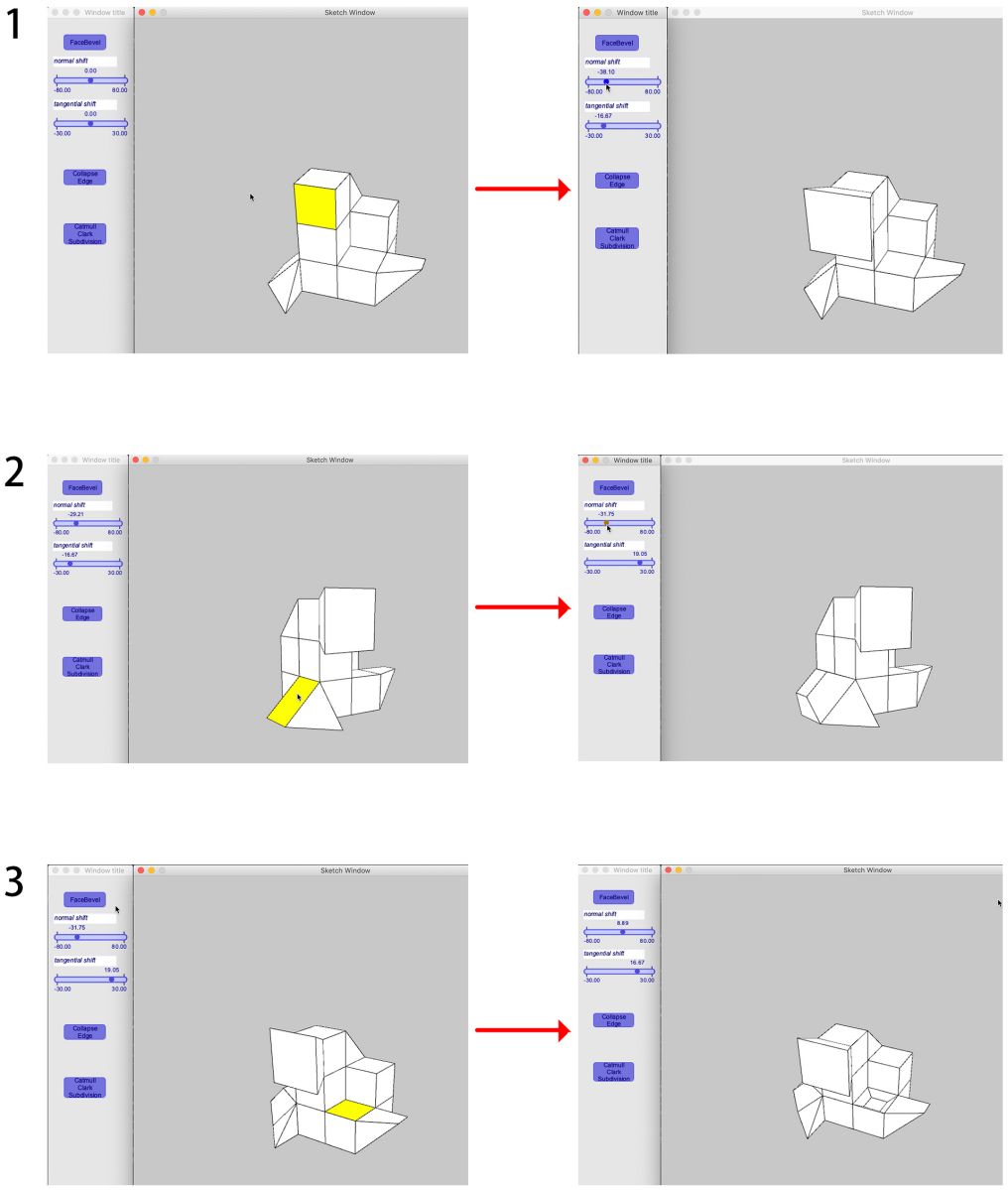


Figure 44. Use face bevel to edit the model

Edge Collapse (see figure 45):

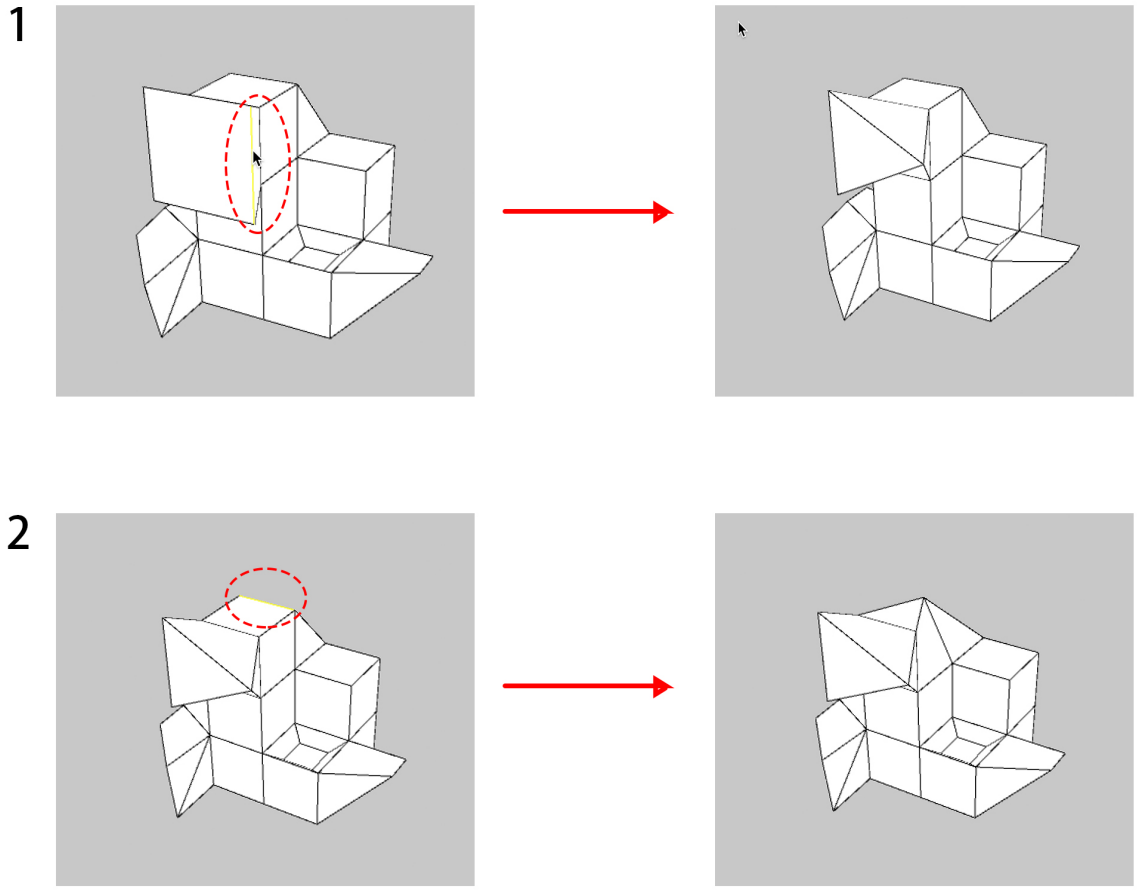


Figure 45. Use edge collapse to edit the model

Tangible Interaction: keep adding cubes in the physical space

After the model is modified using the graphical interface, users can still interact with the tangible interface by replacing, moving, or adding voxels. In this example, a cube is added in the physical space and the digital model updates itself accordingly in real-time.

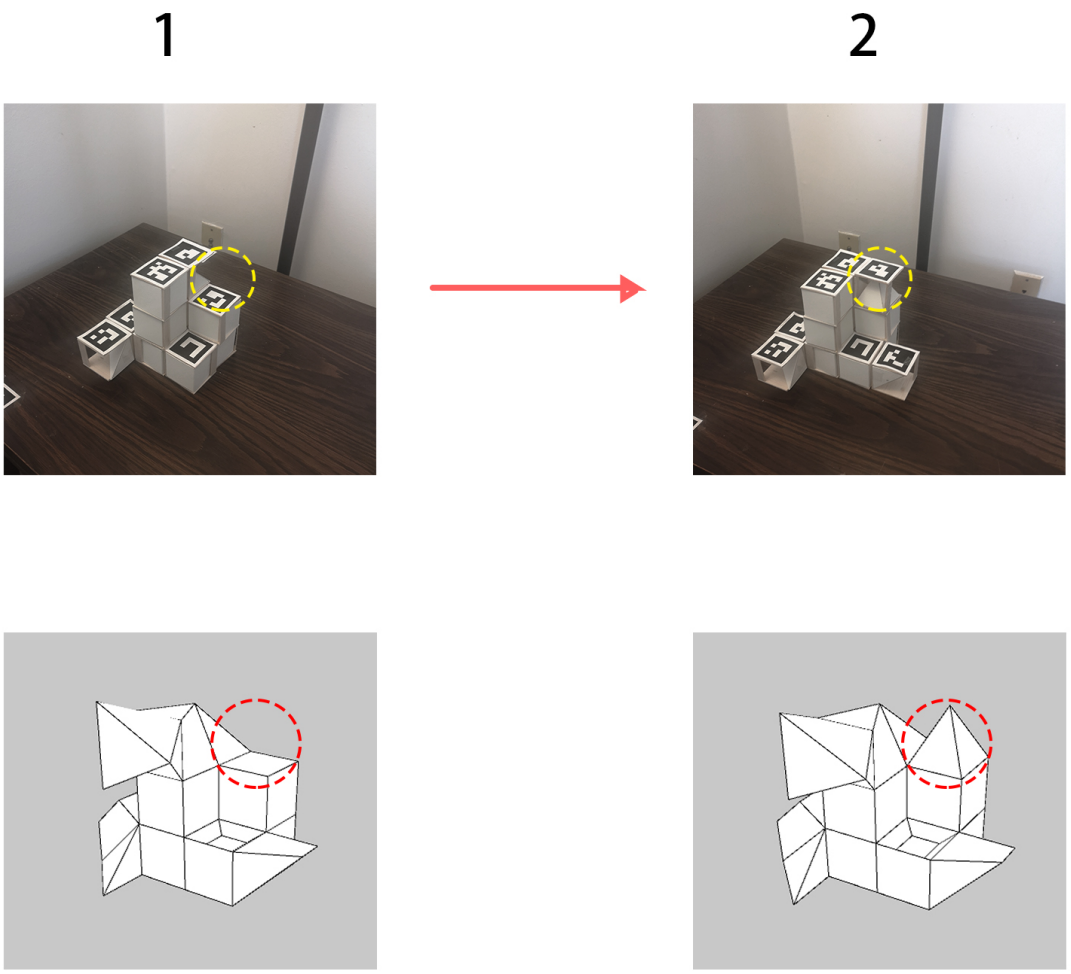


Figure 46. Add cubes in the physical world to update the model

Digital Interaction: use subdivision to smooth the model

If users are still not satisfied with the quality of the mesh and want to improve it, they can use global operations. For example, the Catmull-Clark subdivision is used to subdivide the model and make it smooth (as figure 47 shows).

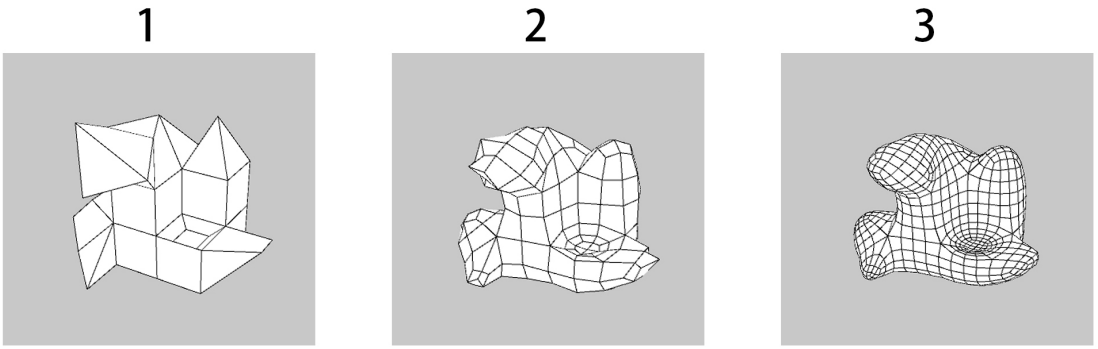


Figure 47. Catmull-Clark subdivision of the model

The above sequence of interactions shows the affordance of both the tangible and digital interface. Users can modify the voxel model while playing with tangible cubes and adjust details with the graphical interface. Moreover, global operations like subdivision offer a chance to break away from the limitation of voxels and generate smooth shapes out of that.

4.6.2 Failed Attempts

The kind of behavior that causes the system to fail is that users challenge the system by making non-manifold meshes with the tangible interface. In this condition, the digital system will fail to rebuild the geometries as half-edge meshes, and subsequent operations on the model will fail. Some examples are shown below:

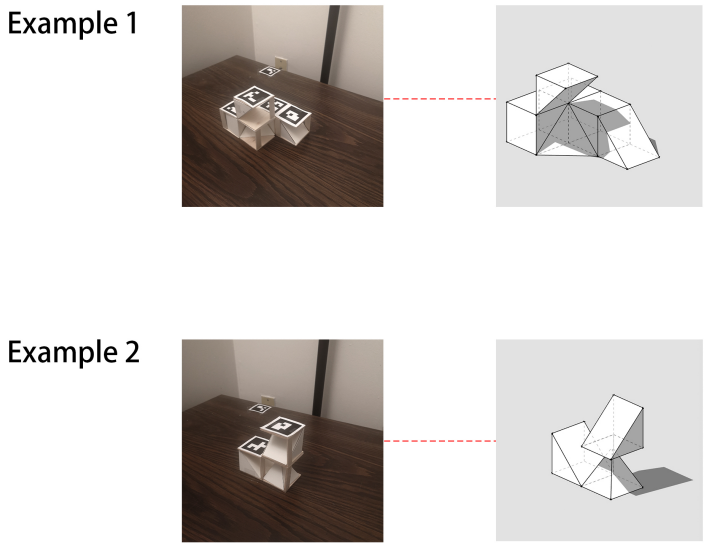


Figure 48. Examples that challenges manifoldness

Chapter 5. Discussion

After setting up the mixed modeling environment and use it for a real design task, we have a chance to experience and evaluate the benefits and limitations of this approach.

5.1 Benefits

Easy To Launch

Due to the algorithm supported by the marker detection library, it's easy to set up the tangible system since no position is hardcoded. The only thing that we need is to make sure the Kinect provides a top view of the workspace and is at least 1.5m from the table. Then the algorithm is going to be able to found the exact position of the workspace by locating the 3 markers on its corners.

No Significant Distraction Caused By Switching Between The Two Worlds

When using the modeling environment, there's no significant time loss observed when switching between the tangible and graphical user interfaces. The explanation is that since the tangible interface is responsible for constructing a low-resolution voxel model, and it's up to the graphical interface to add more details and do some post-processing, which means their roles can be split and users' attention doesn't have to switch between both worlds frequently. For example, users only need to switch from the digital world to the physical world when they find they need to iterate the low-level topological structure they built with the tangible interface, which is not easy to accomplish with the graphical interface.

Real-time, Interactive And Collaborative Experience

Since any change in both the physical world and the digital world will be visualized in real-time, users can have immediate feedback and adjust their design accordingly.

Also, the tangible interface shows the potential for collaborative design because the interaction logic of the tangible user interface is just like assembling toy bricks, which can be done in collaboration.

5.2 Contribution

The thesis explores the affordance of tangible interfaces as well as graphical interfaces regarding their strength. I developed a mixed modeling system that leverages the affordance of both. The tangible interface enables users to stack, remove, replace, and aggregate units and the graphical interface enables users to sculpt the voxel representation by adding details and do some post-processing.

Besides, the thesis proposes a way to merge the two media: use a voxel approach to encode geometries in tangible cubes and interpret them into half-edge meshes in the digital world. In this way, we will be able to switch back and force between the two interfaces and interact with the model. And this approach also proposes a framework that can be further extended by others.

The project also focus on new ways to iterate design without significant time cost or comprehension cost for both professional and non-professional users to create geometric models. In addition, the tangible interface opens up space for collaboration.

5.3 Limitations

Restriction Of The Grid And Gravity

Since the tangible system uses a voxel approach, the users have to follow the grid to make it work. In addition, gravity means that no cubes can hang in the space by sticking to others, which limits the kinds of geometries that the system can generate.

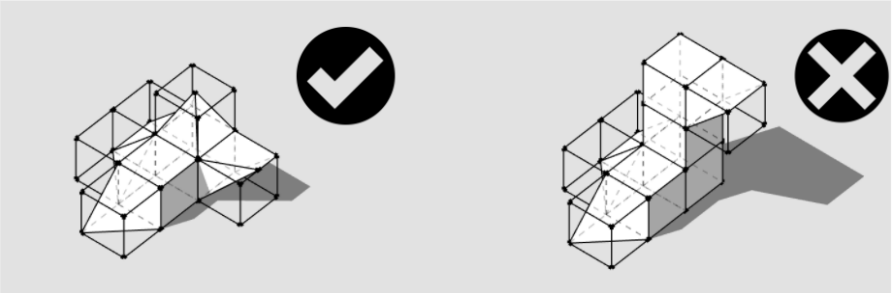


Figure 49. Examples of allowed and disallowed behaviour

In addition, the size of the physical space is limited by Kinect. Since Kinect only has a 640 X 480 output image and there's perspective distortion around the edge, each cube has to be as large as 5.5cm X 5.5cm X 5.5cm to be detected. This significantly reduces the number of cells that the tangible interface can hold. Therefore, the tangible interface can't hold hundreds of voxels to create something like a rabbit (as figure 50 shows).

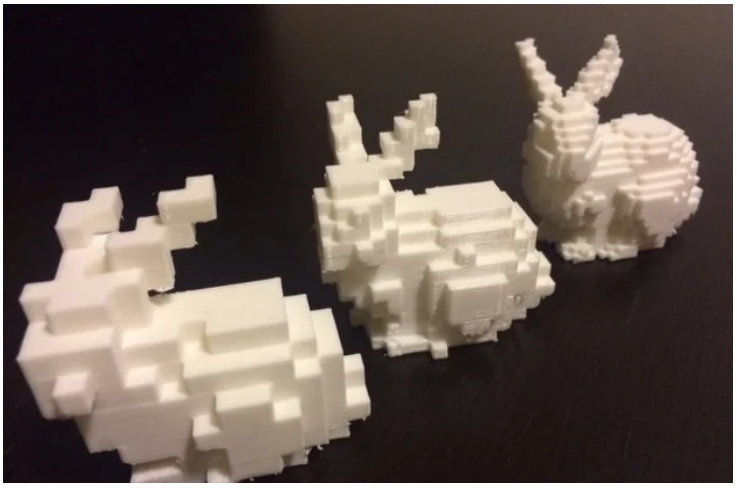


Figure 50. Low voxel rabiit, source: https://www.thingiverse.com/thing:1052991

Non-manifold Mesh Probelm

As is explained in section 4.5.2, users can challenge the system by ignoring the restrictions on manifold meshes. By combining tangible units to form a non-manifold mesh, they can cause an error in the digital model since that geometry is not compatible with the half-edge data structure. Therefore, most of the local operations and all global operations will fail.

Interactions In the Graphical Interface Can Be Improved

Some interactions in the graphical interface are not implemented perfectly. For example, providing the normal shift and tangential shift is supported by dragging a slide bar. However, using transformation widgets could be a better choice for graphical interactions (see figure 51).

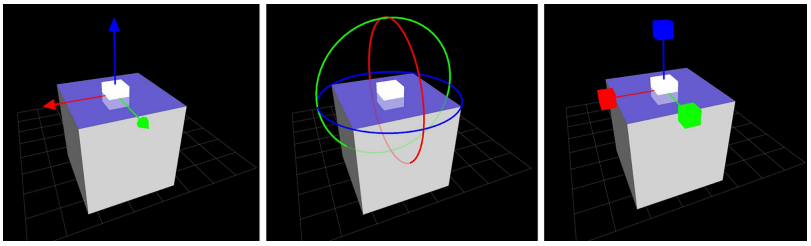


Figure 51. transformation widgets, source: https://github.com/cmu462/Scotty3D/wiki/User-Guide-(MeshEdit)

5.4 Next Steps

There are some potentials to explore further to improve the system:

Improve the tangible interface

There can be more ways to improve the tangible interface. For example, using a high-resolution RGBD camera will significantly increase the number of cubes that the physical space can hold by decreasing the size of each cube.

Another possibility is using embedded electronics instead of computer vision. It has the potential to solve the limitation of the grid as well as gravity. For example, users can have various well designed tangible blocks that can be plugged to each other and thus breaks away from the gravity problem. Also, better interaction for the graphical interface can be explored based on studying the design of existing design software, which will make it more user friendly.

Address the non-manifold mesh problem

Users can challenge the modeling system by creating non-manifold meshes with the tangible interface and make subsequent digital operations fail. One way to solve this problem is to split the non-manifold mesh into several manifold meshes. This algorithm is even more complex than buildMesh(), and there are some conditions to be taken care of. For example, a user first made a manifold mesh with the tangible interface and edited the model with the graphical interface, then the user changes the tangible representation into a non-manifold mesh by replacing some cubes. In this case, the system needs to rebuild it as several manifold meshes and keep the change done by the graphical interface.

Testing

There can be well-designed experiments and metrics to evaluate the design of such a mixed modeling environment. For example, researchers can quantify the time users waste when switching back and force between the physical and digital worlds, and use this to improve the interaction logic in both the tangible and graphical interfaces.

Chapter 6. Conclusion

In conclusion, my thesis prototyped a mixed modeling environment that leverages the strengths of both tangible and graphical user interfaces. It provides physical affordance to build a low-resolution voxel model just like playing with toy bricks and offers more refined editing operations and post-processing functions in the graphical user interface. Users can interact with both worlds to create the geometries they want and iterate their designs easily.

I also believe in the potential that such a mixed modeling system can support collaborative design modeling for the design field, especially for those non-experts to participate in the creation of some design work and be able to have a medium to express their ideas.

As a low-fidelity thesis prototype, it has some limitations and the digital operations provided are not very rich, which can be improved by future researches, for example exploring units with embedded electronics or something beyond a voxel representation. I hope that this thesis can inspire future research with its framework for a mixed modeling environment and successors can add more features or explore alternative approaches for either the tangible interface or the graphical one.

References

Abdelmohsen, S. M., & Do, E. Y. L. (2007). TangiCAD: Tangible interface for manipulating architectural 3D models. In Proc. CAADRIA (Vol. 7, pp. 1-8).

Al-Qattan, E., Galanter, P., & Yan, W. (2016). Developing a tangible user interface for parametric and BIM applications using physical computing systems. Proc. eCAADe 2016.

"Barcode and QR Code Scanner Using ZBar and OpenCV | Learn OpenCV." n.d. Accessed April 25, 2020. https://www.learnopencv.com/barcode-and-qr-code-scanner-using-zbar-and-opencv/.

Fukuda, Tomohiro, et al. "A Dynamic Physical Model Based on a 3D Digital Model for Architectural Rapid Prototyping." Automation in Construction, vol. 72, 2016, pp. 9–17., doi:10.1016/j.autcon.2016.07.002.

Garrido-Jurado, S., R. Muñoz-Salinas, F.J. Madrid-Cuevas, and M.J. Marín-Jiménez. 2014. "Automatic Generation and Detection of Highly Reliable Fiducial Markers under Occlusion." Pattern Recognition 47 (6): 2280–92. https://doi.org/10.1016/j.patcog.2014.01.005.

Han, Lu (2017): A Concurrent Physical and Digital Modeling Environment / Exploring Tactile and Parametric Interactions in Design Modeling. figshare. Thesis.

Hughes, C. E., Stapleton, C. B., Hughes, D. E. and Smith, E. (2005). Mixed Reality in Education, Entertainment and Training: An Interdisciplinary Approach. IEEE Computer Graphics and Applications, 26,6, 24-30.

Ishii, H., & Ullmer, B. (1997, March). Tangible bits: towards seamless interfaces between people, bits and atoms. In Proceedings of the ACM SIGCHI Conference on Human factors in computing systems (pp. 234-241). ACM.

Jordà, Sergi, Günter Geiger, Marcos Alonso, and Martin Kaltenbrunner. 2007. “The ReacTable: Exploring the Synergy between Live Music Performance and Tabletop Tangible Interfaces.” In Proceedings of the 1st International Conference on Tangible and Embedded Interaction, 139–146.

Kim, M.J. and Maher, M.L.: 2005, Comparison Of Designers Using A Tangible User Interface And A Graphical User Interface and the Impact On Spatial Cognition, in Proceedings of International Workshop on Human Behaviour in Designing, Melbourne, Victoria, Australia, KCDCC, University of Sydney, pp. 81-94.

McGuire, Max. 2000. “The Half-Edge Data Structure.” Website: Http://Www. Flipcode. Com/Articles/Article Halfedgepf. Shtml.

Milgram, P., & Kishino, F. (1994). A taxonomy of mixed reality visual displays. IEICE TRANSACTIONS on Information and Systems, 77(12), 1321-1329.

Nakagaki, Ken, Artem Dementyev, Sean Follmer, Joseph A. Paradiso, and Hiroshi Ishii. 2016. “ChainFORM: A Linear Integrated Modular Hardware System for Shape Changing Interfaces.” In Proceedings of the 29th Annual Symposium on User Interface Software and Technology - UIST ’16, 87–96. Tokyo, Japan: ACM Press. https://doi.org/10.1145/2984511.2984587.

Nakagaki, Ken, Luke Vink, Jared Counts, Daniel Windham, Daniel Leithinger, Sean Follmer, and Hiroshi Ishii. 2016. “Materiable: Rendering Dynamic Material Properties in Response to Direct Physical Touch with Shape Changing Interfaces.” In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, 2764–72. San Jose California USA: ACM. https://doi.org/10.1145/2858036.2858104.

Petrasova, A., Harmon, B., Petras, V., & Mitasova, H. (2014, June). GIS-based environmental modeling with tangible interaction and dynamic visualization. In Proceedings of the 7th International Congress on Environmental Modelling and Software, San Diego, CA, USA (pp. 15-19).

Salim, F. D., Mulder, H. M., & Burry, J. R. (2011). Form fostering: A novel design approach for interacting with parametric models in the embodied virtuality. Journal of Information Technology in Construction (ITcon), 16(9), 135-150.

Soon, Tan Jin. 2008. “QR Code.” Synthesis Journal 2008: 59–78.

Suchman, L., & Suchman, L. A. (2007). Human-machine reconfigurations: Plans and situated actions. Cambridge university press.

Tangible Modeling with Open Source Gis. 2018. New York, NY: Springer Science+Business Media.

Ullmer, Brygg, and Hiroshi Ishii. “The MetaDESK.” Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology - UIST '97, 1997, doi:10.1145/263407.263551.

Underkoffler, J., & Ishii, H. (1999, May). Urp: a luminous-tangible workbench for urban planning and design. In Proceedings of the SIGCHI conference on Human Factors in Computing Systems (pp. 386-393). ACM.

Appendix

Project code

Project code for both the tangible part and the digital part can be accessed at:

https://github.com/hongtaooMA/thesis

Additional Modeling Examples

Here are two other shapes created using this modeling environment together with the corresponding interactions. Although users can switch between the physical world and the digital world frequently when using it, it's not manifested in the examples just for convenience.

Example 1

1 Place cubes in the physical space

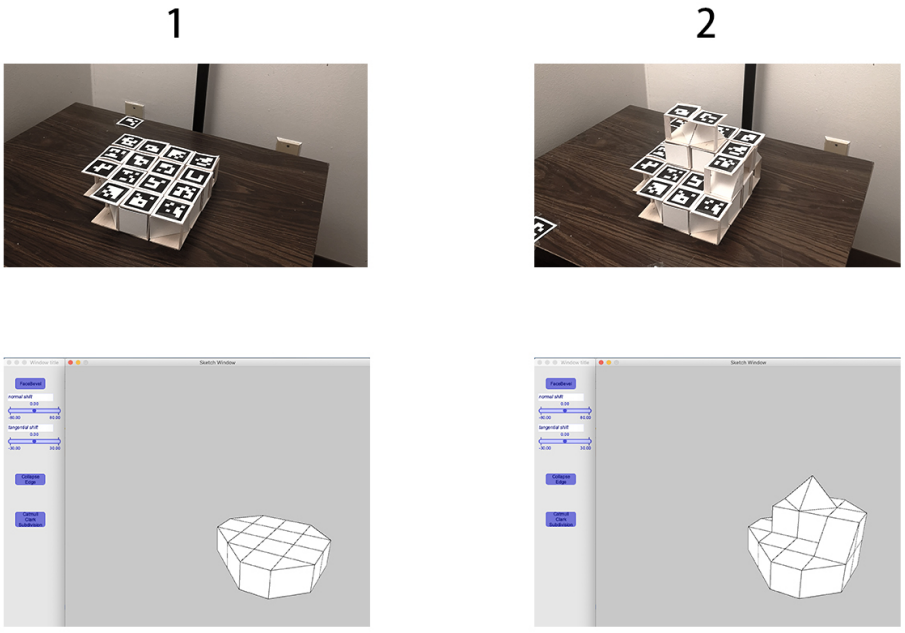


Figure 52. Example 1, step 1

2 Face Bevel

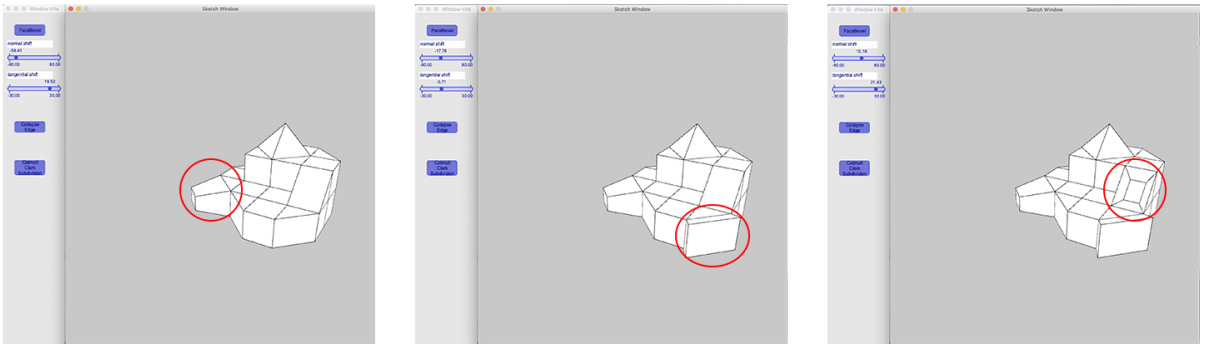


Figure 53. Example 1, step 2

3 Edge Collapse

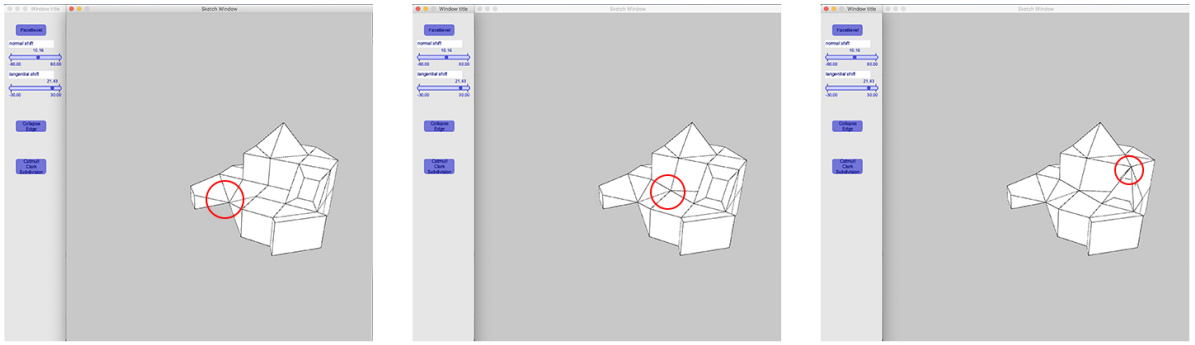


Figure 54. Example 1, step 3

4 Catmull-Clark subdivision

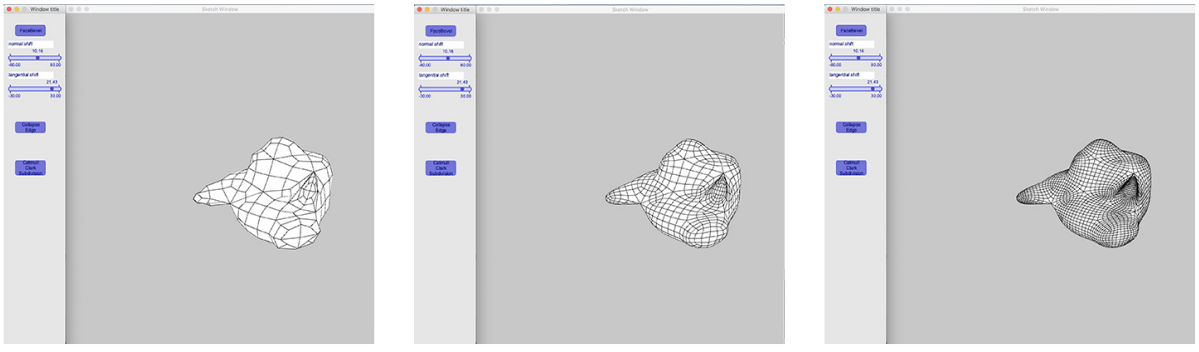


Figure 55. Example 1, step 4

Example 2

1 Place cubes in the physical space

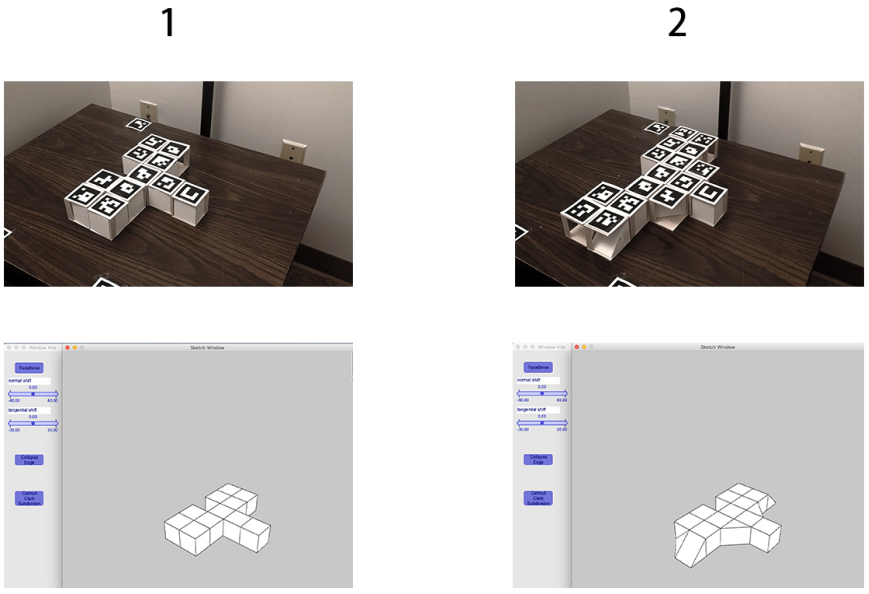


Figure 56. Example 2, step 1

2 Edge Collapse

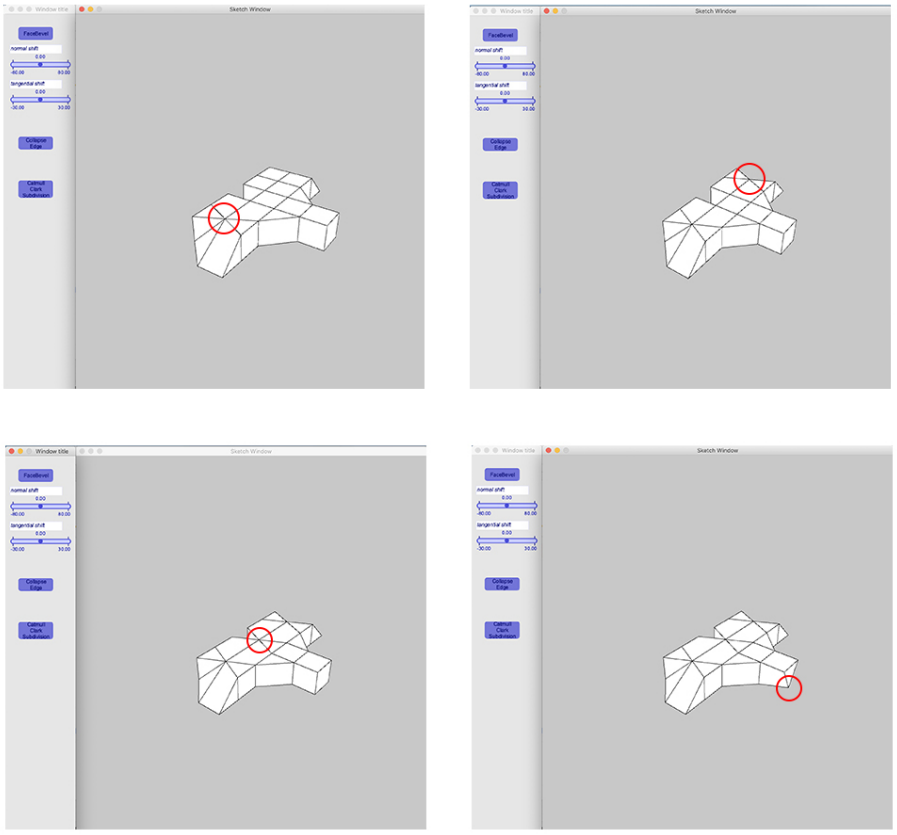


Figure 57. Example 2, step 2

3 Face Bevel

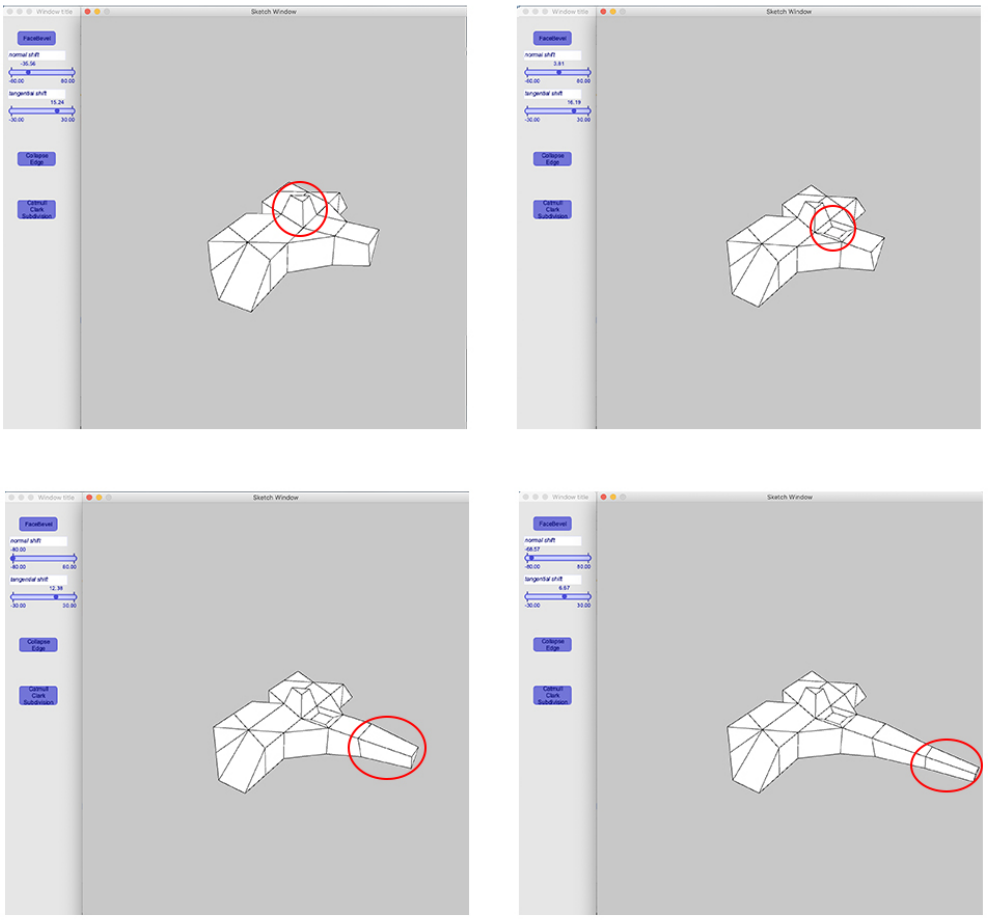


Figure 58. Example 2, step 3

4 Catmull-Clark subdivision

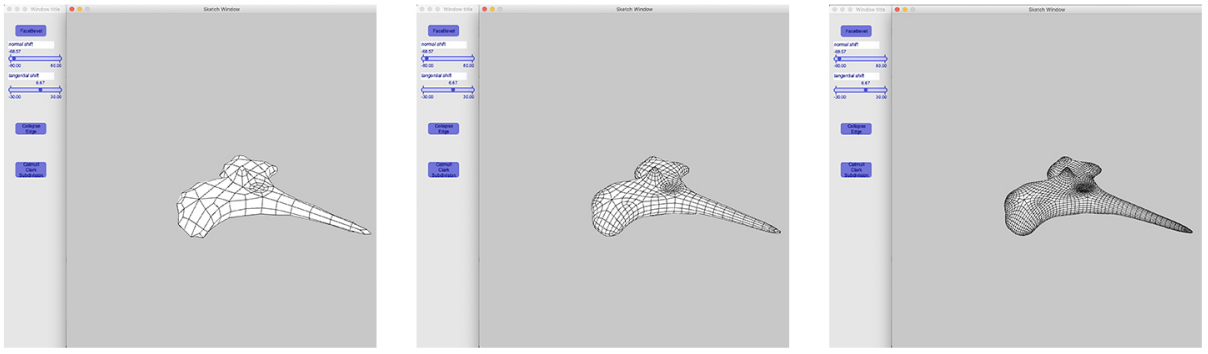


Figure 59. Example 2, step 4