공학석사학위논문

Hand motion based 3D printing slicer

손동작 기반의 3D Printing Slicer 개발

2018 년 2 월

서울대학교 대학원

기계항공공학부

이 태 석

손동작 기반의 3D Printing Slicer 개발

Hand motion based 3D printing slicer

지도교수 이 건 우

이 논문을 공학석사 학위논문으로 제출함

2017 년 10 월

서울대학교 대학원

기계항공공학부

이 태 석

이태석의 공학석사 학위논문을 인준함

2017 년 12 월

위 원 장 _______ 주 종 남 _______ (인)

부위원장 _______ 이 건 우 _______ (인)

위 원 _______ 박 용 래 _______ (인)

ABSTRACT

Hand motion based 3D printing slicer

Taeseok Lee
School of Mechanical and Aerospace Engineering
The Graduate School
Seoul National University

With the advancement of computer-aided design (CAD), 3D modeling has been extended not only in manufacturing and product design but also in architecture, civil engineering, and even art. The public become familiar with 3D modeling as self-production technologies, such as 3D printing and 3D scanning, become popular. However, 3D modeling is still considered a difficult task for users. One of the reasons for this stereotype is the conventional 2D work environment, operated by mouse, which results in a dimensional gap with the 3D model in the stereoscopic workspace with which the user wants to interact.

This paper describes an intuitive, easy-to-use 3D work environment for 3D modeling, using a 3D printing slicer as an example of a 3D modeler. The proposed 3D printing slicer selects a user's hand for 3D input device and operates by hand motion. Because a user's hand is an optimal input device that can transmit user intention through simple gesture without distortion, this research has mainly focused on providing intuitive human hand interaction with 3D models.

The proposed 3D printing slicer collects data regarding the hand and each finger joint through a ready-made hand tracking device, Leap Motion, which it uses to recognize hand gestures. Then, mesh processing function associated with the recognized hand gesture is activated, which enables easy deformation of the model for 3D printing. As a result, hand motion makes it possible to implement the functions that were difficult to use in the existing 2D work environment. Furthermore, hand motion makes it possible to interact with 3D model more efficiently by conveying the user's intuition to slicer functions that were not efficient with simple algorithms. Therefore, the contribution of this paper is more than just an implementation of the hand-controlled slicer function. It proves that, in interaction with 3D models, hand motion can be superior to existing 2D input devices.

Keyword : 3D modeling, 3D printing slicer, CAD, Hand gesture recognition, Hand tracking, Hand motion interface, Leap Motion

Student Number : 2016-20711

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1.
Introduction

The introduction of computer-aided design (CAD) has resulted in the digitization and automation of the product design and manufacture process reducing the considerable amount of time and money spent on product development and production [1]. Among various fields inside CAD, 3D modeling or shape modeling is the basis of CAD which defines the product's geometric shape. By enabling direct manipulation and modification of 3D figures, 3D modeling frees users from dealing with large numbers of 2D drawings and reduces the time necessary for product design.

Recently, the diversification of 3D modeling functions and the development of supporting IT technologies has produced more complicated and sophisticated shapes. Thus, the influence of 3D modeling has spread not only within manufacturing industry, but within the fields of large-scale architecture and civil engineering [2], as well as the visual arts industry [3, 4], particularly, animation, film, and video games. Nowadays, as the public has become familiar with self-production technologies such as 3D printing or scanning, which facilitates the creation of customized products, attention to 3D modeling and demand for 3D models has increased.

However, in the public eye, 3D modeling is still perceived as a difficult task requiring long training and professional expertise. One reason for this stereotype, which serves as a barrier to public use of 3D modeling, is that 3D modelers (i.e., 3D modeling software packages) are complex to use. The various 3D model creation functions of well-known, authentic CAD packages such as SolidWorks and AutoCAD or sculpting software such as Zbrush make them extremely complicated. However, the actual reason why users find 3D modelers difficult to use is the limited 2D workspaces in which 3D geometric shapes must be described. Although CAD has enabled product design to move away from 2D drawings of paper toward 3D models in virtual space, all the modeling is still performed in a 2D environment. The widely used 3D modelers require use of a monitor and a mouse, and users must undertake multiple processes to create 3D models by moving a mouse on a plane table while viewing the projected shape on a flat screen.

One can accurately visualize the desired model in one's head, but conventional 2D operation tools make it difficult to reproduce the desired shape. In short, the dimensional gap between the workspace and the work product is to be blamed for creating barriers to entry to this complex but powerful 3D modeling software. Therefore, constructing an easy-to-use and intuitive 3D environment for 3D modeling is an appropriate solution that can encourage users to approach this challenge without hesitation.

In the general CAD system, the 3D modeler working environment consists of a display device and an input device [1]. A display device is a tool, such as a monitor, that allows the user to view the 3D model's shape and its deformation. The other component, the input device, is a tool such as a mouse that gives specific orders for interacting with the 3D model. Adapting these two elements to the 3D environment means that the display device must be able to view the stereoscopic shape from any direction without projecting on the plane. Although holographic technology is appropriate, it has not yet been developed much, so recently developed VR and AR technologies could be an alternative method.

However, unlike the display device, which is merely used to show the current situation, the input device actually handles the 3D model. Therefore, a 3D modeler will still be inconvenient if the input device remains in a 2D environment. This also means that the selection of a 3D input device is also an important step for user convenience. When choosing an input device, important factors to consider are simplicity of control and clarity of input as well as the capability of 3D control. In this respect, the human hand is a more appropriate input device for dealing with 3D models than any other tool. The abundant hand control experiences obviate the need for extra training and energy consumption. Further, numerous hand motions exist [5], so the hand can represent the user's intention clearly.

This paper proposes a slicer program for 3D printing as an example of a 3D modeler that recognizes hand motions and operates through them. The hand motion based 3D printing slicer that this paper introduces provides easy and intuitive interaction with 3D models required for 3D printing. It uses Leap Motion [6] to track and capture hand gesture data; recognized hand gestures are then coupled with 3D model processing functions. The user could perform the 2D environment slicer's functions in the 3D environment just as same by using hand motions, and it demonstrated an advantage over specific 2D slicer functions.

The order of this paper is as follows. Chapter 2 describes related research on hand gesture interfaces and recognition methods. Chapters 3 and 4 explain the construction of the hand motion based slicer and the processes implemented for gesture recognition. Chapter 5 presents the results of user interaction with 3D models using hand motion in comparison with use of existing 2D based slicers. Finally, Chapter 6 presents conclusions, discusses limitations, and suggests future research directions.

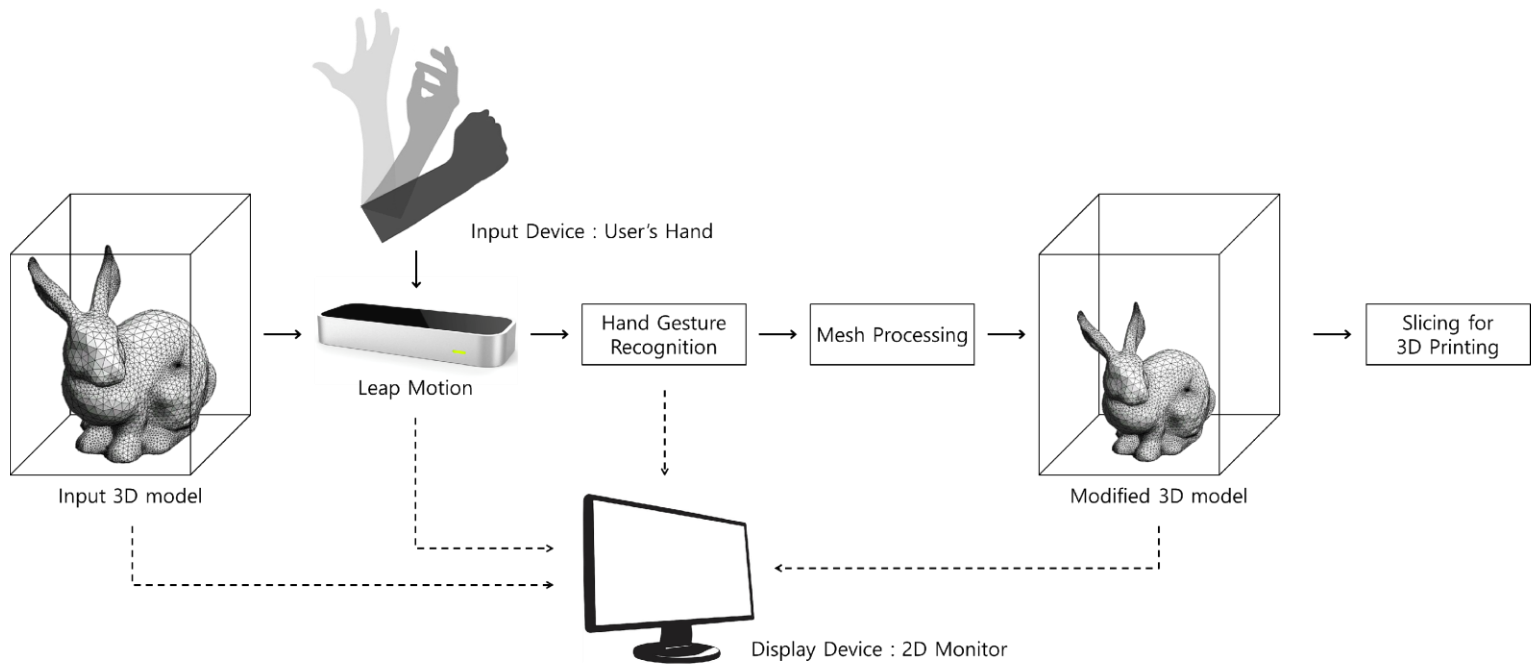


Fig. 1.1 Schematic diagram of hand motion based 3D printing slicer

CHAPTER 2.
Related Works

2.1. Hand motion based interface

Hand tracking and hand gesture recognition have been already studied in many areas such as human-computer interaction (HCI) [7], robotics [8], and motion capture [9]. These two technologies are fundamental to hand motion based interfaces that enable users to interact with the computer using their own hands instead of other input devices. The hand based interface is one of the most popular subjects in research on control systems as efficient and easy-to-use interaction with the computer has become increasingly important. Moreover, the development of practical real-world applications and adoption of devices using such interfaces are increasing.

Application of hand motion based interfaces is broad and widely varying [10], from video gaming and entertainment in the case of Nintendo Wii and Kinect [11] and DJI Spark [12] to complex robot control [13, 14] to even patient rehabilitation [15].

The human hand is regarded as a leading operator for control because the hand is a simple and familiar tool that does not need to study or practice. Users are already trained with long experiences. Many studies [7, 16] have demonstrated that the hand is an easy and intuitive device for any kind of purposes. Briefly, interaction using hand motion is considered a natural user interface (NUI) [17, 18], which emphasizes the convenience and naturalness of user interaction.

The hand can be an input device for 2D environments, such as a smartphone with a touch screen. However, it is in a 3D manipulation situation that the hand proves its worth. Several case studies describe user hand control of stereoscopic movement. Research has been conducted on user hand manipulation of cranes [19] or wheelchairs [20]. More complex hand based control of drones has been researched either [21]. A user's hand can serve as an enjoyable haptic feedback gaming device [22] or manipulate a 3D painting tool on floating air [23]. Furthermore, even operators in the early stage of a robotic surgery system are controlled with hand gestures [24].

Using a hand motion based interface for control is also advantageous because no equipment other than the hand-tracking device is necessary. With its simplicity, hand motion based interfaces are preferred in augmented reality (AR) and virtual reality (VR) [25-28]. Moreover, VR and AR are pure 3D environments where stereoscopic interaction with a user is important. Therefore, existing 2D-based tools such as mice or joysticks are less capable than a hand that can fully manipulate 3D motions.

2.2. Interaction with Hand motion in 3D modeling

The influence of interaction through hand motion is also valid in 3D modeling. For the CAD system, one study describes a haptic system that provides feedback and deforms the 3D model when the position-tracked pen is touched [29]. It has actually been commercialized in products such as haptic devices from 3D Systems

[30]. Although the system does not interact with any motion of the hand, it is still one type of hand tracking product since it captures the pen's position and provides haptic feedback to the user.

Other studies on hand gesture recognition for 3D modeling include Araùjo's 3D positioning equipment based research [31] and Wang's [32] and Murugappan's [33] image-based studies. Araùjo's research obtained the trajectory of thumb and index finger by touch sensor on screen and by 3D positioning equipment. If the finger touches the screen, the system sketches its trajectory, and if not, the system extrudes or sweeps the selected trajectory and creates a new 3D model. Wang's and Murugappan's studies adopted the stereo camera and depth camera each to extract the hand motion. Then Wang performed 3D assembly of multiple 3D models and Murugappan created and deformed 3D model with diverse methods.

However, the papers described above recognized only limited types of hand gestures. Araùjo tracked only the thumb and index finger, whereas Wang and Murugappan recognized one and three hand gestures, respectively. They overcame the lack of function by assigning different roles for each hand or by combining the hand's motion and direction of movement. Their approaches seem quite reasonable and easy because only few hand motions are required. However, it might hinder the intuition of the hand gesture because the function and the hand motion are lack of relevance since there is no similarity between the desired function and hand gesture. In this paper, however, hand gesture and specific function are intimately related.

Another 3D modeling tool, Sculpting [34], can easily deform and create 3D models through hand tracking using Leap Motion. Although interacting with a desired region of 3D model is possible, it is not a genuine hand motion based interface because it uses only position information rather than hand gestures.

2.3. Hand gesture recognition

For a sensible, responsive hand motion based interface, hand gesture recognition should first be available. Because the computer does not understand the hand motions currently taken from the outside, it collects and interprets information about the gestures by various methods and means. Hand gesture recognition can be broadly classified as inertial-based, vision-based, and others depending on the sensor acquiring an external signal from a human hand and information from it.

Inertial-based hand gesture recognition uses an IMU sensor with accelerator and a gyroscope as a receiver of information. As seen in the previous studies [20, 35, 36], the IMU sensor is attached to the hand or wrist to determine the acceleration and roll, pitch, and yaw changes and process these signals with their own algorithms.

However, because IMU sensor is vulnerable to noise, signal processing has another big problem to solve, such as drift or filtering. Moreover, to detect multiple hands or to receive more signals not only from the hand but also from the fingers and wrist, more sensors are necessary, which becomes inconvenient for users.

Another widely used method is vision-based hand gesture recognition, which captures the image of a nearby hand, detects its position, and recognizes its gesture. Vision-based hand gesture recognition depends on whether or not a marker is present and by the type of camera used to get the image [16, 25, 37]. The marker tracking vision-based method uses active/passive and single/multiple markers as descriptors for finding the identical region of the hand to track hand more easily

[38]. Sometimes, color can be used as a marker [39, 40].However, markerless hand gesture recognition, adopts time-of-flight depth camera [41] or single/stereo, RGB/IR camera [42, 43] to track the user hand, and often both depth camera and RGB camera are used [44].

This method is vulnerable to occlusion because image data for the hand can be lost. But it only requires a camera, so it is relatively simple compared to other methods. Therefore, vision-based hand gesture recognition has recently seen wide application, and this paper also adopts it.

Another hand gesture recognition method is the glove device. Gloves are frequently used instruments for hand motion capture, because a glove is a wearable device that does not give the user an uncomfortable and constrained feeling. It can also provide a platform for various sensors for hand data such as an optic fiber cable [45], an accelerometer [46], or even color [40]. An EMG sensor is another well-known equipment for hand gesture recognition that uses electromyography to record muscle activity during hand gestures [47].

Multiple sensor systems consist of two or more data acquisition devices are exist, such as a combination of pressure sensors and EMG sensors [48] or the set of a glove, a force sensor and an EMG sensor [49]. Finally, as machine learning stands out in the field of hand recognition, recent research has focused on classifying and training hand gestures for diverse sensor input [50, 51].

CHAPTER 3.
System Configuration

This chapter introduces the hardware and software environments required for implementing and operating of the proposed hand motion based 3D printing slicer.

The 3D printing slicer can be seen as a kind of a CAD system with the ability to edit a 3D model to be 3D printed, so the ultimate goal of this research is to apply an interacting function of 3D models with a proper input. Then, an input device is needed to give an order to the 3D model about what function will be applied to which degree. In addition, hardware as a medium that accepts signals from outside input device is necessary. Also computing component is required that interprets accepted signals from the medium and a transmitting component that sums up the comprehensive status of the slicer and delivers it to the user. Finally, for visualizing the collected information, a display device is necessary.

This system consists of three subcomponents:

1) Hardware: consists of the *display device*, *input device*, and the *sensor*, a medium hardware device that accepts signals from the input device.
2) Software: handles two key parts, the *analyzing unit* and the *transferring unit*. The former interprets the information received from the sensor, and the latter transmits the collected information to the display device.
3) Mesh processor: actually modifies the 3D model with a specific algorithm connected to each order from the input device.

3.1. Hardware

The 3D printing slicer hardware system consists of a display device, an input device, and a sensor. The display device informs the processing situation and settings of the current 3D printing slicer, and the input device provides order that represents how the user intends to modify the 3D model. Lastly, the sensor which can be specified among diverse products is an agent that accepts the commands of the input device and transmits to the software's analysis component. Fig. 3.1 shows the complete 3D printing slicer system, with its three major hardware components.

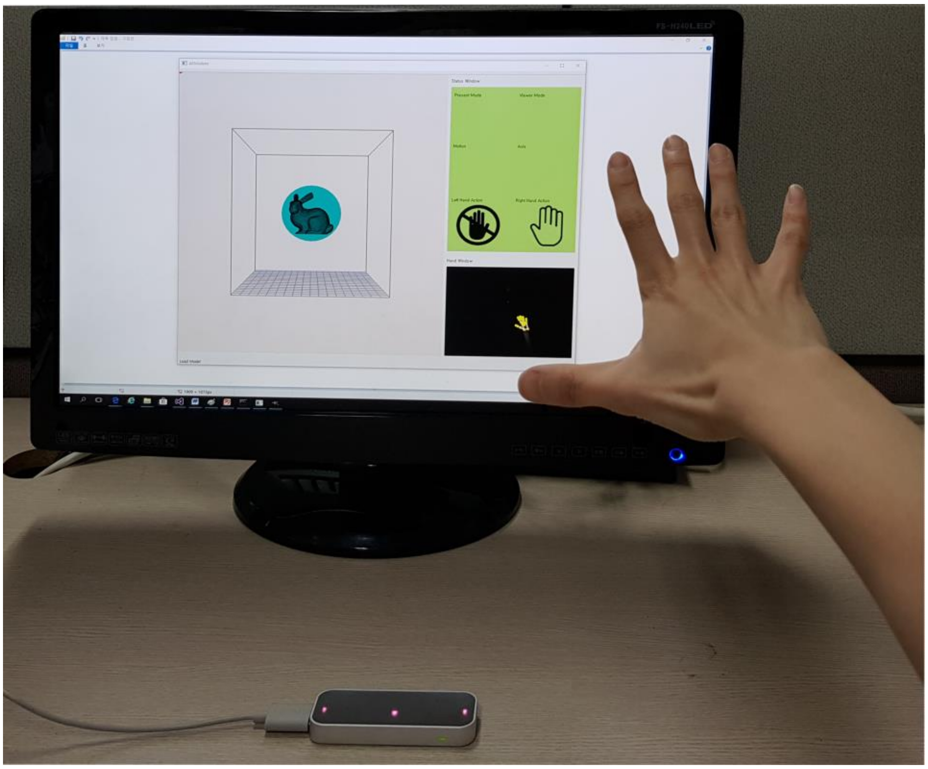


Fig. 3.1 Hardware configuration of proposed 3D printing slicer system: monitor (display device), hand (input device), and Leap motion (sensor)

3.1.1. Display Device

Because the role of display device is simply to provide information about the 3D printing slicer’s current context, this system uses a common 2D monitor. For a perfect 3D visualizing environment AR device such as a Hololens or a VR device such as Oculus could be used.

Fig. 3.2 (a) 2D monitor as a display device for proposed 3D printing slicer, (b) Hololens AR device, and (c) Oculus VR device for the future display device

3.1.2. Input Device

The user's hand replaces a conventional 2D input device such as a mouse. The hand is an optimal 3D input device that can convey user intention through simple and intuitive behavior without distortion. At the same time, the human hand has an unlimited degree of freedom to demonstrate various movements.

A device that employs a joystick, such as a drone manipulator, is also another tool for 3D movement control. However, the joystick's motion is rather close to the 2D interface with limited front/back and left/right movement. Therefore, intuitive control is difficult and requires a lot of practice. Although degree of freedom has improved with the advent of the 3D joystick, to include more various functions, the board contains a number of annoying and confusing buttons. On the contrary, the number of possible hand gestures is so numerous that the hand has no restrictions on giving diverse 3D motion without such auxiliaries, as Fig. 3.3 shows.

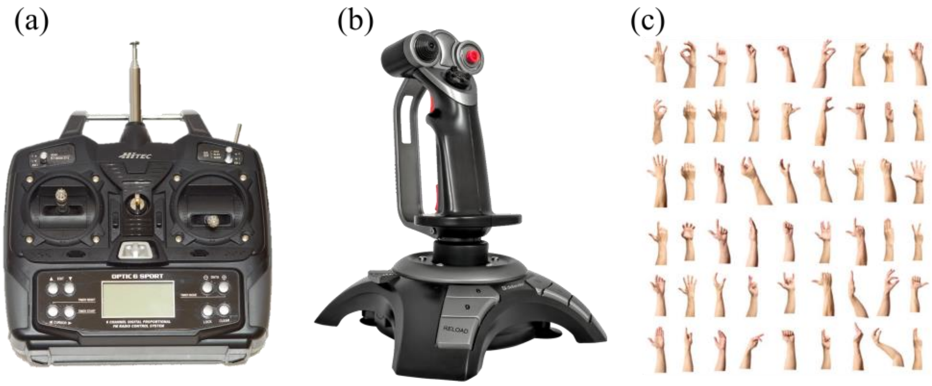


Fig. 3.3 Limited motion of (a) 2D joystick (b) 3D joystick, and (c) unlimited motion of bare hand

However, unlike the mouse, which is already connected to the , the hand operates outside of the computer. Therefore, to deliver the analog signals from the hand movements to the analyzing unit, medium component, or simply, sensor is required.

3.1.3. Sensor

A sensor has long been an unnecessary element for existing 3D modelers. However, for the proposed slicer system, it is now an essential component because the serves as a mediator that captures the user’s hand movement and converts digital signals that the analyzing unit can interpret. Converted digital signals include many useful data. Most of them are several features that can describe the status of the hand, such as position or velocity, normal of hand or even folding and unfolding of fingers. Captured data are then transmitted to the analyzing unit to be processed for recognizing the hand gesture.

One thing that should be considered is that human hand is a not a static input device but a dynamic one that continuously changes its status. Thus, the sensing system must be able to track the hand’s every movement in real time. Overall, the required sensor component must satisfy two criteria: data collection and real time tracking. The proposed 3D printing slicer system adopts the ready-made product Leap Motion because it is a specialized real time hand tracker. Chapter 4.1 will explain the implementation of hand tracking.

3.2. Software

The software consists of an analyzing unit that supports intuitive user hand motion by interpreting signals from the input device, and a transmitting unit that collects overall information about the slicer program and then delivers it to the display device.

The analyzing unit infers hand gesture based on the processed data from the sensor that are first captured form the user's hand. Then, it recognizes the user's intended hand gesture with an interpretation algorithm and finally transmits this result to the mesh processor, so that the related 3D modeling function can be executed, enabling users to interact with the 3D model with the help of the analyzing unit. Chapter 4.2 will describe the detailed process of interpretation or recognition of hand gestures.

Fig. 3.4 Hand tracking (left) and 3D model interaction (right) examples with Leap Motion

The transmitting unit is responsible for collecting current situations of the slicer program and sends this context to users via the display device. Although the interpreted hand gesture data from the analyzing unit is the most fundamental information, context data for the 3D printing slicer are gathered together from whole inside the slicer program, not just from the analyzing unit. Fig. 3.5 depicts a viewer that visualizes the context of the slicer program. The viewer is divided into three areas to contain each group of context to be treated. The first is a workspace area where the user can see the edited 3D model and the mesh processing result. Another section is a tracking area for visualizing the tracked hand data from sensor. Finally, there is a status window area where mesh processing functions connected to the current hand gesture and slicer program settings are displayed. These data are aggregated and classified so that users can refer to them at one glance.

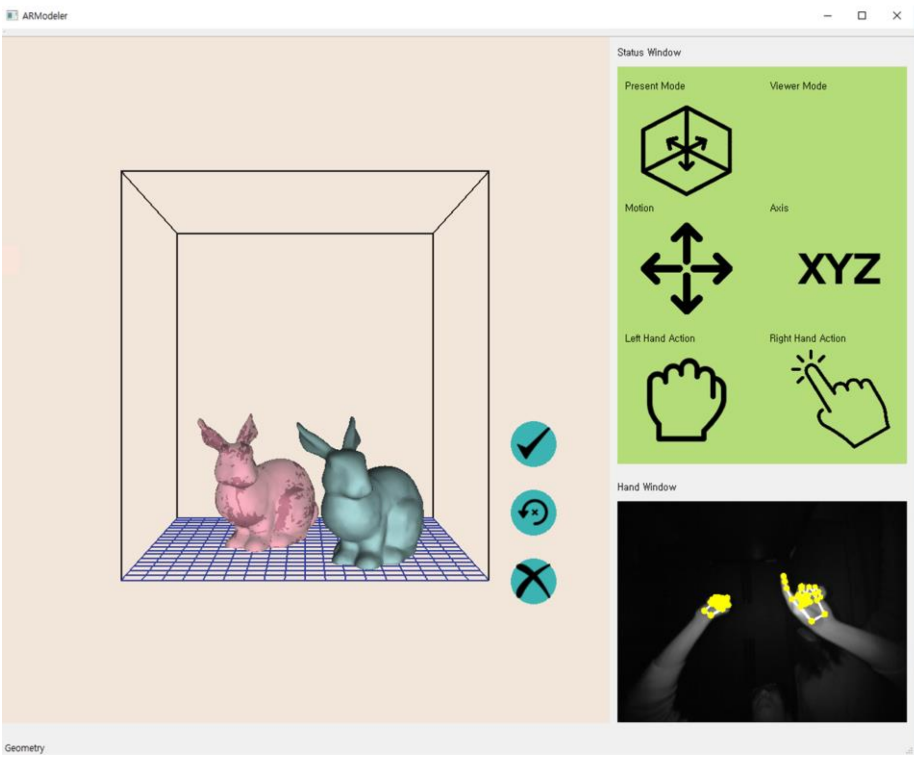


Fig. 3.5 Viewer from the transmitting unit of hand motion based 3D printing slicer

3.3. Mesh Processor

Mesh processing is a procedure for modifying a 3D model with a specific algorithm. In 3D printing, the 3D model is edited to an optimized shape for stable and fast printing. Each mesh processing function is connected with one hand gesture. Therefore, if one hand gesture is triggered, a mesh processor deforms the 3D model based on the specific mesh processing algorithm with determined values.

This paper focuses on a few generally used functions in 3D printing. These include transformation functions such as translation, rotation, scaling, as well as commonly used or required functions such as splitting of the model, layflat, and manual support generation (Table 3-1). In particular, the selected functions are intended to assist intuitive use with appropriate hand motions that can depict the characteristic of each function. Chapter 4.3 provides a detailed explanation.

Table 3-1 Implemented mesh processing functions and their descriptions

Mesh Processing	Functional Description
Geometric deformation	Basic transformation of 3D model including translation, rotation, and scaling
Splitting	Dividing of a given model with a specified 3D plane
Layflat	Function that estimates the bottom surface of a 3D model and attaches it to print bed
Manual Support generation	Creation of support structure that sustains the region floating in the air during 3D printing

Selected mesh processing algorithms are implemented through OpenMesh [52] and OpenFlipper [53] libraries that can deform 3D models based on the geometric and topologic information stored in the half-edge data structure.

CHAPTER 4.
Hand Gesture Recognition

Chapter 4 describes the hand gesture recognition process. With the proposed slicer system, users can interact with the 3D model by performing proper hand motion as they want. However, hand motion itself cannot transmit any orders. Converting the hand motion process into a computer-interpretable message is required, and this procedure is called hand gesture recognition. Interpreting the hand gesture involves two steps. First, hand tracking acquires information about the hand; second, the hand gesture recognition analyzes the prepared hand information to find out what hand motion is being performed.

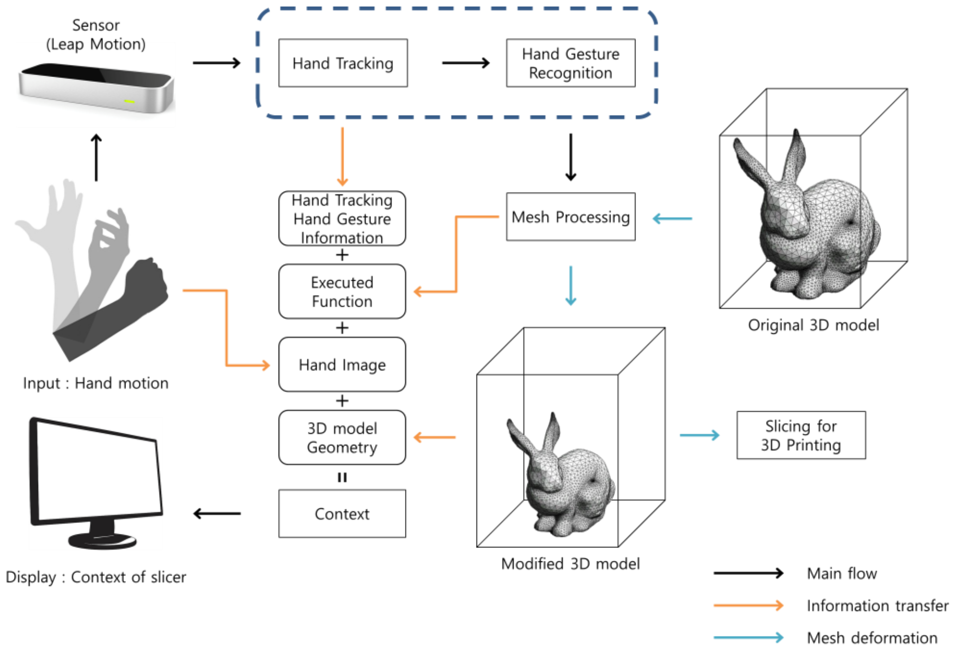


Fig. 4.1 Hand gesture recognition process from the arranged schematic diagram of hand motion based 3D printing slicer

4.1. Hand Tracking

To understand hand gestures, it is necessary to extract the information of the continuous state of hand and analyze these data for any type of hand gesture recognition method. It means that figuring out where the hand exists and observing how the hand's state has changed must be preceded before hand gesture recognition. Additionally, the human hand is continually wandering. This requires hand tracking to acquire the hand's geometry and motion information in real time. In other words, hand motion must be obtained after accumulating the stationary pose of the hand at every moment.

A corresponding component for hand tracking in the proposed 3D printing slicer was a sensor, and for which purpose ready-made product Leap Motion was specifically adopted. A detailed explanation of Leap Motion's characteristics and performance follows.

Leap Motion consists of pre-calibrated stereo IR cameras that acquire the 3D position of each pixel from the captured two images based on the principles of binocular disparity (Fig. 4.2). Therefore, the software can track hand shape through the images received from the stereo image. Moreover, its inherent rigging algorithm detects every finger joint and fingertips, and then marks their position. From 3D positions of hand, distances, angles and normals of palm and each fingertip are calculated (Fig. 4.3). The hand's velocity and that of other joints can be calculated using frame-per-second rates [54].

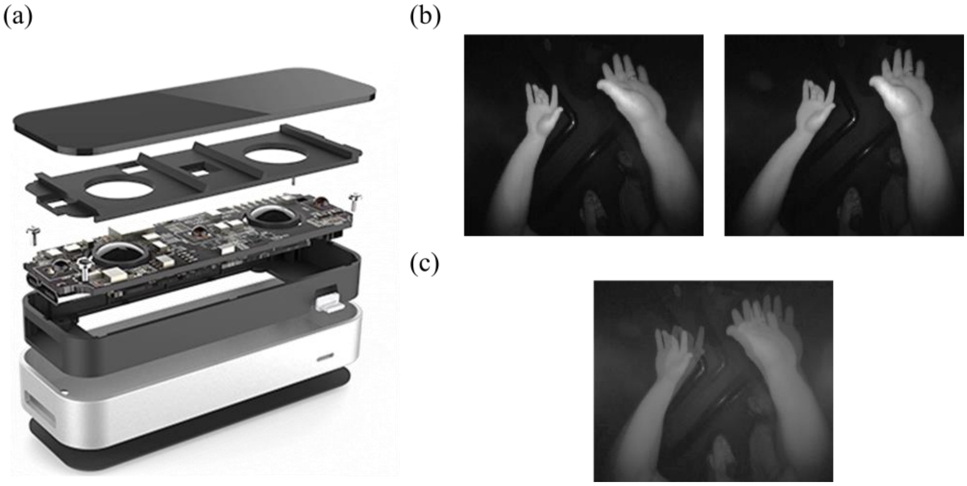


Fig. 4.2 (a) Stereo IR camera of Leap Motion, (b) acquired IR images, and (c) overlapped IR images for confirming the binocular disparity

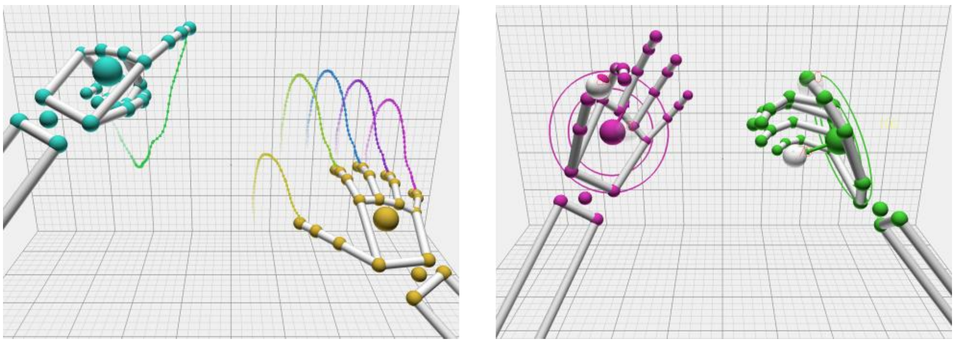


Fig. 4.3 Continuous position data of hand and fingertips (left) and normal data of tracked hand (right)

Chapter 2 described various hand tracking methods and instruments, including inertial-based hand tracking with acceleration and gyroscope, vision-based hand tracking with or without marker and depth/stereo camera, and hand tracking with EMG signal from muscles. Each method adopts different sensors for hand data acquisition because types and principles of obtaining data are fundamentally different.

One study on inertial-based hand tracking described an experiment involving a smart ring [55]. It used a single tri-axis accelerometer for its data acquisition. In the case of vision-based hand tracking with marker or with depth camera, commercial products exist: an example of the, former is the optical tracking marker and motion capture system of Vicon Nexus, and an example of the latter is Kinect [56]. Finally, Myo is a commercial EMG-based hand tracker [57].

All the introduced sensors are effective, quality devices for hand tracking, and they have been verified through several studies. However, this study uses markerless stereo camera based Leap Motion because other sensors are must be attached to the hand. For the proposed 3D printing slicer, interaction with the visible 3D models is important. Therefore, auxiliaries such as accelerometers, markers, and EMG armbands that might disturb or occlude user should be removed. The depth-based camera Kinect is the exception because, like Leap Motion, it has only a camera and not a wearable device.

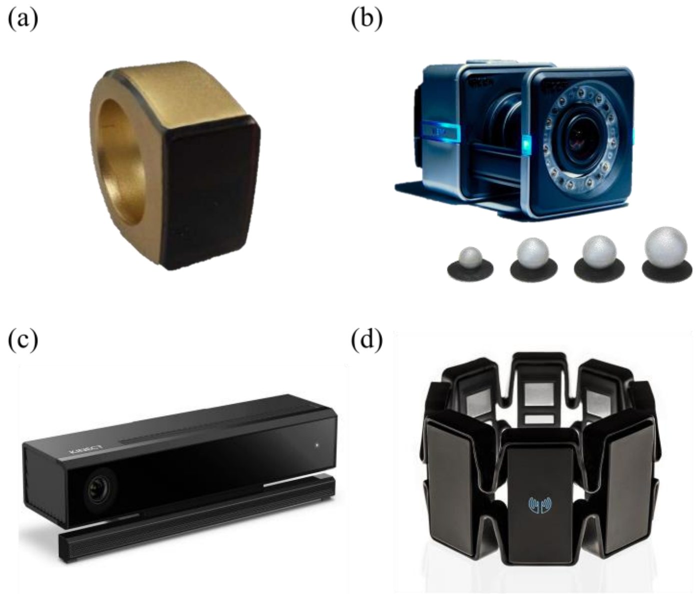


Fig. 4.4 Various hand tracking devices: (a) smart ring [55], (b) Vicon nexus with optical tracking marker, (c) Kinect, and (d) Myo

However, the direction and the tracking range requirements eliminated Kinect from use. According to the initial design, the hand tracker should look up the user hand because it is convenient for users when they sit down at the desk as Fig. 3.4 shows rather than Fig. 3.1 does. However, Kinect looks forward while Leap Motion looks up. Although Kinect can be rotated 90 degrees, Kinect has a much bigger sensing range than Leap Motion. Its closest range was at least 0.5m to 0.8m [11, 41], which is too far for users to raise their hands. Therefore, Leap Motion was finally selected as the hand tracker.

Furthermore, Leap Motion is specialized for hand tracking in that it provides deeper information that other hand trackers do not offer. Based on the basic hand data above, including finger folding, grabbing angle, palm curvature, palm-originated 3D axis, and even captured left and right hand images are acquired without any need for accessories. This simple but convenient sensor also tracks multiple hands at the same time, and its accuracy is guaranteed to be adequate [58]. So, Leap Motion is easily found in a system applied with the hand movements because it reduces the effort on hand tracking [59, 60].

Through the Leap Motion, hands were tracked at about 120 frames per second, and the tracked hand information was stored for a certain time (usually 1 second) and used as a data set for hand gesture recognition. Figure 4.5 presents the Leap Motion hand tracking results. Captured hand image and tracked hand and joint positions are overlapped to confirm whether they are tracked well.

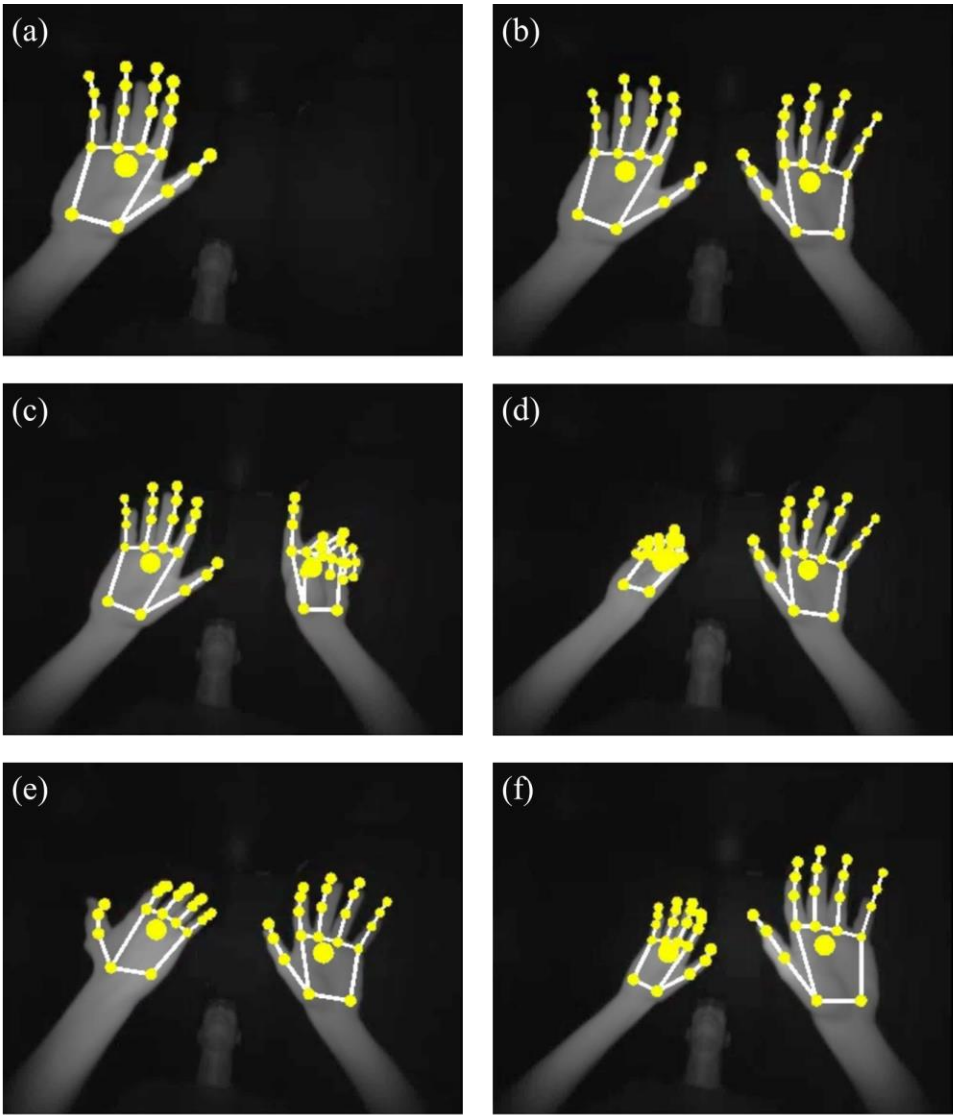


Fig. 4.5 Overlapped sensor images and various tracked hands: (a) one hand, (b) two hands, (c) pointing hand, (d) grabbing hand, (e) reversed hand, and (f) near and far hands.

One thing to notice is that because Leap Motion is an image-based hand tracker, hands are well traced in the specific region near the IR cameras called the interaction box. When hands are too close or far from the sensor, locating the hand from the captured IR images is difficult, which leads to huge errors. Moreover, occlusions, an inevitable phenomenon of image-based tracking systems, also present a problem. If the hand is covered by another object or another hand, hand tracking is impossible and results in irrational outcomes. Fig. 4.6 shows examples of failures in hand tracking.

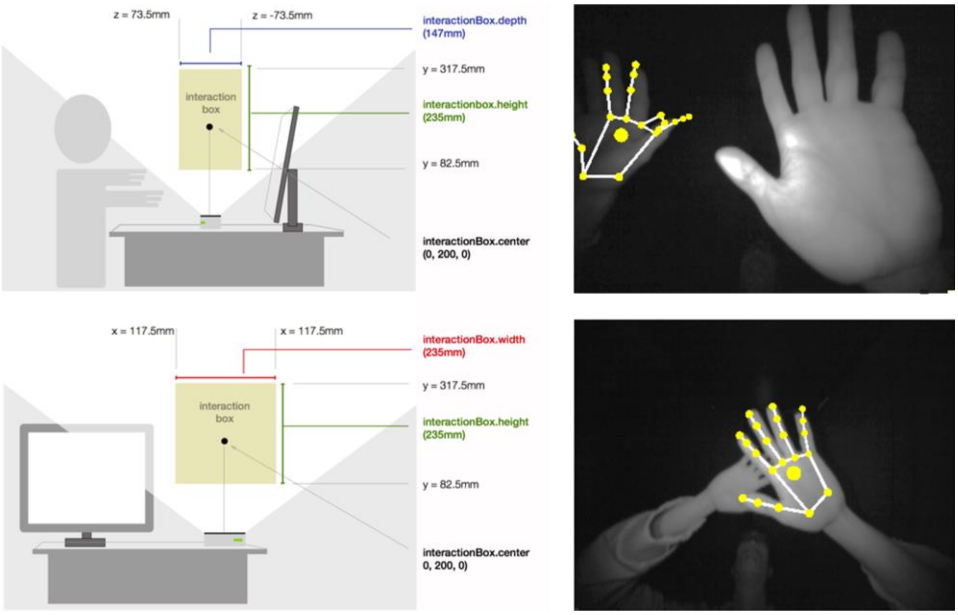


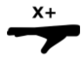
Fig. 4.6 Interaction box of Leap Motion (left) and failed hand tracking images (right)

4.2. Hand Gesture Recognition

4.2.1. Selection of Hand gesture

Table 4-1 lists the six major hand gestures that the proposed 3D printing slicer identifies: untracking, grabbing, pinching, pointing, nodding, and swiping. Furthermore, if each motion is closely related to direction, the system reflects it. For example, nodding which is the action rotating the wrist, has six different directions: roll, pitch, and yaw in both clockwise and counterclockwise directions. A swiping gesture also has six directions: up, down, left, right, front, and back.

Table 4-1 Recognized hand gestures in proposed 3D printing slicer

Hand Gesture	Action	Icon
Untracking	No hand is detected near the sensor	
Grabbing	Hand is changing into fist	
Pinching	Thumb and index finger is closing together	
Pointing	Only index finger is extending and touches virtual screen	
Nodding	Rotating hand with pivoted wrist (six directions : roll, pitch, and yaw in cw/ccw)	
Swiping	Translating hand with constant direction (six directions : up, down, left, right, front, back)	

The proposed printing slicer focuses on these hand gestures because for users they are frequent, familiar actions. Moreover, they are the hand gestures that can easily change the magnitude of the action or the intensity of input such as speed and distance. Although this number of recognizable hand gestures seems small, too many kinds of hand gestures might confuse the user, as in the case of joystick buttons. Moreover, pointing for menu selection and nodding for mode change to control the overall process have compensated the user's insufficient freedom with limited six hand gestures.

4.2.2. Hand gesture recognition algorithm

From hand tracking, sensor collected diverse hand related data. Based on the 3D positions of hand, fingertip and finger joints, the tracked information included velocity, angle, and normal of hand elements. Furthermore, Leap Motion provided more developed data such as angle of the grab, curvature of hand, correlated palm based axis, and even finger folding.

With sufficient hand data accumulation, we can infer what hand gesture has taken place. In the proposed slicer system, hand gestures are recognized by extracting the distinctive features of the hand movement pattern from prepared hand tracking data. Hand motion features are mainly the trends of increasing or decreasing of specific hand data. For example, a pinching motion shows a pattern in which the distance between the thumb and the index finger gradually decreases. Thus, a corresponding gesture was recognized based on the position information of the thumb and index finger. Fig. 4.7 and 4.8 show all the recognized hand gestures, followed by Table 4-2 that summarizes the noteworthy features of hand gestures.

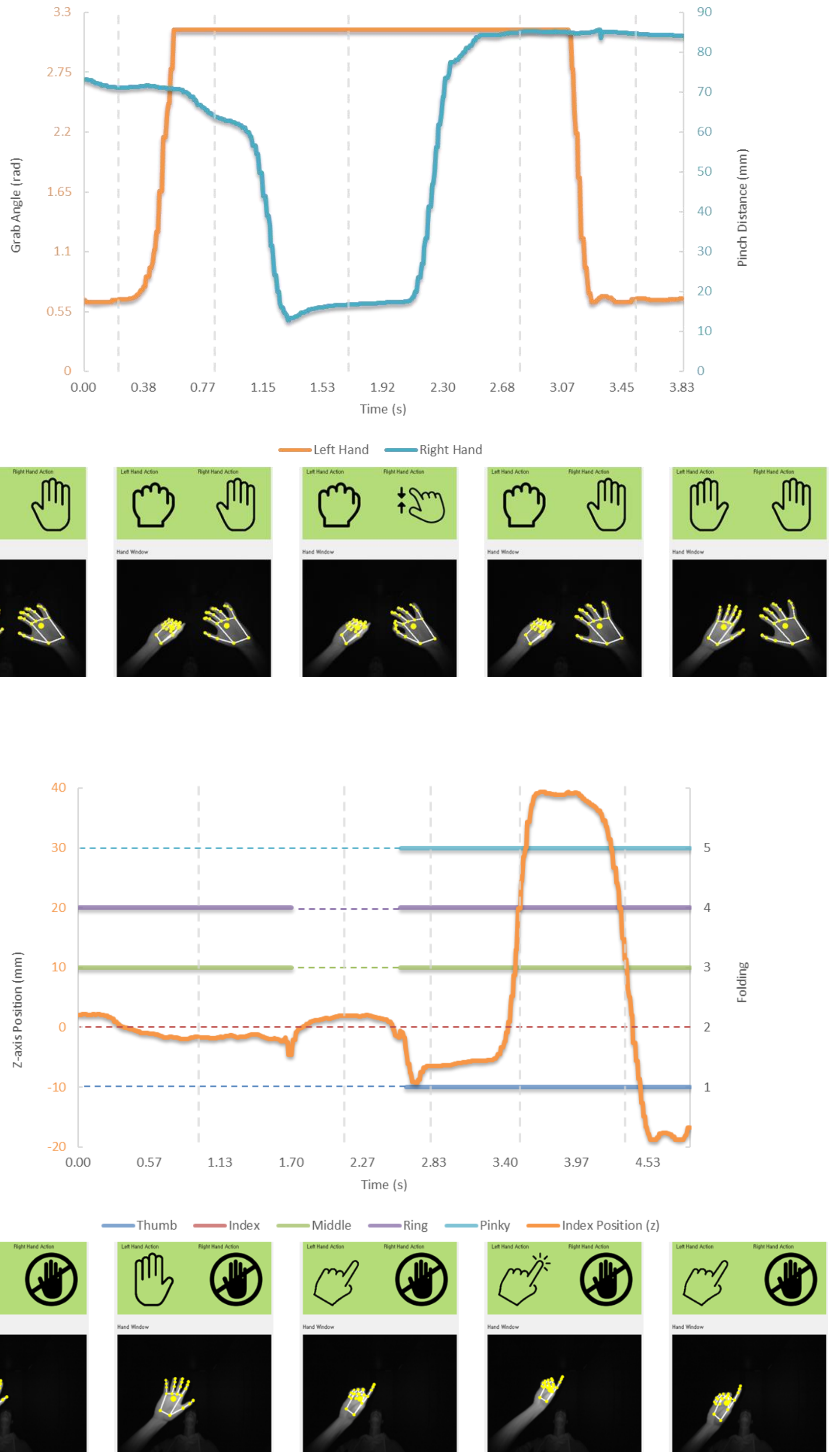


Fig. 4.7 Hand motion and the recognition of hand gesture: grabbing and pinching (up) and pointing (down)

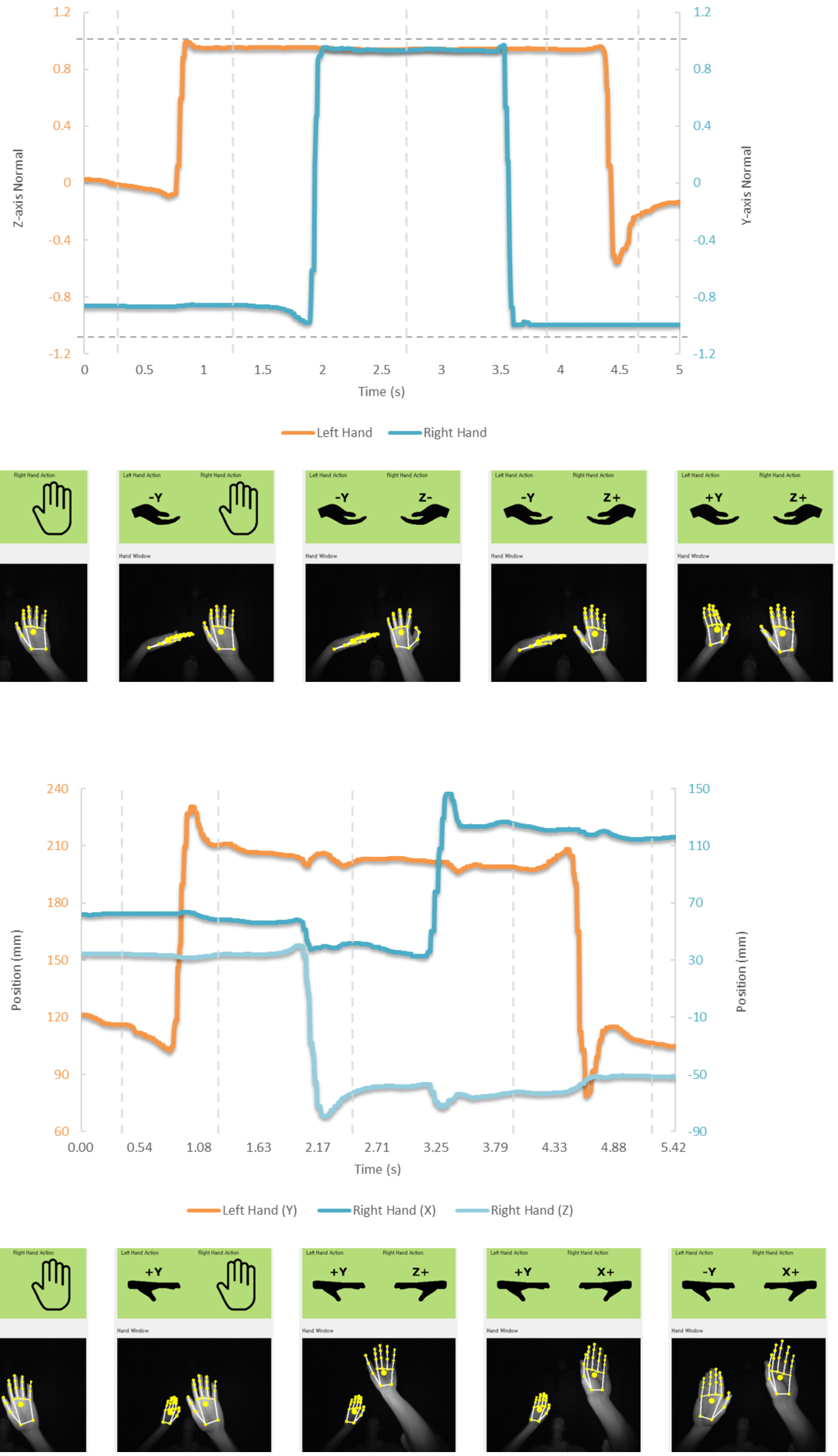


Fig. 4.8 Hand motion and the recognition of hand gesture: nodding (up) and swiping (down)

Table 4-2 Hand gestures and their corresponding features for gesture recognition

Hand Gesture	Features of hand gesture
Untracking	No data available
Grabbing	The grab angle that indicates the angle between palm and finger decreases from π to $-\pi$ (rad)
Pinching	The distance between thumb and index finger decreases
Pointing	The index finger is extended, whereas other fingers are folding, also index finger moves toward the front
Nodding	The normal of palm changes along the axis of roll, pitch, and yaw
Swiping	The position of palm moves along the x, y, and z axis

4.2.3. Improving in Hand gesture recognition

One obstacle to hand gesture recognition is the unintended hand movement, or noise, that occurs during the process of gesturing in the air [61]. This does not mean the error obtained during the acquisition of the hand information by the sensor but refers to an inevitable error from using the human hand as an input device. The input devices such as a mouse or touch pen send discrete signals when clicked or pressed, continue to follow the order when held, and finally end operation when released. However, the hand's input is continuous, and hand motion keeps changing. Therefore, hand movement is not a successive group of meaningful gestures. Valid gestures must be distinguishable from noise.

Frequent types of noise include the instantaneous changes in velocity or normal, inconsistent hand rotation direction, and blurred feature clarity from similarity between different gestures or from the hand and arm fatigue. These are types of errors that make the user's command ambiguous. The problem appeared with these errors can be improved by filtering out insignificant data. The trend of the specific hand data is more important than one value of data for recognizing hand gesture. Therefore, omitting of outliers improved recognition.

One example of solution for outlier is the separation of grabbing and pinching. Those two hand gestures have similar characteristic that the distance of the thumb and index finger decreases as they gets closer during the gesture. Fig. 4.9 is the chart for plotting the similar tendency in pinch distance of grabbing and pinching.

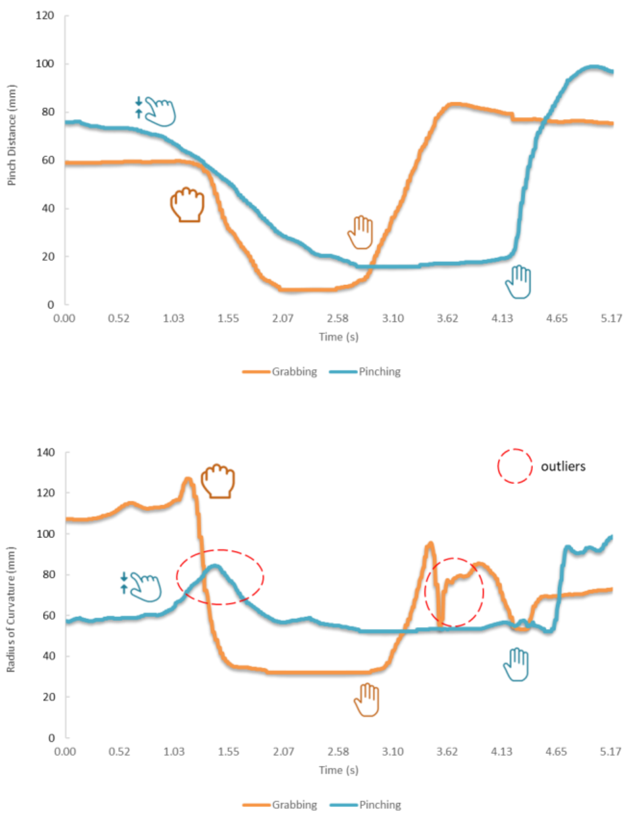


Fig. 4.9 Characteristics of grabbing and pinching gesture

However, in case of radius of curvature, the grabbing fluctuates more drastically than the pinching does. Although some outliers exist in both gestures, radius of curvature decreases drastically in grabbing gesture while it maintains constant. The outliers appear because occlusion is occurred between fingers and hand when gesturing. Therefore, the tendency of changing in radius of curvature can be the criteria for distinguishing the grabbing and the pinching gesture.

Another type of error is the pollution of gestures. In the process of performing the next gesture, the intermediate gesture should be ignored but is instead recognized as another gesture. Therefore, the system interprets another gesture incorrectly even if the user performed it correctly. This type of error is critical because it distorts the user’s intention. Therefore, applying appropriate thresholds to changes of hand data is important to distinguish meaningless noise from intended gestures. After all, adding motion filters and adjusting the correlated threshold values improved the system’s recognition rate.

In determining the appropriate threshold, discrimination of a nodding and a swiping can be a proper example. In the use of proposed slicer, the most frequently occurred problem was misrecognizing of nodding and swiping gesture. Fig. 4.10 shows the four data that describes the nodding and swiping gestures.

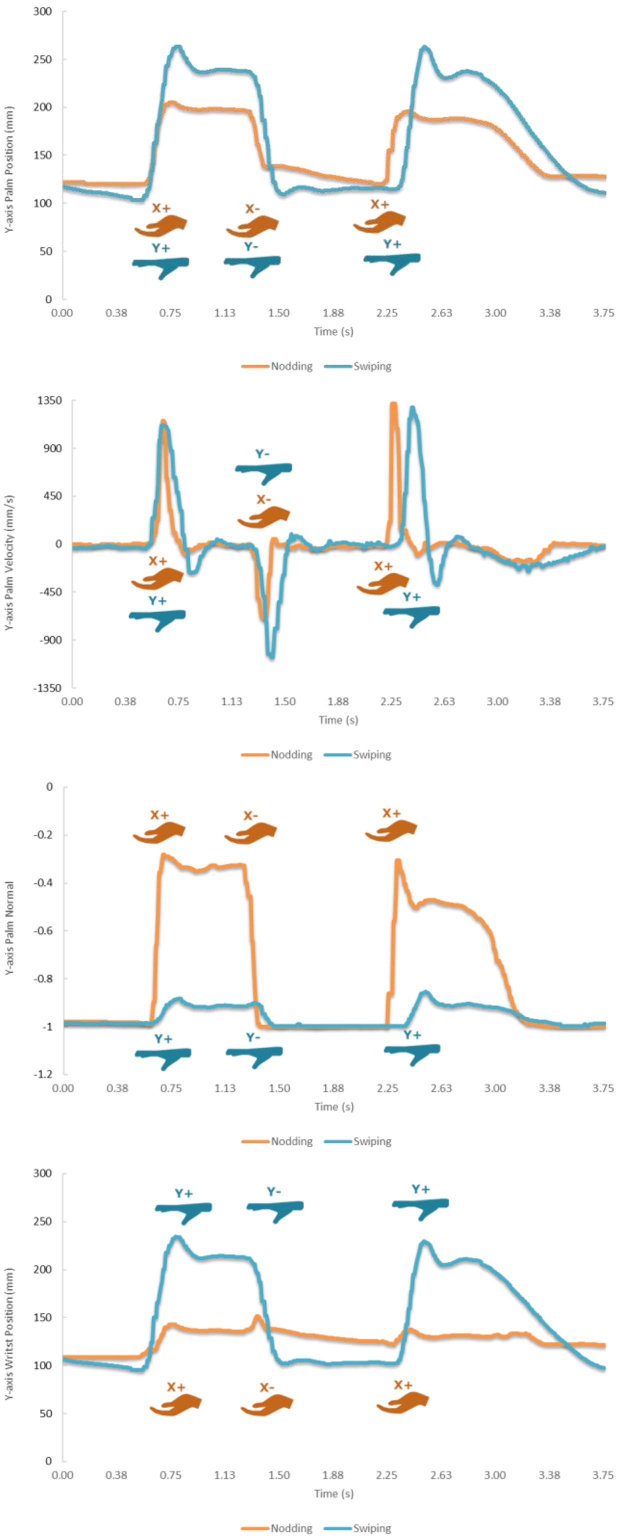


Fig. 4.10 Characteristics of nodding and swiping gesture

If direction is identical, nodding and swiping shows great similarity as it is shown in the Fig. 4.10. In the example for comparison, both gestures contain the up and down motion since nodding is x-axis rotating and swiping is y-axis moving. Therefore, trends in Y-axis position and Y-axis velocity are almost identical. However, the biggest difference of two gestures is the constraint from the wrist. When the hand is nodding, the wrist is pivoted so that it exposes two characteristics. Firstly, movement of wrist is nearly ignorable in nodding, while hand and wrist are moving together in swiping. Secondly, by pivoting the wrist, normal component changes consistently as palm rotates. On the other hand, normal of palm remains almost same in swiping. Therefore, to separate each gesture from other, proper threshold for normal and wrist position would be effective for a motion filtering.

4.3. Connection between Hand gesture and Mesh processing function

The recognized hand movements were connected with each function of mesh processing. Making reasonable and intuitive connections between the mesh processing functions and the hand gestures was the most important focus. For example, the model enlarging function executed with a pinching gesture, just like the zooming on a smartphone. In addition, the splitting function cuts the model along with the trajectory of the index finger like a knife, and pinching and pulling the corresponding region of the model like clay results in support generation. Table

4-3 shows the associations between hand gestures and mesh processing functions.

Table 4-3 Match of mesh processing function and the hand gesture

Mesh Processing	Hand Gesture	Analogy
Geometric deformation*	Grabbing (Translation) Pinching (Scaling)	Moving objects by hand Smartphone zooming with pinching
Splitting	Swiping	Cutting with a knife
Layflat	Nodding	Balancing object not to fall by giving torque like seesaw
Support generation**	Pinching	Playing with clay to create desired shape

* Rotation is simply performed by the rotation of hand.
** For support generation, pinching hand must move.

CHAPTER 5.
Results

5.1. Results of Hand motion based Mesh processing

A given 3D model, the Stanford bunny, was processed with hand motion based 3D printing slicer and then 3D printed with the obtained modified 3D model. Four different mesh processing functions were applied: geometric deformation of translation, rotation and scaling, splitting to remove the unwanted region, layflat, and manual support generation, all frequently used or wanted gestures for 3D printing slicers.

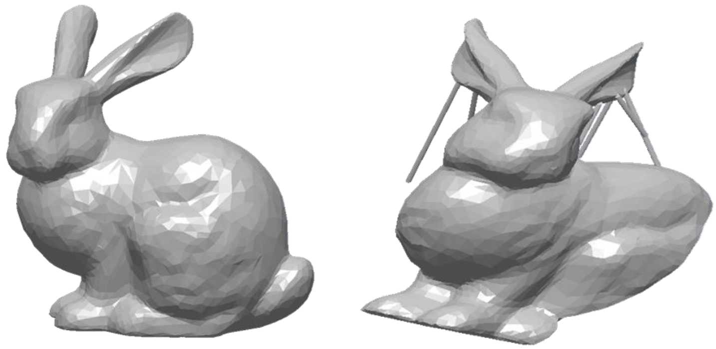


Fig. 5.1 Stanford bunny as an input model (left) and deformed result with the hand motion based 3D printing slicer (right)

5.1.1. Geometric Deformation

Geometric deformation interacts with the 3D model by translating, rotating, and scaling it. Fig. 5.2 shows the procedure and result for each deformation.

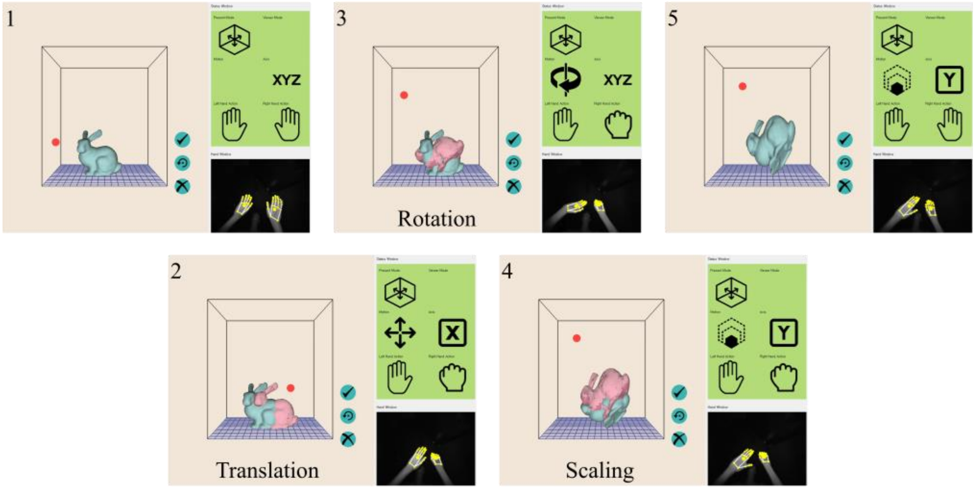


Fig. 5.2 Geometric deformation: translation, rotation and scaling by each hand motion of moving, rotating, and pinching and unpinching

5.1.2. Splitting

In addition, splitting can leave only the desired region of the 3D model by designating the cutting plane that divides the model, as Fig. 5.3 shows.

5.1.3. Layflat

Most of the 3D models have flat bottom surfaces and are attached on the print bed for stable printing. For this reason, slicers provide a layflat function that corrects the model's posture by attaching to the model's surface, which is expected to be the bottom, on the bed. Fig. 5.4 is the result of layflat function with hand motion.

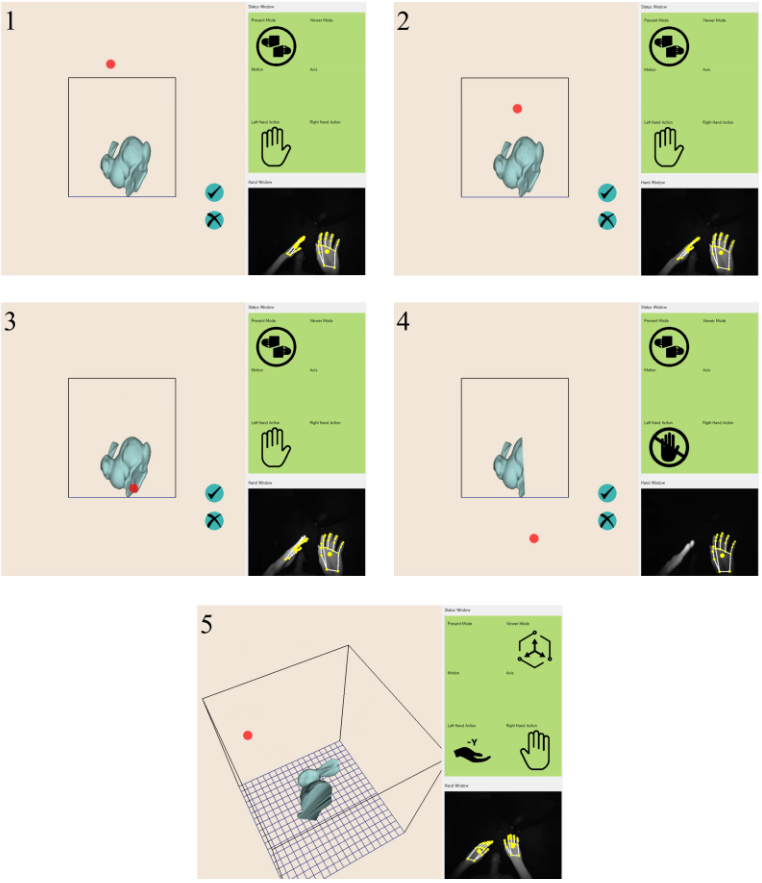


Fig. 5.3 Split of 3D model with a hand moving or a swiping

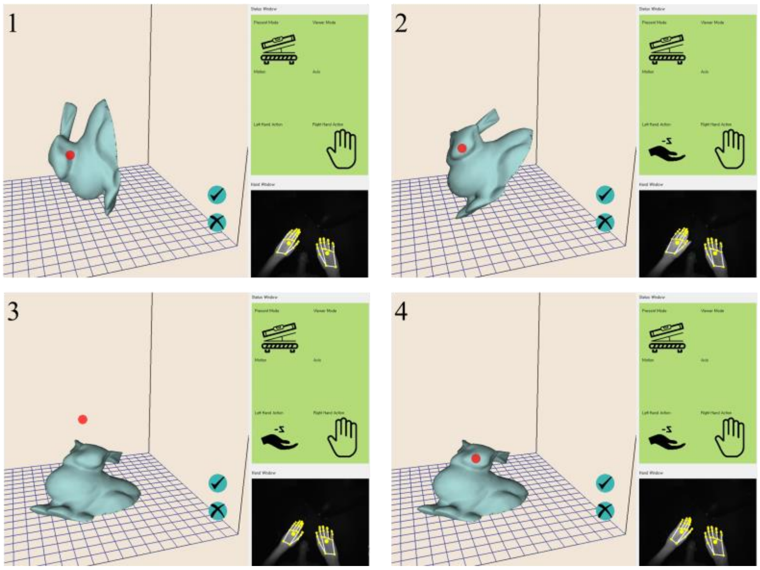


Fig. 5.4 Layflat function with the help of a nodding gesture

5.1.4. Manual support generation

The last mesh processing function for slicer is manual support generation. Support is a structure for sustaining the region that is floating in the air because of its shape while printing. Therefore, automatic or manual support generation is indispensable for a 3D printing slicer. Likewise, if support is needed, a support structure can be created and edited manually with hand motion as shown in Fig. 5.5.

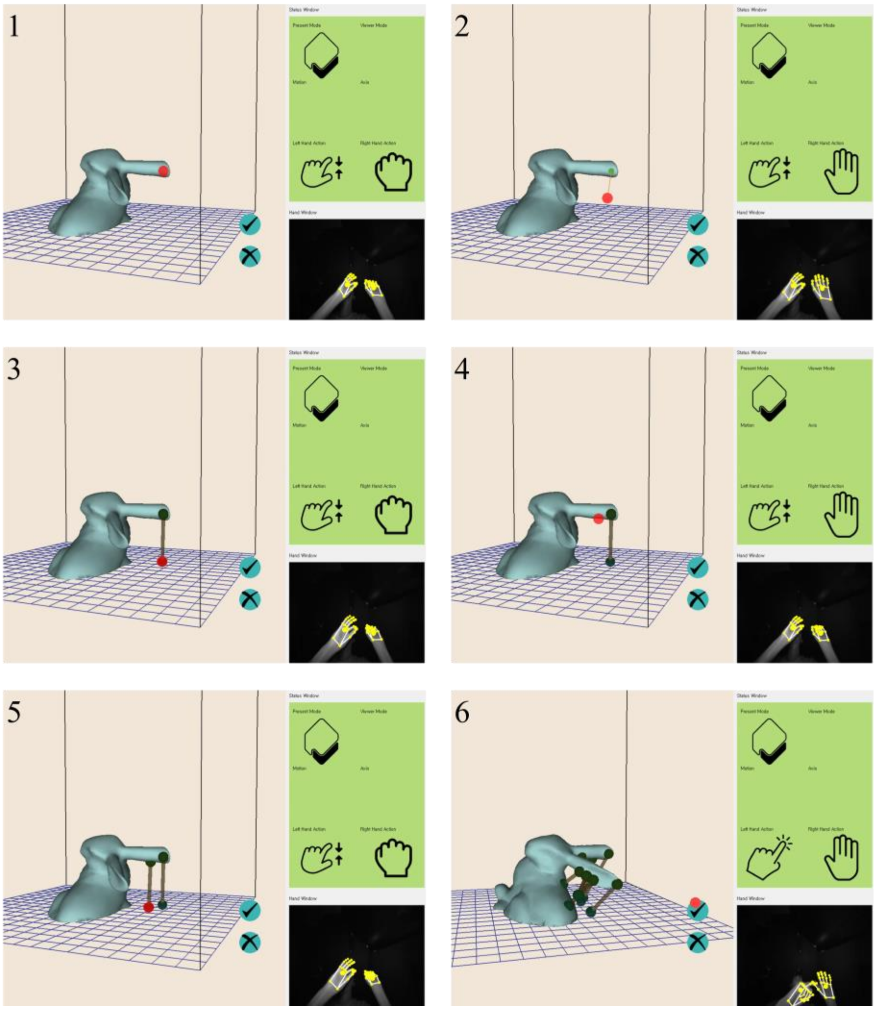


Fig. 5.5 Manual support generation that resembles the interaction with clay

5.1.5. 3D printed model

After the end of the hand motion interaction through previous steps, the deformed 3D model has actually been 3D printed (Fig. 5.6).

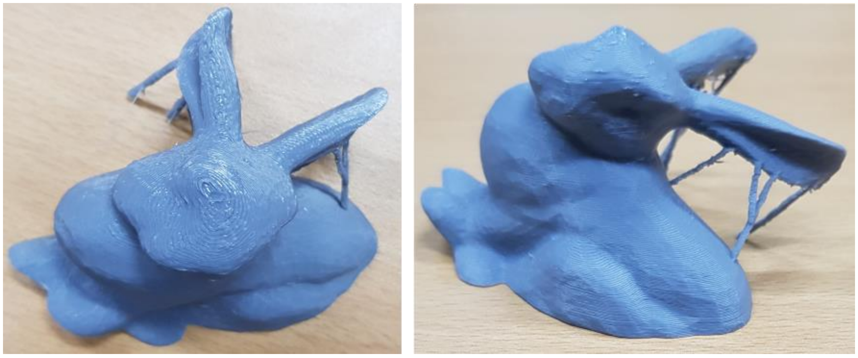


Fig. 5.6 3D printed Standard bunny model deformed by hand motion interaction

5.2. Comparison with 2D based slicer

To verify its functional improvement over existing 2D based slicers, the proposed 3D printing slicer was compared with them. In experiments, above four functions were conducted, which use clearly different manipulation in performing functions with respect to existing slicer. The experiment excluded other functions whose operational details did not differ but just modified the input device from mouse to hand. The slicer used in the comparison is the Cura [62] from Ultimaker and 3D WOX Desktop [63] from Sindoh, and both are common 2D environment slicer programs.

5.2.1. Geometric Deformation

The first comparison experiment was the rotation efficiency of the 3D model. Most slicers can easily change the position and size of the 3D model, but not the rotation. In particular, it was easy to rotate in a single direction along the roll, pitch, and yaw axis. However, in the case of complex rotation, it was difficult to find a desired posture by repeatedly rotating along the roll, pitch, and yaw axis several times. In comparison with conventional manipulation, the normal of the palm was directly reflected to the model, making easy, intuitive control of the proposed slicer possible (Fig. 5.7)

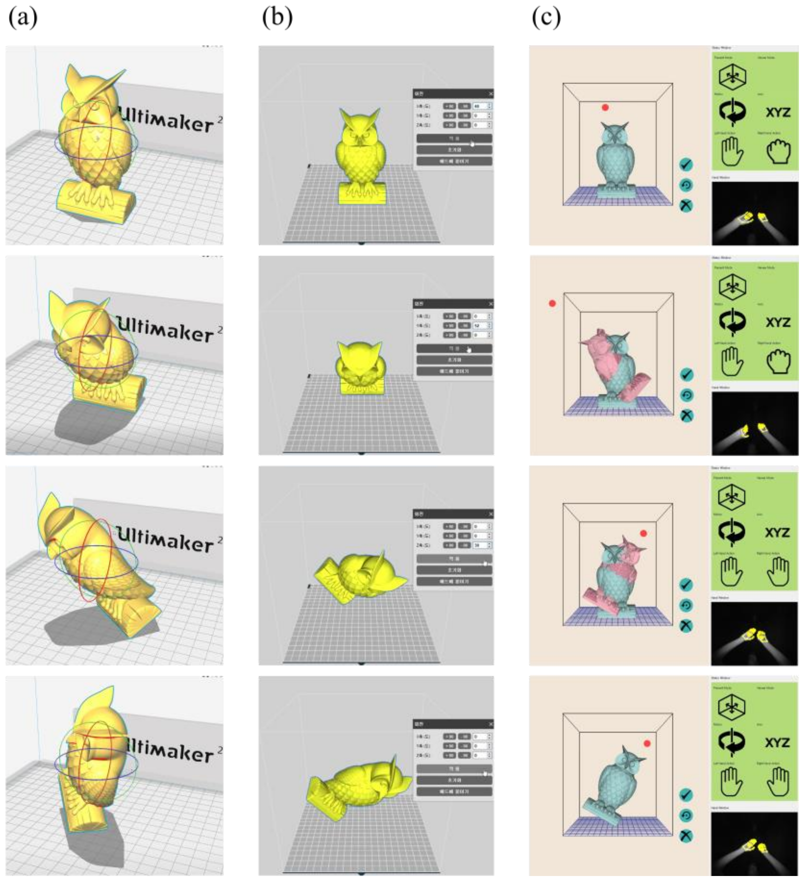


Fig. 5.7 Comparison of 3D model rotation from (a) Cura, (b) 3D Wox Desktop, and (c) proposed slicer

5.2.2. Splitting

The next experiment involved splitting. Comparison was not perfectly possible as most slicers do not support splitting because for 2D-based input devices, specifying the 3D cutting plane requires preconditions. Also, Cura slicer was only able to split the model below the print bed. However, as the hand position is tracked in 3D space, the cutting plane is easily defined according to the movement of the hand. Fig. 5.8 shows how to split the model in the hand motion based slicer.

(a)

(b)

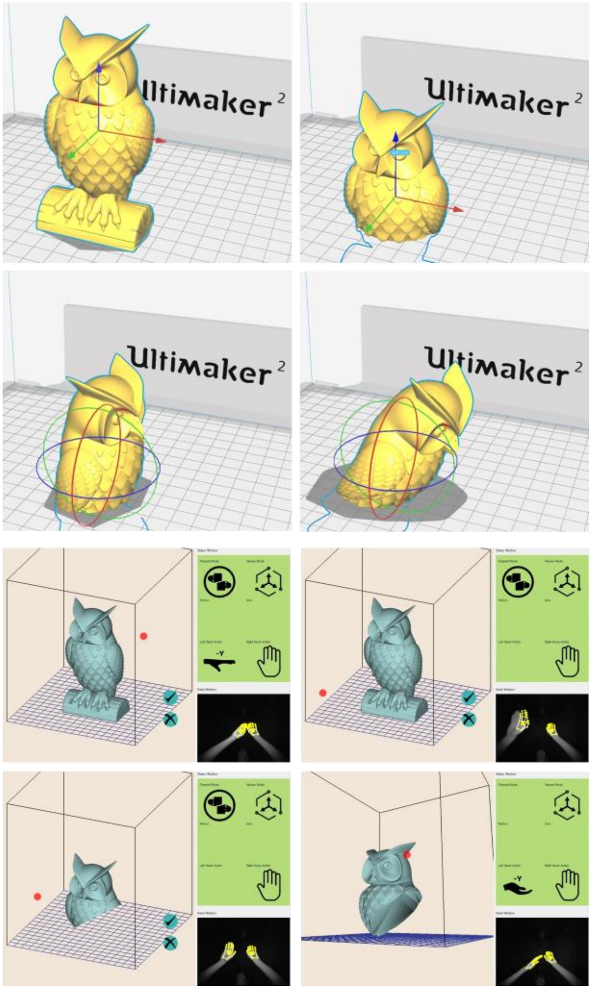


Fig. 5.8 Splitting function experimented in (a) Cura, and (b) proposed slicer. 3D WOX Desktop does not provide splitting function

5.2.3. Layflat

When 3D printing the models, for various reasons users wanted to print them with proper posture, mostly an upright posture. It can be stability, or less support structure for reducing material, printing time, and aesthetic requirements. Thus, finding the most suitable posture for given 3D model and transforming of the posture has been researched [64]. For this reason, the slicer offers the layflat function, which estimates the bottom surface and attaches the bottom to the print bed by rotating the model.

Finding the proper bottom surface is easy with human intuition. However, it is not the case for algorithms. In the extreme case when the model's geometry is complex or the initial pose of the model is far from the upright, selecting the floor of the 3D model with a simple algorithm cannot guarantee a high probability of success. With the proposed slicer, the user performed the layflat function more successfully by giving hint through hand gestures at the orientation to be rotated. Fig. 5.9 and Fig. 5.10 present a comparison of results for the layflat function for a given model and situation.

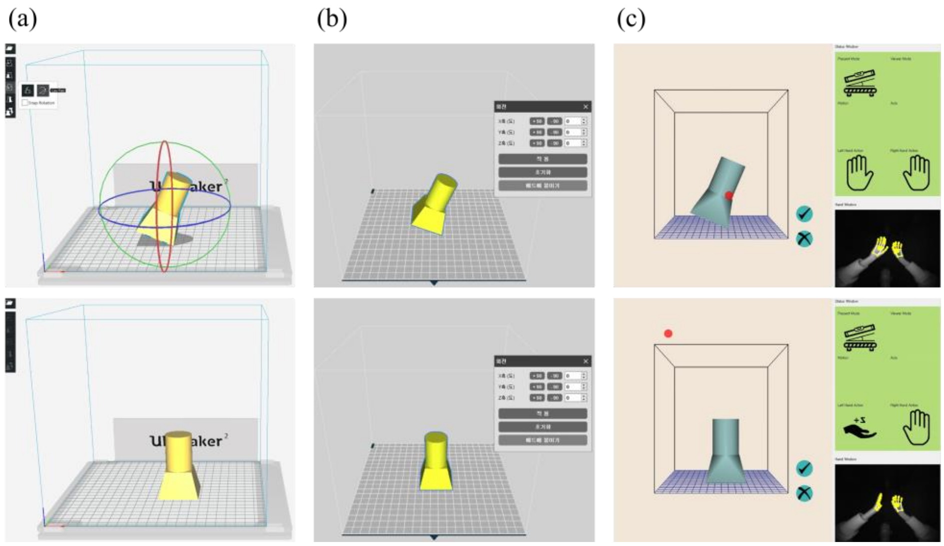


Fig. 5.9 Layflat function for a given model with easy condition: (a) Cura, (b) 3D Wox Desktop, and (c) proposed slicer. All 3D printing slicers succeeded

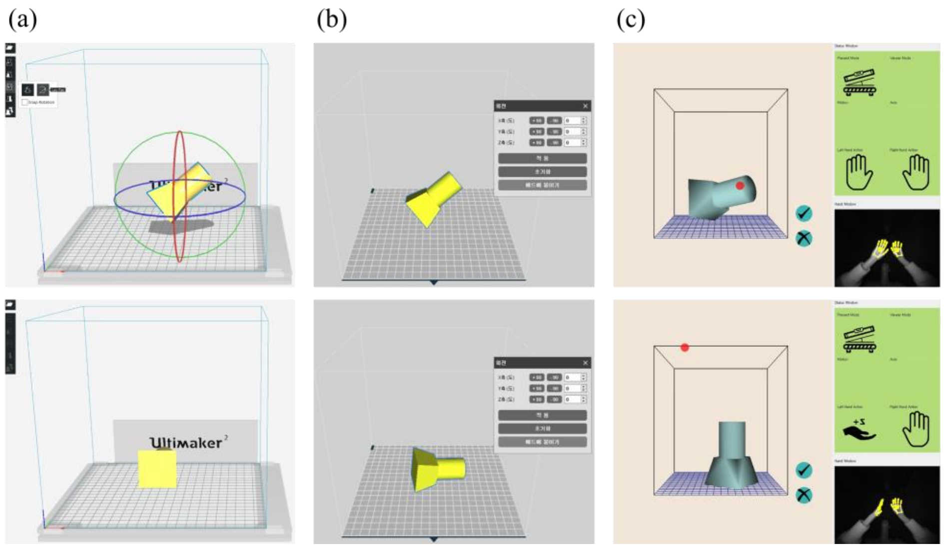


Fig. 5.10 Layflat function for a given model with harsh condition: (a) Cura, (b) 3D Wox Desktop, and (c) proposed slicer. Only proposed slicer succeeded with the help of user intuition, through hand gesture

5.2.4. Manual support generation

Support is an essential structure for stable printing without failure, but it consumes considerable printing time [65]. Therefore, an optimized structure with less amount of support is preferred. However, most of the slicers provide automatic generated support with a predefined grid or line type that is not efficient. Some slicers offer more advanced support editing options such as half-manual support generation that can specify the desired area with a brush tool. However, the support structure’s grid or line type remains inefficient.

In contrast, the proposed slicer can generate support in any position and with any direction through a simple hand gesture, allowing users to freely create efficient support as if they were dealing with clay. For example, slicers usually generate supports between the model and the printing bed. They do not construct supports between 3D model either automatically or manually. Therefore, the proposed slicer can provide a proper method for optimal support that user imagines.

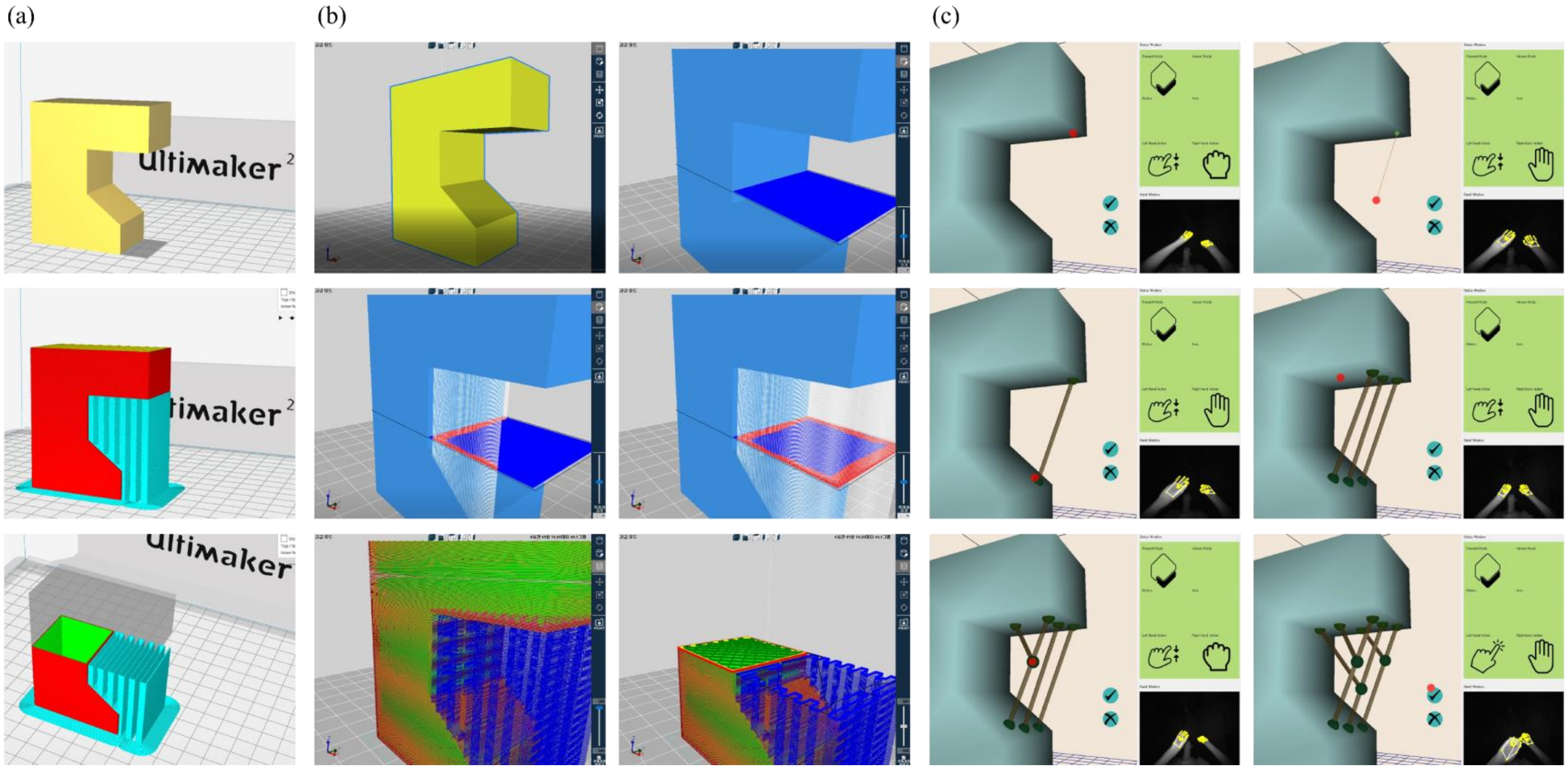


Fig. 5.11 Support generation methods: (a) automatic support generation of Cura, (b) manual brush support generation of 3D Wox Desktop, and (c) manual support generation of proposed slicer. Support from the proposed slicer is more efficient structure with the user intuition.

CHAPTER 6.
Conclusion

This paper proposes a slicer that can interact with a 3D model via the user's hand motion. It recognizes hand gestures based on the hand tracking data collected with Leap Motion, then, executes the mesh processing function connected with the designated hand motion to deform the given 3D model.

In particular, the proposed slicer does more than just modifying already-existing functions for hand control, adding or modifying several functions that can be performed effectively with hand gestures and connecting each function with an intuitive hand gesture.Through palm rotation, changes in normal of palm facilitate the rotation of 3D models. The splitting function, which was difficult to include because of the 2D work environment for determining the cutting plane, is implemented by defining the cutting plane with extracted hand position. Further, the layflat function, which showed limited success rate through an algorithm, was improved through hints with hand gestures. Finally, the proposed slicer allowed users to create their own optimized support directly through hand gesture as if they were dealing with clay.

The hand motion based 3D printing slicer contributed to the construction of an intuitive and easy-to-use 3D modeler controlled with stereoscopic hand gestures as an input. It helped users move away from conventional 2D work environments, which do not work effectively for the 3D model with which users want to interact. At the same time, it also proved that hand motion can be more beneficial in function when the user interacts with 3D models.

Limitation and Future works

However, some aspects of the system could be improved. To reduce the influence of occlusion in hand tracking, constructing a system that tracks hands at various angles or developing a predictive algorithm that can track the hidden hand [66] could be the solution. Moreover, the application of machine learning could improve the accuracy of both hand tracking and hand gesture recognition [67].

Another important future research task to continue is to perform a qualitative test of the proposed slicer. The current research does not evaluate whether a hand motion based slicer is an actually user-friendly interface. Although mesh processing results were better than those of the conventional slicer, it does not mean the true convenience for the user. Therefore, performing a qualitative test on the rationality and usability of the proposed slicer is important as in the case of Song [68] or Hackenberg [69].

Finally, this paper has examined 3D input device of 3D CAD environments using the example of a 3D printing slicer. To build a complete 3D environment for the CAD system, research on 3D display device is also necessary. Application of AR or VR devices [70, 71] or low-level hologram technology could achieve a more realistic and intuitive stereoscopic work environment, and encourage more users to manipulate 3D modeling freely with lower entry barriers.

REFERENCES

[1] Lee, K. (1999). Principles of cad/cam/cae systems.

[2] Mao, B. (2011). Visualisation and generalisation of 3D City Models (Doctoral dissertation, KTH Royal Institute of Technology).

[3] Borosán, P., Jin, M., DeCarlo, D., Gingold, Y., & Nealen, A. (2012). Rigmesh: automatic rigging for part-based shape modeling and deformation. ACM Transactions on Graphics (TOG), 31(6), 198.

[4] Ratner, P. (2012). 3-D human modeling and animation. John Wiley & Sons.

[5] Chaudhary, A. (2017). Robust Hand Gesture Recognition for Robotic Hand Control.

[6] Leap Motion. (2015). https://www.leapmotion.com/

[7] Cheng, H., Yang, L., & Liu, Z. (2016). Survey on 3D hand gesture recognition. IEEE Transactions on Circuits and Systems for Video Technology, 26(9), 1659-1673.

[8] Xu, D., Wu, X., Chen, Y. L., & Xu, Y. (2015). Online dynamic gesture recognition for human robot interaction. Journal of Intelligent & Robotic Systems, 77(3-4), 583-596.

[9] Ballan, L., Taneja, A., Gall, J., Van Gool, L., & Pollefeys, M. (2012). Motion capture of hands in action using discriminative salient points. Computer Vision–ECCV 2012, 640-653.

[10] Wachs, J. P., Kölsch, M., Stern, H., & Edan, Y. (2011). Vision-based hand-gesture applications. Communications of the ACM, 54(2), 60-71.

[11] Francese, R., Passero, I., & Tortora, G. (2012, May). Wiimote and Kinect:

gestural user interfaces add a natural third dimension to HCI. In Proceedings of the International Working Conference on Advanced Visual Interfaces (pp. 116-123). ACM.

[12] DJI Spark (2017), https://www.dji.com/spark

[13] Lins, R. G., Gomes, E. W. M., Corazzim, M., & Beaulieu, A. (2015, April). Development and implementation of a natural interface to control an industrial hydraulic robot arm. In Systems Conference (SysCon), 2015 9th Annual IEEE International (pp. 766-772). IEEE.

[14] Tang, G., Asif, S., & Webb, P. (2015). The integration of contactless static pose recognition and dynamic hand motion tracking control system for industrial human and robot collaboration. Industrial Robot: An International Journal, 42(5), 416-428.

[15] Airò Farulla, G., Pianu, D., Cempini, M., Cortese, M., Russo, L. O., Indaco, M., ... & Vitiello, N. (2016). Vision-based pose estimation for robot-mediated hand telerehabilitation. Sensors, 16(2), 208.

[16] Rautaray, S. S., & Agrawal, A. (2015). Vision based hand gesture recognition for human computer interaction: a survey. Artificial Intelligence Review, 43(1), 1-54.

[17] Wojciechowski, A. (2012). Hand’s poses recognition as a mean of communication within natural user interfaces. Bulletin of the Polish Academy of Sciences: Technical Sciences, 60(2), 331-336.

[18] Krupka, E., Karmon, K., Bloom, N., Freedman, D., Gurvich, I., Hurvitz, A., ... & Bar-Hillel, A. (2017, May). Toward Realistic Hands Gesture Interface: Keeping it Simple for Developers and Machines. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (pp. 1887-1898).

ACM.

[19] Peng, K. C. C., Singhose, W., & Frakes, D. H. (2012). Hand-motion crane control using radio-frequency real-time location systems. IEEE/ASME Transactions on Mechatronics, 17(3), 464-471.

[20] Kundu, A. S., Mazumder, O., Lenka, P. K., & Bhaumik, S. (2017). Hand Gesture Recognition Based Omnidirectional Wheelchair Control Using IMU and EMG Sensors. Journal of Intelligent & Robotic Systems, 1-13.

[21] Fernández, R. A. S., Sanchez-Lopez, J. L., Sampedro, C., Bavle, H., Molina, M., & Campoy, P. (2016, June). Natural user interfaces for human-drone multi-modal interaction. In Unmanned Aircraft Systems (ICUAS), 2016 International Conference on (pp. 1013-1022). IEEE.

[22] Foottit, J., Brown, D., Marks, S., & Connor, A. M. (2014, December). An intuitive tangible game controller. In Proceedings of the 2014 Conference on Interactive Entertainment (pp. 1-7). ACM.

[23] Schkolne, S., Pruett, M., & Schröder, P. (2001, March). Surface drawing: creating organic 3D shapes with the hand and tangible tools. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 261-268). ACM.

[24] Wen, R., Tay, W. L., Nguyen, B. P., Chng, C. B., & Chui, C. K. (2014). Hand gesture guided robot-assisted surgery based on a direct augmented reality interface. Computer methods and programs in biomedicine, 116(2), 68-80.

[25] Sagayam, K. M., & Hemanth, D. J. (2017). Hand posture and gesture recognition techniques for virtual reality applications: a survey. Virtual Reality, 21(2), 91-107.

[26] Miura, T., Yoshii, A., & Nakajima, T. (2016, July). Designing Affordances for

Virtual Reality-Based Services with Natural User Interaction. In International Conference of Design, User Experience, and Usability (pp. 266-277). Springer International Publishing.

[27] Fuge, M., Yumer, M. E., Orbay, G., & Kara, L. B. (2012). Conceptual design and modification of freeform surfaces using dual shape representations in augmented reality environments. Computer-Aided Design, 44(10), 1020-1032.

[28] Lv, Z., Halawani, A., Feng, S., Ur Réhman, S., & Li, H. (2015). Touch-less interactive augmented reality game on vision-based wearable device. Personal and Ubiquitous Computing, 19(3-4), 551-567.

[29] IX, F. D., Qin, H., & Kaufman, A. (2001). A novel haptics-based interface and sculpting system for physics-based geometric design. Computer-Aided Design, 33(5), 403-420.

[30] 3D Systems Haptic devices. https://www.3dsystems.com/scanners-haptics#haptics-devices

[31] De Araùjo, B. R., Casiez, G., & Jorge, J. A. (2012, May). Mockup builder: direct 3d modeling on and above the surface in a continuous interaction space. In Proceedings of Graphics Interface 2012 (pp. 173-180). Canadian Information Processing Society.

[32] Wang, R., Paris, S., & Popović, J. (2011, October). 6D hands: markerless hand-tracking for computer aided design. In Proceedings of the 24th annual ACM symposium on User interface software and technology (pp. 549-558). ACM.

[33] Murugappan, S., Liu, H., & Ramani, K. (2013). Shape-It-Up: Hand gesture based creative expression of 3D shapes using intelligent generalized cylinders. Computer-Aided Design, 45(2), 277-287.

[34] Sculpting (2017), https://gallery.leapmotion.com/sculpting/

[35] Xu, C., Pathak, P. H., & Mohapatra, P. (2015, February). Finger-writing with smartwatch: A case for finger and hand gesture recognition using smartwatch. In Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications (pp. 9-14). ACM.

[36] Asokan, A., Pothen, A. J., & Vijayaraj, R. K. (2016, December). ARMatron—A wearable gesture recognition glove: For control of robotic devices in disaster management and human Rehabilitation. In Robotics and Automation for Humanitarian Applications (RAHA), 2016 International Conference on (pp. 1-5). IEEE.

[37] Hasan, H., & Abdul-Kareem, S. (2014). Human-computer interaction using vision-based hand gesture recognition systems: a survey. Neural computing & applications, 25(2).

[38] Chang, L. Y., Pollard, N. S., Mitchell, T. M., & Xing, E. P. (2007, October). Feature selection for grasp recognition from optical markers. In Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on (pp. 2944-2950). IEEE.

[39] Siam, S. M., Sakel, J. A., & Kabir, M. (2016). Human Computer Interaction Using Marker Based Hand Gesture Recognition. arXiv preprint arXiv:1606.07247.

[40] Schröder, M., Elbrechter, C., Maycock, J., Haschke, R., Botsch, M., & Ritter, H. (2012, November). Real-time hand tracking with a color glove for the actuation of anthropomorphic robot hands. In Humanoid Robots (Humanoids), 2012 12th IEEE-RAS International Conference on (pp. 262-269). IEEE.

[41] Li, Y. (2012, June). Hand gesture recognition using Kinect. In Software

Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on (pp. 196-199). IEEE.

[42] Kim, D., Hilliges, O., Izadi, S., Butler, A. D., Chen, J., Oikonomidis, I., & Olivier, P. (2012, October). Digits: freehand 3D interactions anywhere using a wrist-worn gloveless sensor. In Proceedings of the 25th annual ACM symposium on User interface software and technology (pp. 167-176). ACM.

[43] Yeo, H. S., Lee, B. G., & Lim, H. (2015). Hand tracking and gesture recognition system for human-computer interaction using low-cost hardware. Multimedia Tools and Applications, 74(8), 2687-2715.

[44] Sridhar, S., Oulasvirta, A., & Theobalt, C. (2013). Interactive markerless articulated hand motion tracking using RGB and depth data. In Proceedings of the IEEE International Conference on Computer Vision (pp. 2456-2463).

[45] Nishiyama, M., & Watanabe, K. (2009). Wearable sensing glove with embedded hetero-core fiber-optic nerves for unconstrained hand motion capture. IEEE Transactions on Instrumentation and Measurement, 58(12), 3995-4000.

[46] Kim, J. H., Thang, N. D., & Kim, T. S. (2009, July). 3-d hand motion tracking and gesture recognition using a data glove. In Industrial Electronics, 2009. ISIE 2009. IEEE International Symposium on (pp. 1013-1018). IEEE.

[47] Kim, J., Mastnik, S., & André, E. (2008, January). EMG-based hand gesture recognition for realtime biosignal interfacing. In Proceedings of the 13th international conference on Intelligent user interfaces (pp. 30-39). ACM.

[48] McIntosh, J., McNeill, C., Fraser, M., Kerber, F., Löchtefeld, M., & Krüger, A. (2016, May). EMPress: Practical hand gesture classification with wrist-mounted EMG and pressure sensing. In Proceedings of the 2016 CHI

Conference on Human Factors in Computing Systems (pp. 2332-2342). ACM.

[49] Liu, H., Ju, Z., Ji, X., Chan, C. S., & Khoury, M. (2017). Human hand motion analysis with multisensory information. In Human Motion Sensing and Recognition (pp. 171-191). Springer Berlin Heidelberg.

[50] Tompson, J., Stein, M., Lecun, Y., & Perlin, K. (2014). Real-time continuous pose recovery of human hands using convolutional networks. ACM Transactions on Graphics (ToG), 33(5), 169.

[51] Ahsan, M. R., Ibrahimy, M. I., & Khalifa, O. O. (2011, May). Electromygraphy (EMG) signal based hand gesture recognition using artificial neural network (ANN). In Mechatronics (ICOM), 2011 4th International Conference On (pp. 1-6). IEEE.

[52] Botsch, M., Steinberg, S., Bischoff, S., & Kobbelt, L. (2002). Openmesh-a generic and efficient polygon mesh data structure.

[53] Möbius, J., & Kobbelt, L. (2010). OpenFlipper: An Open Source Geometry Processing and Rendering Framework. Curves and Surfaces, 6920, 488-500.

[54] Leap Motion Developer. (2015). https://developer.leapmotion.com/

[55] Xie, R., Sun, X., Xia, X., & Cao, J. (2015). Similarity matching-based extensible hand gesture recognition. IEEE sensors journal, 15(6), 3475-3483.

[56] Pfister, A., West, A. M., Bronner, S., & Noah, J. A. (2014). Comparative abilities of Microsoft Kinect and Vicon 3D motion capture for gait analysis. Journal of medical engineering & technology, 38(5), 274-280.

[57] Benalcázar, M. E., Jaramillo, A. G., Zea, A., Páez, A., & Andaluz, V. H. (2017, August). Hand gesture recognition using machine learning and the Myo armband. In Signal Processing Conference (EUSIPCO), 2017 25th European (pp. 1040-1044). IEEE.

[58] Weichert, F., Bachmann, D., Rudak, B., & Fisseler, D. (2013). Analysis of the accuracy and robustness of the leap motion controller. Sensors, 13(5), 6380-6393.

[59] Rosa, G. M., & Elizondo, M. L. (2014). Use of a gesture user interface as a touchless image navigation system in dental surgery: Case series report. Imaging science in dentistry, 44(2), 155-160.

[60] Chen, S., Ma, H., Yang, C., & Fu, M. (2015, August). Hand gesture based robot control system using leap motion. In International Conference on Intelligent Robotics and Applications (pp. 581-591). Springer International Publishing.

[61] Xu, P. (2017). A Real-time Hand Gesture Recognition and Human-Computer Interaction System. arXiv preprint arXiv:1704.07296.

[62] Cura. (2012). https://ultimaker.com/en/products/ultimaker-cura-software

[63] 3D WOX Desktop. (2016). http://3dprinter.sindoh.com/Downloads

[64] Fu, H., Cohen-Or, D., Dror, G., & Sheffer, A. (2008, August). Upright orientation of man-made objects. In ACM transactions on graphics (TOG) (Vol. 27, No. 3, p. 42). ACM.

[65] Schmidt, R., & Umetani, N. (2014, July). Branching support structures for 3D printing. In ACM SIGGRAPH 2014 Studio (p. 9). ACM.

[66] Mueller, F., Mehta, D., Sotnychenko, O., Sridhar, S., Casas, D., & Theobalt, C. (2017). Real-time Hand Tracking under Occlusion from an Egocentric RGB-D Sensor. arXiv preprint arXiv:1704.02201.

[67] Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., & Kautz, J. (2016). Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. In Proceedings of the IEEE Conference on

Computer Vision and Pattern Recognition (pp. 4207-4215).

[68] Song, J., Cho, S., Baek, S. Y., Lee, K., & Bang, H. (2014). GaFinC: Gaze and Finger Control interface for 3D model manipulation in CAD application. Computer-Aided Design, 46, 239-245.

[69] Hackenberg, G., McCall, R., & Broll, W. (2011, March). Lightweight palm and finger tracking for real-time 3D gesture control. In Virtual Reality Conference (VR), 2011 IEEE (pp. 19-26). IEEE.

[70] McGraw, T., Garcia, E., & Sumner, D. (2017, July). Interactive swept surface modeling in virtual reality with motion-tracked controllers. In Proceedings of the Symposium on Sketch-Based Interfaces and Modeling (p. 4). ACM.

[71] Maimone, A., Georgiou, A., & Kollin, J. S. (2017). Holographic near-eye displays for virtual and augmented reality. ACM Transactions on Graphics (TOG), 36(4), 85.

초 록

CAD (Computer Aided Design)의 발달로 3차원 모델링은 생산·설계분야뿐만 아니라 건축·토목 및 예술분야에 이르기까지 넓게 확장되었다. 특히, 3D 프린팅과 3D 스캐닝 등 자가 생산 기술이 보급되면서 3차원 모델링은 대중들에게도 친숙해졌다. 하지만 여전히 3차원 모델링은 어려운 작업이라는 인식이 남아있는데, 이러한 편견의 원인 중 하나는 사용자가 다뤄야 할 입체 공간에서의 3차원 모델과 괴리되는 2차원 작업 환경 때문이라 할 수 있다.

본 논문에서는 3차원 모델링에 적합한 직관적이고 간편한 3차원 작업 환경 구축을 목표로 삼았으며 이에 따른 적절한 3차원 입력 도구로써 사용자의 손을 선정하고, 손동작을 통해 동작하는 3D 모델러의 예시로 3D 프린팅을 위한 slicer를 제안하였다. 특히, 사용자의 의도를 왜곡 없이 간단한 동작을 통해 전달할 수 있는 입력 도구로서의 손의 장점을 살려 3차원 모델과의 직관적인 상호작용을 제공하는 것에 초점을 맞추었다.

제안한 3D 프린팅 slicer는 hand tracking 장비인 립 모션을 통해 손과 각 관절들의 주요한 데이터를 수집하고, 이를 기반으로 손동작을 인식하였다. 이후, 인식된 손동작과 연계된 mesh processing 기능이 작동함으로써 3D printing할 모델을 쉽고 간편하게 변형할 수 있었다. 그 결과, 기존의 2D 작업 환경에서 작업하기 까다로웠던 기능을

손동작을 통해 구현할 수 있었고, 단순 알고리즘만으로는 효율적이지 않았던 기존 slicer 기능들을 손동작을 통해 사용자의 직관적인 사고를 전달함으로써 더 효율적인 상호작용이 가능하도록 만들었다. 이에 따라 본 논문은 slicer의 기능을 단순히 손동작으로 구현한 것에 그치지 않고, 3D 모델을 가공함에 있어 사용자의 손동작이 우위를 지닐 수 있음을 보였다.

주요어 : 3차원 모델링, 3D 프린팅 slicer, CAD, 립 모션, 손동작 인식,

Hand motion based interface, Hand tracking,

학번 : 2016-20711