Realizing Multi-Touch-Like Gestures in 3D Space

Chunmeng Lu(✉), Li Zhou, and Jiro Tanaka

Graduate School of Information, Production and System,
Waseda University, Tokyo, Japan
luchunmeng@fuji.waseda.jp

Abstract. In this paper, our purpose is extending 2D multi-touch interaction to 3D space and presenting a universal multi-touch gestures for 3D space. We described a system that allows people to use their familiar multi-touch gestures in 3D space without touching surface. We called these midair gestures in 3D as *3D multi-touch-like gestures*. There is no object or surface for user to touch in 3D space, so we use depth camera to detect fingers' state and estimate whether finger in the "click down" or "click up", which show user's intention to interact with system. We use machine learning to recognize hand shapes. While we do not need to precessing the recognition all the time, we only recognize hand shape between "click down" or "click up".

Keywords: Gesture · Human-computer interaction · Machine learning

1 Introduction

Multi-touch extends the vocabulary of interaction and has been used in many devices, such as smart phone and laptop. People use 2D multi-touch gestures to accomplish interaction with these devices. After learning to use a set of 2D multi-touch gestures, people could perform the same gestures in different kinds of machines. The universality and ease of use make multi-touch interaction become a revolutionary technology.

Future human-computer interfaces will enable more nature, intuitive communication between people and all kinds of sensor-based devices. Gesture-based interaction, as a kind of nature interaction, will be popular in the future [11]. Obviously the present multi-touch gestures have some limitations, because many devices will not provide object surfaces for users to touch, like some VR and AR devices. Thus non-touch interaction system cannot apply the 2D multi-touch gestures directly. In this case, a gesture-based interaction system needs to realize the hand gestures in 3D space.

In our study, inspired by the 2D multi-touch interaction, we intend to design a new system to realize 3D multi-touch-like gestures for 3D space. "Multi-touch-like gestures" here means that the hand shapes and functions of 3D gestures are similar to 2D multi-touch gestures. Our system will allow users to perform

S. Yamamoto and H. Mori (Eds.): HIMI 2018, LNCS 10904, pp. 227–239, 2018.
https://doi.org/10.1007/978-3-319-92043-6_20

their familiar multi-touch gestures in 3D space. In the system, we extend five typical 2D multi-touch gestures to 3D space to become multi-touch-like gestures: zoom in/out, rotate, scroll, swipe and drag, as showed in Fig. 1. People are often comfortable with traditional interaction methods, and our method that extending 2D multi-touch gestures to 3D space will decrease the difficulty and time to learn a new interaction style for people.

In order to recognize these 3D multi-touch-like gestures, we present a method using machine learning. However, only using machine learning is not good enough. It is because machine learning can recognize the hand shape, however, it does not tell when the gestures start and when they end. If we cannot recognize the exact timing of gestures, it makes difficult to return the response to the given gesture in the right timing. So we also need to realize the state of fingers. We use depth camera to detect the state of fingers to realize the starting and ending of gestures.

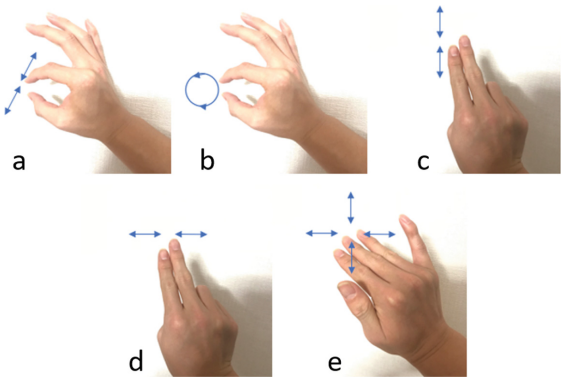

Fig. 1. Five 3D multi-touch-like gestures: (a) zoom in/out, (b) rotate, (c) scroll, (d) swipe and (e) drag

2 Related Works

Gesture-based interaction provides a nature way for user to interact with machines and has been researched for over three decades. There are two kinds of approaches to capture gesture data and recognize gestures: "data-glove based" and "vision-based". When applying data-glove based method, users need to wearing a glove-like device which is equipped sensor to collect data of hand and finger motions [5]. However, the extra devices are not cheap and not convenient. Thus, applying vision-based methods can be a good choose.

Many researches show that using video camera and depth sensor to recognize hand gesture is a effective way [5,9,10]. Wilson [12] presented a novel touch screen technology. They use two video cameras to capture hand and process the images. In their system, the a pair of video cameras could get depth information of hand gesture and give feedback for interface. Boussemart et al. [3] present a

framework for 3D visualization and manipulation in an immersive space. Their work can be used in AR and VR system. Wachs et al. [11] summarizes the requirements of hand-gesture interfaces and the challenges when applying hand gestures in different application. They divide these applications into four classes: medical systems and assistive technologies; crisis management and disaster relief; entertainment and human-robot interaction. These applications reveal the rich vocabulary of hand gesture.

In our system, the important part is realizing the state of fingers. This part will tell system when to start analyze hand gestures and when to end. There are also previous works to research this issue. Karam et al. [6–8] carry out a series of studies about finger click detection in 3D space. They present two-hand interactive menu and introduce how to use depth based selection techniques [6]. They design a selection mechanism approach, "three fingers clicking gesture", to detect intention from user [7]. Their approach is using depth camera to capture hand shapes and analyze the relative positions of the fingers. They present "Xpli Click" to improved midair finger click detection method [8]. The "Xpli Click" allow user to perform more nature click action in their system.

3 System Design

Our system combines theses three parts to realizing multi-touch-like gestures in 3D space:

(1) Mechanical click detection.
(2) Hand shape recognition.
(3) Motion recognition.

In the first part, mechanical click detection, system will detect whether user want to interact with machines. In the second part, hand shape recognition, system will realize which hand shape that user performs. And in the third part, motion recognition, system will recognize the movement of fingers to find which gesture that user want to use.

3.1 "Click Down" and "Click Up" in 3D Space

In the case of 2D gesture, we start gesture by touching the screen and end gesture by leaving hands. Similarly, there is a click action in mouse gesture.

Therefore, we need to catch the timing of when the gesture starts and ends in our 3D gestures. We introduce click/touch action in our 3D gestures for such purpose. In 2D multi-touch, system could recognize gestures by detecting click action and movements of fingers on surface. While in 3D space, there is no midair physical surface for user to click.

In order to detect the timing of starting and ending of 3D gestures in 3D space, "click state" and "neutral state" are defined to show whether user want to perform gestures [8]. In 3D space, user's hand moves freely in "neutral state" when having no intention to interact with computer. As for "click state", we

define “click down” and “click up” to distinguish the states. “Click down” means that user has the intention to interact with computer and fingers are getting into “click state”. “Click up” means fingers are going back to “neutral state”.

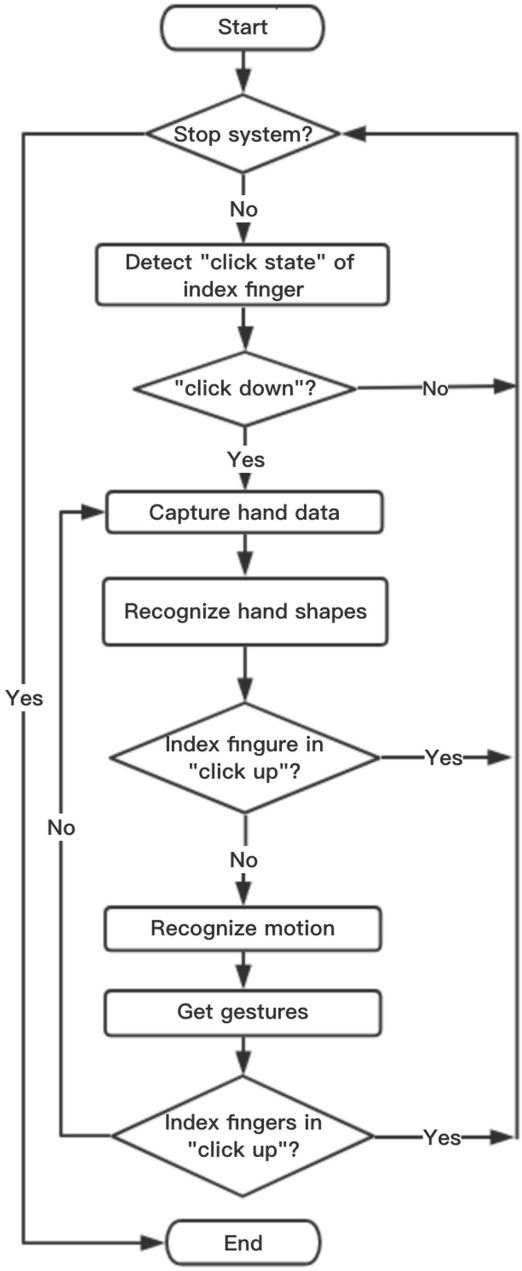

Fig. 2. System overview

3.2 System Overview

From Fig. 1, we could find that index finger is used in these five gestures that we need to realize. It means that we could only detect the state of index finger to know whether user is performing a gesture.

The system is designed to follow these steps, as showed in Fig. 2:

(1) System detects the "click state" of index finger all the time.
(2) Once detecting the "click down", system will capture hand data and start machine learning to recognize hand shape.
(3) If getting a hand shape of one of the five gestures, system will capture the motion of the fingertips and get gestures until detecting index fingers in "click up" or stopping system. If not, system will continue recognize hand shape until getting a hand shape of the five gestures or detecting index finger in "click up" or stopping system.

4 Mechanical Click Detection

In our system, we detect "click down" and "click up" of index finger by mechanical calculation. We could calculate the angle θ related with palm center and index finger, as showed in Fig. 3. There is a range of the angle θ when index finger in "click down" [8]. So, we could calculate angle θ to confirm whether index finger is in "click down" or "click up".

The mechanical calculation follows these steps:

(1) Use Leap Motion as depth sensor to capture the coordinates of fingertip of index finger and palm center.
(2) Create the vector from palm center to fingertip of index finger basing on the coordinates.
(3) Think of the palm as a plane and calculate the angle θ between palm and the vector in step (2).
(4) If the angle in the range, index finger is in "click down". Otherwise, index finger is in "click up".

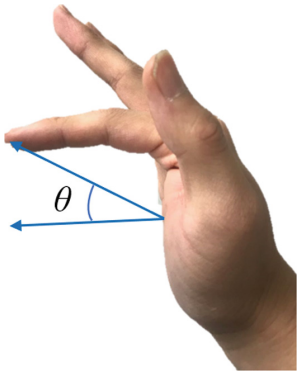

Fig. 3. Angle θ

5 Hand Shape Recognition

Multi-touch-like gestures start from "click down" and end at "click up". Between them, there are hand shape performing and movement. Hand shape and fingers movements actually decide gestures.

System needs to classify the hand shapes. We use SVM (Support Vector Machine) to achieve the purpose.

Adapting SVM method to our system, we design four steps for hand shape recognition:

(1) data collection
(2) normalization and scale
(3) model training
(4) predicting.

5.1 Data Collection

In the first step, data collection, we need to decide what kind of data of hand that we need to capture.

In a hand of human, there are nineteen bones related to five fingers. We could use the endpoints of these bones and palm center as "key points" to describe these hand shapes, as showed in Fig. 4. There are total 26 key points. Then we use depth camera to capture the coordinates of key points of user's hand.

Fig. 4. 26 Key Points: two blue points represent palm center and wrist joint; red points represent the endpoints of bones in hand (Color figure online)

In one frame of Leap Motion, all coordinates of key points become a group of original data. As we use Leap Motion, the center of Leap Motion is the origin of coordinate system, and the data shows the key points' positions relative to the center of Leap Motion. Using the data, we could draw the skeleton of hand.

5.2 Normalization and Scale

In 3D space over Leap Motion, hand moves around freely. Thus the data of hand shape can be captured in any position in that 3D space. So we can make the palm center as the origin coordinate. Then we calculate other key points positions relative to palm center.

Data normalization follows these steps:

(1) Translate all the points until the palm center is on the origin coordinate.
(2) Rotate the points around the palm center until the palm parallel to the x-z axis plane.
(3) Rotate again the points around the y coordinate axis until the palm points the - z axis.

There is still a problem that the sizes of hand model are different because of the different distances between hand and Leap Motion. So we scale all the data to $[-1, 1]$. After that, we also could use the new data to draw the skeleton of hand and exclude the effect of Leap Motion's position. This step will eliminate noise and improve accuracy.

5.3 Model Training

The third step of hand shape recognition is model training.

Our work is to recognize 5 kinds of gestures: zoom, rotate, scroll, swipe and drag. While the zoom and rotate gestures have the same hand shape when moving, we mark this hand shape as hand shape 1. And there is the same situation with scroll and swipe, and we mark this hand shape as hand shape 2. We also mark the hand shape of drag as hand shape 3. So we only need to classify 3 kinds of hand shapes.

There are many SVM models and we use Classification SVM Type 1 (also known as C-SVM classification) in our system. As there are three kinds hand shapes, we use one-vs.-one (OvO) method to classify them. In OvO method, we need to get three classifiers: classifier 1 for hand shapes 1 and 2, classifier 2 for hand shapes 1 and 3, and classifier 3 for hand shapes 2 and 3. Then combining the classifiers, we will get a 3-label classifier model.

There are three steps when applying SMV method: (1) making training set; (2) getting 3-label classifier model through training; (3) predicting hand shapes with trained model in our system. In this three parts, making training set and getting 3-label classifier model through training have to be finished before building our system.

Here are the steps to get the 3-label classifier model:

(1) Capture 50 groups coordinates of the key points for every hand shape.
(2) Normalize and scale the data and get 3 groups training sets.
(3) Through training sets, get classifier 1, classifier 2 and classifier 3.
(4) Combine this three classifiers to get a 3-label classifier model.

The model training part is preparation work for system. The 3-label classifier model will be used to realize hand shapes in time in our system.

5.4 Predicting

The final part of hand shape recognition is predicting.

We use 3-label classifier model to predict hand shapes. The 3-label classifier model works following these steps:

(1) Input a hand shape.
(2) Start classifier 1, 2 and 3 in order. Then we will get three result numbers of labels of hand shapes.
(3) Count the times of occurrences of the labels.
(4) Put out the highest times of occurrences of label, and label shows which gesture is performed.

In this part, we collect 20 data groups of every hand shapes to test the accuracy of the 3-label classier model. The total 60 data groups become the testing set.

The predicting part plays an important role in our system. In the system, once detecting "click down", system collects a group of hand data immediately. Then using 3-label classifier model that we get in model training part, predict part will tell system what hand shape that this captured group of data represents.

6 Motion Recognition

After knowing the hand shape, the system needs motion recognition because gestures are defined by both hand shape and hand motion together.

Once getting the hand shape, system starts motion recognition step. After the frame when hand shape recognized, system will continuously calculate and record the positions of fingers in every frame captured by depth camera. For different hand shapes, system detects motion of different fingertips to realize gestures, as showed in Fig. 5.

In case of knowing the hand shape 1, system will detects movements of thumb and index finger. We create a vector v from thumb fingertip to index fingertip. Firstly, we calculate and record the v in the frame where system gets hand shape 1. We mark this original v as v_0. After that, in every frame, We compares v with v_0. If v is longer/shorter than v_0, gesture will be zoom in/out and system calculates the distance d between the two fingers. If System also calculate the angle between v and v_0 in every frame. If fingertips of thumb and index finger rotate around a center, the angle between v and v_0 will change continuously and system will realizing gesture Rotate.

In case of knowing the hand shape 2, system will detects movements of index finger and middle finger. In every frame, system calculates the coordinate of the midpoint of index finger and middle finger. We mark this midpoint as p_0. System calculates the relative distance d_0 between the p_0 in present frame and the p_0 in the frame when system gets hand shape 2. The d_0 will be used for interaction. Besides, system could realize movement directions through comparing the coordinates the p_0 in present frame and the p_0 in previous frame. If p_0 moves

up or down, the gesture can be scroll; if p_0 moves left or right, the gesture can be swipe; if p_0 moves to other directions, system will not give response.

In case of knowing the hand shape 3, system will detects movements of index finger, middle finger and ring finger. In every frame, system calculates the coordinate of the midpoint of index finger, middle finger and ring finger. We mark this midpoint as p_1. System calculates the relative distance d_1 and realize the movement direction p_1 through comparing the p_1 in present frame and the p_1 in the frame where system gets hand shape 3. d_1 and p_1 will be used for interaction.

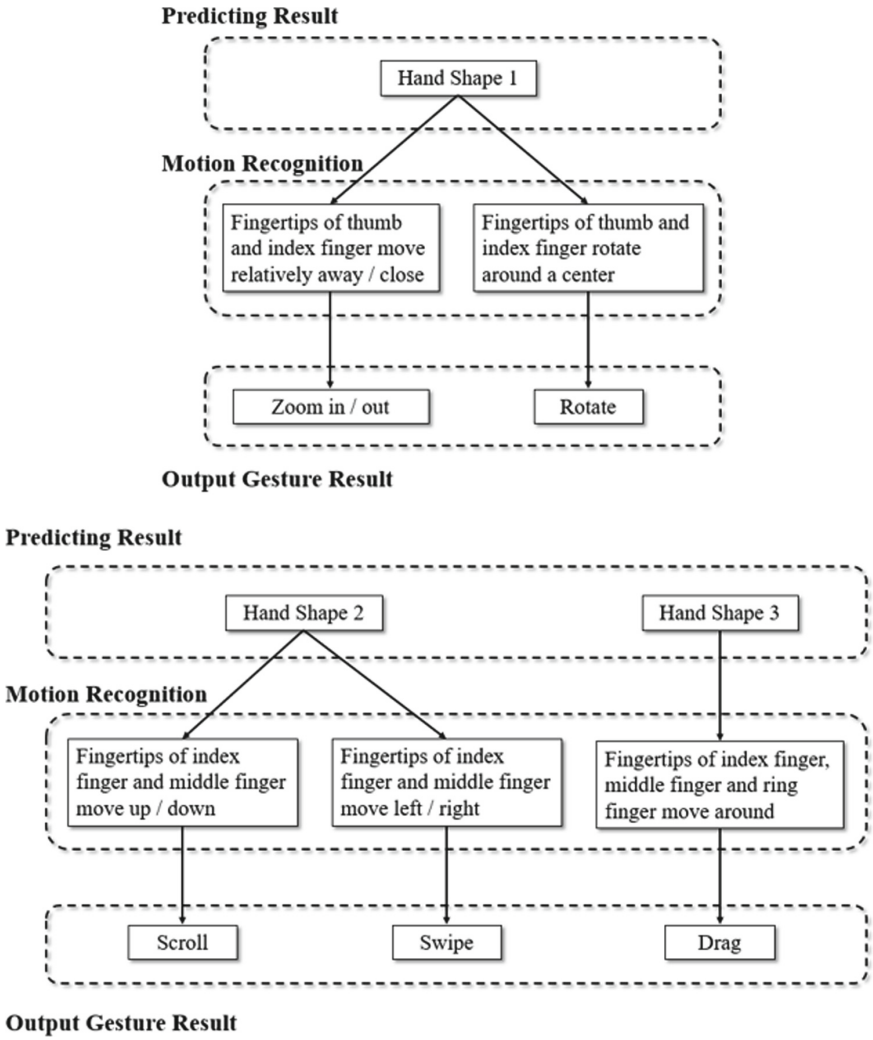

Fig. 5. Motion recognition and getting output

7 Implementation

7.1 Hardware Overview

In our system, we use Leap Motion as the depth camera [1]. Leap Motion could recognize and track hands and fingers in a 3D space. The 3D space is over the sensors and the best scope of sensor is 25 mm to 60 mm. The average frame rate of sensor is about 90 fps. We use APIs from Leap Motion to recognize the joints of hands, which allow us to collect data of key points in system. Leap Motion works with a Thinkpad laptop (Intel(R) Core(TM) i7-6500 CPU @2.5 GHz, RAM 8 GB, 64 bit windows 10).

7.2 Software Overview

We use Qt [2] as the development environment to build our system. Qt provides a cross-platform software development environment to create innovative devices, modern UIs and applications for multiple screens. This character will help us apply our system in other devices.

In our system, we use LIBSVM [4] as the machine learning tools in hand shape recognition part. LIBSVM is an open source library for support vector machines. We combine the open source codes written by C++ and Leap Motion APIs in Qt development environment to design the system, experiences and application.

8 Evaluation

8.1 Accuracy Rate of Click Detection

In system, hand shape recognition starts when detecting "click down" of index finger and ends when detecting "click up" of index finger. Therefor accuracy of click detection decides the usability of our system.

When interacting with system, hands will perform click action in state of stopping and in state of motion. Besides, different users are habituated to different speeds. Considering these two factors, we design a experience to evaluation the accuracy of click detection. In the experience, we choose three click speeds: low speed is one click per two second, normal speed is one click per second and fast speed is two clicks per second. Here are the experience steps:

(1) User move hand around in a 3D space over depth sensor and make click actions with index finger randamly for 100 times in fast speed, 100 times in normal speed and 100 times in low speed.
(2) User stops his hand over Leap Motion and make click actions with index finger randomly for 100 times in fast speed, 100 times in normal speed and 100 times in low speed.

The Accuracy rate are showed in Table 1.

Table 1. Test accuracy rate of click detection

State	Moving			Stopping		
Click speed	Fast	Normal	Low	Fast	Normal	Low
Accuracy rate	96%	98%	98%	99%	100%	100%

8.2 Accuracy Rate of Hand Shape Recognition

The system needs to start SVM method recognize the hand shapes after detecting "click down" of index finger. So there are two situations that recognizing hand shape of a hand in state of motion and in state of stopping.

Here are the experience steps:

(1) A user moves his hand around in a 3D space over the depth sensor and perform the three hand shapes. He needs to perform every hand shape for 100 times.
(2) User stops his hand over depth sensor and perform every hand shape for 100 times.

The accuracy rates of hand shape recognition are showed in Table 2.

Table 2. Test accuracy rate of hand shape recognition

State	Moving			Stopping		
Accuracy rate	93%	96%	95%	95%	98%	96%

8.3 Accuracy Rate of Realizing Multi-Touch-Like Gestures in 3D Space

As explained in motion recognition part, we need to realizing the five gestures basing on movement trajectories. We designed a experience to test the accuracy rate of realizing the five gestures.

In this experience, user put his hand with neutral stat over the sensor. Then he performs the five gestures: zoom in/out, rotate, scroll, swipe and drag. User needs to perform every gestures for 100 times one by one. As for zoom in/out, user performs zoom in and zoom out for 50 times respectively. In every times, the system outputs which gesture it captures. Table 3 shows the accuracy rate of realizing these five multi-touch gestures.

8.4 Simple Application

We design a picture viewer to test the usability of our system. As showed in Fig. 6, we place Leap Motion on the touchpad of a laptop to replace touchpad and mouse. User performs 3D gestures over Leap Motion without touch anything.

In this simple application, the five typical gestures: zoom in/out, rotate, scroll, swipe and drag are applied in the application to control the picture:

Table 3. Accuracy rate of realizing five gestures

Gestures	Zoom in	Zoom out	Rotate	Scroll	Swipe	drag
Accuracy rate	96%	95%	95%	98%	98%	98%

(1) When user performs gesture zoom in/out, system will detect the distance d between the thumb fingertip and index fingertip. In every frame, system will calculate the difference value between d in present frame and d in the frame when system gets hand shape. As the window size of picture viewer is not big, in this application, if the difference value is extended or reduced by $x$cm, the length of diagonal of picture in viewer will be extended or reduced $2x$cm.
(2) When user performs gesture rotate, picture will rotate following the angle between v and v_0 in every frame, which is described in Motion Recognition part.
(3) When user performs gesture scroll, swipe or drag, picture will move following the d_0 or d_1, which is described in Motion Recognition part. If d_0 or d_1 is extended or reduced by $x$cm, the picture in viewer will move $x$cm to corresponding direction.

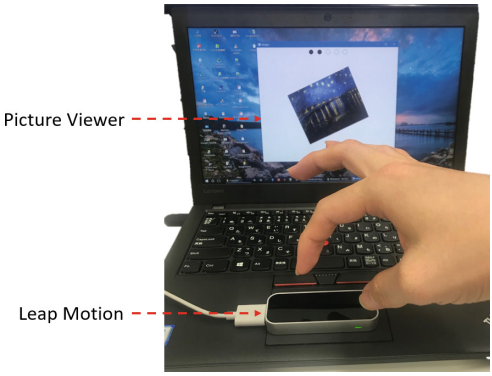

Fig. 6. Picture viewer: user is performing rotate gesture

9 Conclusion

With the rapid development of 3D technology, many researchers design new 3D gestures for their own system. Thus, people need to learn different gestures in different systems. In this situation, a universal 3D gesture set is need for 3D interaction system. In this paper, the experiences and simple application reveal the availability of our idea that expands the vocabulary of 2D multi-touch gestures to 3D space to improve non-touch interactions. In recent years,

augmented reality and virtual reality become more and more popular. In future work, we will apply the multi-touch-like gestures in these fields to enhance the interaction with virtual environment.

References

1. Leap Motion. https://www.leapmotion.com. Accessed 5 Feb 2018
2. Qt. https://www.qt.io. Accessed 5 Feb 2018
3. Boussemart, Y., Rioux, F., Rudzicz, F., Wozniewski, M., Cooperstock, J.R.: A framework for 3D visualisation and manipulation in an immersive space using an untethered bimanual gestural interface. In: Proceedings of the ACM Symposium on Virtual Reality Software and Technology, VRST 2004, pp. 162–165. ACM, New York (2004)
4. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 27:1–27:27 (2011)
5. Garg, P., Aggarwal, N., Sofat, S.: Vision based hand gesture recognition. World Acad. Sci. Eng. Technol. **49**(1), 972–977 (2009)
6. Karam, H., Tanaka, J.: Two-handed interactive menu: an application of asymmetric bimanual gestures and depth based selection techniques. In: Yamamoto, S. (ed.) HCI 2014. LNCS, vol. 8521, pp. 187–198. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07731-4_19
7. Karam, H., Tanaka, J.: Finger click detection using a depth camera. Procedia Manuf. **3**, 5381–5388 (2015)
8. Karam, H., Tanaka, J.: An algorithm to detect midair multi-clicks gestures. Inf. Media Technol. **12**, 340–351 (2017)
9. LaViola, Jr., J.J.: An introduction to 3D gestural interfaces. In: ACM SIGGRAPH 2014 Courses, SIGGRAPH 2014, 42 p. ACM, New York (2014)
10. Lee, U., Tanaka, J.: Finger identification and hand gesture recognition techniques for natural user interface. In: Proceedings of the 11th Asia Pacific Conference on Computer Human Interaction, pp. 274–279. ACM (2013)
11. Wachs, J.P., Kölsch, M., Stern, H., Edan, Y.: Vision-based hand-gesture applications. Commun. ACM **54**(2), 60–71 (2011)
12. Wilson, A.D.: TouchLight: an imaging touch screen and display for gesture-based interaction. In: Proceedings of the 6th International Conference on Multimodal Interfaces, ICMI 2004, pp. 69–76. ACM, New York (2004)