

A Hand Gesture Recognition Library for a 3D Viewer Supported by Kinect's Depth Sensor

Khoa Thi-Minh Tran¹ and Seung-Hyun Oh^{2*}

*Department of Computer Science, Dongguk University, Gyeongju, South Korea
{ttmk84¹, shoh²}@dongguk.ac.kr*

Abstract

In modern life, human–computer interaction has received the most interest from researchers, especially in developing new interaction methods. Microsoft's Kinect sensor, which has integrated red-green-blue and depth (RGB+D) cameras, opens up new possibilities. As well, many three-dimensional (3D) applications have been developed fashioning the computer world more natural and real for the user. In this paper, we propose and develop a 3D viewer application for heritage, which we call an HT3DViewer, using information received from the RGB+D camera of the Microsoft Kinect sensor. This viewer provides a hand gesture recognition library for users to control the object in the viewer by hands. Hand gestures are detected and defined using tracked information from the Kinect depth camera. The prototype of our hand gesture recognition library and 3D viewer application was built using the Microsoft Kinect software developer's kit with C# programming language on the Microsoft .NET platform. Moreover, we investigate and select a simple and fast process to generate 3D models of heritage items from 2D images captured by cameras.

Keywords: *Kinect sensor, depth camera, hand gesture recognition, 3D viewer, 2D-to-3D conversion*

1. Introduction

Human-computer interaction (HCI) is an area of research on planning and design the interaction between people and computers. Its aim is to improve the interactions between users and computers by making computer more usable to users' needs. However, traditional computer input devices, such as the keyboard, mouse, or touch screen, are not very effective in human–computer interaction. They only exchange small amounts of information between people and computers. Hence, it is left to researchers in discovery new input methods to interact between people and computers. The most user-friendly method is users use their own actions to interact with the computers. In modern life, the use of human gestures to control the computer is one of the principle topics in HCI. Several approaches to gesture recognition have been developed [1]. Microsoft's Kinect is a motion sensing input device for the Microsoft (MS) X-box 360 video game console. With the integration of low-cost sensors such as high-resolution depth and visual sensing, the Kinect has become available for widespread use as an off-the-shelf technology [2]. Moreover, the Kinect camera opens

* Corresponding author

up new kinds of HCI methods, such as tracking [3, 4]; reconstruction [5, 6]; voice, gesture, motion and activity detection and recognition [7-10], and so on.

Nowadays, more and more three-dimensional (3D) applications have been developed for the computer, and 3D technology allows the computer to offer more reality to users who want to see the world through their computers. The 3D technique displays three full dimensions, increasing information about 3-dimensional objects being displayed by observer head and eye movements. The 3D technique is also left many challenges to scientists creating, generating 3D models of real objects as well as developing more natural and effective 3D applications for computers, television, smartphone, and so on.

In this paper, we propose a 3D viewer application for controlling 3D heritage objects, which we call the Heritage Tourism 3D Viewer (HT3DViewer) and which supports a hand gesture recognition library. A hand gesture library is developed by using user's tracked information from Kinect device. The 3D viewer is used to show high-quality 3D images of real heritage objects which are generated by our suggested method. To interact with HT3DViewer, a user use their hands to control a high-quality 3D model of a heritage item to observe it. There are some approaches and techniques for generating a 3D model [10-12]. However, we investigate a more simple and faster process to generate a 3D model from series of 2D images using existing applications instead of proposing a new algorithm for 3D creation. Our study more focuses on develop hand gesture recognition library using user tracked information from Kinect depth sensor.

Our paper contains five main sections. Section 2 covers some related research. In Section 3, we discuss our 2D-to-3D conversion process, building a hand gesture recognition library, and developing the Heritage Tourism 3D Viewer. Experiment results and discussions are presented in Section 4. Finally, conclusions and future work are presented in Section 5.

2. Related Research

In this section, we briefly discuss some research papers related to the Microsoft Kinect sensor. In study [2], the authors gave an overview of other researchers working with the Kinect and its applications for reconstruction and synthesis, detection, tracking and recognition, processing and estimation.

Paper [6] addresses the problem of detecting human postures. The paper consists of three main parts. First, authors propose a new method to measure the reliability of the tracked body parts by Kinect sensor. Second, they use the reliability measurement to suggest a new method to correct broken postures with a motion database. Third, their proposed framework is possible to utilize Kinect for application such as real-time sport training, in which the users usually need to interact with large external objects. They also show in their experimental results that their method performs effectively under extreme situation, such as when a large portion of the body is being occluded.

Other authors introduced a method for real-time gesture recognition from a noisy skeleton stream extracted from the Kinect depth sensor [8]. They proposed a scheme for gesture recognition based on decision forests. The decision forest is learned during a training phase. They also used a decision state machine to provide natural and robust temporal alignment. Their results show that the whole process is robust, even with a noisy depth-based skeleton. However, this method met with some limitations, such as generating different skeletons for different individuals performing the same pose, and composing gestures of distinctive, key poses.

Some researchers discussed the design of, and investigation into, a human gesture recognition system that uses a Microsoft Kinect sensor and a neural network to recognize gestures in a 3D space [10]. Those authors manually collected historical data of gesture

motions for training and testing the neural network. Gesture recognition and interpretation are performed by using a trained neural classifier in two ways: single-hand motion gestures are captured in free 3D space, and people’s head coordinates in 3D are used as reference points for recorded hand gestures. Their experiment results showed that when a human head is used, the gesture recognition system provides better recognition quality, as well as more correct recognition, than with single-hand tracking. However, some gestures were difficult to recognize because of the wide variety of individual human motions used in the study.

Creating an efficient and enhanced 3D image reconstruction system based on the combination of higher quality images from a webcam and faster computation of depth information from the Kinect input device is another proposed method [11]. These authors used the Kinect for 3D depth mapping, along with 3D geometrical theory, to construct real-world objects in 3D models from 2D images captured by cameras. They combined the use of the Kinect with a high-definition webcam to deliver high-quality 3D image reconstruction for real-time video streaming. Then, the high-quality 3D image was used for hand tracking and finger detection and recognition.

3. Heritage-Tourism-3D-Viewer, Hand Gesture Recognition Library, and 2D-to-3D Conversion

The Heritage Tourism 3D Viewer is an alternative method for interaction between humans and computers. Instead of using a mouse or keyboard as the input device to interact with the computer, the HT3DViewer lets people control and observe a heritage item or site using their hand gestures that are recognized by the depth sensor of the Kinect device. Consequently, it is necessary to generate high quality 3D images of the heritage objects and build a hand gesture recognition library to control those 3D images. In this section, we briefly discuss our HT3DViewer development process.

Figure 1 shows the overall development process.

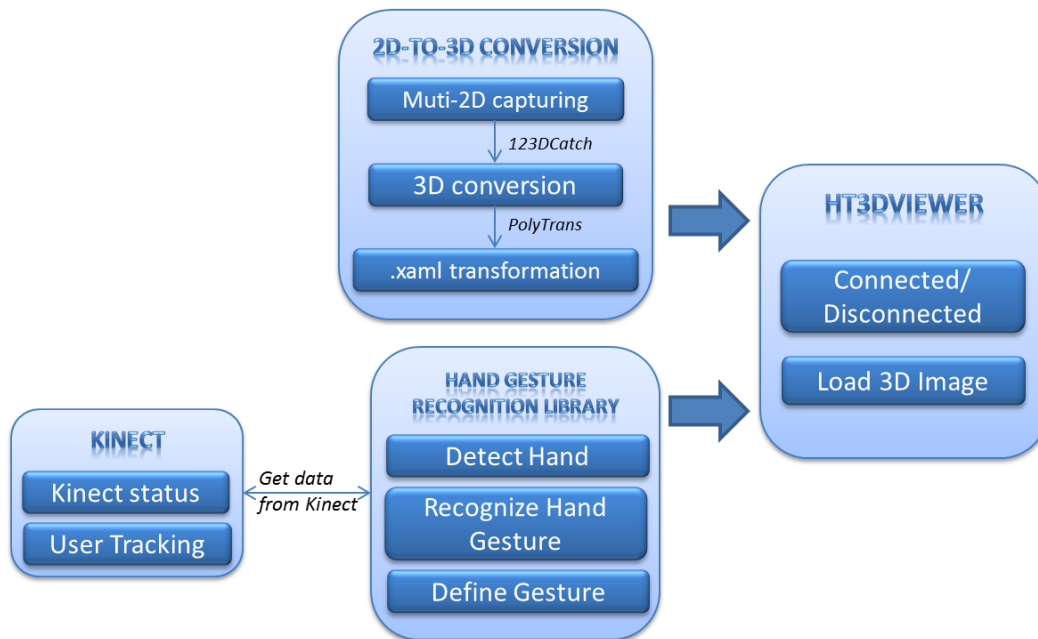


Figure 1. HT3DViewer developing process

Our development process consists of three sub-processes: 2D-to-3D conversion, a hand gesture recognition library, and the HT3DViewer development process. The details of each sub-process will be explained in sections 3.1, 3.2, and 3.3.

The Kinect device is connected to the computer to track the user status and activities, especially the gesture and position of hands. The hand gesture recognition library and the HT3DViewer are built using the Microsoft Kinect software developer's kit (SDK) and the C# programming language on the Microsoft .NET platform. The hand gesture recognition library is developed to detect and recognize every hand gesture of a user when the Kinect is connected and start to record the user activities. Then, all recognized hand gestures are defined in the library for later use. The HT3DViewer displays 3D images of heritage sites and items. A user controls and observes a heritage site or item (moving it, rotating it, zooming in/out on it, etc.) using their hands. For the 2D-to-3D conversion process, we consider to utilize a simple process for generating such 3D images. Our suggested process consists of three small steps: capture, conversion, and transformation. To keep the time for this conversion process but still generate high quality 3D images, we have used existing applications for interaction between each pair of steps.

3.1. The 2D-to-3D Image Conversion Process

In this subsection, we describe our suggested process for generating a 3D image of a heritage item or site. We combined the use of some existing applications instead of implementing a new algorithm or application. It is not only a simple process that can be done easily by anyone, but it also saves time in creating a high quality 3D image from ordinary images (2D images). We can generate the 3D image of any heritage item or site if we can capture around it. Moreover, transforming the 3D image format to XAML (Extensible Application Markup Language) one significantly reduces the size of the 3D image when we import it to the viewer application.

The 2D-to-3D image conversion process consists of three main steps: capturing a series of 2D images of a heritage object, converting to 3D images from series of 2D images, and transforming 3D image file to XAML file. Figure 2 shows an example of our 3D image making process:

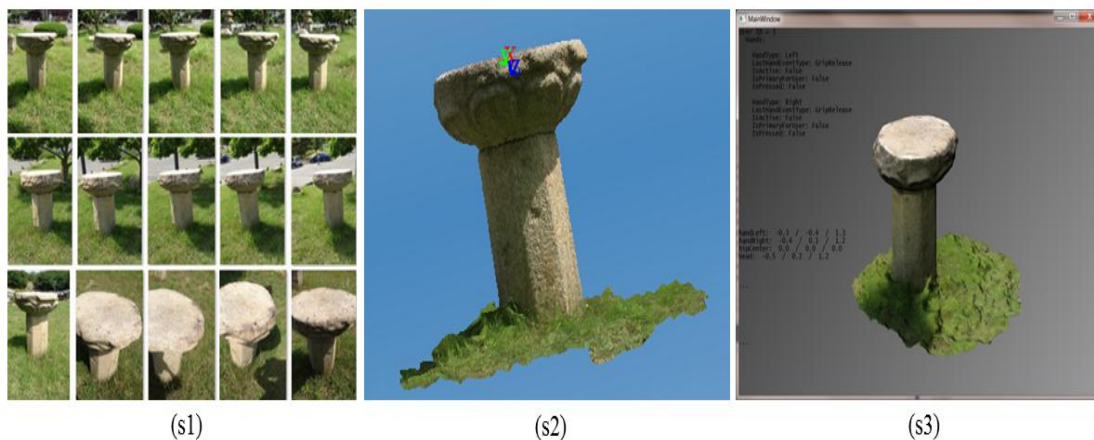


Figure 2. The 3D image conversion process: (s1) ordinary pictures captured by camera, (s2) 3D image view in 123DCatch, (s3) 3D image view in HT3DViewer by loading XAML file

(s1) Capturing a heritage object: For each heritage object, we capture various ordinary images around it. The more images are captured, the higher the quality of the 3D image that will be generated.

(s2) Converting to a 3D image: All ordinary images captured in (s1) are used to create 3D images by using 123DCatch application [13]. The 123DCatch software is an application to easily turn ordinary images into extraordinary 3D models. Simply use any camera to capture places, people, and things and turn them into 3D images. However, this application limits the number of ordinary photos to 40. In order to get more beautiful 3D image, user can use 123DCatch to modify the 3D image such as delete unneeded parts, reset the coordinates, and so on.

(s3) Transforming to XAML: In order to import the 3D images into the HT3DViewer, we first transform the .3dp format files (generated from step s2) to XAML. The tool we used to switch between the two different file types is PolyTrans, which is a commercial tool developed by Okino Computer Graphics [14]. This tool also supports conversion between many file types. We can also configure the parameters inside the tools in order to get an appropriate XAML file.

3.2. Hand Gesture Recognition Library









To control the HT3DViewer, we proposed and developed a hand gesture recognition library. This library allows users to control the viewer by using their hands instead of a mouse or keyboard. The Microsoft Kinect SDK does not include a hand gesture detection engine. It is left to developers to define their own engine using information tracked by the Kinect sensor. The hand gesture detection engine was built on an algorithmic approach—a simple process of defining rules and conditions that must be satisfied to produce the result. In this study, we have built our own hand gesture library and various hand gestures have been defined in the library, such as hands gripped, hands released, and hands swipe. Moreover, we combine hand gesture recognition and hand position with hand movement in order to expand the number of commands. It means one control command is not depend on one function, but the corporation of two or three functions. This makes a command more accuracy.

The positions of the user hands are recorded as a 3-dimensional vector so we can simply determine the distance between the hands or between the hands and the Kinect device. To be tracked, users need to stand within a radius of 2 meters in front of the Kinect device. Although our application can track many users at the same time if they all stand within the track area, only the user standing closest to the Kinect device can control the viewer. This is because all information of all users is being recording by Kinect.

Table 1 lists all states of hand gestures that can be captured and the commands to control the current 3D image of the HT3DViewer. For example, a user stands in front of the Kinect device and raises two hands in front of the body to start interact with the application. When a user wants to move the 3D image, the left hand is placed over the head and the right hand is moved up, down, left, or right. The 3D image will be moved in the direction of the right hand. The 3D image is zoomed in when the right hand is pushed toward the Kinect device and zoomed out when the right hand is pulled backward to the user. In those above cases, the state of two hands is open/released state.

While the two hands are gripped, a user can move the right or left hand up/down or toward/backward in order to rotate the displayed 3D image.

Table 1. Hand gesture recognition library

Name	Description	Explanation
Start		Hands in front of the body
Move up		Hands released Left hand over head Right hand moving up
Move down		Hands released Left hand over head Right hand moving down
Move left		Hands released Left hand over head Right hand moving left
Move right		Hands released Left hand over head Right hand moving right
Zoom in		Hands released Left hand over head Right hand moving toward the Kinect device
Zoom out		Hands released Left hand over head Right hand moving backward to user
Rotate up		Hands gripped Left hand moving up Right hand moving down

Rotate down		Hands gripped Left hand moving down Right hand moving up
Rotate left		Hands gripped Left hand moving toward the Kinect device Right hand moving backward to the user
Rotate right		Hands gripped Left hand moving backward to the user Right hand moving toward the Kinect device

3.3. Heritage Tour 3D Viewer Development

The HT3DViewer is built using the latest Kinect SDK version 1.7 with the C# language on the Microsoft .NET platform. It is a WPF application that displays the heritage item or site in 3D form and allows users to control the 3D image with hand gestures. Each 3D image of the heritage object is described in an .XAML file (generated from the 2D-to-3D conversion process). Then, we load all the .XAML files into the viewer. So, we can add any 3D image of a heritage object to the HT3DViewer if we have its .XAML files.

The hand gesture recognition library was developed with the Microsoft Kinect SDK (section 3.2) and is imported into the viewer. Each recognized hand gesture corresponds to a command implemented to control the object in the viewer. For example, when hands are first gripped and then the left hand moves towards the Kinect while the right hand moves backwards towards the user, the 3D image rotates to the left (around the y-axis). Or, when hands are released and then the left hand is placed over the head, the right hand is pushed toward to the Kinect device to zoom in the displayed 3D object.



Figure 3. HT3DViewer application screenshot

Figure 3 shows one of the screenshots from the HT3DViewer after the user chooses what object to be observed. The 3D image is loaded and can be moved around, rotated, or zoomed in on within the display screen. The upper left corner shows user information (primary status, hands status, etc.), the information of multi-users also can be shown when they are being tracked. Lower down shows the positions (3-dimensional vector) of the primary user's hands and information about the skeleton, distance measures between user hands or between the user and the Kinect device, and so on. As we mentioned above, our application also tracks and displays information of many users at the same time. However, we only the primary user (closest to the Kinect) has priority to control the viewer. When users change their position, their priority also is changed.

4. Experiment Results and Discussion

In this work, we have captured several heritage sites around our university, following the 2D-to-3D conversion process to generate 3D images and transforming them to XAML formats. Then, we imported them into the HT3DViewer. We can add any heritage item or site to the viewer if we can capture its likeness. In this experiment, we only captured some small and medium-sized heritage objects. Figure 4 shows several execution screens from the HT3DViewer. You can see all the 3D heritage objects are displayed in high-quality. However, in Figure 4f and Figure 4g, the tops of the objects are not displayed because they were too high for us to capture.

To observe and control the 3D image, a user must stand within a radius of 2 meters in front of the Kinect device and follows instructions on using the hand gesture recognition library to control the 3D image which is currently shown in the viewer, for example, move the 3D object around within the viewer, zoom the 3D object in and out, or rotate the 3D object left/right or up/down.

Figure 5a shows the zoom-out state with the 3D image. To obtain this state, a user opens both hands, places the left hand above the head, and moves the right hand backward towards the body. Figure 5c is the zoom-in state for the 3D image. To do this, the user first opens both hands, then replaces the left hand above the head, and moves the right hand towards the Kinect device. We estimated and set the initial position of the user's hands—the position of the hands when a user wants to be recognized. Therefore, the 3D image is not affected when a hand from any position moves back to the initial position.

In Figure 5b, the 3D image has been moved down and to the right. There are two ways for the user to move the 3D image to this position. First, the user moves the right hand down from the initial position to move the 3D image downwards, and then directly moves the right hand to the right to move the 3D image to the right side. In the second way, the user moves the right hand down, and back to the initial position, and then to the right; the 3D image will be moved to the lower right side of the application screen.

When both hands are gripped, users move the left hand up and the right hand down to rotate the 3D image up, as seen in Figure 5d. Similarly, the 3D image is rotated down when both hands are gripped, then the left hand is moved down, and then both hand move up, as seen in Figure 5e. The rotate up and rotate down functions in HT3DViewer correspond to the rotate functions around the x-axis in the background. In Figure 5f, the 3D image is being rotated around the y-axis. The HT3DViewer displays the 3D image being rotated left and right. These functions are called when both hands are gripped, the left hand moves toward the Kinect device (or backward to the user), and the right hand moves backward to the user (or toward the Kinect device).

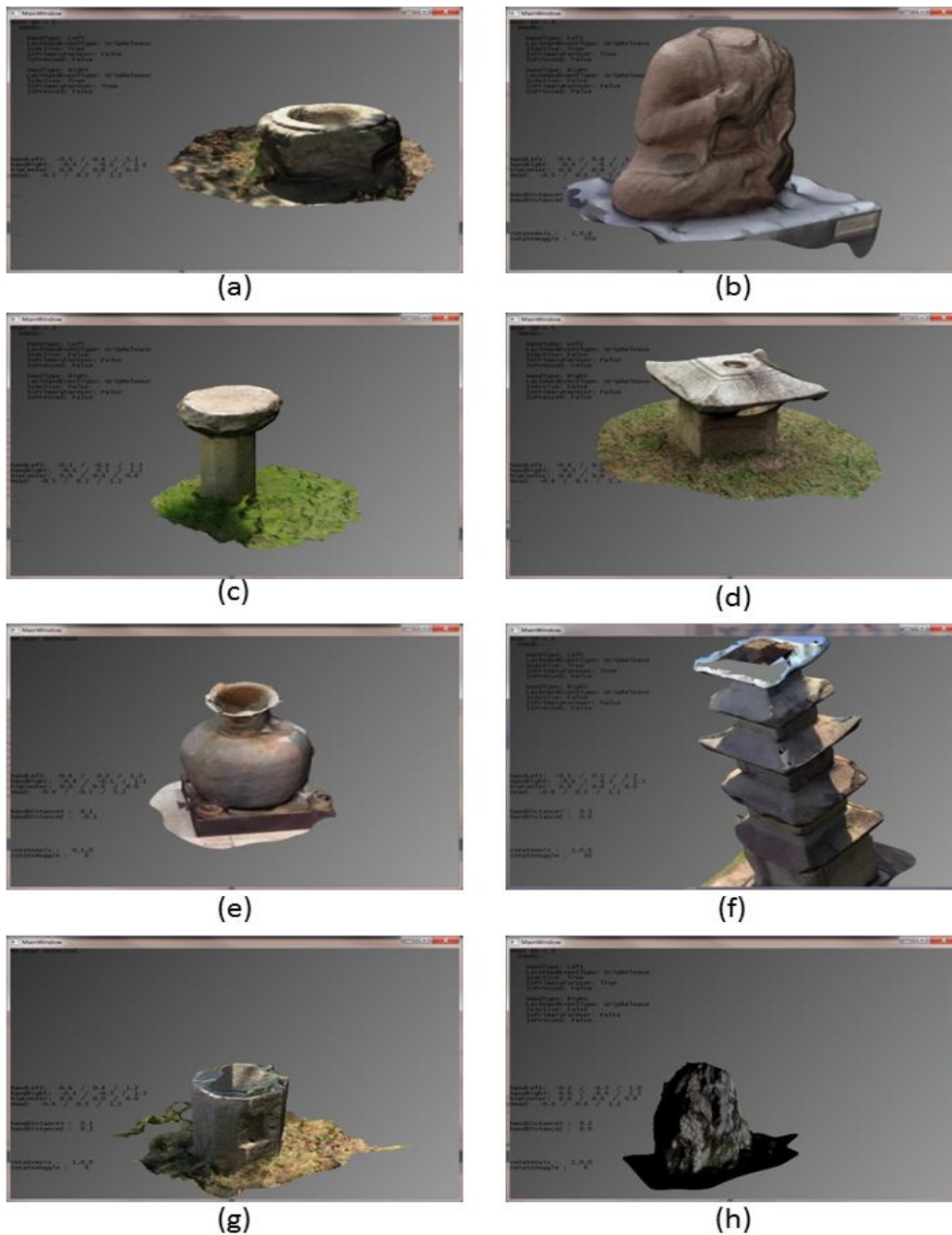


Figure 4. Several screenshots of HT3DViewer application

The three-dimensional coordinates system provided by the Kinect consists of x-axis directions to the right of the user, the y-axis directed up, and the z-axis directed towards the user. However, the three-dimensional coordinates system of the 3D image is not exactly the same. It was skewed during the capturing, converting, and transforming processes but not significantly. Hence, the viewer motion was not smooth enough during object control in some cases.

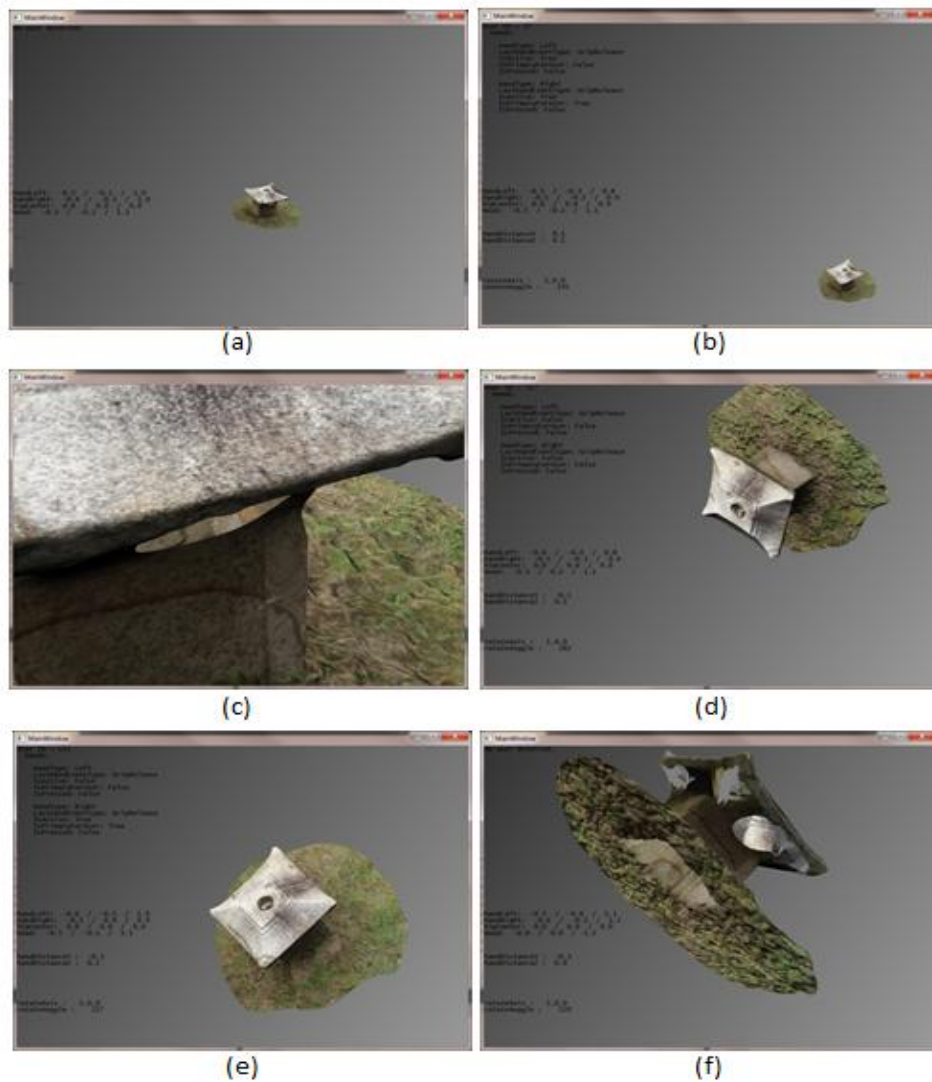


Figure 5. Screenshots of controlling a 3D image

5. Conclusions and Future Works

In this paper, we studied how to use the Microsoft Kinect RGB+D sensors to recognize hand gestures, create a hand gesture recognition library, and use this library to support controlling a 3D image in viewer. We also worked with user tracking information received from the Kinect, extracted coordinates information of the screen, identified the positions of the user's hands, and measured the distances. Then, we implemented a 3D viewer application in which user can use basic hand gestures to control the viewer easily. In some cases, the viewer motion was not smooth enough during object control because the three-dimensional coordinates system of the Kinect and the 3D image are not exactly the same.

In future work, we intend to extend the hand gesture recognition library by adding some new hand gestures. We will also improve the effectiveness and smoothness of our viewer by research on how to match the two three-dimensional coordinates systems. Also, we are going to finding another simple 2D-to-3D conversion process to deal with some limited we have met during using existing applications.

References

- [1] M. Kolsch and M. Turk, "Fast 2D Hand Tracking with Flocks of Features and Multi-Cue Integration", IEEE Conference on Computer Vision and Pattern Recognition Workshop, (2004), pp. 158-165.
- [2] L. Shao, J. Han, D. Xu and J. Shotton, "Computer vision for RGB-D sensors: Kinect and its applications [special issue intro.]," IEEE Transactions on Cybernetics, vol. 43, no. 5, (2013), pp.1314-1317.
- [3] J. L. Raheja, A. Chaudhary and K. Singal, "Tracking of Fingertips and Centers of Palm Using KINECT", 2011 Third International Conference on Computational Intelligence, Modeling and Simulation (CIMSIM), (2011), pp. 248-252.
- [4] S. Monir, S. Rubya and H. S. Ferdous, "Rotation and scale invariant posture recognition using Microsoft Kinect skeletal tracking feature", 2012 12th International Conference on Intelligent Systems Design and Applications (ISDA), (2012), pp. 404-409.
- [5] S. Kim and J. Kim, "Occupancy Mapping and Surface Reconstruction Using Local Gaussian Processes With Kinect Sensors", IEEE Transactions on Cybernetics, vol. 43, no. 5, (2013), pp. 1335-1346.
- [6] H. P. H. Shum, E. S. L. Ho, Y. Jiang and S. Takagi, "Real-Time Posture Reconstruction for Microsoft Kinect", IEEE Transactions on Cybernetics, vol. 43, no. 5, (2013), pp. 1357-1369.
- [7] V. Tam and L. -S. Li, "Integrating the Kinect camera, gesture recognition and mobile devices for interactive discussion", 2012 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE), pp.H4C-11-H4C-13, (2012).
- [8] L. Miranda, T. Vieira, D. Martinez, T. Lewiner, A. W. Vieira and M. F. M. Campos, "Real-Time Gesture Recognition from Depth Data through Key Poses Learning and Decision Forests", Graphics, Patterns and Images (SIBGRAPI), Conference on 25th SIBGRAPI 2012, (2012), pp. 268-275.
- [9] B. Megha D, "Human activity recognition using body pose features and support vector machine", 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI), (2013), pp. 1970-1975.
- [10] K. Rimkus, A. Bukis, A. Lipnickas and S. Sinkevicius, "3D human hand motion recognition system", The 6th International Conference on Human System Interaction (HIS) 2013, (2013), pp. 180-183.
- [11] J. Weidi, W. -J. Yi, J. Saniie and E. Oruklu, "3D image reconstruction and human body tracking using stereo vision and Kinect technology", IEEE International Conference on Electro/Information Technology (EIT), (2012), pp. 1-4.
- [12] Z. Ruifang and L. Chengda, "3D model generation of complex objects from multiple range images", 2011 International Conference on Electric Information and Control Engineering (ICEICE), (2011), pp. 1-4.
- [13] Autodesk 123, <http://www.123dapp.com/catch>.
- [14] PolyTrans, <http://www.okino.com/conv/conv.htm>.

Authors



Khoa Thi-Minh Tran

She received the Bachelors of Science in Information Technology from University of Science, Ho Chi Minh City, Viet Nam in 2008 and completed her Master in Computer Networking in 2012 from Donguk University, Gyeongju Campus, South Korea. Currently she is a Ph.D graduate student at Dongguk University, Gyeongju Campus, South Korea. Her research interests are in the area of communication protocols in Underwater Wireless Sensor Networks. During her PhD course she has joined a project for Kinect technology.



Seung-Hyun Oh

He received the B.S. degree in computer science from Dongguk University, Seoul, Korea, in 1988, and the M.S. and Ph.D. degrees in computer engineering from Dongguk University, Seoul, Korea, in 1998 and 2001, respectively.

He has been a Professor with the Department of Computer Engineering, Dongguk University Gyeongju Campus since 2002. Since 2011, he is serving as Chairperson of the Department of Computer Engineering, Dongguk University Gyeongju Campus. His current research interests include wireless communications and sensor networking system.