

# Scan Modeling: 3D Modeling Techniques using Cross Section of a Shape

Tatsuhito Oe, Buntarou Shizuki, and Jiro Tanaka

University of Tsukuba

Tennoudai 1-1-1, Tsukuba, Ibaraki, Japan 305-8571

{tatsuhito,shizuki,jiro}@iplab.cs.tsukuba.ac.jp

## ABSTRACT

In clay modeling, a creator makes a model by using the shapes of objects, including hands. In contrast, in traditional 3D modeling environments, shapes are assigned by the systems a priori, i.e., no real world object's shape is used. In this paper, we present *Scan Modeling*, in which the creator performs 3D modeling by scanning any real object. To realize Scan Modeling, we developed an input device called “Wakucon”. Wakucon is square-shaped with 245 millimeters per side. The creator uses it to scan a cross section of a shape by placing it inside of the device. By moving the Wakucon or the objects inserted into the device spatially while scanning these objects, the creator can perform 3D modeling. In this paper, we show interaction techniques and implementation of Scan Modeling. Additionally, we present an evaluation of the accuracy of re-constructed 3D models when using the Wakucon.

## Author Keywords

3D modeling; 3D user interface; input device; cross section; shape reconstruction; virtual reality; rapid prototyping; interaction techniques.

## ACM Classification Keywords

H.5.2. User Interfaces: Input Devices and Strategies; H.5.1. Multimedia Information Systems: Artificial, Augmented, and Virtual Realities

## General Terms

Design; Human Factors.

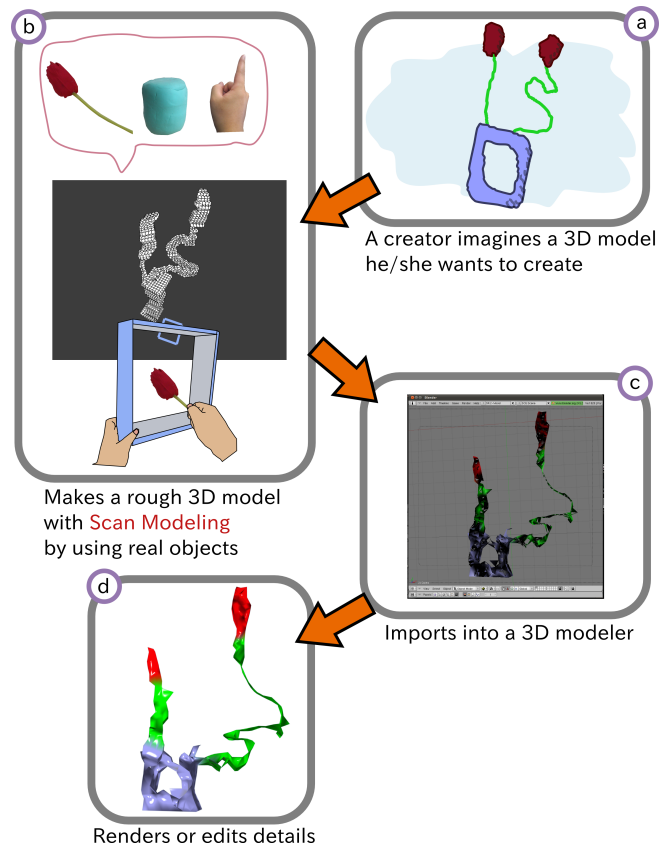
## INTRODUCTION

In clay modeling, which is used as a method for rapid prototyping, a creator makes a model by using the shapes of objects, including hands. Examples of such objects are tools such as spatulas, knives, and prick punches and materials such as wood, leaves, and metals. By using these objects, which include hands, the creator shaves, marks, and smoothes the model. Note that these modifications are performed by direct contact of an object's or hand's shape to the model, i.e.,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

APCHI'12, August 28–31, 2012, Matsue-city, Shimane, Japan.

Copyright 2012 ACM 978-1-4503-1496-1/12/08...\$15.00.



**Figure 1.** A typical 3D modeling process using Scan Modeling. a: The creator imagines a 3D model that he/she wants to create. b: The creator makes a rough 3D model using Scan Modeling. c: The creator imports the model into a popular 3D modeler from Scan Modeling. d: The creator renders or edits details by using the 3D modeler.

the shape becomes the interface that affects the target. This interface has a rich power of expression in rapid prototyping for three reasons: changing the object's or hand's contact point can be performed quickly since the creator is used to using these tools in the real world, changing the object's shape (i.e., changing the object itself) or the hand's shape can be performed rapidly, and the shape can be freely chosen.

In contrast, in traditional digital 3D modeling, a mouse or a stylus is used. In such modeling, the interface affecting a target is a single point whose position is controlled by using a mouse or a stylus. Moreover, a creator performs 3D modeling by selecting one of multiple modification modes given by

the system a priori. Due to these limitations, the amount of freedom for expression is limited.

To address this problem, 3D modeling environments that use real objects have been researched. This research includes 3D modeling environments in which the shape of a hand is used and ones in which real tools are used. In these 3D modeling environments, the interface to a 3D model becomes a surface represented by a hand's or device's shape. Therefore, the amount of freedom for expression increases compared to traditional 3D modeling. Nevertheless, the shapes used in these 3D modeling environments are assigned by the systems a priori, i.e. no real world object's shape is used.

Our goal is to realize a 3D modeling environment in which a creator can make rough 3D models rapidly by using the shape of any real world object. To achieve this goal, we introduce Scan Modeling, in which the creator performs 3D modeling by scanning real objects readily available. As shown in Figure 1, an example of a typical 3D modeling process with Scan Modeling is:

- a: The creator imagines a 3D model that he/she wants to create. In Figure 1a, he/she wanted to make a 3D model of a holed vase with two flowers.
- b: The creator makes a rough 3D model by using Scan Modeling. In this case, the creator scans the clay, the flower, and his/her finger in the real world.
- c: The creator imports the 3D model into a popular 3D modeler from Scan Modeling to edit its details. In Figure 1c, he/she imports it into Blender<sup>1</sup>.
- d: After editing, he/she renders the 3D model.

We developed an input device called “Wakucon” to realize Scan Modeling. The Wakucon is square-shaped with 245 millimeters per side. A creator uses it by inserting an object inside of it to scan a cross section of the shape of the object. To scan a cross section, the Wakucon has four cameras attached on each inner side of the it that face inward. Additionally, our 3D modeling environment recognizes the Wakucon's 3D position and rotation. To recognize these, we use a depth camera and a degrees-of-freedom (9DOF) sensor, respectively. The creator can perform 3D modeling by moving either the Wakucon or objects inserted into the device while scanning these objects.

## RELATED WORK

Scan Modeling consists of 3D modeling techniques performed by moving the Wakucon or objects inserted into the device while scanning these objects. Therefore, related works relate Scan Modeling to interaction techniques for 3D modeling or an implementation of our system. These works include research on 3D modeling environments in which hands' or devices' spatial movement, position, posture, and shape are used and research on scanning an object's shape.

## Spatial 3D Modeling

There is research on 3D modeling environments in which various devices are used. Saso et al. presented the Beyond-Me Site, in which 3D modeling is performed by using a camera held in one hand and a marker held in the other [12]. Keefe et al. presented CavePainting, in which 3D models are created by using a real brush in the CAVE environment [5]. Sachs et al. presented 3-Draw, in which a creator holds a stylus and a palette in the hands and draws 3D curves by moving the stylus [11]. A common feature among this research is that the shape of a vertex or a surface to be added into VR space is given by each 3D modeling environment a priori; that is, a creator makes a 3D model by selecting modes that define how to modify the 3D model. In contrast to this research, in Scan Modeling, 3D modeling is performed by scanning a cross section of a real object dynamically. The creator makes a rough 3D model by changing the position of an object for scanning or its posture or by changing the Wakucon's position or rotating it.

Some research has tried to use the shapes of hands in 3D modeling. Gloss et al. presented Gesture Modeling, in which a creator makes a 3D model by using hand postures recognized with computer vision [4]. Moritz et al. presented a 3D modeling environment in which a creator wears Data Gloves on both hands and performs boolean operation on 3D models by using a model of a plane, a sphere, a cube, or a cone [8]. Schkolne et al. presented Surface Drawing, in which a creator wears a glove interface and draws 3D models by shaping with his/her hands [13]. By contrast, in Scan Modeling, not only are hand shapes used but also any object's shape is used. Therefore, Scan Modeling extends this research in which 3D modeling is performed by shaping with the hands.

Malleable device shaping also has been tried. A 3D modeling environment was presented in which a creator holds a malleable device in his/her hands and deforms a 3D model by deforming the device [9, 15]. In an environment created by Sheng et al., a creator edits a 3D model not only by deforming a malleable device with his/her hands but also by pressing a knife against the device [14]. All of this research focused on deforming a 3D model by using press manipulation. In contrast to this research, in Scan Modeling, the press like manipulation is supported by subtracting a 3D model spatially by using a scanned cross section. In addition, we support reconstructing a part or the whole of a real object.

There are pieces of research that use real objects' shapes in 3D modeling environments. Anabuki et al. presented AR-Jig, in which a creator makes a 3D model by using a pin array called a “jig” [1]. With AR-Jig, the creator can copy a real object's contour by pushing the jig against the object and paste the contour to a 3D model. Anderson et al. presented a 3D modeling environment in which a creator builds a model by using physical blocks, and these blocks are reconstructed in VR space [2]. Also, in Scan Modeling, the real object is reconstructed in VR space by using scanning. In addition to reconstruction, complex interaction techniques including holding, shaving, and cutting with a real object as a tool are uniformly supported in Scan Modeling.

<sup>1</sup><http://www.blender.org/>

### Scanning an Object's Shape

Methods for scanning a real object are classified into two classes:

- scanning by pressing an object against a deformable device.
- scanning an object with sensors, which observe the object from different positions in an environment.

Our method belongs in the second class, in which an object's cross section is scanned by using cameras.

The first class is used to reconstruct an object's surface. Pieces of research using this method were shown in the previous section, such as research with deformable devices [9, 15, 14] or the AR-Jig presented by Anabuki et al. [1]. In addition, Sean et al. presented a malleable surface that reconstructs the surface of an object pressed against it [3].

In the second class, light sensors, magnetic sensors, cameras, etc. are used. Moeller et al. presented a multi-touch interface that scans the cross section of an object inserted into a device by using an IR sensors array [7]. Reed presented a method to scan the entire shape of a clay object [10]. In this research, wireless magnetic transceivers were embedded into the clay, and to track these transceivers' positions and postures with a magnetic receiver, the clay's shape was scanned. In contrast to the first class, these pieces of research reconstruct a cross section or the whole of an object.

The first is suitable for texturing a 3D model because an object's surface can be scanned. In contrast, the second suits the prototyping because it can reconstruct a whole object rapidly in VR space by scanning. Therefore, we adopt the second, in which an object's cross section is scanned by using cameras.

### OVERVIEW OF SCAN MODELING

Scan Modeling's interaction is performed by spatially moving the Wakucon or a real object inserted into the device. Figure 2 shows an overview of Scan Modeling. The bottom of Figure 2 illustrates both the Wakucon and a real object scanned in our system. The top shows our 3D modeler's display. In the 3D modeler, the created 3D model, the cross section of the shape inserted into the device, and the cursor, which represents the Wakucon's position and rotation, are displayed. 3D modeling is performed by drawing or subtracting 3D voxels by using a real object's cross section.

### INTERACTION TECHNIQUES OF SCAN MODELING

There are three manipulations with Scan Modeling: *Scan Subtraction*, *Scan Draw*, and *View Change*. The first is a manipulation that holes, shaves, or cuts a 3D model by using a cross section of a shape. The second is a manipulation that generates a 3D model by drawing voxels spatially by using a cross section. The third is a manipulation that changes the view toward a 3D model in VR space.

#### Scan Subtraction

Scan Subtraction is a manipulation that holes, shaves, or cuts by subtracting existing voxels by using a cross section. This is similar to modification by using a tool's surface in real clay modeling. In Scan Subtraction, a creator determines how to

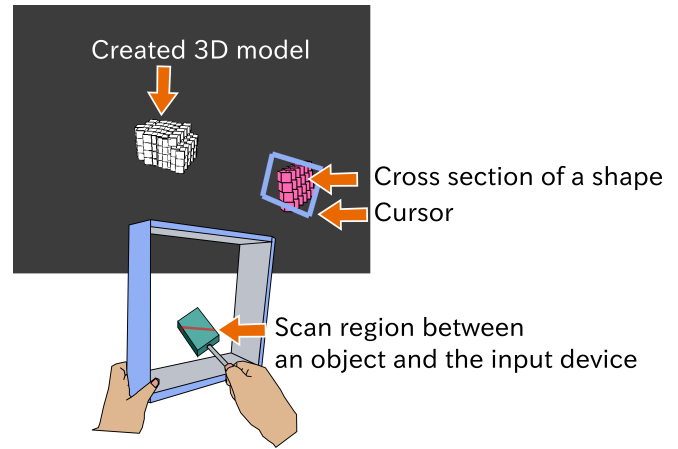


Figure 2. Overview of Scan Modeling.

modify a 3D model by changing a scanned shape or the way to push against the target, like in real clay modeling.

Figure 3 shows a technique of Scan Subtraction, holing. Using this technique, it is possible to hole an arbitrarily shaped hole. A case involving the holing process is described below and is shown in Figure 3.

- The creator holds the Wakucon in one hand and the object in the other.
- The creator pushes both hands on the 3D model while scanning the object.
- The 3D model is holed.

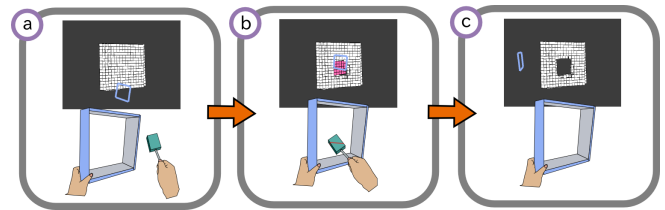


Figure 3. A technique for holing by using Scan Subtraction.

Figure 4 shows a shaving technique. A case involving the shaving process is described below and is shown in Figure 4.

- The creator holds the Wakucon in one hand and clenches the other.
- The creator shifts both hands around the right side of the 3D model while scanning.
- The 3D model is shaved.

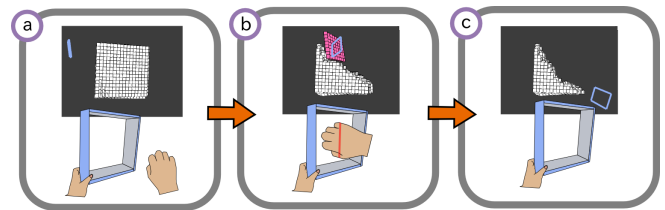


Figure 4. A technique for shaving by using Scan Subtraction.

Figure 5 shows a cutting technique. A case involving the cutting process is described below and is shown in Figure 5.

- a:** The creator holds the Wakucon and the object.
- b:** The creator shifts both hands in a longitudinal direction while scanning.
- c:** The 3D model is cut.

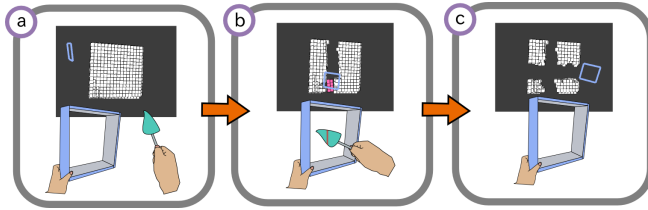


Figure 5. A technique for cutting by using Scan Subtraction.

### Scan Draw

Scan Draw is a manipulation that generates one or more 3D models by drawing voxels spatially by using a cross section. A technique with Scan Draw is shown in Figure 6. In Figure 6, the creator is scanning cross sections of two of his/her fingers and shifting both hands while maintaining a relative position between the Wakucon and fingers. This results in two parallel lines in VR space.

Another technique with Scan Draw is shown in Figure 7. This technique copies an entire real object into VR space by scanning a stationary target object. In Figure 7, the creator scans his/her entire hand by making his/her one hand into a specific shape and moving the Wakucon through this hand by using the other hand.

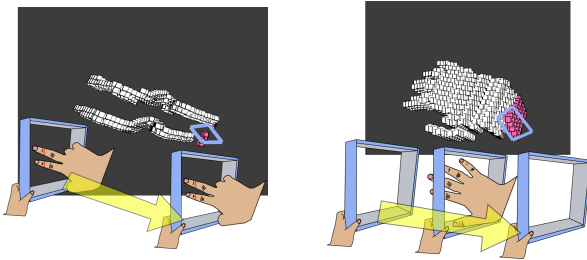


Figure 6. A technique with Scan Draw. The creator draws two 3D lines with two fingers.

Figure 7. Another technique with Scan Draw. The creator scans his/her entire hand.

### View Change

View Change is a manipulation that changes the view to a 3D model in VR space by using Wakucon's pose. By using View Change, a creator can view a 3D model from any direction by rotating the camera in VR space along the x, y, z axes. For example, in the top of Figure 8, the creator is rotating the camera along the x axis, rotating along the y axis on the left, and rotating along the z axis on the right.

### IMPLEMENTATION

In this section, we describe an implementation of Scan Modeling. First, we present the system configuration. Second, we present the Wakucon and 3D reconstruction method that uses the device. Finally, we show a method for rotation and position detection.

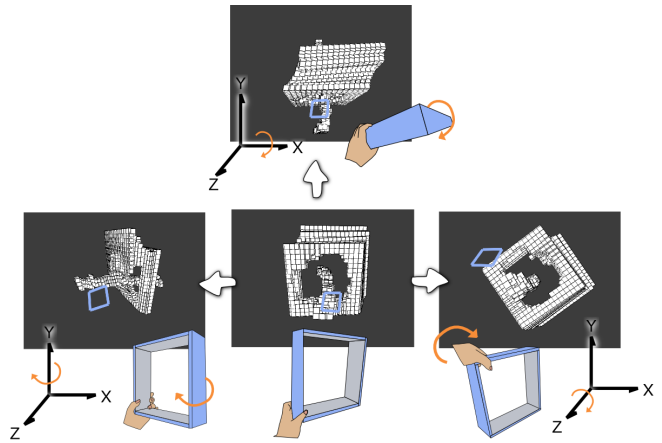


Figure 8. An example of View Change. At the top, the creator is rotating the camera along the x axis, rotating along the y axis on the left, and rotating along the z axis on the right.

### System Configuration

As shown in Figure 9, the system consists of the Wakucon, a depth camera, and a display. The Wakucon is used to scan a cross section of a real object. The depth camera is used to detect the 3D position of the Wakucon. At current implementation, we used Microsoft's Kinect<sup>2</sup> as the depth camera. Finally, the display is used to show the 3D modeler.

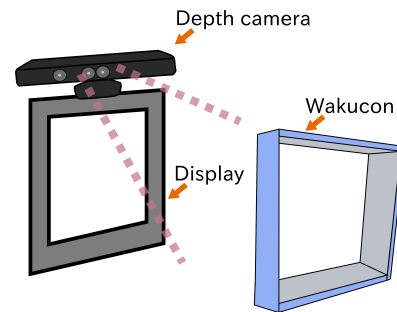


Figure 9. System configuration of Scan Modeling. Our system consists of the Wakucon, a depth camera, and a display.

### Wakucon

As shown in Figure 10, the Wakucon consists of four cameras, infrared (IR) LEDs, a microcontroller, a degrees-of-freedom (9DOF) sensor, and buttons. Each element's arrangement and purpose is described below.

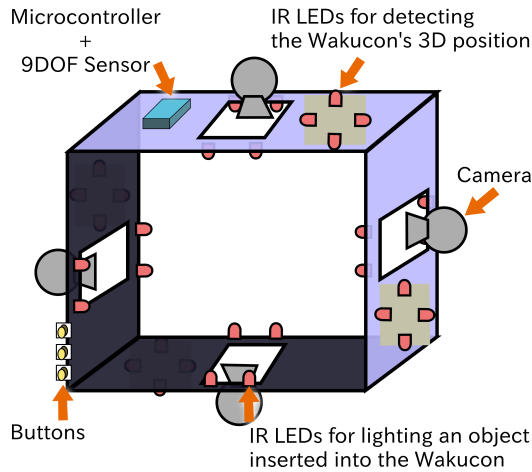
**Four cameras** A camera is attached on each inner side of the Wakucon and faces inward. The camera views the objects inserted into the Wakucon from each position. The frames captured by the cameras are used to reconstruct the cross section of an object. In the current implementation, the FPS of each camera was set to 30.

**IR LEDs** As shown in Figure 10, the Wakucon has two types of IR LEDs:

**IR LEDs for lighting an object** In our 3D reconstruction method, an object's silhouette, which is grabbed from

<sup>2</sup><http://www.xbox.com/kinect>





**Figure 10. Wakucon configuration.** The Wakucon consists of four cameras, IR LEDs, a microcontroller, a 9DOF sensor, and buttons.

a camera frame, is used as an input source. Since detecting the silhouette sometimes fails when the object's color is dark or due to the lighting environment, we use the reflected light of the IR LEDs that are attached around on each camera's lens as the camera's input.

#### **IR LEDs for detecting the Wakucon's 3D position**

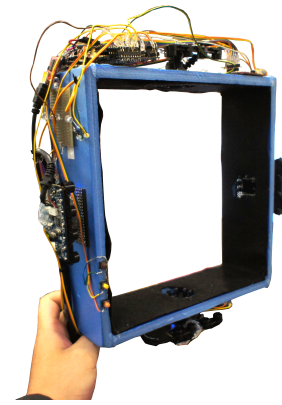
Wakucon 3D position is detected by tracking with the IR LEDs attached on the outside of the device by using the depth camera.

**Microcontroller** We use the microcontroller to turn on/off the IR LEDs used for lighting an object inserted into the Wakucon. Because turning on all the IR LEDs is too bright, we only turn on the IR LEDs around the camera used for capturing a frame while turning off all the other IR LEDs. At current implementation, we used an Arduino Pro Mini<sup>3</sup> as the microcontroller.

**9DOF sensor** The 9DOF sensor attached on the Wakucon consists of a triple-axis accelerometer sensor, a triple-axis magnetometer sensor, and a triple-axis gyroscope. By using the sensor, Wakucon's pitch, roll, and yaw are computed. At current implementation, we used Sparkfun's SEN-10736<sup>4</sup> as the 9DOF sensor.

**Buttons** Three buttons are attached around the grip of the device. These buttons are triggers of Scan Subtraction, Scan Draw, and View Change.

Our prototype Wakucon is shown in Figure 11. The prototype weighs about 400 grams without cables and about 700 grams with cables connecting the device to a PC. Each side of the device is 245 millimeters.



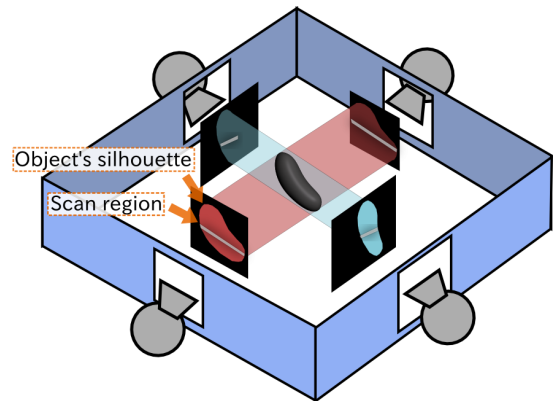
**Figure 11. Prototype Wakucon.**

### **Reconstruction of a Cross Section**

Scan Modeling is a 3D modeling technique that uses a scanned cross section. The Scan Modeling system repeats the following steps: 1) reconstruct a cross section and 2) represent the cross section as voxels in VR space. In this section, first, we describe Shape-From-Silhouette [6] which is an algorithm we use for the first step. Second, we describe how to reconstruct a cross section by using the algorithm as well as how to calculate the rotation and position of the input device for the second step.

#### *Shape-From-Silhouette*

Shape-From-Silhouette [6] is a reconstruction method that uses a target object's silhouettes, which are extracted from camera frames captured from different angles. Figure 12 shows the concept of Shape-From-Silhouette. In Figure 12, a camera is attached on each of the sides of the Wakucon and detects the silhouette of an object inserted into the center of the device. Shape-From-Silhouette first defines a volume by extending each silhouette toward the depth direction. The cross region of all the volumes will be the reconstructed model. In our implementation, we use Figure 12's scan region as the cross region, enabling the creator to widen/narrow the width of scanning.



**Figure 12. Reconstructing a cross section by using Shape-From-Silhouette.** A white line within each object's silhouette is used to represent a scan region in this figure.

<sup>3</sup><http://arduino.cc/it/Main/ArduinoBoardProMini>

<sup>4</sup><http://www.sparkfun.com/products/10736>

### Reconstruction of a cross section by using Shape-From-Silhouette

In our implementation, a silhouette is detected in four steps. Figure 13 shows the result of each step. Cam\_0~3 represent frames grabbed from each camera. The frames, shown in Figure 13, are raw, synchronized, moving averaged, and binarized frames from left to right. A detailed description of each step is given below.

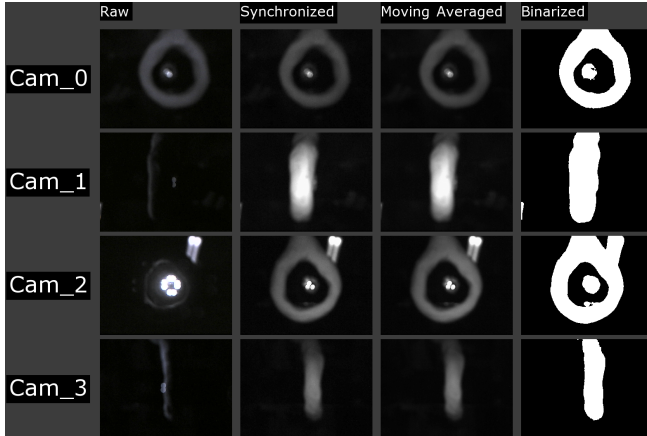


Figure 13. Frames captured from cameras attached on the Wakucon. In this picture, Cam\_0~3 represent each camera's frame. Moreover, shown from the upper left are raw, synchronized, moving averaged, and binarized frames.

**Raw** A raw frame is captured with a camera. At this step, the capture is not synchronized with LEDs. Therefore, as is shown under "Raw" in Figure 13, the object is lit by LEDs' from various directions.

**Synchronized** Synchronized frames are frames captured under synchronized lighting. That is, the system turns on the LEDs attached around the camera and then captures a frame with the camera.

**Moving averaged** Moving averaged frames are frames that are taken as a moving average by using some synchronized frames. This is computed to stabilize brightness variations that occur from the blinking of the LEDs.

**Binarized** Binarized frames are generated by binarizing moving averaged frames.

Since the binarized frames show silhouettes, we use them as the input to the Shape-From-Silhouette algorithm.

An example of reconstruction is shown in Figure 14. In Figure 14, a torus's cross section is reconstructed. Another example is shown in Figure 15. In Figure 15, the creator widens the scan region to the maximum size; that is, the whole of the camera frame. In this case, the torus shaped object and the hand grasping it are reconstructed instantly.

### Calculating Rotation and Position of the Input Device

The Wakucon's rotation is computed from the attached 9DOF sensor. The device's position is computed by attached tracking IR LEDs for position detection. Figure 16 shows the process that utilizes the Kinect's RGB depth camera, making the tracking simple but robust. First, an RGB frame is captured with the RGB camera (Figure 16a). To track only the IR

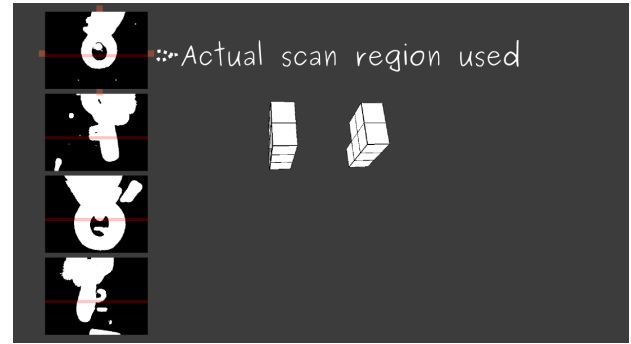


Figure 14. Reconstruction result by using our system. In this result, the torus's cross section is reconstructed.

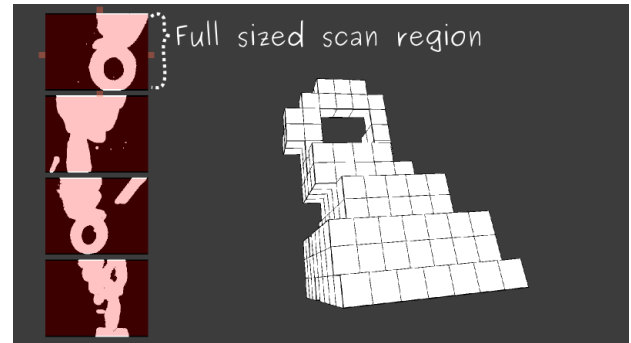


Figure 15. Reconstruction result by using our system. In this result, the torus shaped object and the hand grasping it are reconstructed instantly with a full sized scan region.

LEDs, an IR pass filter is attached to the RGB camera. At the same time, an depth frame is captured with the depth camera (Figure 16b). Second, binarization and dilation are applied to the RGB frame (Figure 16c) to make a mask frame for later use. In the mask frame, white pixels correspond to the Wakucon's IR LEDs. Next, by masking the depth frame with the mask frame shown in Figure 16c as a mask, we can obtain each IR LED's  $(x, y, z)$  position (Figure 16d). Finally, from these positions, the Wakucon's  $(x, y, z)$  position is computed. In this position, drawing or subtracting voxels is performed in Scan Modeling.

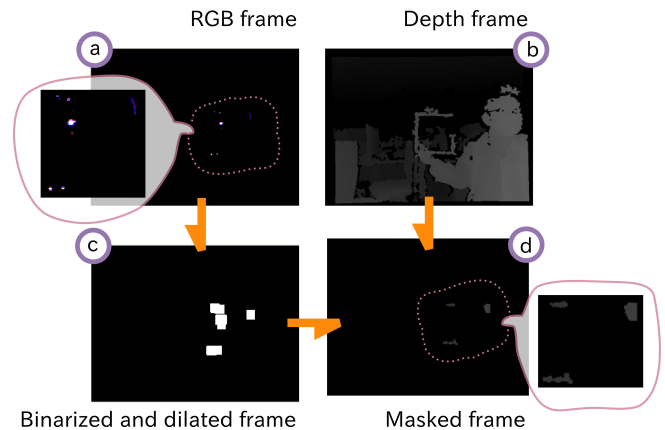


Figure 16. Image processing to obtain a device's 3D position.

## EVALUATION OF RECONSTRUCTED 3D MODELS

Our research goal is to realize a 3D modeling environment in which a creator can make rough 3D models rapidly by using any real world object's shape. Because reconstructed 3D models are main tools in Scan Modeling, the accuracy of the reconstructed 3D models is important. Therefore, we examined this accuracy.

### Method

We scanned four primitive objects by using Scan Draw. These four objects are shown in Figure 17. From the left, a sphere, a triangle, a rectangle, and a torus are shown. Each object was made of clay beforehand, and a stick was inserted as a grip.



Figure 17. The four objects used in this evaluation. From the left, a sphere, a triangle, a rectangle, and a torus are shown.

We scanned these objects by using Scan Draw multiple times, and the 3D model which was thought to be closest to the source object was used as an experimental result. Additionally, the range for object scanning was from the top of each object to a point attached on the stick.

The experiment was conducted by one of the authors. We used a computer with an Intel Core2 Quad Q9550 CPU, 4GB of RAM, and an ATI Radeon HD 3600 GPU that runs Ubuntu Linux.

### Results and Discussion

The experimental result is shown in Figure 18. Figure 18a, b, c, d represent the scanned sphere, triangle, rectangle, and torus, respectively. Each figure shows the source object in the upper left for comparison. Moreover, each figure is the 3D modeling's result created from the following steps.

1. Create 3D voxels by using Scan Modeling with a source object.
2. Import into meshlab<sup>5</sup> to build meshes by using Delaunay triangulation from 3D voxels.
3. Import into Blender to render meshes.

In addition, the time taken to reconstruct the 3D models was 2.6 seconds for the sphere, 1.5 seconds for the triangle, 2.0 seconds for the rectangle, and 3.3 seconds for the torus. Note that each time represents the time taken to create 3D voxels by using Scan Modeling.

As shown in Figure 18, all the reconstructed 3D models roughly reflected the corresponding source objects' shapes. For example, in Figure 18b, the model represents the triangle

<sup>5</sup><http://meshlab.sourceforge.net/>

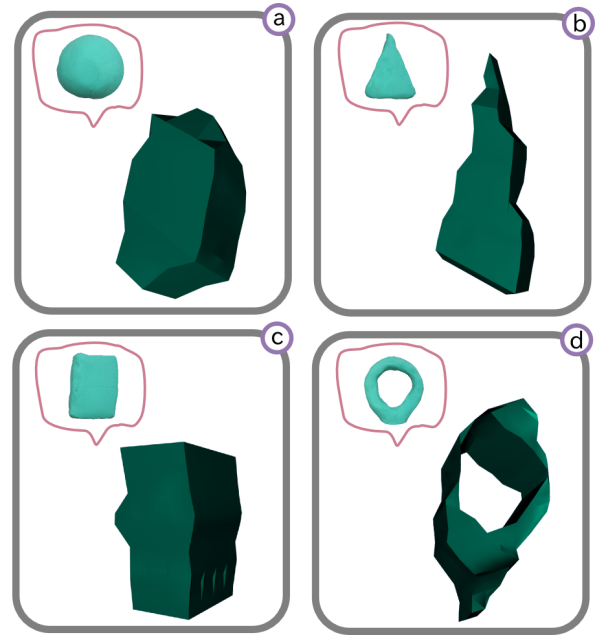


Figure 18. Reconstructed 3D models of real objects by using Scan Draw; a: the sphere, b: the triangle, c: the rectangle, and d: the torus.

shape. In Figure 18d, the model represents the torus, which has a hole in the center. However, problems were found and are described below.

- In current implementation, a sphere was difficult to reconstruct, as shown in Figure 18a.
- A distortion along the scanned direction occurred in the reconstructed 3D models. For example, the rectangle shown in Figure 18c curves loosely in a longitudinal direction. Moreover, the noises of vertices are shown around the center of the model.
- There is variance in the density of the vertices in the reconstructed 3D models. For example, in the torus shown in Figure 18d, the density of the vertices of the tube's left side are low compared to those of the right side.

The first problem was caused by how the cross section scanning was implemented. In current implementation, we use silhouette frames with lower resolution as input sources to reduce the calculation cost of Shape-From-Silhouette. Specifically, we reduced the resolution of raw frames  $320 \times 240$  into  $16 \times 12$  to detect the silhouette. This problem can be addressed by using higher resolution; reconstructed 3D models become closer to the source objects. However, real time reconstruction becomes difficult because it takes more time to detect the intersection of the silhouettes' volumes. Therefore, we plan to modify the algorithm to detect an intersection by using parallel computation of CPUs or by using a GPU based Shape-From-Silhouette [16].

The second and third problems were caused by a variance of the Wakucon's position and move speed, respectively. To solve these problems, a creator can scan the whole object instantaneously with a full sized scan region, as shown in Figure 15, in case of reconstructing a whole object. To scan in-

stantaneously, the effect of the variance of the Wakucon's position or speed must be reduced. Moreover, we plan to apply filters, such as the Kalman filter, to smooth the variance.

## CONCLUSION AND FUTURE WORK

In this paper, we presented 3D modeling techniques that use Scan Modeling to realize a 3D modeling environment in which a creator can make rough 3D models rapidly by using any real world object's shape. Scan Modeling's manipulations consist of Scan Subtraction, Scan Draw, and View Change. By using these manipulations, a creator can make a 3D model rapidly. We presented an implementation of a square shaped input device called "Wakucon", a method to reconstruct a cross section of a shape, and a method to calculate the rotation and position of the device.

Our future work is to make the precision of the cross section's reconstruction and the Wakucon's position tracking higher, as shown in the "Evaluation of Reconstructed 3D Models" section. By making precision higher, a creator can prototype a 3D model closer to what he/she wants to create with only one interaction. For this reason, we believe that a creator can make a 3D model more rapidly. We also plan to conduct two evaluations in order to validate Scan Modeling's effectiveness and possibility of expression. In the first evaluation, we will employ creators, including 3D modeling novices, as subjects and let them create specific 3D models in a given amount of time by using a typical 3D modeler or Scan Modeling. As a result of the first evaluation, we will confirm that creators can make 3D models rapidly by using Scan Modeling. In the second evaluation, we will employ as subjects professional creators who are used to creating 3D models and let them create 3D models freely by using Scan Modeling. As a result of the second, we will confirm the possibility of expression by using Scan Modeling and analyze various points to make improvements for practical realization.

## REFERENCES

1. Anabuki, M., and Ishii, H. AR-Jig: A Handheld Tangible User Interface for Modification of 3D Digital Form via 2D Physical Curve. In *Proc. ISMAR 2007*, IEEE (2007), 1–10.
2. Anderson, D., Frankel, J. L., Marks, J., Agarwala, A., Beardsley, P., Hodgins, J., Leigh, D., Ryall, K., Sullivan, E., and Yedidia, J. S. Tangible Interaction + Graphical Interpretation: A New Approach to 3D Modeling. In *Proc. SIGGRAPH 2000*, ACM (2000), 393–402.
3. Follmer, S., Johnson, M., Adelson, E., and Ishii, H. deForm: an Interactive Malleable Surface for Capturing 2.5D Arbitrary Objects, Tools and Touch. In *Proc. UIST 2011*, ACM (2011), 527–536.
4. Gross, M., and Kemp, A. Gesture Modelling: Using Video to Capture Freehand Modeling Commands. In *Proc. CAAD Futures 2001*, Kluwer Academic Publishers (2001), 271–284.
5. Keefe, D. F., Feliz, D. A., Moscovich, T., Laidlaw, D. H., and LaViola, Jr., J. J. CavePainting: A Fully Immersive 3D Artistic Medium and Interactive Experience. In *Proc. I3D 2001*, ACM (2001), 85–93.
6. Martin, W. N., and Aggarwal, J. K. Volumetric Descriptions of Objects from Multiple Views. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1983), 150–158.
7. Moeller, J., Kerne, A., and Damaraju, S. ZeroTouch: a Zero-Thickness Optical Multi-Touch Force Field. In *Ext. Abstracts CHI 2011*, ACM (2011), 1165–1170.
8. Moritz, E., Kuester, F., Hamann, B., Kenneth, I. J., and Hagen, H. Towards Immersive Clay Modeling: Interactive Modeling with Octrees. *Proc. SPIE 3957* (2000), 414–422.
9. Murakami, T., and Nakajima, N. Direct and Intuitive Input Device for 3-D Shape Deformation. In *Proc. CHI 1994*, ACM (1994), 465–470.
10. Reed, M. Prototyping Digital Clay as an Active Material. In *Proc. TEI 2009*, ACM (2009), 339–342.
11. Sachs, E., Roberts, A., and Stoops, D. 3-Draw: A Tool for Designing 3D Shapes. *IEEE Computer Graphics and Applications 11* (1991), 18–26.
12. Saso, T., Tamayama, T., and Inakage, M. Beyond-Me Site: Two-Handed Interface for 3D Drawing in MR. In *Proc. GRAPHITE 2003*, ACM (2003), 275–276.
13. Schkolne, S., Pruett, M., and Schröder, P. Surface Drawing: Creating Organic 3D Shapes with the Hand and Tangible Tools. In *Proc. CHI 2001*, ACM (2001), 261–268.
14. Sheng, J., Balakrishnan, R., and Singh, K. An Interface for Virtual 3D Sculpting via Physical Proxy. In *Proc. GRAPHITE 2006*, ACM (2006), 213–220.
15. Smith, R. T., Thomas, B. H., and Piekarski, W. Digital Foam Interaction Techniques for 3D Modeling. In *Proc. VRST 2008*, ACM (2008), 61–68.
16. Yous, S., Laga, H., Kidode, M., and Chihara, K. GPU-Based Shape from Silhouettes. In *Proc. GRAPHITE 2007*, ACM (2007), 71–77.