Tangible and Visible 3D Object Reconstruction in Augmented Reality

Yi-Chin Wu* Liwei Chan† Wen-Chieh Lin‡

Department of Computer Science
National Chiao Tung University, Taiwan

ABSTRACT

Many crucial applications in the fields of filmmaking, game design, education, and cultural preservation — among others — involve the modeling, authoring, or editing of 3D objects and scenes. The two major methods of creating 3D models are 1) modeling, using computer software, and 2) reconstruction, generally using high-quality 3D scanners. Scanners of sufficient quality to support the latter method remain unaffordable to the general public. Since the emergence of consumer-grade RGBD cameras, there has been a growing interest in 3D reconstruction systems using depth cameras. However, most such systems are not user-friendly, and require intense efforts and practice if good reconstruction results are to be obtained. In this paper, we propose to increase the accessibility of depth-camera-based 3D reconstruction by assisting its users with augmented reality (AR) technology. Specifically, the proposed approach will allow users to rotate/move a target object freely with their hands and see the object being overlapped with its reconstructing model during the reconstruction process. As well as being more instinctual than conventional reconstruction systems, our proposed system will provide useful hints on complete 3D reconstruction of an object, including the best capturing range; reminder of moving and rotating the object at a steady speed; and which model regions are complex enough to require zooming-in. We evaluated our system via a user study that compared its performance against those of three other state-of-the-art approaches, and found our system outperforms the other approaches. Specifically, the participants rated it highest in usability, understandability, and model satisfaction.

1 INTRODUCTION

A means of digitalizing the physical world, 3D reconstruction is often divided into two types, scene and object reconstructions, both of which are important building blocks of various applications in robotics, self-driving vehicle-making and other forms of manufacturing, cultural-heritage preservation, digital museums, and mixed reality, among other fields. More recently, personal fabrication/digitalization [2] has been calling for 3D reconstruction, which leads to the demand for accessible, easy-to-use reconstruction tools aimed at novice users.

Existing approaches have been using portable 3D scanners that are also made possible by smartphones, but these require the user to walk the scanner around the object, and the results are usually unevenly guaranteed due to unstable hand motions. Moreover, their users have no way of knowing during scanning that such problems are occurring or if the eventual reconstruction will be successful. Other approaches use motorized rotating platforms to capture objects at a controlled pace. Though less prone to operator error than their hand-held equivalents, such systems are incompatible with mobile digitalization/fabrication [27]. In an attempt to strike a balance between mobility and accuracy, Tzionas and Gall [37] proposed a method for reconstructing an object held and rotated in the user's hand during tracking and reconstruction. However, their initial exploration was forced to place limits on users' handling of the object, both in terms of speed and specific hand posture, and did not provide them with guidance on how to achieve the best results.

This work presents a wearable object-reconstruction system centered on video see-through AR glasses, which is indeed a VR headset equipped with an RGBD camera. The main advantage of AR glasses is that they allow the display of reconstruction guidance, as well as progress, directly on the objects in the users' hands, thus allowing them to focus completely on the task, rather than dividing their attention between the objects and external screens. The visual guidance provided to streamline the process and improve the results includes 1) the appropriate distance between the object and the AR glasses, 2) the suitable speeds at which to move and rotate the object, and 3) which parts of a given object may require more exposure due to their complex geometries. In other words, our system is intended to help users improve their reconstructions at the time of capture rather than repairing them later. As such, it offers novice users a more intuitive and effective means of creating 3D models of objects.

Like Tzionas and Gall's [37], our system relies on the user's hand as a motorized platform for the target object during scanning. However, our focus was on improving user interaction by providing interactive guidance on how best to move the object using augmented reality (AR), in the expectation that this would lead to improved reconstruction results. The goal of our paper is not developing a new 3D reconstruction algorithm. Rather, the main goal and contribution of this paper resides in proposing the concept of AR-assisted 3D reconstruction with visual guidance in a first-person view (tangible and visible way) and nicely integrating existing methods, including 3D reconstruction, hand detection and segmentation, and model analysis. Furthermore, the proposed concept and systems were carefully evaluated and compared with the state of the art systems qualitatively and quantitatively.

In sum, this paper makes three contributions. First, it proposes a 3D reconstruction system that represents a new departure in both intuitiveness and user-friendliness. Second, it makes a novel use of AR technology to reconstruct and manipulate objects from the perspectives of users' eyes and hands; and third, it improves 3D reconstruction quality and the user experience by providing timely object-manipulation guidance.

2 RELATED WORK

This section reviews relevant prior studies on 3D object reconstruction. We have divided their approaches according to whether the object to be captured is constrained to a surface or kept in the user's hand during the capture process. The provision of user guidance aimed ultimately at improving the quality of the results is also discussed.

2.1 Object on Surface during 3D Reconstruction

Typical 3D reconstruction based on image sequences requires users to walk the camera around an object to acquire images of it from all directions. Early versions of this process had to be conducted offline [8] due to computational issues that prevented users from

*e-mail:chinapipa.cs04g@nctu.edu.tw
†e-mail:liweichan@cs.nctu.edu.tw
‡e-mail:wclin@cs.nctu.edu.tw

978-1-7281-4765-9/19/$31.00 ©2019 IEEE
DOI 10.1109/ISMAR.2019.00-30

reviewing their partial results during capture. KinectFusion [11, 17] was an early effort to enable reconstruction during runtime, and led to the development of new algorithms for reconstructing large-scale scenes [18] and approximating such performance on mobile devices [13]. Kähler et al. dealt with the loop-closure problem [12] caused by alignment error accumulation; and the system proposed by Cao et al. [3] achieved high-fidelity 3D reconstruction with consumer RGBD cameras in real time.

To spare users' efforts in acquiring images from a multitude of angles, turntable-based reconstruction system was proposed by Fremont and Chellali. [5]. An object to be captured would be placed on a platform whose rotation can be synchronized with image recording – i.e., images can be recorded at known angles. This approach has subsequently been supported by commercial software such as Intel Cappasity Easy 3D Scan [1]. However, since the object is placed on the platform throughout the recording process, the bottom of it is typically excluded. HP Sprout[2] solved this problem by adding a second capture phase, during which the object is placed in a different position.

Since all the aforementioned methods reconstruct whole scenes, their users are expected to segment object models manually after reconstruction. For that reason, Golodetz et al. proposed a reconstruction system that included a semantic painting function to aid such segmentation [7, 38] while Tateno et al. worked on a simultaneous reconstruction, segmentation and recognition system [33, 34].

2.2 Object in Hand during 3D Reconstruction

Rather than constraining the object on a surface, another line of research allows users to manipulate objects' positions directly with their hands during capture. In this vein, Tsukizawa et al. [36] demonstrated extraction of rough shape and texture of an object in hand with the introduction of hand-object-view relationship. Rusinkiewicz et al. [29] and Weise et al. [40, 41] proposed the wearing of black gloves to facilitate the removal of users' hands from the resulting images. Without relying on colored gloves, Ren et al. [24] proposed a method robust to missing data, occlusion and outliers that incorporated a primitive shape model of the target object; however, their results were insufficiently detailed. Tzionas and Gall [37] focused on the reconstruction of symmetrical objects rotated by hand. Panteleris and Kyriazi [22] exploited real-time bi-manual hand tracking to devise a system that mitigated hand occlusion by allowing the user to manipulate the object with either hand. Rünz and Agapito [28] proposed a co-fusion system that segments each user's hands as an object model, and thus allows the tracking of multiple objects in real time. Our proposed system does not utilize hand tracking, yet detects users' hands via the reconstruction process, allowing not only a more flexible capture process (insofar as users can freely move their hands in and out of the camera's field of view) but also the use of a midair touch interface.

2.3 User Guidance during 3D Reconstruction

Object-in-hand reconstruction allows users greater freedom than its object-on-surface equivalent, but its results may be marred by user manipulation of the object at the point of capture, e.g., by hand occlusions or inappropriate manipulation speeds, especially when the user is a novice. Guidance in proper user manipulation has previously been provided in the form of augmented arrows [6, 21], a health bar to show the quality of frame alignment [4], next position of an image frame to take [26], and geometry information about the current result [16]. In the realm of commercial software, Autodesk 123D Catch[3] makes suggestions for optimal camera poses during the reconstruction. The present research also presents user guidance. Specifically, our interface focuses on objects with complex geometry that often challenges novice users, providing interactive guidance about such geometry based on real-time detection of both the object's texture and its position.

We summarize the differences between our work and the aforementioned ones as follows: 1) Compared to the object-on-surface reconstruction methods, ours is more straightforward as the target object is tangible to users; 2) Compared to the object-in-hand reconstruction methods except [36], ours is more intuitive as the target object is constructed in users' first-person view via augmented reality; 3) Compared to user-guidance reconstruction methods, ours provides additional guidances that consider target object's geometric and textural complexity.

3 Methods

We designed our system based on the following requirements: 1) an AR-based user interface to facilitate hand-held manipulation and reconstruction; 2) a mid-air touch interface to ease operations in the AR environment; and 3) timely and appropriate user guidance to assist 3D reconstruction.

3.1 System Overview

Our system allows a user to reconstruct a 3D model of an object by freely rotating the object by hand. We attached an Intel Realsense SR300 to a Oculus Rift DK2 to create video see-through AR glasses, as shown in Figure 1. This ensures that the viewpoints of the user and the camera are the same, and offers users an intuitive way to examine his/her reconstruction results by manipulating the object.

Figure 1: Our video see-through AR glasses combining Oculus Rift DK2 and Intel Realsense SR300 with Velcro

Figure 2 presents the four main modules of our system with a functional diagram: object extraction, 3D reconstruction, model analysis, and user interface. Given an object to be 3D scanned, the camera captures RGBD images of it from multiple views as the user rotates or moves it. The object-extraction module extracts the object region from the resulting images by removing the background and the user's hands.

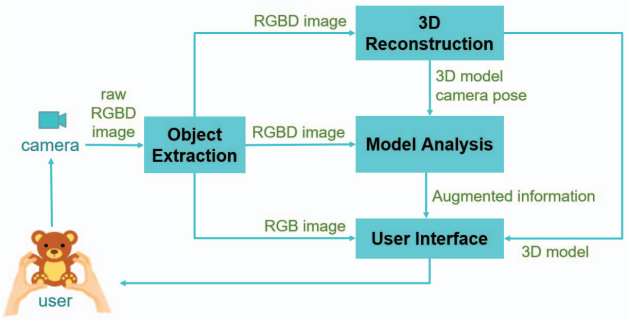


Figure 2: System Overview

The 3D reconstruction module was developed based on the Semantic Paint system [7], which used the InfiniTAM v2 fusion engine developed by Kähler et al. [13]. Our system adopts

[1] **Intel Cappasity Easy 3D Scan for Intel RealSense**:https://devmesh.intel.com/projects/cappasity-easy-3d-scan

[2] **HP sprout**:https://www8.hp.com/uk/en/sprout/3d.html

[3] **Autodesk 123D Catch for iOS 7**:https://www.youtube.com/watch?v=OxsmnDKO7D0

InfiniTAM v3, which adds loop closure, improved camera tracking, and a relocalizer. The 3D reconstruction module operates in three main steps: tracking, fusion, and rendering. The first localizes the camera relative to the object model. The next fuses novel depth data to the model under reconstruction whenever the fusion toggle is on. Lastly, the rendering step raycasts the model from the current camera pose. The three steps are repeated each time a new input image is received, until a fully 3D model has been obtained.

In the model-analysis module, the system analyzes input images and the model-in-progress to remove noise and/or provide guidance to users, which includes instructions on how to place the object at a more appropriate distance; highlighting of regions with complex geometry that need additional care; hints that one or more faces of the object have not been captured; and alerts that the object is being moved too fast. The user-interface module then displays this augmented guidance alongside the 3D model itself.

3.2 Object Extraction

The system extracts an object's depth data by removing its background and the user's hands. Figure 3 shows an example of background removal in the depth image, with the depth threshold being determined during the user calibration process (see section 3.5).

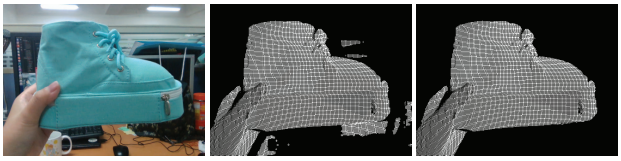


Figure 3: Left: color image for reference; Middle: depth image; Right: depth image after removal of depth values exceeding the background removal distance.

Skin-color detection. Our hand-detection and segmentation method integrates and modifies several existing methods. First, it identifies hand regions by detecting skin pixels, using the skin-color detection approach proposed by Kukharev and Nowosielski. [14], who suggested based on 25 face examples that the color ranges of skin in the YC_bC_r color space are ($Y > 80$, $85 < C_b < 135$, $135 < C_r < 180$). In our system, if any pixel's color values fall within that range, it is deemed to be skin. As well as YC_bC_r, however, we consider skin-color ranges in the HSV space as suggested by Tsekeridou and Pitas [35] and Oliveira and Conci [20]: $0 \leq H \leq 20$ or $170 \leq H \leq 180$, and $51 \leq S \leq 153$. Because skin detection using YC_bC_r is rapid, we also use it to determine if the user has selected a virtual menu on the AR screen (HMD).

Nevertheless, given that the every user's skin color will be slightly different, our system adjusts the above skin-color ranges to each individual user's skin color, to facilitate more accurate hand segmentation during the user calibration process (section 3.5).

Hand detection. The aforementioned pixel-based skin-detection method could miss some skin pixels, especially due to the influence of lighting. Therefore, to increase the robustness and accuracy of hand detection, we further developed a hand-segmentation method by 'growing' hand regions from skin superpixels, based on the hand-detection method proposed by Huang et al. [10], who first over-segmented an RGB image to superpixels using SLIC [1], and found several seeds that were definitely human hands. They then searched the superpixels neighboring the seeds, and performed region-growing from these seeds by merging neighboring superpixels that were highly similar.

We made several modifications to Huang et al.'s [10] technique to fit the particular needs of our system. First, we used gSLICr [25], which is GPU-accelerated and thus faster than SLIC [1], to over-segment the input depth images; and only treated a given superpixel as a seed if more than 80% of the pixels within it were previously detected as user skin.

Second, we did not elect to perform hand detection using the motion pattern relied on by Huang et al. [10]. This tends to make our system more flexible and robust, insofar as the user's hands can freely move, and hence frequently be occluded by the object, and sometimes be off-camera altogether. Such a design also facilitates the detection of hands by our midair touch interface.

Third, Huang et al. [10] described each superpixel using only the HSV histogram and the center position. To ensure better hand detection, we added depth information and Gabor features [15] to such descriptions. Our system computes a histogram with $P = 28$ bins to represent a superpixel, including eight for hue, four for saturation, four for depth and 12 for Gabor features at 0°, 45°, 90° and 135° from the HSV channel, respectively, following the recommendations of Li and Kitani [15].

To segment the hand region by merging similar superpixels, we used four egocentric cues proposed by Huang et al. [10], and those authors' approach, to compute a seed's similarity to its neighbors. These four cues are contrast, location, position consistency, and appearance continuity. Specifically,

$$contrast: S_1(s_k,\hat{s}_u) = \exp(-\sum_{j=1}^{P}(h_{k,j}\log\frac{h_{k,j}}{\hat{h}_{u,j}} + \hat{h}_{u,j}\log\frac{\hat{h}_{u,j}}{h_{k,j}}))$$

$$location: S_2(s_k,\hat{s}_u) = \exp(-\frac{\|c_k - \hat{c}_u\|_2}{l})$$

$$position\ consistency: S_3(s_k,\vec{s}'_w) = \exp(-\frac{\|c_k - \vec{c}'_w\|_2}{l})$$

$$appearance: S_4(s_k,a_v) = \exp(-\sum_{j=1}^{P}(h_{k,j}\log\frac{h_{k,j}}{a_{v,j}} + a_{v,j}\log\frac{a_{v,j}}{h_{k,j}})),$$

where s_k is the kth neighbor of the uth seed $\hat{s}_u \in \hat{S}$; a neighbor is a superpixel that is adjacent to $\hat{s}_u$ and contains skin-color pixels; $\hat{S}$ is the set of all seeds; $h_{k,j}$ and $\hat{h}_{u,j}$ are the value in bin j of s_k and $\hat{s}_u$, respectively. $c_k/\hat{c}_u/\vec{c}'_w$ is the center of a superpixel; s'_w is a superpixel in the user hand region of a previous frame; $a_v \in A$ is a superpixel in the hand region acquired during user calibration (section 3.5); and $a_{v,j}$ is the value in bin j of a_v.

The weighted sum of the maximal similarity scores of these cues among different seeds and hand superpixels are arrived at using Eq. 1, as proposed in Huang et al. [10], but modified the weights κ_2 from 0.2 to 0.15, and κ_3 from 0.2 to 0.25, because continuity across successive frames was more important in our application. $\kappa_1 = \kappa_4 = 0.3$ were set at the same values used in [10].

$$S(s_k) = \max_{\hat{s}_u \in \hat{S}}[\kappa_1 S_1(s_k,\hat{s}_u) + \kappa_2 S_2(s_k,\hat{s}_u)] + \max_{\vec{s}'_w \in \vec{S}'} \kappa_3 S_3(s_k,\vec{s}'_w) + \max_{a_v \in A} \kappa_4 S_4(s_k,a_v) \quad (1)$$

Finally, for each seed, the superpixel with the highest similarity score among its neighbors is added iteratively until there are either no neighbors, or the highest score among all remaining neighbors is too small. After locating all hand regions from among all seeds, we connect them and remove tiny regions containing fewer than 1,600 pixels. Figure 4 presents examples of superpixels and segmentation results.

3.3 Model Analysis and User Guidance

Model analysis comprises computation of the information needed for user guidance, which includes four types of information: capture-distance guidance, complex-geometry detection, object-movement overspeed warning, and occluded-face detection. Figure 5 illustrates the visual representation of each.

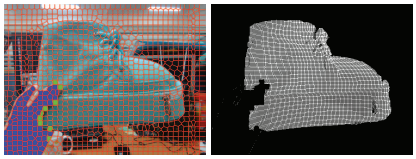


Figure 4: Left: region growing results (blue superpixels are seeds and green ones are growing neighbors); Right: object extraction result.

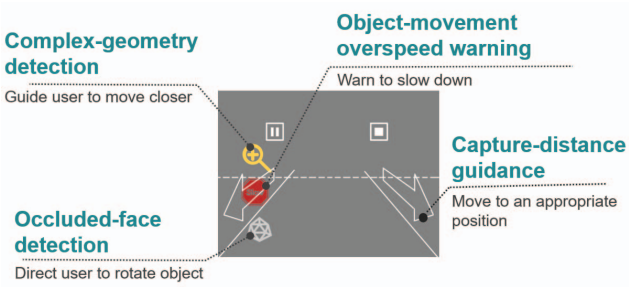


Figure 5: The user interface's guidance-information display. The arrows on both sides instruct the user to move the object to an optimal capture distance. The magnifier icon hints that the user should move the object closer because complex geometries have been detected in the current view. The 'Slow' sign alerts the user that the object is being moved too fast; and the icosahedron icon indicates that a face of the object is occluded and has not been captured enough data

3.3.1 Capture-distance guidance

Users need to put the target object at an appropriate distance away from the depth camera, but the precise distance in question can be difficult to determine, and this often leads to poor capture quality. Therefore, our system counts the number of pixels whose depth values are not within the minimal and maximal capture distances obtained during the user calibration process (section 3.5). According to this pixel count, the system reminds the users to move the object, as follows: 1) if more than 10% of the object volume (measured by pixel count) exceeds the maximal distance, a 'far' line and two inward-pointing arrows are displayed (Figure 5); or 2) if more than 10% of the object is nearer than the minimal capture distance, a 'near' line and two outward-pointing arrows are displayed (Figure 6(d)).

3.3.2 Complex-geometry detection

When an object has some regions that contain more geometric detail than others, it is desirable to move those regions closer to the user, so that the spatial resolution of the camera can be utilized more efficiently — i.e., such that more RGBD data can be acquired in those geometrically complex regions. Since complex geometry often exhibits complex texture (due to more complicated reflections), our system also exploits color information to detect regions with such geometry. This is particularly helpful when the object is farther away from the camera, such that the RGB channels have higher resolutions than the depth channel.

Accordingly, we developed a complex-geometry detection algorithm, which operates as follows. First, the extracted object region (section 3.2) is divided into 2D grids to facilitate analysis of the complexity of both its color and depth data (Figure 7). This analysis is then accomplished via computation of the entropy and gradient orientation in each grid, using Eqs. 2 and 3, respectively. The depth complexity C_d of a grid is the average of its depth entropy and depth gradient orientation. Similarly, the color complexity C_{rgb} is the average of color entropy and color gradient. In our system, the size of each grid is 20 by 20 pixels. More specifically,

$$Entropy = \sum_{i=1}^{M} x_i \log \frac{1}{x_i} = -\sum_{i=1}^{M} x_i \log x_i, \quad \sum_{i=1}^{M} x_i = 1, x_i \geq 0 \tag{2}$$

$$Gradient\ orientation = 1 - \frac{\sqrt{\sum_{i=1}^{m} g(i) - \frac{1}{m}}}{\sigma_{gmax}}, \quad \sigma_{gmax} = \sqrt{(1-\frac{1}{m})^2 + (\frac{1}{m})^2 (m-1)}, \tag{3}$$

where M is the number of pixels with nonzero color value or depth value in each grid; x_i is the normalized color value or depth value of the ith pixel [39]; m is the number of histogram bins; and $g(i)$ is the frequency of the ith bin in the histogram [42]. Entropy measures the complexity within the current view, and gradient orientation measures the perceptual chaos of the image. If both of them have large values, the current view of the object is deemed to contain complex geometries or textures.

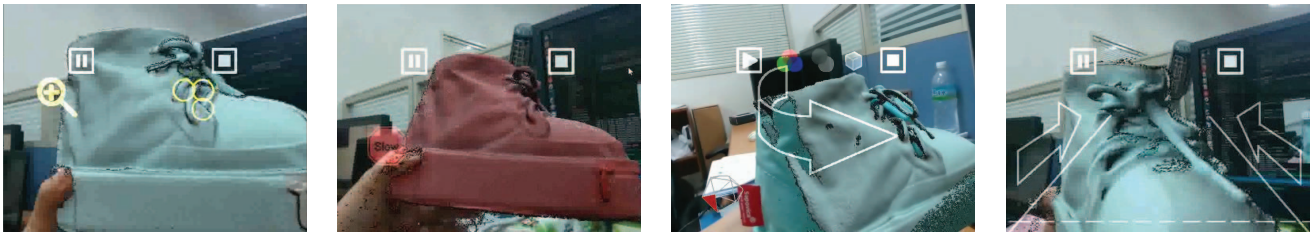
When determining the complexity of a grid, the complexity of the color and depth data are computed separately, and then integrated based on the object's distance from the camera. As the spatial resolution of a depth image is higher when the object is nearer, we consider depth data to be more reliable than color data, and thus assign a higher weighting to the former. Conversely, when the object is farther from the camera, color data are more reliable, and color complexity is given a larger weighting (Figure 8). The complexity C in each grid is calculated as

$$C = w_d * C_d + (1 - w_d) * C_{rgb}, \quad w_d = \frac{D_{removal} - D_{object}}{D_{removal}}, \tag{4}$$

where C_d and C_{rgb} are the depth and color complexity, respectively; D_{min}, D_{max}, and $D_{removal}$ are the minimal capture distance, maximal capture distance, and background removal distance obtained in the user calibration process; and D_{object} is the average depth of all pixels in a grid.

It is worth noting that, when the depth map of an object is complex (even though the depth has been used in the current reconstruction), the object can still be moved further closer to improve acquisition quality in the later reconstruction iterations. On the other hand, color complexity, C_{rgb}, is useful when the object is farther from the camera, where the depth resolution is lower and less reliable. However, the color complexity could be misleading when there is complex texture on a smooth surface.

The complexity computed from the 2D grids in an RGBD image is view-dependent. To integrate the complexities of multiple views during the reconstruction process, therefore, our system maintains a data structure of voxels, each of which corresponds to a 3D point on the object model and stores both a complexity state and a measured distance. As shown in Figure 9, a voxel records the complexity state of a 3D point updated to the most recent frame containing that point. For the current frame, the most up-to-date complexity of each grid is computed; and if such complexity is above a threshold C_h, then the complexity states of those voxels that are within the grid and whose measured distances are greater than the average depth of the grid are marked as 'high'. Those voxels' measured distances are updated as the average depth of the grid; and the complexity states and measured distances of the remaining voxels, i.e., those with measured distances less than the average depth of the grid, are not changed. If a grid's complexity is less than C_h, the system only updates the measured distances of those voxels that are farther than



(a) (b) (c) (d)

Figure 6: All guidance in our system. (a) Grids with complex geometry marked by yellow circles and user hint; (b) 'Slow' icon and red color used to signal the user to slow the movement of the object; (c) Arrow instructing the user to rotate the object. A regular icosahedron with a red face indicates the rough position of an occluded face on the object; (d) Arrows instruct the user to move the object farther.

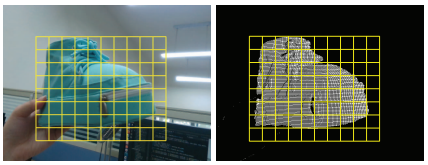


Figure 7: Left: grids on color image; Right: grids on depth image

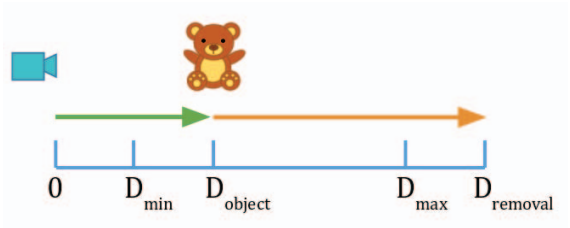


Figure 8: The weights assigned to the depth complexity and color complexity of a grid depend on the average depth of the pixels in that grid

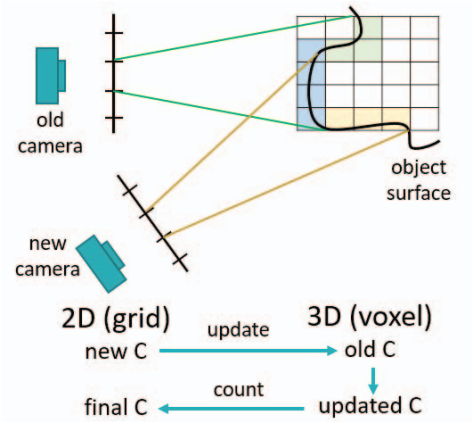


Figure 9: Integration of geometrical-complexity states from multiple views by iteratively updating the corresponding states in the 2D grids and 3D voxels (top view). Complexity in the green voxels are old. Complexities computed from new camera position are used to update the complexity in the blue voxels, whose original complexity were computed from the old camera position, or to set the complexity in the orange voxels, which contained no complexity originally.

the average depth of the grid. We set $C_h = 0.63$ as suggested by Yin et al. [42]. After all voxels have been updated, all those that have high complexity states within each grid are counted, and a given grid is deemed 'complex' if its high-complexity voxels make up more than half the total voxels within it. So that users can recognize these complex grids, which are indicated by circles in the user interface (Figure 6(a)).

It should be noted here that, once a complex grid has been moved closer for capture, it is labeled as 'complete' based on two criteria and will not be modified again. First, if the complexity of a grid is decreased and smaller than a threshold C_h, the complex geometry should have been acquired. Second, if a grid's distance to the camera has reached the minimal capture distance, the grid is labeled as 'complete'. These two criteria avoid the geometry in the grid being affected by other data that are captured farther away from the camera or characterized by low confidence.

3.3.3 Object-movement overspeed warning

When a user manipulates the object, the system estimates the camera position at each frame using the iterative closest point (ICP). The camera position at time k is represented as a transformation matrix:

$$T_k = \begin{bmatrix} R_k & t_k \\ 0 & 1 \end{bmatrix}. \quad (5)$$

$R_k \in SO(3)$ and $t_k \in \mathscr{R}^3$ are the orientation and position of the camera at time k. We use two consecutive camera positions to calculate the camera movement $T_d = T_k^{-1} T_{k+1}$.

Having obtained the translation from t_d, and converted R_d to an axis angle, the system analyzes the camera movement in each frame, smoothing it every five frames to prevent sudden shaking from affecting the results. When the mean translational or mean rotational speeds exceed their respective thresholds (see below), the interface warns the user to move the object more slowly. We set the default translation threshold as $V_{max} = 0.21\ m/s$, averaged speed from TUM datasets [32], while the default rotational speed threshold was set as $\omega_{max} = 90\ deg/s$ empirically. Additionally, the two thresholds are dynamically adjusted to account for the influences of capture distance and the target object's geometrical complexity. Specifically, if the object is farther away, the translational speed threshold is set higher; if there is complex geometry in the current view, both V_{max} and ω_{max} will be reduced to 60% of their original values.

If any movements exceed the thresholds, the interface shows a 'Slow' icon and re-colors the model red to alert the user that too-fast movements render the captured data unreliable (Figure 6(b)).

3.3.4 Occluded-face detection

Pan et al. [21] used an icosahedron to record occluded faces, and showed a 3D arrow to guide their users to complete 3D reconstruction. However, because their system displayed this arrow at all times, users could not rotate the target objects freely. We adopted their

algorithm, but modified their approach by providing users with hints only when strictly necessary: i.e., when he/she pauses the capture process to view the current reconstruction result. To enable users to better understand our system's hints for rotating/moving the object, we also simplified Pan et al.'s original omni-directional arrow to one with just four directions — left, right, up, and down — with the goal of guiding the user as quickly as possible to rotate the face with the highest occlusion score into view. The occlusion score of a given face depends on the angle between the surface normal of that face and the optical axis of the camera coordinate [21].

The system paints the face of the icosahedron corresponding to the side of the target object with the highest occlusion score in red, and shows that face to the user (Figure 6(c)). Specifically, the user can see the arrow in the middle of the interface screen, and the icosahedron with a red face at its bottom left.

3.4 User Interface

As briefly noted above, our user interface overlaps a color image of the 3D model and the augmented guidance information onto the real-world background. Figure 6 presents four examples of what a user might see through the system's AR glasses. Please see also the accompanying video for a demonstration of the system.

To avoid situations in which a reconstructed model fully overlays the real object in the interface to the point that users cannot judge whether the model is a correct reconstruction of the real object, our system allows users to shift between three modes — RGB, depth and mixed — by putting a hand on a virtual button, as shown in Figure 10.

When a user puts his/her hand on a virtual button, the button will turn green and then a progress bar will appear to hint the user s/he needs to keep touching the button for a short duration to confirm this selection. In our implementation, there are two main menu buttons: calibration and reconstruction; and three control buttons: pause, start and stop the reconstruction; and three view mode changing buttons: RGB, depth and mixed mode. All user guidance are provided in the reconstruction stage except that occluded-face detection is executed in the pause stage. In addition, view mode toggle buttons are only shown in the pause stage for user to check the correctness of model. During the reconstruction stage, system is set at the mixed mode. System will save the model after the user presses the stop button.

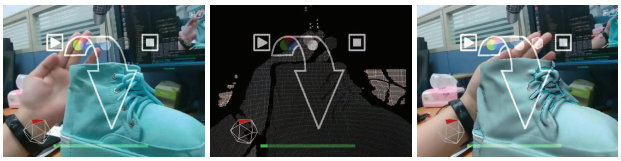


Figure 10: Our system provides three view modes. Left: RGB mode; Middle: depth mode; Right: mixed mode.

3.5 User Calibration for Skin Color and Capture Range

To accommodate the system users' individual differences, the system is capable of obtaining a user's skin-color information, and using it to 1) adjust the thresholds of hand detection, and 2) find the best range of distances at which to capture RGBD images, in light of camera resolution and the individual user's reach. Figure 11 illustrates a user executing this calibration process. The system instructs each user to place the palm and then the back of his/her hand into a hand-shaped guide at four different distances: i.e., 16-22cm, 25-30cm, 35-40cm and 42-47cm. In this process, skin-color ranges and capture ranges are acquired simultaneously.

Skin color ranges. Having acquired the user's skin-color information, our system adjusts the C_bC_r thresholds for that person as follows:

$$\begin{aligned} T_{max} &= 0.3 * U_{max} + 0.7 * O_{max} \\ T_{min} &= 0.3 * U_{min} + 0.7 * O_{min}, \end{aligned} \tag{6}$$

where T's are the adjusted thresholds, U's are the user's skin-color information, and O's are the default thresholds. Assuming that the user follows the system's instructions to place his/her hand at the center of the screen and within the hand-shaped guide lines, there will be only hand data within that shape. Then, we can obtain the maximum and minimum values in the C_b and C_r channels. As the Y values are mainly influenced by lighting rather than by user skin, we elected not to modify the default Y threshold. Similarly, hue (H) and saturation (S) thresholds are adjusted according to users' skin information using Eq. 6. The intensity (V) is influenced by lighting, so the default V threshold was not modified.

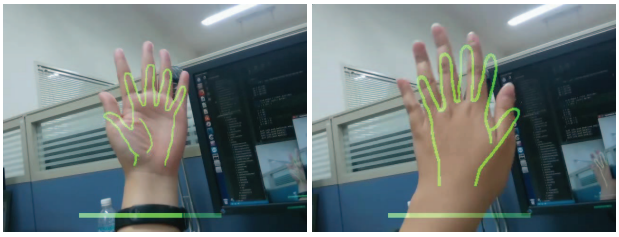


Figure 11: The system guiding a user to calibrate their skin color and capture a set of camera distances most suitable to that individual, with (a) being the palm of the hand at a farther distance (35-40cm), and (b), the back of the hand at a nearer distance of 25-30cm.

Capture range. The spatial resolution of a depth image decreases as the target object's distance from the camera increases. Hence, to best exploit the spatial resolution of an RGBD camera, it is desirable to put an object to be reconstructed as close to it as possible without placing part of the object outside the shot.

Therefore, to determine the best range of distances at which to capture a particular object, we compute the spatial resolution at four different distances as follows:

$$point\ density = \frac{number\ of\ points}{point\ area}. \tag{7}$$

The numerator is the number of points of a region in the depth image, and the denominator is the area of the region in the real world. As such, for computing point density, we need an object or region whose size is known in advance . We use the user's hand as such an object, and the average size of human hands as reported in TheAverageBody[4] as the point area.

The minimal capture distance D_{min} has been set as the distance at which the point density is maximal, and the maximal capture distance D_{max} as the distance beyond which such density is minimal. Additionally, the distance (depth value) at which to remove the background from the depth image, $D_{removal}$, was set as slightly longer than the maximal capture distance. If a user does not perform calibration, the default settings of capture range are 12cm for minimal capture distance, 40cm for maximal capture distance, and 50cm for background-removal distance.

4 Experimental Results

To evaluate our system, we conducted a user study that compared its user experience and reconstruction quality against those of other 3D reconstruction systems. The goals were to analyze whether 1) our system is more user-friendly; 2) its instructions are useful in

[4] **TheAverageBody**:http://www.theaveragebody.com/average_hand_size.php

3D object reconstruction; and 3) AR-assisted reconstruction is more accurate, as compared to its non-AR-assisted counterparts.

All experiments were performed using hardware with the same specifications: i.e., a CPU that supported OpenMP (Intel Core i7-6700 CPU@3.40GHz with four cores and 16GB RAM); a GPU that supported Nvidia CUDA GeForce GTX 1060 with 6GB GRAM; an Intel RealSense SR300 RGBD camera; and Oculus Rift DK2 video see-through AR glasses. The operating system we used was a 64-bit Ubuntu 16.04.

4.1 Experimental Design

We recruited four female and six male participants, all aged 18 to 34. In a within-subjects design, all were asked to operate five state-of-the-art 3D reconstruction systems. As shown in Figure 12, the experimental procedures were as follows: 1) watch an instructional video; 2) practice for five minutes; 3) reconstruct two objects in five minutes apiece; 4) score the system by filling out a seven-point Likert-scaled questionnaire; and 5) provide feedback about the five systems in an interview.

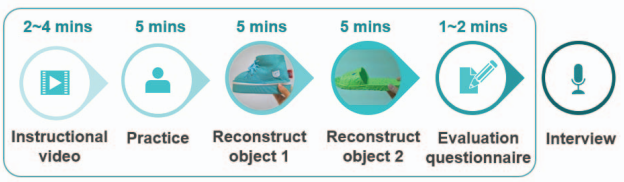


Figure 12: All participants followed the above procedures to test each reconstruction system, after which we interviewed them about their experience

The five reconstruction systems we tested are listed below.

- Ours: A stripped-down version of the proposed system, lacking the user-guidance features.
- Ours_G: Our complete system, including the user guidance features.
- InfiniTAM [13]: A standardized system that allows the user to capture an object by walking a hand-held camera around it.
- Semantic Paint [7]: users wearing a VR glasses can walk around the target object to reconstruct it. Unlike our system, this system only showed the reconstructed model on the screen excluding the color image. This may affect the user's perception to the environment.
- In hand scanner [9, 30]: This system is based on the work of Weise et al. [40], with some modifications. Like ours, it allows the user to rotate the target object in hands during image capture, but with very limited guidance. Via an external screen, it shows a fixed box representing the allowable capture region; a colored-point cloud denoting lose-tracking (red); uncertain noise (white); and valid model points (green).

Apart from In Hand Scanner, which used a different camera (ASUS Xtion Pro Live) and a different 3D reconstruction algorithm, all systems used Intel Realsense SR300 and the same 3D reconstruction engine, InfiniTAM. All of the compared systems were correspondent authors' implementations released on GitHub. We use a counterbalanced design (Latin square) to reduce the influence of learning effects.

The objects to be scanned in the experiment were 1) a blue canvas shoe, its size being 18.7cm $\times$ 7.4cm $\times$ 12.7cm, and 2) a green plastic slipper, 23.5cm $\times$ 9.2cm $\times$ 6.8cm. The shoe had a more complicated structure than the slipper (e.g., included a shoelace).

4.2 User Study Results

4.2.1 Questionnaire

Each participant filled out a seven-point Likert-Scale survey, with 10 items divided into three dimensions: system usability (six questions), e.g., "Was the system easy to use?"; system understanding (two questions), e.g., "Were the system operations for reconstruction easy to understand?"; and model quality (two questions), e.g., "Are you satisfied with the reconstruction results?"

Because our experiment violated the equal-variance assumption, a Welch's t-test was first conducted to check if our systems were significantly different from others, followed by pairwise post-hoc tests. For system usability, we used a Games Howell post-hoc test; and for the other dimensions, we used Dunnett T3, because there were fewer than 50 trials. Main effects were found in all three dimensions ($F(4,147.16)=30.2$, $p<0.001$ for system usability; $F(4,47.28)=7.5$, $p<0.001$ for system understanding; and $F(4,47.25)=10.26$, $p<0.001$ for model quality). Figures 13-15 show the post-hoc test results.

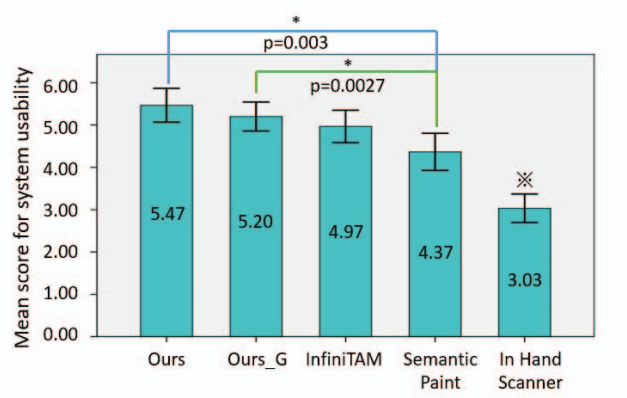


Figure 13: Games Howell post-hoc test results for system usability, indicating that there were significant differences both between Ours and Semantic Paint, and between Ours_G and Semantic Paint. ※denotes that In Hand Scanner was significantly more difficult to use than the other four systems; in all comparisons, $p<0.001$.

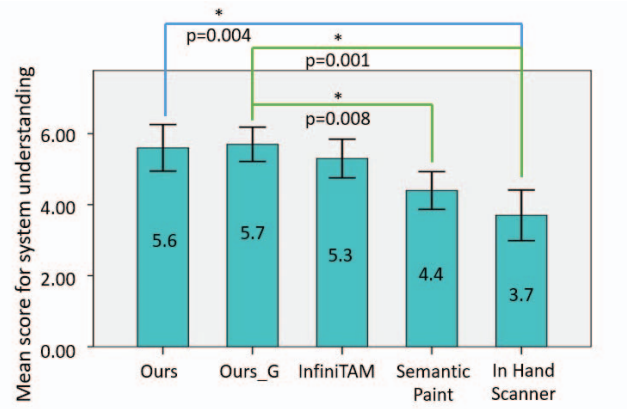


Figure 14: Dunnett T3 statistical analysis results for system understanding, indicating that there were significant differences between Ours and In Hand Scanner; between Ours_G and In Hand Scanner; and between Ours_G and Semantic Paint.

Ours and Ours_G were rated significantly easier to use than Semantic Paint, as shown in Figure 13. Regarding system understanding (Figure 14), Ours was significantly more understandable than

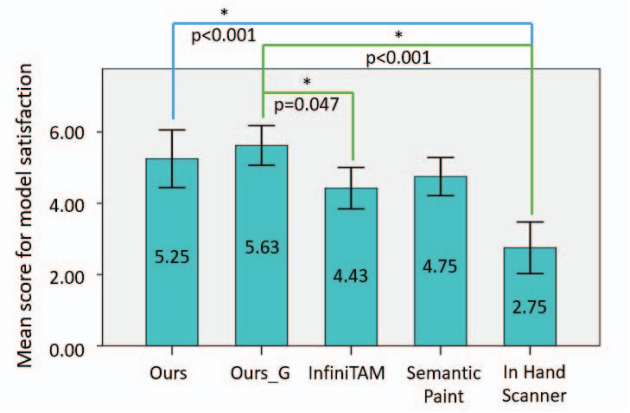


Figure 15: Dunnett T3 statistical analysis results for model quality, indicating that there were significant differences between Ours and In Hand Scanner; between Ours_G and InfiniTAM; and between Ours_G and In Hand Scanner.

In Hand Scanner, while Ours_G was rated significantly better than either Semantic Paint or In Hand Scanner. Lastly, when it came to model quality (Figure 15), the participants were significantly more satisfied with the models reconstructed by Ours than those reconstructed using In Hand Scanner, while the model quality of Ours_G was more rated as more satisfactory than that of either InfiniTAM or In Hand Scanner. In general, our system is considered beneficial among the four systems, as it exceeded each of the other three systems in at least one aspect of the evaluations.

4.2.2 Reconstruction Error

To gain more insight into model quality, we compared it against the ground truth acquired with a HP 3D Structured Light Pro S3 scanner[5], which allowed a resolution/precision up to 0.05% of the scan size. The average error of each system is shown in Figure 16. We adopted Hausdorff Distance to compute the error between the ground-truth model and user-reconstructed models. Both the root mean square (RMS) and mean error of our systems were lower than those of the other three systems. We further performed Dunnett T3 statistical analysis on the reconstruction errors. The results, $F(4, 45.9)=6.36$, $p<0.001$ for RMS error and $F(4, 46.0)=6.96$, $p<0.001$) for mean error, also showed that Ours_G is significantly better than InfiniTAM and Semantic Paint in terms of mean and RMS reconstruction error. Figure 17 shows the reconstruction results of the shoe and slipper by participant P10 and P6, respectively. For all reconstrcution results of the participants, please see the accompanying video.

4.2.3 Interviews

When asked about the preferable system if they planned to engage in such activity, eight of the 10 participants chose our systems; and among those eight, three chose Ours_G. The remaining two participants both chose InfiniTAM.

Regarding the reasons of their choice, participants who selected Ours and Ours_G commented the systems being intuitive and convenient because the virtual model was overlaid on the real object all the time, even during rotation. Five of the eight stated that our user-calibration process for setting parameters was easy to use. Those who selected Ours were also satisfied with the fluency and the model quality of the Ours system, and remarked *"The operation of Ours was convenient and the reconstructed model was good. I could fully*

[5]**HP 3D Structured Light Pro S3**:https://www8.hp.com/us/en/campaign/3Dscanner/overview.html

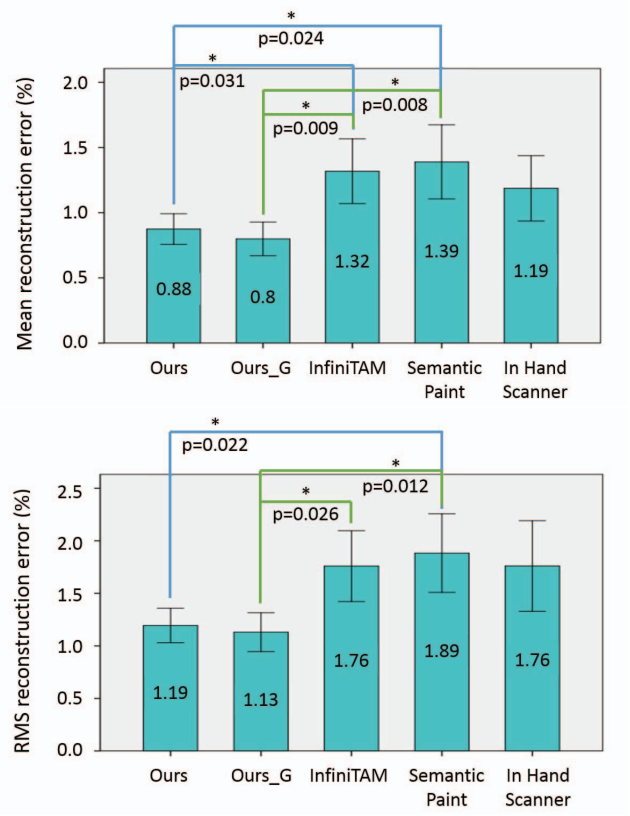


Figure 16: Dunnett T3 statistical analysis results for reconstruction error, indicating that there were significant differences between Ours and Semantic Paint; between Ours_G and InfiniTAM; and between Ours_G and Semantic Paint. The mean and RMS reconstruction errors of each system were measured in percentages (relative error). Although there were no significant differences between In Hand Scanner and the other systems, the quality of models by In Hand Scanner were unstable due to the fact that it could not be learned quickly for novice users.

manipulate the object by myself based on the view shown in the AR glasses" (P1); and *"I think that Ours was easy to use and fluent during reconstruction"* (P2).

All three participants who selected Ours_G commented the helpfulness of its augmented guidance, as follows: *"It [Ours_G] was easier to manipulate the object with those augmented instructions"* (P4); *"The effects of the system [Ours_G] surprised me. The model was beautiful after I followed the instructions"* (P7); and *"I would not have known what to do without [Ours_G's] instructions"* (P9). Interestingly, additional three participants who did not select Ours_G still thought the guidance and augmented instructions were helpful: e.g., *"The guidance would tell me where the reconstruction was incomplete"* (P1), and *"I knew which part of the reconstruction needed to be improved or enhanced, due to the guidance"* (P10).

Of the two participants who did not concern our systems more preferable, one said that despite the higher model quality she could achieve, *"I felt uncomfortable when wearing the AR glasses, so I preferred not to use it"* (P6). P10 who selected InfiniTAM wished to integrate the interactive guidance from Ours_G into InfiniTAM, as she liked the visual hints but could not control objects well in our systems due to trembling hands.

Participants also answered questions related to the user experience, the pros and cons of reconstruction from the user's viewpoint,

Figure 17: Ground truth and some reconstruction results by participants using the five systems in the user study. The results of blue shoe were reconstructed by P10 and the results of green slipper were reconstructed by P6.

how reconstruction based on manipulating the object differed from that based on moving the camera, and their understanding of the systems' processes. We coded their replies, and summarized the results in Table 1. The number after each comment indicates the number of participants who shared the same opinion. The manipulation was deemed intuitive and the visual guidance comprehensive in both Ours and Ours_G. Six participants felt that the guidance in Ours_G was helpful even though it sometimes resulted in information clutter on the screen. Some participants thought InfiniTAM (N=6) and Semantic Paint (N=3) were convenient because they just needed to control the camera, but noted that both those systems readily lost tracking, and did not allow the target object to be moved.

In Hand Scanner provided rich settings that, while not friendly to novice users, allowed expert users to optimize their final results. Ours, Ours_G and In Hand Scanner systems were all criticized as causing hand fatigue. However, both versions of our system, Ours and Ours_G, allowed users to break down the capture process into multiple sessions, between which they could put down the target object and take a rest.

Our system was able to automatically recover tracking based on previously captured model data to continue the reconstruction. Although In Hand Scanner also provided a re-tracking function, its users needed to align the model manually whenever it lost tracking.

Summary of Interview Data		
System Name	Pros	Cons
Ours	Intuitive manipulation (8) Good model (6) Comprehensible icons (6)	Hand fatigue (4)
Ours_G	Intuitive manipulation (8) Good model (7) Comprehensible icons (6) Helpful instructions (6)	Too much info (5) Hand fatigue (5) Latency (2)
InfiniTAM	Convenient (6)	Loses tracking (7) Object not moveable (6)
Semantic Paint	Clear view (4) Convenient (3)	Loses tracking (8) Object not moveable (4) No directions (3)
In Hand Scanner	Flexible (2)	Complicated settings (10) Bad model (4) Hand fatigue (3)

Table 1: *Note.* Numbers in parentheses indicate how many participants shared the same opinion.

4.3 Discussion

This section analyzes possible causes of the differences in the reported usability of the tested systems, and makes some observations about the user behaviors collected during the user study, before proceeding to discuss the present work's limitations and possible directions for future research on this topic.

4.3.1 System usability and reconstruction quality

Ours vs. Ours_G. Ours_G was slightly better than Ours in system understanding, model satisfaction, and reconstruction error, but slightly worse than Ours in system usability. The differences between Ours and Ours_G were not statistically significant though. A possible explanation could be that the usability questionnaire measures easy-to-use, conveniences, and intuitiveness of a system in general and the only difference between Ours and Ours_G is whether visual guidance is provided. Hence, the visual guidance may not significantly affect the overall usability since both Ours and Ours_G are intuitive and easy to use. Nevertheless, the interview results still revealed that visual guidance indeed offered helpful instructions (6 participants), but that it would be better if its presentation could be improved (5 participants). Further study on the influences of visual guidance should be conducted in the future to better understand and improve its effect.

InfiniTAM & Semantic Paint vs. Ours & Ours_G InfiniTAM was significantly worse than Ours_G on the model satisfaction and reconstruction error. Semantic Paint was significantly worse than Ours_G on the system understanding and reconstruction error. These suggest that visual guidance could assist the participants to reconstruct better models.

The worse reconstruction error of InfiniTAM and Semantic Paint could also be partly attributed to that they were intended for 3D scene reconstruction rather than for object reconstruction. Therefore, the input images in the two systems were not processed in ways aimed at separating the foreground object from the scene; and this could negatively impact the quality of the resulting object models in both cases.

The usability of Semantic Paint was rated significantly lower than that of Ours and Our_G. This reveals that the 3D reconstruction engine might also have an effect to the usability as both Ours and Ours_G adopted InfiniTAM as the core 3D reconstruction algorithm.

In Hand Scanner. This system, which provides expert settings with many adjustable parameters, received the worst scores in our user study. Particularly, it was significantly worse than Ours and Ours_G in system usability, system understanding, and model satisfaction. This was because novice users who lacked the relevant background knowledge had difficulty even getting started with it, let alone becoming competent at using it within the limited time availability. Besides, as the viewpoints of camera and a user were different and the background image were not shown to the user, the novices had to spend more time on practicing this system.

4.3.2 User behaviors

Manipulating an object by hand gives rise to stability and fatigue issues. As briefly noted above, one of our participants had a hand tremor that negatively impacted the quality of the reconstructions. Two participants worked around the fatigue, resting their elbows against a surface during the reconstruction, which also served to reduce involuntary hand trembling. The system can assist users with e.g., a recommendation of the elbow posture; suggesting the user take breaks during the capture process to avoid fatigue; and/or adding detection and stabilization for trembling motions.

4.3.3 Limitations and directions for future research

The present research has several limitations that should be acknowledged. First, our system relied on several CPUs and GPUs for real-time reconstruction; and AR glasses also require a powerful PC to run. Both these factors mean that the designed system requires a considerably higher computer specification than its counterparts if it is to outperform them. Nevertheless, the total financial cost of our system's components remain lower than that of a professional 3D scanner, while also being more beginner-friendly.

Second, some objects cannot be reconstructed using our system: for example, materials that are too shiny or transparent for depth cameras to capture them, and textureless objects whose colors are similar to the user's hands. The latter problem might be solved by improving the hand detection method.

Third, our object-extraction method only removed backgrounds and hands, so a scene that is too close to the target object would be reconstructed at the same time, resulting in annoying noise. This limitation could be eliminated through a better object segmentation algorithm.

Fourth, our study did not detect significant difference when introducing visual guidance to the system in terms of the user experience captured by our questionnaire. More rigorous questionnaire designed for evaluation of handheld augmented reality interfaces suggested in Santos et al.'s research [31] and also in [19] may help to better detect the potential benefits by incorporating different aspects, such as comprehensibility. Moreover, we measured user performance based on quality of the reconstructions. Other measures such as the needed time and the amount of movement may be considered in future study as recommended by [23].

In the future, we would like to improve our system in the following ways. First, although most of the participants in our user study expressed an interest in augmented instructions, they also thought that there was room for improvement in the existing instructions' timing, quantity, and presentation methods — which at worst could interfere with reconstruction operations, leading to an unsatisfactory user experience. How to provide these augmented guidance notes at the right moments, in appropriate numbers, and in ways that do not interfere with user comfort are important issues for an AR-assisted system. And second, as noted above, the object-extraction algorithm we used could not work well if the object was too close to the background. To solve this problem, improved hand-detection and object-segmentation algorithms should both be developed. Finally, we would also like to improve our reconstruction algorithm to handle shiny or transparent objects. We hope these improvements in the future, including hand detection, object extraction and reconstruction, will make further technical contributions that had not been addressed in this paper.

5 Conclusion

This paper has presented an AR-assisted 3D object-reconstruction system with a user-friendly interface and visual guidance, aimed at assisting users to create more satisfactory and accurate models intuitively. By integrating the user's own viewpoint with that of the input camera, the system allows him/her to focus entirely on the task at hand: i.e., without needing to check whether the virtual object is still on the screen, and whether the camera is still targeting the object simultaneously. Additionally, this type of overlapping allows users to notice more quickly when their models lose tracking. Last but not least, the proposed system analyzes users' models during reconstruction and presents visual suggestions for improvements, including capture-distance guidance, complex-geometry detection, object-movement overspeed warnings, and occluded-face detection. This, too, saves the user time by enabling timely adjustments to be made, rather than restarting from scratch if anything has gone wrong.

We conducted a user study to verify the effects of our methods. Most of the participants preferred one or the other version of our system, citing the intuitiveness of its object-manipulation approach, its good reconstruction results, and the usefulness of the visual guidance. Analysis of the evaluation questionnaire results indicated that our system was significantly better than the other three methods in system usability, system understanding, and model satisfaction. while our systems' reconstruction errors were much smaller than those of the other three state-of-the-art systems that were tested.

Acknowledgments

We thank Prof. Chin-Tien Wu and Dr. Chien-Lun Lai for helping us acquire the ground-truth models.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, Nov 2012.

[2] P. Baudisch and S. Mueller. Personal fabrication: State of the art and future research. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pp. 936–939, 2016.

[3] Y.-P. Cao, L. Kobbelt, and S.-M. Hu. Real-time high-accuracy three-dimensional reconstruction with consumer RGB-D cameras. *ACM Trans. Graph.*, 37(5):171:1–171:16, 2018.

[4] H. Du, P. Henry, X. Ren, M. Cheng, D. B. Goldman, S. M. Seitz, and D. Fox. Interactive 3D modeling of indoor environments with a consumer depth camera. In *Proceedings of the 13th International Conference on Ubiquitous Computing*, pp. 75–84, 2011.

[5] V. Fremont and R. Chellali. Turntable-based 3D object reconstruction. In *IEEE Conference on Cybernetics and Intelligent Systems, 2004.*, vol. 2, pp. 1277–1282, Dec 2004.

[6] K. Fudono, T. Sato, and N. Yokoya. Interactive 3-D modeling system using a hand-held video camera. In *Scandinavian Conference on Image Analysis*, pp. 1248–1258. Springer, 2005.

[7] S. Golodetz, M. Sapienza, J. P. C. Valentin, V. Vineet, M.-M. Cheng, A. Arnab, V. A. Prisacariu, O. Kähler, C. Y. Ren, D. W. Murray, S. Izadi, and P. H. S. Torr. SemanticPaint: A Framework for the Interactive Segmentation of 3D Scenes. Technical Report TVG-2015-1, Department of Engineering Science, University of Oxford, October 2015. Released as arXiv e-print 1510.03727.

[8] K. Häming and G. Peters. The structure-from-motion reconstruction pipeline–a survey with focus on short image sequences. *Kybernetika*, 46(5):926–937, 2010.

[9] D. Holz, A. E. Ichim, F. Tombari, R. Rusu, and S. Behnke. Registration with the point cloud library - a modular framework for aligning in 3-D. *IEEE Robotics & Automation Magazine*, 22(4):110–124, 2015.

[10] S. Huang, W. Wang, S. He, and R. W. H. Lau. Egocentric hand detection via dynamic region growing. *ACM Trans. Multimedia Comput. Commun. Appl.*, 14(1):10:1–10:17, Dec. 2017.

[11] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: Real-time 3D reconstruction and interaction using a moving depth camera. pp. 559–568. ACM, October 2011.

[12] O. Kähler, V. A. Prisacariu, and D. W. Murray. Real-time large-scale dense 3D reconstruction with loop closure. In *14th European Conference on Computer Vision*, pp. 500–516, 2016.

[13] O. Kähler, V. A. Prisacariu, C. Y. Ren, X. Sun, P. H. S. Torr, and D. W. Murray. Very high frame rate volumetric integration of depth images on mobile device. *IEEE Transactions on Visualization and Computer Graphics (Proceedings International Symposium on Mixed and Augmented Reality 2015)*, 22(11), 2015.

[14] G. Kukharev and A. Nowosielski. Visitor identification-elaborating real time face recognition system. In *WSCG*, 2004.

[15] C. Li and K. M. Kitani. Pixel-level hand detection in ego-centric videos. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 3570–3577, 2013.

[16] D. S.-M. Liu and T.-L. Chang. Interactive guidance and navigation for facilitating image-based 3D modeling. In *WSCG*, 2014.

[17] R. A. Newcombe, S. Izadi, O. Hilliges, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE ISMAR*, 2011.

[18] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3D reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (TOG)*, 2013.

[19] B. Nuernberger, K.-C. Lien, L. Grinta, C. Sweeney, M. Turk, and T. Höllerer. Multi-view gesture annotations in image-based 3D reconstructed scenes. In *22nd ACM Conference on Virtual Reality Software and Technology*, pp. 129–138, 2016.

[20] V. Oliveira and A. Conci. Skin detection using HSV color space. In *H. Pedrini, & J. Marques de Carvalho, Workshops of Sibgrapi*, pp. 1–2, 2009.

[21] Q. Pan, G. Reitmayr, and T. W. Drummond. Interactive model reconstruction with user guidance. In *2009 8th IEEE International Symposium on Mixed and Augmented Reality*, pp. 209–210, Oct 2009.

[22] P. Panteleris, N. Kyriazis, and A. A. Argyros. 3D tracking of human hands in interaction with unknown objects. In *Proceedings of the British Machine Vision Conference*, pp. 123.1–123.12, September 2015.

[23] J. Polvi, T. Taketomi, G. Yamamoto, A. Dey, C. Sandor, and H. Kato. Slidar: A 3d positioning method for slam-based handheld augmented reality. *Computers & Graphics*, 55:33 – 43, 2016. doi: 10.1016/j.cag.2015.10.013

[24] C. Y. Ren, V. Prisacariu, D. Murray, and I. Reid. STAR3D: Simultaneous tracking and reconstruction of 3D objects using RGB-D data. In *2013 IEEE International Conference on Computer Vision*, pp. 1561–1568, Dec 2013.

[25] C. Y. Ren, V. A. Prisacariu, and I. D. Reid. gSLICr: SLIC superpixels at over 250hz. *ArXiv e-prints*, sep 2015.

[26] A. Richardson, J. Strom, and E. Olson. Aprilcal: Assisted and repeatable camera calibration. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1814–1821, 2013.

[27] T. Roumen, B. Kruck, T. Dürschmid, T. Nack, and P. Baudisch. Mobile fabrication. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pp. 3–14, 2016.

[28] M. Rünz and L. Agapito. Co-fusion: Real-time segmentation, tracking and fusion of multiple objects. In *IEEE International Conference on Robotics and Automation*, pp. 4471–4478, May 2017.

[29] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy. Real-time 3D model acquisition. *ACM Transactions on Graphics (TOG)*, 21(3):438–446, 2002.

[30] R. B. Rusu and S. Cousins. 3D is here: Point cloud library (pcl). In *2011 IEEE International Conference on Robotics and Automation*, pp. 1–4, May 2011.

[31] M. E. C. Santos, T. Taketomi, C. Sandor, J. Polvi, G. Yamamoto, and H. Kato. A usability scale for handheld augmented reality. In *Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology*, pp. 167–176, 2014.

[32] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 573–580, 2012.

[33] K. Tateno, F. Tombari, and N. Navab. When 2.5D is not enough: Simultaneous reconstruction, segmentation and recognition on dense slam. In *IEEE International Conference on Robotics and Automation*, pp. 2295–2302, 2016.

[34] K. Tateno, F. Tombari, and N. Navab. Large scale and long standing simultaneous reconstruction and segmentation. *Computer Vision and Image Understanding*, 157(Supplement C):138 – 150, 2017.

[35] S. Tsekeridou and I. Pitas. Facial feature extraction in frontal views using biometric analogies. In *9th European Signal Processing Conference*, pp. 1–4, 1998.

[36] S. Tsukizawa, K. Sumi, and T. Matsuyama. 3D digitization of a hand-held object with a wearable vision sensor. In *Computer Vision in Human-Computer Interaction*, pp. 129–141, 2004.

[37] D. Tzionas and J. Gall. 3D object reconstruction from hand-object interactions. In *International Conference on Computer Vision*, 2015.

[38] J. Valentin, V. Vineet, M.-M. Cheng, D. Kim, J. Shotton, P. Kohli, M. Niessner, A. Criminisi, S. Izadi, and P. H. S. Torr. SemanticPaint: Interactive 3D Labeling and Learning at your Fingertips. *ACM Transactions on Graphics*, 34(5), 2015.

[39] H. Wang, J. Duan, X.-H. Han, and B. Xiao. Research on image complexity evaluation method based on color information. In *LIDAR Imaging Detection and Target Recognition 2017*, vol. 10605, p. 106051Q, 2017.

[40] T. Weise, T. Wismer, B. Leibe, and L. V. Gool. In-hand scanning with online loop closure. In *IEEE 12th International Conference on Computer Vision Workshops*, pp. 1630–1637, Sept 2009.

[41] T. Weise, T. Wismer, B. Leibe, and L. V. Gool. Online loop closure for real-time interactive 3D scanning. *Computer Vision and Image Understanding*, 115(5):635 – 648, 2011.

[42] K. Yin, L. Wang, and Y. Guo. Fusing multiple visual features for image complexity evaluation. In *Pacific-Rim Conference on Multimedia*, pp. 308–317. Springer, 2013.