Interactive Room Capture on 3D-Aware Mobile Devices

Aditya Sankar
University of Washington
Seattle, WA, USA
aditya@cs.washington.edu

Steven M. Seitz
University of Washington
Seattle, WA, USA
seitz@cs.washington.edu

(a) Real Scene (b) User interacting with our system (inset) (c) Generated Model

Figure 1: With our system, a casual user can generate a representative 3D CAD model of a room, in a few minutes, on a '3D-aware' mobile device. The generated model is editable and contains scene semantics, enabling interior design applications.

ABSTRACT

We propose a novel interactive system to simplify the process of indoor 3D CAD room modeling. Traditional room modeling methods require users to measure room and furniture dimensions, and manually select models that match the scene from large catalogs. Users then employ a mouse and keyboard interface to construct walls and place the objects in their appropriate locations. In contrast, our system leverages the sensing capabilities of a 3D aware mobile device, recent advances in object recognition, and a novel augmented reality user interface, to capture indoor 3D room models in-situ. With a few taps, a user can mark the surface of an object, take a photo, and the system retrieves and places a matching 3D model into the scene, from a large online database. User studies indicate that this modality is significantly quicker, more accurate, and requires less effort than traditional desktop tools.

ACM Classification Keywords

H.5.2 [Information Interfaces and Presentation]: User Interfaces - Graphical User Interfaces (GUI); I.3.6 [Computer Graphics]: Methodology - Interaction Techniques

Author Keywords

Design; Human Factors; 3D Modeling; CAD; Interactive Modeling; Mobile; Augmented Reality; User Study;

UIST 2017, October 22–25, 2017, Quebec City, QC, Canada

 ISBN 978-1-4503-4981-9/17/10…$15.00

DOI: https://doi.org/10.1145/3126594.3126629

INTRODUCTION

The advent of '3D aware' mobile devices (phones, tablets, headsets, etc.) such as Google Tango [31], Microsoft Hololens [15], and Meta2 [1] opens up interesting new opportunities to map and understand our world. In this work, we explore the specific problem of 3D CAD room model capture using such devices.

A captured CAD room model is a representative, editable model of a real room, that is composed of CAD models of room elements such as walls, furniture, doors, windows, and artwork. These individual object models can be authored from scratch with interactive modeling tools such as AutoCAD or Sketchup. Recent large-scale efforts [45, 8] have collected and indexed several thousands of these user-generated object models. Leveraging this large corpus of object models, the task of CAD room capture can be reduced to selecting, for each object in the scene, a closely matched CAD model and placing it at an appropriate location within the scene. The resulting 3D CAD room models enable a variety of compelling applications ranging from interior design, to virtual walkthroughs, and more recently, interactive virtual and augmented reality experiences.

Several desktop and web tools [16, 2, 28] have been developed based on this idea, and they usually contain a curated set of object models to choose from. However, creating CAD room models is still a time consuming task for users even with these specialized tools. This is due to a few reasons: 1) the user needs to manually measure a scene in order to construct an accurate model, 2) finding matching furniture in large catalogs is frustrating and time consuming, and 3) CAD modeling interfaces tend to be complex and pose a steep learning curve for new users.

In this work, we present a new approach to CAD room modeling by interactively recognizing and placing representative object models in the scene, by harnessing the power of 3D aware devices and recent advances in object recognition. By recognizing and placing the objects in-situ, via an augmented reality interface, this method mitigates the need for manual measurement of the scene, and facilitates selection of models from large databases.

A 3D aware device is one that can accurately track its 3D position and orientation in the world (6DoF tracking) and can estimate the depth of points in a scene. Equipped with sensors that enable motion tracking and depth sensing, these devices enable 'scanning' a room by capturing depth maps and fusing them into a point cloud or mesh model. However, the resulting depth-based models are typically incomplete (with lots of holes), are time consuming to create, lack semantics, and cannot be easily edited/rearranged. In contrast, a CAD room model overcomes many of these issues – it represents the room in terms of the same kinds of CAD primitives that architects and designers are accustomed to. Because the sensing capabilities of these devices alone are not sufficient to produce a CAD model, we therefore introduce an interactive technique that additionally employs a human-in-the-loop in order to gain a better semantic understanding of the scene and efficiently capture a CAD model of the environment.

The primary contribution of our work is the design of a novel user interface for in-situ CAD room modeling that combines the above mentioned elements into a robust, end-to-end, functional 3D modeling tool that allows users to create rich semantic models of their environment. As part of our system, we also contribute a novel CAD object model alignment algorithm that is able to accurately place a 3D object in a scene in the matter of a few seconds, despite the presence of noise in the sensor data. Finally, we present results from a within-subjects user study that compares the usability of our system to a popular desktop room modeling tool. Quantitative and qualitative observations from this study indicate that 3D aware in-situ CAD room modeling offers significant advantages over traditional desktop tools.

RELATED WORK

3D Aware Devices and Applications: Tango-enabled [31] smartphones like Lenovo Phab 2 Pro [30] and AsusZenFone AR [5], and head-mounted devices such as HoloLens [15] and Meta2 [1] have just come to market in the last year. They provide powerful new spatial awareness capabilities, notably the abilities to measure depth and accurately self-localize in 3D space, that open up many new opportunities in AR and 3D interaction research. Our paper is the first to leverage Tango for CAD-based furniture recognition and placement.

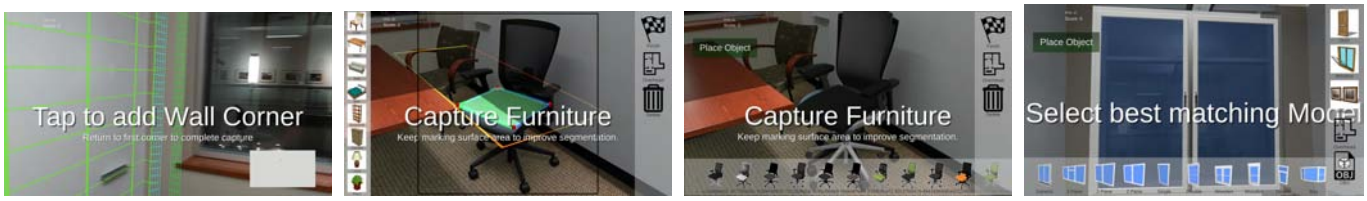
MagicPlan [24] is an augmented reality floor plan capture app that has been ported to Tango. The app is able to reconstruct room shape and dimensions using the spatial mapping data, but to model furniture, they rely on a manual drag and drop interface similar to desktop tools. Lowe's Vision [19] and AR Home Designer [10] are spatially aware apps for placing new virtual items into a scene. They do not, however, capture existing objects in the room. SnapToReality [26] extracts physical constraints, such as planes, from a scene in a manner similar to ours, but also does not capture existing objects.

Virtual Home Design Software: Several desktop [40, 2] and web-based tools [16, 28, 29, 3, 32] are available to capture 2D and 3D building and furniture models. They are, almost exclusively, 'drag and drop' interfaces where blocks of furniture, doors, windows, appliances etc. can be manually positioned around the scene. A main drawback of these tools is the need for manual measurement and object model selection. These tools serve as a useful baseline for our work, and we compare results.

In-Situ 3D Modeling: In-situ indoor modeling has been explored by several groups. Ishikawa et al [18] demonstrate interactive plane based 3D modeling on images but require that all objects be approximated by planes. Others [27, 20, 21] enable building simple polyhedral shapes using a sketching interface. Others have explored free form in-situ sketching of virtual [46] and physical [4] objects. While these approaches work well for relatively simple objects, modeling more complex geometries is time-consuming, and unnecessary in our context, when a corresponding CAD model already exists. In our previous work [36] we generated CAD-based room and furniture models using an ordinary tablet, but the lack of 3D-aware sensing restricts the user to rotational motions (you can't walk around the scene freely), and cannot produce metrically accurate models without manual scale adjustment. Most related to our work, Shao et al. [39] use depth sensing and matching to a database of CAD models for furniture recognition and placement. A key difference, however, is that their system does not operate on a mobile device – rather they take an RGBD photo and use desktop computer for user-guided modeling. In contrast our contribution is a UI and system in which all interactions occur in-situ, on a 3D-aware mobile device, enabling users to interact with the system (and the room) in real-time, while the images are being captured. Our system leverages ShapeNet [8], a large-scale dataset of CAD models curated from the Internet which contains several thousand furniture models for each category, more than an order of magnitude more objects than is supported by [36] (65 objects), and [39] (145 objects). Finally, we note that neither [36] nor [39] have been user tested and compared to existing commercial systems, one of the contributions of this paper.

3D Object Recognition: Our work leverages recent advances in 3D object recognition in images, from large exemplar databases. In particular, we build upon work by Li et al. [22] in order to select matching furniture models from ShapeNet. A variety of other techniques [23, 6, 17] have focused on automatically aligning 3D furniture models to single images. All of these approaches focus heavily on automatic detection and placement, which can be error prone. An advantage of interactive systems such as ours is the ability for users to correct retrieval errors in real-time, therefore increasing the likelihood of success in their modeling task.

Fully Automatic 3D Reconstruction: Several stereo [41, 14, 44] and depth-based [25, 9, 11, 37] methods exist for "scanning in" mesh models of objects and scenes. Acquir-



(a) Capture Walls (b) Capture Surface (c) Placed Furniture (d) Window Added

Figure 2: User's view of the system, during various stages of the modeling process. Bold text in the center provides the user with instructions for the current phase of capture. The inset in Fig 2a shows a top down view of the walls as they are captured. The thin square frame in Fig 2b provides a guide for image capture. The thumbnails in Figs 2c & 2d show alternate model choices.

ing detailed and hole-free models, however, is quite time-consuming, as you need to pass the sensor over every surface including the undersides of objects where it may be difficult to move the camera. The resulting models typically consist of millions of polygons that lack semantics and are not easily editable. Whereas our system produces clean, hole-free, semantic, editable CAD models that can be efficiently compressed/transmitted and are compatible with a variety of existing tools. Salas-Moreno et al. [34] propose a hybrid technique consisting of a real-time SLAM++ system that recognizes previously reconstructed objects and inserts them into the scene in real-time. While their system only supports a handful of objects and does not run on a mobile device, it is purely automatic. SemanticPaint [43] uses interaction to segment and label fused point clouds, but their models are not as clean and easy to edit as CAD models.

DESIGN GOALS AND OBSERVATIONS

Our overall goal is to construct a 3D model that is useful for room-scale interior design projects such as furniture rearrangement, remodeling and redecorating. We gain inspiration from popular room modeling tools [16, 2, 28], and attempt to produce similar results, but with a fraction of the manual effort.

We therefore define the following features for our system: 1) Record dimensions of walls to create a floor plan layout of the room, 2) Select representative 3D models of existing furniture and place them at appropriate locations within the virtual floor plan, 3) Model structural elements in the room, such as doors/windows, and finally 4) Transfer artwork, wallpaper and other textural information from the scene into the model.

We also make a number of simplifying assumptions, based on observations made during the process of system development. These assumptions help to simplify the system and user interactions.

First, we restrict our attention to furniture items that contain a dominant horizontal surface. Most functional furniture items contain such a surface. For instance, chairs provide a place to sit, tables provide a surface upon which to work or eat, beds provide a surface to lie on, and so forth. Furthermore, this horizontal surface uniquely determines the position, scale, and orientation of the furniture item within the scene, making it a useful primitive for interaction.

Second, we assume that all furniture in the scene can be well-approximated by furniture models in the ShapeNet database. By aligning the dominant horizontal surfaces of the model and the real object, we can therefore place the model at the appropriate scale and location within the scene.

Third, we assume that the room interior is bounded by a connected set of flat, vertical walls (i.e., there is no opening other than the doors).

Finally, we assume that doors, windows, and artwork lie in/on vertical walls, and bounded by axis-aligned rectangles, allowing them to be specified by two corners on a wall. We also observed that windows and artwork, in particular, often contain glass or other specular materials that cannot be reliably detected by depth sensors.

In the discussion section, we evaluate how these assumptions affect the generality of our technique, discuss limitations and propose possible improvements for future work.

USER'S VIEW OF THE SYSTEM

We first describe our system from a user's perspective. To get a complete understanding of the interactive aspects of our system, we request readers to view the accompanying supplementary video. Our system is implemented on a Tango-enabled [31] Lenovo Phab 2 Pro, but the underlying technique is designed to work on other emerging 3D-aware platforms.

Modeling with our system proceeds in three phases (note that the order of phases 2 and 3 is interchangeable):

1) Floor, Ceiling, and Wall Capture: (Figure 2a) To begin, the user launches our application and is presented with a live view of the scene as seen from the device's RGB camera. Bold text in the center of the screen provides instructions for the user. First, the user is asked to capture the floor and ceiling by pointing the device at them and holding still for a second. A visible UI cue and an audible click indicate that these have been scanned. The cursor then changes to resemble the edge between two walls and the user is asked to aim the device at a wall edge. They can then tap the screen to being wall capture. As the user moves and rotates around the room, they mark each subsequent wall edge. A bright grid visualizes the walls as they are being captured. Upon returning to the first wall edge, the system automatically completes the room

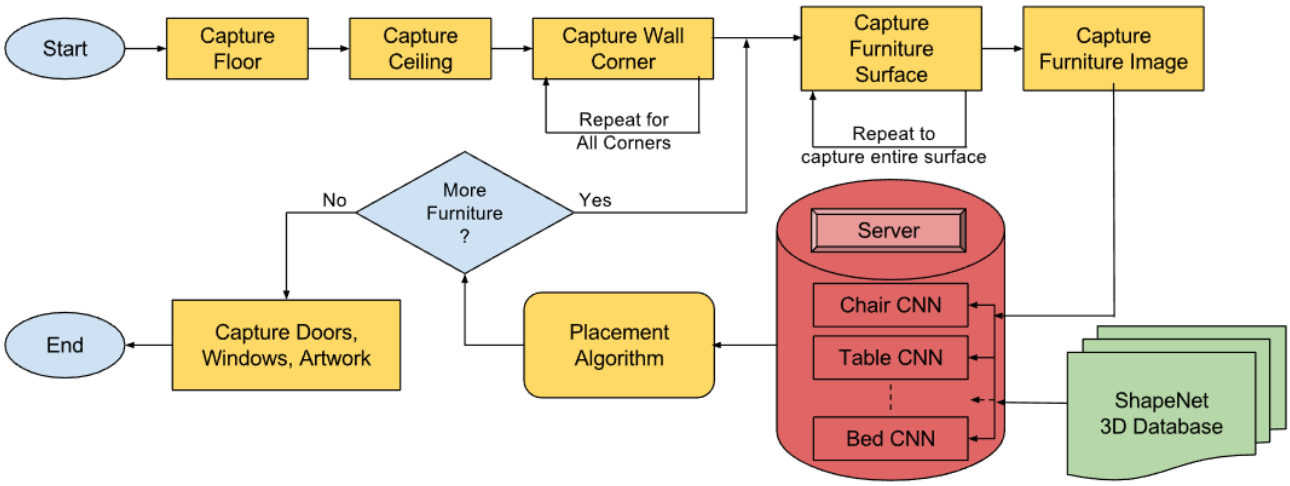


Figure 3: System overview and workflow. Elements in yellow are implemented as part of our system and run entirely on the mobile device hardware. Elements in red and green are components from related work that reside on the server.

shape and the captured walls are rendered in a solid color, to indicate that wall scanning is complete.

2) Furniture Capture: (Figures 2b and 2c) To capture furniture, the user aims the device at the dominant horizontal surface of the furniture item, such as the seat of a chair or the top surface of a table. As the user aims the device, a preview outline appears in 3D, highlighting the extents of the currently identified surface. Once the user is satisfied with the preview, they tap the screen to confirm the selection. The selected surface is then represented as a persistent solid green quadrilateral. The user is then instructed to move back to a position where the entire object is within the camera's view and is prompted to take a photograph. A menu appears from the right with buttons corresponding to supported furniture types (Chair, Table, Sofa, Bed, Bookshelf, Cabinet), and the user taps the button corresponding to the type of object being captured. Within 2-4 seconds, an object recognition system returns a list of matching 3D models and they are displayed as thumbnails on the bottom of the screen. The CAD model corresponding to the first result is loaded and is overlaid on the live view of the scene, in the same position and orientation as the original object that was being captured. If the first result is unsatisfactory, the user has the option of trying out various models from the list in order to find the one that best resembles the real object. Once satisfied, they press 'Place Object' to confirm, and repeat this workflow for all furniture items in the scene. For larger surfaces that do not fit within the camera's field of view, such as tables or couches, the user may select the surface in multiple stages, with each tap automatically improving the surface segmentation. The user also has the option of repeating a previously captured item, in cases where the scene contains multiple pieces of identical furniture (e.g. dining chairs).

3) Door, Window, and Artwork Capture: (Figure 2d) Once all furniture objects have been captured, the user proceeds into the final phase of modeling windows, doors and artwork. To capture these objects, the user is prompted to mark their top right and bottom left corners. The cursor changes to resemble the corner shape to assist the user. As they mark the surface, a planar proxy is rendered to indicate the extent of the selection. They then select the type of object and, similar to furniture, they can chose the best match from a list of models. For artwork, the user is additionally prompted to take a photo of the artwork, whose texture is then automatically extracted and transferred into the 3D model.

Once modeling is complete, the user may view the model in one of a three ways: A) by using the augmented reality view to look around the real scene and see the CAD model overlaid on top of it. B) they can also disable the see-through camera and only view the virtual scene, with less visual distraction (as in Figures 1c and 9, column 2), or C) they can view a top-down rendering of the scene to reveal the floor plan and 2D furniture placement (Figure 9, column 3).

SYSTEM COMPONENTS

Our application is built on the Tango platform [31] which provides three key features that we leverage in our system: 1) *Motion Tracking* accurately determines the device's movements in 6DoF space. 2) *Area Learning*: As the device explores an environment, it stores an optimized map that can be re-used later in order to improve tracking and perform loop closure, and 3) *Depth Sensing*: a time-of-flight depth sensing camera on the device determines the distance of every pixel from the camera (barring a few exceptions such as specular or non-reflective surfaces). The application is implemented in Unity3D for Android.

Floor, Ceiling, and Wall Capture

To detect when the user aims the device at the floor and ceiling, the system observes the pitch of the device and triggers a RANSAC [12] based plane fitting algorithm on the depth map, if the pitch falls outside a certain range (-45° for floor, to 45° for ceiling). Calibrating floor plane distance is crucial,

since it serves as a supporting plane for furniture items that will be scanned at a later stage. Ceiling plane distance is currently only used to determine the upper extent of the walls. Walls describe the main shape and structure of the room and modeling them is the first major step in our modeling process. We follow a strategy, similar to our prior work [36, 35], of soliciting user input to mark wall edges. However, unlike [36, 35], users are free to move around the room. Additionally, with the availability of depth data, we are able to uniquely determine the depth of the wall edges in 3D world space. Once the user aligns the 'wall cursor' (See Figure 2a) with a wall edge and a tap is registered, the system records the depth value, in world space, of the center pixel of the screen which corresponds to the location of the wall edge. As the user moves towards the next wall edge, a visualizing mesh dynamically extends to follow the current position of the cursor. When the user returns to the first wall edge and the 3D cursor is within 0.25m of the initial marking, the system automatically snaps to the first edge and closes the room shape.

Furniture Surface Capture

The interface instructs the user to aim the device at the surface they wish to capture. Surfaces are assumed to be near-rectangular and parallel to the ground plane. When the user taps to mark a surface, a seed depth point is chosen at the center pixel of the screen, where the 3D cursor is located. All valid depth points in the frame that lie within a height threshold of 3.5cm and have a normal cosine similarity < 0.6 from the seed point are selected.

They are then quantized into an evenly spaced grid of 1.33cm x 1.33cm with collisions being eliminated by the quantization process. This ensures a more uniform distribution of depth points over the entire surface for the next step of rectangle extraction. By tapping multiple times, the user accumulates depth points into a dictionary data structure (indexed by quantized bin) and a dynamic threshold selects the most frequently occurring quantized depth values based on a cutoff frequency of $0.8 \times$ average frequency.

The y-coordinates (height) of the accumulated points are temporarily dropped, in order to analyze them on the x-z plane. We employ Shamos' rotating calipers algorithm [38] in order to select the minimum area rotated rectangle that encloses all of the 2D points. The returned rectangle is then projected back into 3D by reintroducing the average y-coordinate, of the plane and then rendered in the augmented reality interface. The captured planar surface is stored as a set of four corner vertices, along with the length, breadth, and height. We refer to this as the *user-marked surface*.

A faster version of this algorithm (without quantization and accumulation) is used to render a preview of the currently selected rectangle, to give the user an idea of the area they are about to select. We found that having a preview was important, since the user is not aware of the underlying algorithm and may accidentally select spurious points on the wall or on neighboring objects. Having the preview increased accuracy, but also increased the time taken to capture, since the users sometimes tried to get the 'perfect' preview before confirming their selection.

Image Capture and 3D Model Retrieval

Once the surface area been marked, the user moves to a position where the entire object is visible and captures a photograph. The image is captured from the color camera of the device and encoded as a PNG. Based on the user's selection of type of object, the system uploads the image to one of eight HTTP listen servers that are running Convolutional Neural Networks (CNNs) described in [22]. The Networks are identical in architecture but are individually trained on separate classes of objects from the ShapeNet database [8], consisting of 6778 chairs, 8436 tables, 3173 sofas, 233 beds, 452 bookshelves, and 1571 cabinets. We utilize these networks largely unmodified, except for minor performance optimizations. The output of the CNN is a list of ShapeNet model IDs and distance scores that are then returned to the device in the form of an XML document that also contains meta data, such as the name of the model, and a URL to the source files associated with the model (3D model file and a PNG thumbnail).

Upon receiving the response, the mobile application proceeds to load all thumbnails and displays them in a menu that pops up from bottom of the screen. The menu is designed to avoid unnecessarily occupying screen space when not in use, and the pop up animation directs the users' attention to the availability of more model options. The system then asynchronously loads the 3D model Unity asset bundle corresponding to the first result. Once the model is downloaded, it is sampled and placed in the scene using the model placement algorithm described in the next section.

Model Placement Algorithm

The goal of the furniture placement algorithm is to be fast, accurate, and generalizable to a wide variety of furniture types. The inputs to the placement algorithm are the 3D model to be placed, the current global pose of the camera, a depth map of the scene as sensed by the device, and the *user-marked surface* (stored earlier). The high level idea behind the furniture placement algorithm is to compute the dominant horizontal surface of the CAD model and align it with the *user-marked surface*.

Our approach to automatically extract the dominant horizontal planar surface from the CAD model is to first sample the model as a point cloud (Figure 4a). We consider the model in a frontal pose and fire rays from a virtual camera. For efficiency, rays are only fired into the bounding box of the object. For each ray that collides with the object, we record the position and normal of the collision point in the local space of the object. We also record the lowest and highest collision points on the model. Rays that do not encounter the object are recorded as free space rays.

The y-coordinate (height) of each sampled point is added to a histogram with a bin size of 3.33cm. We pick the most frequently occurring quantized y-coordinate and isolate all 3D points that fall within that range, which correspond to points on the near-planar surface of interest. We apply the same rotating calipers algorithm as before to determine the four corner vertices, along with the length, breadth, and height of the planar surface. We refer to this as the *model-extracted surface*.

(a) Model Sampling (b) Good Alignment (c) Poor Alignment

Figure 4: Visualization of CAD model sampling (4a), along with a good alignment to the depth map (4b) and a poor alignment to the depth map (4c). Inliers are in light green and outliers in dark red.

To align the *user-marked* and *model-extracted* surfaces we determine a scale factor in the x and z axes by dividing the corresponding breadth and length values. Scale in the y-axis is determined by equating the height difference between the lowest sampled model point and the model-extracted surfaces, and height difference the real scene's ground plane to the user-marked planar surface. The translation of the 3D model in the x-z plane is determined by aligning the centroids of the user-marked and model-extracted surfaces. The translation in the y-axis is determined by aligning the lowest sampled point in the model to the height of the ground plane in the real scene.

Note that aligning the quadrangular planar surfaces does not necessarily determine the optimal rotation of the model, particularly in cases where the object is rotationally symmetric. It does, however, limit the rotation to at most one of four directions (along the edges of the quad). If the planar surface of the furniture is nearly square (length and breadth within 50cm of each other), four possible orientations can exist, since the order of rotational symmetry of a square is 4. If the planar surface is rectangular, we only need to check two orientations that are 180 degrees apart.

To determine the optimal rotation of the model, we use a scoring function that determines the fit of the model when compared to the observed depth map of the scene, as visualized in Figures 4b & 4c. The scoring function takes each point in the sampled 3D model and finds the distance to the nearest depth map point based on projective point matching [13]. If the distance is below a certain threshold (empirically determined to be 20cm), the model point is considered to be an 'inlier'. Free space rays that were recorded during model sampling are fired into the depth map. If they encounter a depth point within the threshold distance of the model centroid, they are considered as 'non-inlier', i.e. a point that was expected to be free space. The final score is computed as the percentage of inlier points to the total sample points with a penalty subtracted for the percentage of non-inlier points. We found that a penalty factor of 2.0 improved the relevance of the scoring function by adequately accounting for expected free space in the depth map v/s the model.

For circular surfaces (e.g. a circular table or stool), we approximate the surface to the nearest quadrilateral. In these cases, the rotation of the surface is irrelevant, since all rotations are equivalent.

If the user selects a new 3D model from the menu or wishes to capture the next furniture item, the above steps are repeated to place the new model in the scene.

Door, Window, Artwork Capture

The user manually marks the extents of the object surface by annotating the top right and bottom left corners of the object. Based on the assumption that the object is rectangular and not tilted with respect to the ground, we can uniquely determine the 3D extents of the surface rectangle based on these two depth points.

Once the surface is selected, the user chooses the object type (door, window, artwork) and selects an appropriate model. Currently our implementation is limited to a fixed number of models (10 each) for doors and windows. We believe it will be easy to extend the deep learning framework used for furniture, to recognize doors and windows, but we leave this for future work. Finally, a similar method of scaling and translating the user-marked and model-extracted surfaces of the door/window is used to place the model in the scene.

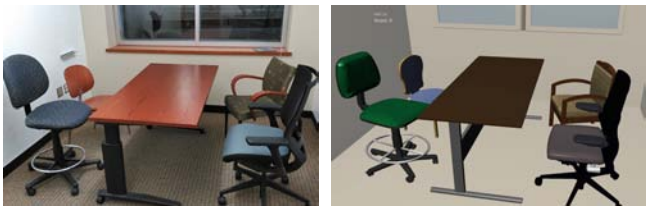
For artwork, we use a 3D planar proxy as the 'model' and transfer the texture of the artwork from the image to the model using a GrabCut algorithm [33].

USER STUDY

To evaluate our system we compare it to Homestyler [16]: an existing, commercially available, web-based CAD modeling tool, specifically intended for modeling indoor scenes. Homestyler features a curated catalog of models to choose from, so users do not have to search and download models from the Internet. The resulting model it produces is very similar in nature to that of our system. We posit that the experiments would be similar for any of the equivalent desktop/web-based indoor modeling tools referred to in the related work section.

The goal of our study was to analyze users' performance, in terms of task time and placement error, while creating 3D CAD models of an indoor scene with either tool, and also to gather qualitative feedback. Our starting hypothesis was that in-situ CAD modeling with Tango would be faster and more accurate than traditional desktop tools.

Participants: Our study was conducted with 10 participants (2 Female) from ages 24-33 (M=27.8 years, SD 4.85). They were all graduate students at the University of Washington and were familiar with desktop computers and touch-based mobile devices. None of the users claimed to be experts or professional 3D modelers. None of the users had used our application or Homestyler prior to the study, and 7 participants claimed to have had some experience with augmented reality before. The users were all volunteers and each study session lasted for approximately 1 hour.



(a) Study Room (b) Generated Model

Figure 5: Study Room Setup. The room contained 4 distinct chairs, 1 desk, 1 window and 1 door.

Study Design: We conducted a 10x2 within-subjects study, where each user was tasked with creating a CAD model of a an office, once each with our tool and Homestyler. The room had dimensions 2.9m x 4.7m, and contained four distinct chairs, a desk, a large window and a door (Figure 5). In order to eliminate any carryover effects caused by practice or fatigue, we randomly assigned half the users our method first.

Study Procedure: The study began with a short introduction to the task followed by demonstrations of each system, by the researcher. The users were then given time to familiarize themselves with the systems and repeat the demonstration tasks. Next, the users were instructed to start over and capture a full 3D model of the room using each system. They were encouraged to think out loud, and were aware that their task times and 3D model results would be recorded for later analysis.

For each method, we recorded time taken to complete subtasks of floor/ceiling/wall capture, furniture capture and door/window capture, and the resulting 3D model was saved. In the case of Homestyler, users were asked to measure the dimensions of the walls, doorway and window, with a tape measure (ignoring height in all cases), so that they may enter it manually into the modeling tool. We did not ask users to measure furniture position, as this was determined to be a cumbersome task that would take a very long time to complete. Manual measurement time was recorded separately. At the end of the tasks, users were asked a series of qualitative questions.

RESULTS

Task Time

Figure 6 shows total time taken by each user to model the room using our system v/s Homestyler. The total time for Homestyler includes the time taken to manually measure the scene, as it is an integral step to create an accurate CAD model on the desktop. In contrast our system does not require the user to make any physical measurements, instead relying on scene measurements from the device.

Results indicate that for all participants, our system (M=336.9s, SD = 100.9s) took significantly ($p < 0.001$) less time to complete the task than Homestyler (M=702s, SD=245.34s) for creating an equivalent model. On average our system was 49.8% faster than Homestyler. For detailed quantitative analysis, please refer to Auxiliary Material.

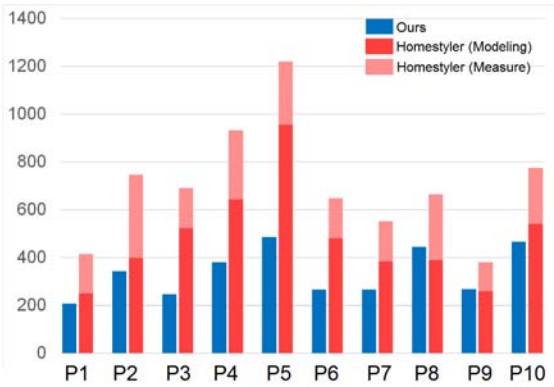


Figure 6: Total time (in seconds) taken by users for the modeling task with either tool. Time for Homestyler is further broken down into measurement and modeling time.

Furniture Placement Error

In order to measure error in furniture placement, we captured a ground truth 3D scan of the room using point-cloud fusion. The point cloud was then rendered from a orthographic top-down view, creating a 2D floor plan of the scene. We rendered similar views of the models captured with our system, and Homestyler and overlaid them on the ground truth scan, as shown in Figure 7.

We then compare manually annotated contours around corresponding furniture items. For each furniture item, we compute Chamfer distance (1), which is defined as the sum of closest point distances between the contours in the model and ground truth. This was symmaterized by (2). We adopt the symmetric Chamfer distance as our primary error metric, since it accounts for difference in contour shapes as well as the displacement of their centroids.

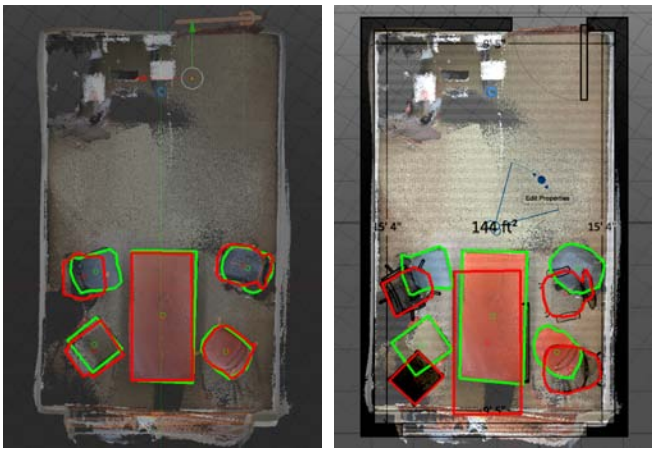
$$Ch(A,B) = \Sigma_{a \in A} min_{b \in B} \|a - b\| \quad (1)$$

$$SymmetricCh(A,B) = Ch(A,B) + Ch(B,A) \quad (2)$$

The results indicate that our system has significantly less placement error ($t(9) = 6.34$, $p < 0.001$), in all cases. We acknowledge that manually measuring the position of each furniture item, and entering it into Homestyler would reduce error, but would also incur a severe time penalty. We therefore chose not to evaluate this scenario, since our system was already much faster than the manual method.

Qualitative Feedback

After completing both tasks, we asked users to reflect on their experience and rate our system when compared to Homestyler, in Mental Demand, Physical Demand, Temporal Demand, Effort, Frustration, Errors in Appearance, and Errors in Position, based on NASA TLX [42]. For each factor, we asked the users if our system was 'Much Lower', 'Slightly Lower', 'Same as Homestyler', 'Slightly Higher', or 'Much Higher'. Responses are shown in Figure 8 and indicate that our system is generally less demanding to use than Homestyler. Users also *qualitatively perceived* our system as being



(a) Our System Overlay (b) Homestyler Overlay

Figure 7: Results from P5 using our system 7a and Homestyler 7b overlaid in a 2D top-down view on a ground truth scan of the scene. Contours in green indicate ground truth and red indicate the user-generated model.

quicker and less error (position) prone than Homestyler, an insight that is in agreement with our quantitative analysis.

We asked participants several open-ended questions. The awnsers, along with summaries are below (raw quotes are included in Auxiliary Material):

"Overall, did you prefer the augmented-reality or desktop modeling system? Please elaborate." 7 out of 10 users strongly preferred our system, stating it was *"much smoother", "easier overall"*, and that they were *"confident in the result"*. 2 users preferred the desktop system since they believed it provided more control. 1 user suggested using a hybrid of the two systems, capturing data with AR and editing on the desktop.

"What did you like or found easy about the AR system?" Users found the AR system intuitive, less time consuming and less frustrating, since it was easier to find and place matching furniture.

"What did you not like, found difficult or frustrating about the AR system?" Some users complained that capturing large surfaces was difficult, and that they experienced arm fatigue after holding the Tango for several minutes. Other complaints included *"small misalignments"* and *"inability to revert changes"*.

"Did the capability to model in augmented reality assist you in the modeling tasks? Please elaborate." All users agreed that AR helped the modeling process, stating for example *"Yes. Didn't have to glance back at references for picking furniture"*.

"Did the model suggestion interface assist you in the modeling tasks?" All users agreed that the model suggestion interface was helpful, as opposed to the manual hierarchical selection required in the desktop tool.

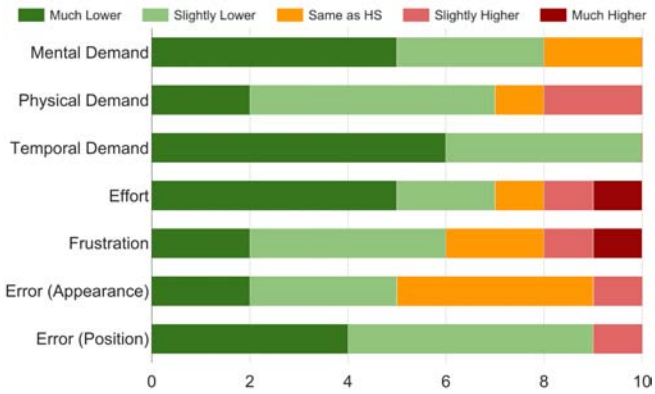


Figure 8: Qualitative feedback. (Best viewed in color)

"If you had complete creative control over this system, how would you improved it?" Several users requested a manual adjustment mode. Others suggested more automation for plane detection and object class selection. Some users wanted an easier way to capture large surfaces.

Other Scenes

We present a variety of other scenes captured with our system to demonstrate that it generalizes to environments beyond those tested in the study. We did not formally evaluate capture performance for these scenes, but instead present visual results in Figures 1 & 9 depicting the real scenes, corresponding captured CAD model view, and overhead views from 4 Living Rooms, 2 Dining Rooms and a Bedroom scene.

DISCUSSION AND LIMITATIONS

The results of the user study indicate that the task of indoor 3D modeling is faster and more accurate with our in-situ, 3D aware mobile device approach, than with a traditional desktop approach. We believe Homestyler is an appropriate tool for comparison since it is a full-featured modeling tool that is popular among casual interior designers today. Overall, users preferred our system and enjoyed completing the modeling task using it. Although the study was conducted in a controlled setting with a fixed layout and furniture arrangement, we also present results on a range of different scenes (Figure 9).

While our simplifying assumptions enable improved performance, they also introduce certain limitations to our system, which we analyze herein. First, the assumption that furniture has a near-planar, horizontal surface works well for many of the dominant furniture types such as tables, chairs, beds, couches, etc. However, objects that lack functional horizontal surfaces such as plants, lamps, refrigerators, and telephones cannot be modeled with this method. Furthermore large functional surfaces (dining tables, beds) cannot be captured in one step and may require the user to move and capture additional parts of the surface. Furniture with a fully occluded surface (such as chair that is tucked under a desk) cannot be modeled with our approach. In this case, we would instruct users to physically move the furniture into a position where the surface is un-occluded. Despite these limitations, we were able

Real Scene Corresponding CAD View Overhead View

Figure 9: Visual results from 4 Living Rooms, 1 Dining Room and 1 Bedroom scene

to capture a wide variety of living room, dining room, bedroom and office scenes. Modeling more general objects is an interesting avenue for future work.

The assumption that doors, windows and artwork are planar, aligned with the wall plane and not tilted with respect to the ground appeared to hold true in almost all of our test scenes. A notable exception is a doors that is ajar and not aligned with the wall, in which case the user would either need to close the door or mark the frame, rather than the body of the door. While we did not formally evaluate the influence of capturing artwork in the scene, we anecdotally observed that adding artwork enhances the sense of realism of the CAD model and makes scenes more identifiable to viewers. In future work, capturing scene lighting [7, 47] may further improve realism of the models.

Due to limitations with current depth-sensing technology, our system is unable to capture the surface of specular or non-IR reflective objects (such as glass or black leather). To overcome this, we used a workaround to capture the glass table in Figure 9, Row 1. A proxy material (a magazine, that is visible to the depth sensor) was placed on the surface during capture. In future work, we could also enable the user to manually mark the surface area in question. Similarly, depth-sensing fails in direct sunlight (e.g. outdoors, or a room awash with sunlight) since the sunlight washes out the IR signal from the depth sensor.

While the ShapeNet database is relatively large, it is not comprehensive, and therefore not all objects can be accurately represented with our CAD-based approach. Also, automatic model retrieval is not perfect, as can be noticed in some of the room models, but allowing the user to select the best match in-situ helps improve results significantly. As model databases and machine learning techniques improve, they will continue to improve our system.

Finally, some users reported arm fatigue when using the hand held mobile device. We acknowledge this limitation and expect it to diminish if our system were to be implemented on a head-mounted device.

CONCLUSION

We have presented a novel system that enables casual users to interactively capture semantic 3D CAD room models on 3D aware mobile devices. Using our system, users can quickly and easily capture walls, furniture and structural elements within the room. We learned from our user studies that such a in-situ capture system offers a significant improvement over traditional desktop CAD modeling software, in terms of speed and accuracy.

Enabling casual users to capture models of their indoor surroundings has several interior design applications including virtual furniture rearrangement or replacement, remodeling, relighting. In the future, such semantic models may also be used to enhance virtual and augmented reality experiences. We believe that our system is a first step in an interesting direction of combining the power of 3D aware mobile devices and state-of-the-art object recognition and we hope that our work motivates further research into this area.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation (IIS-1250793), the University of Washington Animation Research Labs, and Google.

REFERENCES

1. Meta 2. 2017. MetaVision. https://www.metavision.com/. (2017). Accessed: 2017-04-04.
2. Sweet Home 3D. 2016. eTeks. http://www.sweethome3d.com/. (2016). Accessed: 2017-04-04.
3. Planner 5d. 2017. UAB. https://planner5d.com/. (2017). Accessed: 2017-04-04.
4. Harshit Agrawal, Udayan Umapathi, Robert Kovacs, Johannes Frohnhofen, Hsiang-Ting Chen, Stefanie Mueller, and Patrick Baudisch. 2015. Protopiper: Physically Sketching Room-Sized Objects at Actual Scale. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New York, NY, USA, 427–436. DOI: http://dx.doi.org/10.1145/2807442.2807505
5. Zen Phone AR. 2017. ASUS. https://www.asus.com/Phone/ZenFone-AR-ZS571KL/. (2017). Accessed: 2017-04-04.
6. Mathieu Aubry, Daniel Maturana, Alexei A. Efros, Bryan C. Russell, and Josef Sivic. 2014. Seeing 3D Chairs: Exemplar Part-Based 2D-3D Alignment Using a Large Dataset of CAD Models. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*. IEEE, 3762–3769. DOI: http://dx.doi.org/10.1109/CVPR.2014.487
7. Sean Bell, Kavita Bala, and Noah Snavely. 2014. Intrinsic Images in the Wild. *ACM Trans. on Graphics (SIGGRAPH)* 33, 4 (2014).
8. Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015. *ShapeNet: An Information-Rich 3D Model Repository*. Technical Report http://arxiv.org/abs/1512.03012 [cs.GR].
9. Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. 2015. Robust reconstruction of indoor scenes. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 00 (2015), 5556–5565. DOI: http://dx.doi.org/doi.ieeecomputersociety.org/10.1109/CVPR.2015.7299195
10. AR Home Designer. 2017. Elementals. http://www.elementalsweb.com/home_augmented_reality_designer/. (2017). Accessed: 2017-04-04.
11. Hao Du, Peter Henry, Xiaofeng Ren, Marvin Cheng, Dan B. Goldman, Steven M. Seitz, and Dieter Fox. 2011. Interactive 3D Modeling of Indoor Environments

with a Consumer Depth Camera. In *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp '11)*. ACM, New York, NY, USA, 75–84. DOI:http://dx.doi.org/10.1145/2030112.2030123

12. Martin A. Fischler and Robert C. Bolles. 1981. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* 24, 6 (June 1981), 381–395. DOI:http://dx.doi.org/10.1145/358669.358692
13. Robert B. Fisher. 2001. Projective ICP and Stabilizing Architectural Augmented Reality Overlays. In *Proc. Int. Symp. on Virtual and Augmented Architecture (VAA01*. 69–80.
14. Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. 2009. Reconstructing building interiors from images. In *In Proc. of the International Conference on Computer Vision (ICCV '09)*. DOI: http://dx.doi.org/10.1109/ICCV.2009.5459145
15. HoloLens. 2015. Microsoft Corp. https://www.microsoft.com/microsoft-hololens/en-us. (2015). Accessed: 2016-05-27.
16. Homestyler. 2013. Easyhome (Formerly Autodesk). http://www.homestyler.com. (2013). Accessed: 2017-07-13.
17. Qixing Huang, Hai Wang, and Vladlen Koltun. 2015. Single-view Reconstruction via Joint Analysis of Image and Shape Collections. *ACM Trans. Graph.* 34, 4, Article 87 (July 2015), 10 pages. DOI: http://dx.doi.org/10.1145/2766890
18. Tomoya Ishikawa, Kalaivani Thangamani, Masakatsu Kourogi, AndrewP. Gee, Walterio Mayol-Cuevas, Keechul Jung, and Takeshi Kurata. 2009. In-Situ 3D Indoor Modeler with a Camera and Self-contained Sensors. In *Virtual and Mixed Reality*, Randall Shumaker (Ed.). Lecture Notes in Computer Science, Vol. 5622. Springer Berlin Heidelberg, 454–464. DOI: http://dx.doi.org/10.1007/978-3-642-02771-0_51
19. Vision Innovation Labs. 2017. Lowes. http://www.lowesinnovationlabs.com/tango/. (2017). Accessed: 2017-04-04.
20. Tobias Langlotz, Stefan Mooslechner, Stefanie Zollmann, Claus Degendorfer, Gerhard Reitmayr, and Dieter Schmalstieg. 2012. Sketching Up the World: In Situ Authoring for Mobile Augmented Reality. *Personal Ubiquitous Comput.* 16, 6 (Aug. 2012), 623–630. DOI: http://dx.doi.org/10.1007/s00779-011-0430-0
21. Manfred Lau, Masaki Hirose, Akira Ohgawara, Jun Mitani, and Takeo Igarashi. 2012. Situated Modeling: A Shape-stamping Interface with Tangible Primitives. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction (TEI '12)*. ACM, New York, NY, USA, 275–282. DOI: http://dx.doi.org/10.1145/2148131.2148190
22. Yangyan Li, Hao Su, Charles Ruizhongtai Qi, Noa Fish, Daniel Cohen-Or, and Leonidas J. Guibas. 2015. Joint Embeddings of Shapes and Images via CNN Image Purification. *ACM Trans. Graph.* 34, 6, Article 234 (Oct. 2015), 12 pages. DOI: http://dx.doi.org/10.1145/2816795.2818071
23. Joseph J. Lim, Hamed Pirsiavash, and Antonio Torralba. 2013. Parsing IKEA Objects: Fine Pose Estimation. In *Proceedings of the 2013 IEEE International Conference on Computer Vision (ICCV '13)*. IEEE, 2992–2999. DOI:http://dx.doi.org/10.1109/ICCV.2013.372
24. MagicPlan. 2011. Sensopia Inc. https://www.metavision.com/. (2011). Accessed: 2017-04-04.
25. Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. 2011. KinectFusion: Real-time Dense Surface Mapping and Tracking. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR '11)*. IEEE, 127–136. DOI:http://dx.doi.org/10.1109/ISMAR.2011.6092378
26. Benjamin Nuernberger, Eyal Ofek, Hrvoje Benko, and Andrew D. Wilson. 2016. SnapToReality: Aligning Augmented Reality to the Real World. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1233–1244. DOI: http://dx.doi.org/10.1145/2858036.2858250
27. Patrick Paczkowski, Min H. Kim, Yann Morvan, Julie Dorsey, Holly Rushmeier, and Carol O'Sullivan. 2011. Insitu: Sketching Architectural Designs in Context. *ACM Trans. Graph.* 30, 6, Article 182 (Dec. 2011), 10 pages. DOI:http://dx.doi.org/10.1145/2070781.2024216
28. Floor Planner. 2016. http://www.floorplanner.com. (2016). Accessed: 2017-04-04.
29. Planoplan. 2017. http://planoplan.com/en/. (2017). Accessed: 2017-04-04.
30. Phab 2 Pro. 2017. Lenovo. http://shop.lenovo.com/us/en/tango/. (2017). Accessed: 2017-04-04.
31. Project Tango. 2014. Google Inc., ATAP. https://www.google.com/atap/projecttango/. (2014). Accessed: 2016-05-27.
32. Home Planner. 2013. IKEA. http://www.ikea.com/ms/en_JP/rooms_ideas/splashplanners.html. (2013). Accessed: 2016-05-27.
33. Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. 2004. "GrabCut": Interactive Foreground Extraction Using Iterated Graph Cuts. *ACM Trans. Graph.* 23, 3 (Aug. 2004), 309–314. DOI: http://dx.doi.org/10.1145/1015706.1015720

34. Renato F. Salas-Moreno, Richard A. Newcombe, Hauke Strasdat, Paul H.J. Kelly, and Andrew J. Davison. 2013. SLAM++: Simultaneous Localisation and Mapping at the Level of Objects. *2013 IEEE Conference on Computer Vision and Pattern Recognition* (2013), 1352–1359. DOI: http://dx.doi.org/10.1109/CVPR.2013.178

35. Aditya Sankar and Steven M. Seitz. 2012. Capturing Indoor Scenes with Smartphones. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, 403–412. DOI:http://dx.doi.org/10.1145/2380116.2380168

36. Aditya Sankar and Steven M. Seitz. 2016. In Situ CAD Capture. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16)*. ACM, New York, NY, USA, 233–243. DOI: http://dx.doi.org/10.1145/2935334.2935337

37. T. Schöps, T. Sattler, C. Häne, and M. Pollefeys. 2015. 3D Modeling on the Go: Interactive 3D Reconstruction of Large-Scale Scenes on Mobile Devices. In *2015 International Conference on 3D Vision*. 291–299. DOI: http://dx.doi.org/10.1109/3DV.2015.40

38. Michael Ian Shamos. 1978. Computational Geometry. (1978).

39. Tianjia Shao, Weiwei Xu, Kun Zhou, Jingdong Wang, Dongping Li, and Baining Guo. 2012. An Interactive Approach to Semantic Modeling of Indoor Scenes with an RGBD Camera. *ACM Trans. Graph.* 31, 6, Article 136 (Nov. 2012), 11 pages. DOI: http://dx.doi.org/10.1145/2366145.2366155

40. Sketchup. 2016. Trimble Navigation Limited. http://www.sketchup.com/. (2016). Accessed: 2016-05-27.

41. Noah Snavely, Steven M. Seitz, and Richard Szeliski. 2006. Photo Tourism: Exploring Photo Collections in 3D. *ACM Trans. Graph.* 25, 3 (July 2006), 835–846. DOI:http://dx.doi.org/10.1145/1141911.1141964

42. TLX. 2009. NASA. http://www.nasatlx.com/. (2009). Accessed: 2016-05-27.

43. Julien Valentin, Vibhav Vineet, Ming-Ming Cheng, David Kim, Jamie Shotton, Pushmeet Kohli, Matthias Nießner, Antonio Criminisi, Shahram Izadi, and Philip Torr. 2015. SemanticPaint: Interactive 3D Labeling and Learning at Your Fingertips. *ACM Trans. Graph.* 34, 5, Article 154 (Nov. 2015), 17 pages. DOI: http://dx.doi.org/10.1145/2751556

44. Michael Waechter, Mate Beljan, Simon Fuhrmann, Nils Moehrle, Johannes Kopf, and Michael Goesele. 2017. Virtual Rephotography: Novel View Prediction Error for 3D Reconstruction. *ACM Trans. Graph.* 36, 1, Article 8 (Jan. 2017), 11 pages. DOI: http://dx.doi.org/10.1145/2999533

45. Sketchup 3D Warehouse. 2016. Trimble Navigation Limited. https://3dwarehouse.sketchup.com/. (2016). Accessed: 2017-04-04.

46. Brandon Yee, Yuan Ning, and Hod Lipson. 2009. Augmented Reality In-Situ 3D Sketching of Physical Objects. (2009).

47. Edward Zhang, Michael F. Cohen, and Brian Curless. 2016. Emptying, Refurnishing, and Relighting Indoor Spaces. *ACM Trans. Graph.* 35, 6, Article 174 (Nov. 2016), 14 pages. DOI: http://dx.doi.org/10.1145/2980179.2982432