In Situ CAD Capture

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) Captured Virtual Scene (c) Overhead View

Figure 1: Our system captures a CAD-like 3D model of an indoor scene, in a few minutes, on a commodity tablet.

ABSTRACT

We present an interactive system to capture CAD-like 3D models of indoor scenes, on a mobile device. To overcome sensory and computational limitations of the mobile platform, we employ an in situ, semi-automated approach and harness the user's high-level knowledge of the scene to assist the reconstruction and modeling algorithms. The modeling proceeds in two stages: (1) The user captures the 3D shape and dimensions of the room. (2) The user then uses voice commands and an augmented reality sketching interface to insert objects of interest, such as furniture, artwork, doors and windows. Our system recognizes the sketches and add a corresponding 3D model into the scene at the appropriate location. The key contributions of this work are the design of a multi-modal user interface to effectively capture the user's semantic understanding of the scene and the underlying algorithms that process the input to produce useful reconstructions.

Author Keywords

Interactive 3D Modeling; Sketching; Mobile Augmented Reality; Algorithms; Design; Human Factors

ACM Classification Keywords

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

MobileHCI '16, September 06–09, 2016, Florence, Italy.
 ISBN 978-1-4503-4408-1/16/09 $15.00
DOI: http://dx.doi.org/10.1145/2935334.2935337

INTRODUCTION

3D models of architectural scenes enable several compelling applications. Imagine if you wanted to rearrange the existing furniture in your room, or visualize how a new piece of furniture would fit into your current space. An editable 3D model of your room would allow you to preview the results easily, without having to physically move any furniture around. Such 3D models are also powerful tools to visualize homes and offices, for the purposes of real estate, remodeling and insurance. Also, with the recent surge in consumer-level virtual and augmented reality, the digitization of personal spaces can enable better and more meaningful interactions between the real and virtual worlds.

However, creating these models is still a challenging task. A common approach is to manually create the model using CAD software, typically while ex-situ, i.e. in a different context/setting than the original scene. The modeler must therefore rely on memory, photographs, laser scans, or manual measurements in order to create an accurate representation of the scene. Existing desktop CAD tools also pose a steep learning curve for inexperienced users, often requiring considerable effort to create aesthetically pleasing models. Another approach is to use automatic, computer vision based techniques, but they typically do not capture the semantic nature of a scene, and produce point-clouds or large polygonal meshes with missing regions and no notion of which geometric regions correspond to objects or their physical extents. Hence, these models are very difficult to edit or manipulate.

In this paper, we present a novel in situ interactive modeling technique that combines the positive aspects of the automatic and manual approaches. The system runs end-to-end on a commodity tablet or smartphone without any external computational or hardware resources. Our approach utilizes a

multi-modal user interface based on the following elements: an in situ mixed-reality mobile platform, sketch based furniture placement, and voice commands. Another key insight is to leverage humans for high-level recognition (identifying the class of an object, e.g., table, chair), a task that we find effortless, and use model-driven optimization algorithms for precise selection/placement operations. With the user and system acting in synchrony, we are able to improve the efficiency of indoor 3D modeling by harnessing the best of both worlds.

The modeling proceeds in two stages. First, the user captures a 360 degree panning video of a room, annotating corners as they go along. Based on the annotated panorama, the system reconstructs a Manhattan-world (i.e., axis-aligned) 3D model of the room walls and estimates scale with a constrained structure from motion technique.

Once the empty room has been reconstructed, the user interactively populates the scene with objects while they are still in situ. In order to do this, the system provides an interactive modeling interface using a live feed from the camera. The user can model objects by first issuing a voice command describing the object (e.g., "chair") and then sketching a few lines on the image of the object in front of them. Our system then analyzes the voice and sketch inputs, along with the position and orientation of the handheld device in order to determine the type, 3D position, orientation and scale of the object of interest. A representative 3D model is selected from a database and is placed in the room 3D model. This workflow is repeated, until all objects of interest have been modeled, resulting in a schematic 3D model of the room.

Our mobile, in situ system attempts to overcome some of the limitations of traditional ex-situ desktop software. While in situ, the user needn't rely on memory or photographs, but can rather use an image-guided sketching technique to model objects in the scene. Also, by tracking the motion of the mobile device, the user is able to place the objects at the appropriate scale and location in the scene, without needing any manual measurements. We note, that our system is not intended to fully replace more-versatile desktop CAD software, but rather serve as a tool for users to quickly and easily capture and manipulate 3D models of indoor scenes using widely available mobile hardware.

In the remainder of this paper, we discuss related work, describe our end-to-end system, present results and conclude with the description of two potential applications of our system.

RELATED WORK

CAD: A variety of efforts have attempted to simplify the process of computer aided design of scenes and objects. Trimble SketchUp [29] (formerly Google SketchUp) is one such commercial product that simplifies the 3D modeling process by providing a limited set of controls. AutoDesk Homestyler [9] and IKEA Home Planner [23] are tools designed specifically to model indoor scenes. These systems are designed for desktop use with a mouse and keyboard, while ex-situ.

Sketch-based modeling: Sketching techniques allow more casual forms of user input to compute or retrieve 3D models. Several systems [38, 12, 36] have demonstrated the value of quick, stylus-driven interactions to create 3D models.

Several others [28, 37, 4, 6] have extended this paradigm to model various architectural objects by using sketch-based model retrieval techniques. However, these techniques are not designed to work in situ and require users to draw multiple free hand perspective sketches of objects from memory. Lau et al. [17] use an image-aided sketching interface to create 3D wireframes based on 2D image guides, but focus on modeling simple objects with block-like primitives.

In situ modeling: The in situ aspect of our system is similar to [22, 15, 35]. These approaches allow users to create models from scratch by combining various primitives, whereas ours utilizes preexisting 3D furniture models from the internet and instead focuses on their correct placement in relation to the current scene. In situ indoor modeling has been explored by Kim et al. [14], however their system is limited to box shapes and fully manual placement of furniture, whereas ours can handle more general room shapes and uses a sketch based augmented reality interface. Sukan et al. [31] have used marker-based AR to visualize virtual furniture in a scene, but don't address capture. SemanticPaint [34] uses additional depth hardware and in situ interaction to segment and label fused point clouds.

Multi-modal Modeling UI: Novotny et al. [21] show that using a multi-modal augmented-reality interface for 3D modeling is efficient, though they restrict the workspace to a tabletop and focus on free form 3D modeling with primitives. Tsang et al. [32] also use voice, gesture and a mechanical boom in order to annotate 3D objects in the context of real space but do not address model creation. The voice component of our system relies on Pocketsphinx [11], a free and open source speech recognition system for mobile devices.

Furniture Model Alignment: Recent work [18, 1, 10] has focused on automatically aligning 3D furniture models to single images by using an exemplar database. While promising, these are complex and computationally expensive approaches that may not be well suited for mobile, in situ applications.

Automatic 3D Reconstruction: The visual SLAM and computer vision communities have developed several automatic approaches [30, 7] to reconstruct scenes from images. While these methods have demonstrated potential, they tend to be brittle and do not work well on texture-poor surfaces, e.g. painted walls, which dominate interiors. In our previous work [26] we solicit a few user-marked features such as wall edges, resulting in a well-bounded optimization problem that converges in a few seconds on the mobile platform. However, the work was limited to floorplan generation and does not model furniture, which is the main contribution of this work.

Other approaches [20, 27, 25], overcome the issue of feature matching by utilizing depth information provided by a Kinect camera. While this approach has demonstrated exciting results, our method does not rely on external hardware and can work on widely available commodity hardware. However, in the future we would like to extend our system to emerging depth sensing platforms such as Google's Project Tango [24] and Microsoft HoloLens [8].

USER'S VIEW OF THE SYSTEM

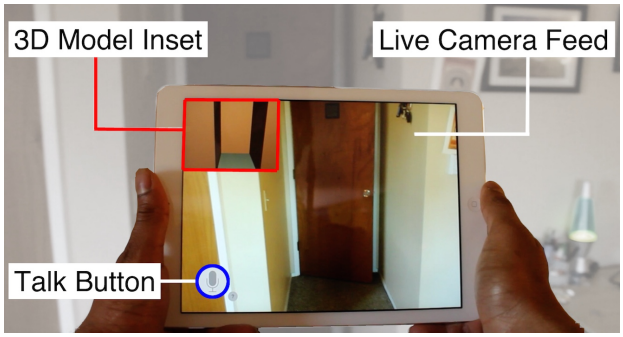


Figure 2: Our mixed-reality modeling interface.

This section describes how a user interacts with our application. Our system is implemented on an iPad Air Retina running iOS 8.3. However, the technique is designed to work on any commodity tablet equipped with a camera and inertial sensors. At this point, we strongly encourage readers to view the accompanying supplementary video in order to get a better understanding of the interactive aspects of the system.

The user begins by capturing a continuous panning video from the center of the room, rotating 360° to capture the entire room. As they pass a wall edge they use a simple tap gesture to indicate the position of a corner. Interface cues notify the user if they're moving too fast or deviating from the ideal camera path. Once the full rotation is complete, the system computes the empty room geometry in a few (< 5) seconds and presents the user with a mixed reality furniture placement interface (Figure 2). The interface consists of a live view of scene along with an inset of the reconstructed room model representing the 3D geometry of the current view.

While standing at the same position as the initial capture, the user simply looks around the room through the device and sketches rough outlines of objects they would like to capture. To aid the search algorithm, they may state out loud the type of furniture they are capturing, say "Table", and proceed to trace the outline of the table that is frozen in the view. In a few tenths of a second, a 3D table model is retrieved and placed at the appropriate location in the *empty room*. This workflow is repeated for various pieces of furniture, artwork, doors, windows, etc., until all objects of interest have been modeled.

Some additional interactions are available to the user. They may add detail to the scene, by using the "Add" voice command. For example they may say "Add curtains" to add a 3D model of curtains to a previously placed window model. By using a command like "Replace IKEA coffee table" users can replace existing generic furniture with specific 3D models provided by retailers, previewing the results before purchasing. Users can also modify the appearance and material properties of their furniture by saying "Material Dark Wood".

After the furniture placement is complete, an overhead view allows users to visualize their space, rearrange furniture and preview the results instantly. They may also add, remove or replace furniture and modify furniture placement. To preview the changes, the user holds the tablet up as though they were viewing a *window* into the scene, and the system transitions smoothly into a live first person virtual view of the scene (see video), allowing the user to get a better sense for the changes they may have made to the scene.

The working of our system, under the hood, is described in the following sections.

SYSTEM COMPONENTS

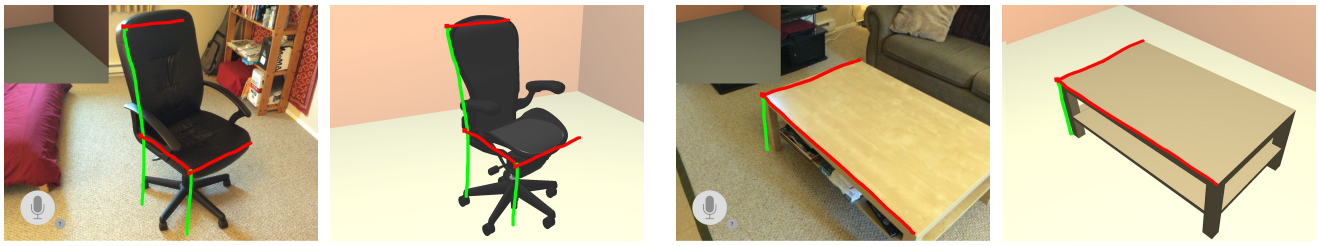
Empty Room Reconstruction

We first reconstruct the shape and scale of the room from a panning video. Building upon our previous work [26], we leverage user-provided corner annotations to recover the Manhattan shape of rooms, i.e., a floorplan consisting of piece-wise planar surfaces with dominant directions. However, we found that for a more accurate mixed reality experience, we additionally need to account for camera translation. This is because it is natural for a user to move the handheld camera roughly along a circular path, while capturing images. Although not by design, this type of camera motion gives rise to significant parallax in the images, particularly indoors where distances are small. For example, in an image frame 1280 pixels wide, horizontal movement objects at distance of 1.2m and 2.4m from the camera differs by about 200 pixels ($\sim$15%). If we did not account for parallax, it would cause misalignment of the 3D model that would adversely affect furniture placement.

Figure 4: Top-down view of the simplified camera motion model, based on the arc radius r and orientation θ.

As illustrated in Figure 4, we assume a circular camera trajectory and known camera height (set manually, per user), based on how users capture panoramas with mobile devices. To assist the user, the interface provides a guide to minimize deviation from the ideal trajectory. Minor deviations in pitch and roll do not affect the reconstruction, as they are not included in our camera model. Any accumulated drift is alleviated by marking the first corner twice (once again at the end) and distributing drift evenly across the frames.

By using the sensor measurements, we are able to estimate extrinsic camera parameters, assuming that the user performs the image capture in accordance to our model. Our model contains two variables (r, θ) that describe the position and orientation of the camera. These two variables are sufficient to determine the extrinsic parameters of a camera that moves along a ring of radius r and that faces outwards at an angle θ from an initial reference. We assume that the other aspects of the camera attitude (pitch, roll) stay constant and that the user remains in the same position while capturing the panorama. In an offline step we use a camera calibration toolbox [2] to determine the intrinsic parameters for the camera. This step is only



(a) Chair sketch (b) Chair model placed (c) Table sketch (d) Table model placed

Figure 3: Sketch based retrieval results. Figures 3a & 3c show user sketches. 3b & 3d show corresponding 3D model placements.

required to be performed once for every camera device and the calibration parameters remains consistent across devices with the same manufacturer, make and model. The camera matrices used by our model are shown in Equations (1) to (4).

$$Camera = Calibration \times Projection \times Extrinsics \quad (1)$$

$$M = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = K \begin{bmatrix} I & 0 \end{bmatrix} T \quad (2)$$

$$M = \begin{bmatrix} \alpha & s & u_0 \\ & \beta & v_0 \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \quad (3)$$

In this case:

$$R = \begin{bmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad t = \begin{bmatrix} 0 \\ 0 \\ -r \end{bmatrix} \quad (4)$$

Our reconstruction algorithm visually tracks the wall edges using the user-annotated template and calculates correspondences across several image frames. These 2D correspondences are then used to reconstruct the absolute depth of the wall edges with structure from motion [30]. Given n known image feature correspondences x_i and corresponding camera matrices M_i, we minimize the following reprojection error, to determine an optimal 3D point $\hat{X}$:

$$min \sum_{i=1}^{n} \|x_i - \hat{x}_i\|^2$$

$$\text{Subject to } \hat{x}_i = M\hat{X}$$

The room shape is scaled according to the calculated 3D locations of walls, while maintaining the manhattan constraint. In this way, we are able to generate an empty 3D room model that closely represents the real scene in terms of geometry, scale and camera motion.

Furniture Placement Interface

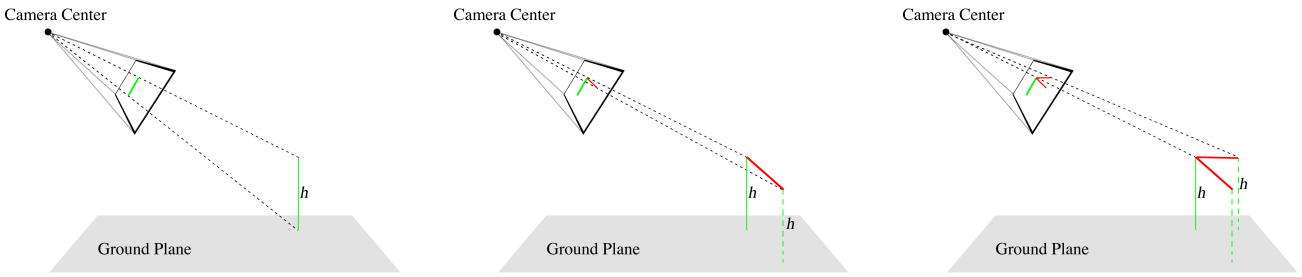
Once we have reconstructed an empty 3D room model the user supplies an initial rotation and translation that aligns the empty room model to the image that is currently being captured by the device camera.

Given the geometry of the room, the user is presented with a mixed reality interface (Figures 2, 3) containing a live view of the device's camera along with an inset of the reconstructed room model from the same viewpoint. We assume that the user stands in the same position as in the room capture phase and rotates the device roughly along a sphere centered at the user. The alignment is maintained by querying the device's gyroscope to determine the current viewpoint on the sphere, and translating and rotating the 3D model to present the scene from this viewpoint, according to the camera model described previously. Over time, gyroscopic drift may cause slight misalignment, but this is alleviated by the user with a simple two-finger pan gesture to realign the model. We have found that minor drift corrections are required in the order of once per minute and can be further mitigated by slow and smooth movement of the device. In the future, we would like to enable more robust tracking using visual features.

Sketch to 3D Model

We use a sketching paradigm to determine the type, position, scale, and orientation of a 3D furniture model that the user wishes to place. The user first issues a voice command or double taps the screen to indicate the start of a sketch. This freezes the live camera view so that the user may more easily draw over the object. Once the display is frozen, the user is able to conveniently sketch the outline of the object. Example sketches are shown in Figure 3. The sketch is recognized and a representative 3D model of the object is retrieved from the database (Described in *Database* section). The 3D model is automatically scaled, positioned and oriented in the 3D room model to correctly align with the live view. The user continues to add furniture in this manner until the modeling process is complete.

Our goal is to recognize and place a 3D furniture object model based on an input 2D sketch. The camera parameters, aligned room model, and camera height of the current view are known and are used as inputs to the algorithm. Reconstructing a general 3D model from a 2D sketch is an ill-posed problem. However, we observe that the major structural components in furniture items such as beds, dressers, tables, chairs etc. are typically planar facets, oriented horizontally or vertically (a notable exception being reclining chairs). If we represent furniture models as a 3D line diagram consisting of horizontal and vertical lines (Figure 8), we can efficiently perform search and retrieval over this compact representation, based on a partial input sketch.



(a) **Step 1:** Vertical line projected into scene to form support

(b) **Step 2:** Non-vertical line projected to intersect with support

(c) **Step 3:** Another non-vertical with the same support

Figure 5: Illustrating the 2D to 3D conversion of the user's sketch

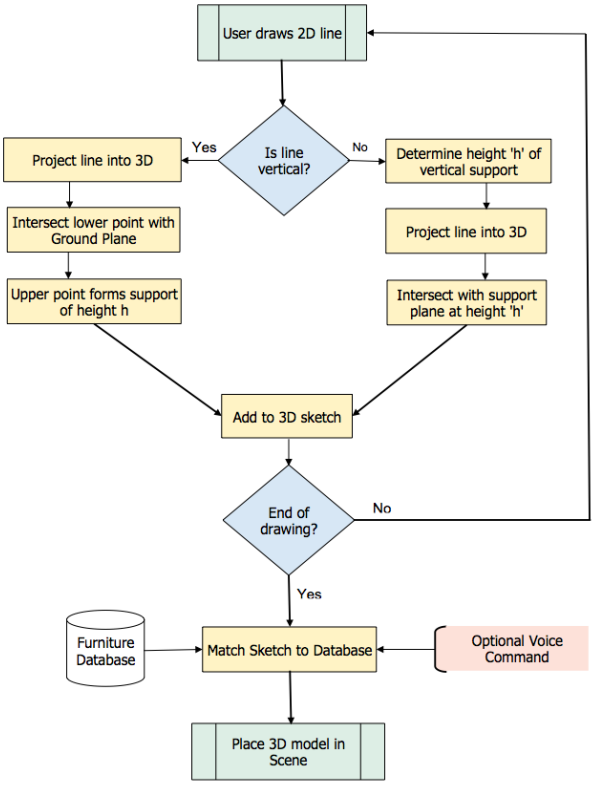


Figure 6: Algorithm to convert a 2D user sketch in to a plausible 3D object sketch.

To place a new object in the scene, the user draws a few line segments on the outline of the object. They may be drawn in any order and there should be enough lines to uniquely scale, position and orient the object. For example, a table would require at least three lines indicating length, depth and height. Three lines are sufficient for a table, because it has 180 degree rotational symmetry about the vertical axis. A chair, however, requires additional lines to uniquely orient the seat back (Figure 3a). For objects that are very thin or are placed on a wall (like artwork, windows etc.), two lines describing

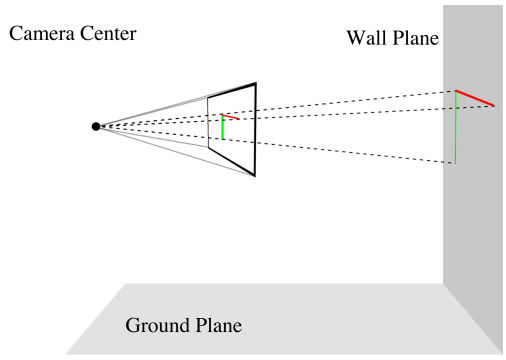


Figure 7: Example of an artwork sketch projected onto a wall.

length and height are sufficient, and a small constant depth is assumed. The lines drawn by the user (henceforth referred to as 'sketch lines') are a 2D projection of horizontal and vertical lines on the actual object (henceforth referred to as 'object lines').

For our system to recognize the user's sketch the following must hold true: 1) Vertical sketch lines must map to vertical object lines and non-vertical sketch lines to horizontal object lines. 2) Vertical lines are supported at the base by the ground plane. 3) Every non-vertical sketch line should be supported by a vertical sketch line, that determines its height above the ground plane.

The reconstruction proceeds according to the algorithm shown in Figure 6. First, the sketch lines are categorized as vertical or non-vertical by sorting them based on slope. Sketch lines that lie within ± 10 degrees of the vertical axis of the image are considered to be vertical, whereas all other lines are considered to be non-vertical. We then cast a ray from our camera center, passing through the base of the vertical sketch line into the 3D scene. The ray is intersected with the ground plane to establish the ground point of the corresponding vertical object line. We then cast a ray passing through the top of the sketch line and intersect it with a vertical plane passing through the ground point, facing the camera. The intersection of the top

ray with this vertical plane determines the upper 3D point of the vertical object line as illustrated in Figure 5a.

Non-vertical sketch lines are projections of horizontal object lines that lie at some height above the ground plane, supported by a vertical sketch line. To determine the supporting vertical sketch line, we find the 2D intersection of the current non-vertical sketch line with all the vertical sketch lines. The selected vertical support has a predetermined height (say h) (as described above), hence the supported horizontal line would also be suspended at the same height h. The non-vertical sketch lines are then projected into the scene and made to intersect with a plane parallel to the ground, suspended at height h, as illustrated in Figures 5a and 5b. We ignore all non-vertical lines that do not have a corresponding vertical support, since we cannot determine their 3D position without ambiguity.

Objects that reside on walls, such as doors, windows and artwork can also be modeled with our sketching framework. After the user has selected the type of object with a voice command, they sketch two lines describing the vertical and horizontal extent of the object. These lines are projected into the scene and intersected with the wall geometry in order to determine their global position as shown in Figure 7. For artwork textures are cropped from the camera image and transferred into the 3D model.

Furniture Database

Once a set of 3D object lines is specified, we perform a database search to determine the furniture item that best matches those lines. Our database consists of a set of 65 3D models of commonly occurring furniture pieces, that have been manually preprocessed to be represented as set of 3D lines, as shown in Figure 8. The database also indexes textual metadata such as the exact name of the object, manufacturer etc. Our database is acquired from freely available online 3D model repositories [29, 33] and consists of both generic and specific (IKEA, Steelcase, Herman Miller etc.) furniture models. Given a partial 3D user sketch, the search problem is therefore two fold: 1) find the furniture model that best matches the partial user sketch & 2) find the best 3D similarity transform that aligns the aforementioned furniture model with the 3D user sketch. By simultaneously finding the closest matching model in the database and the optimal transformation matrix, we can uniquely determine the type, position, orientation and scale of the furniture.

We model rotation only around the vertical axis, non-uniform scale and arbitrary translation as represented by transformation matrix (5). We use an iterative search algorithm, described next, to consider all plausible transformations of the 3D model to fit the partial sketch.

$$M = \begin{pmatrix} s_x \,.\, cos\theta & s_y \,.\, sin\theta & 0 & t_x \\ s_x \,.\, -sin\theta & s_y \,.\, cos\theta & 0 & t_y \\ 0 & 0 & s_z & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5}$$

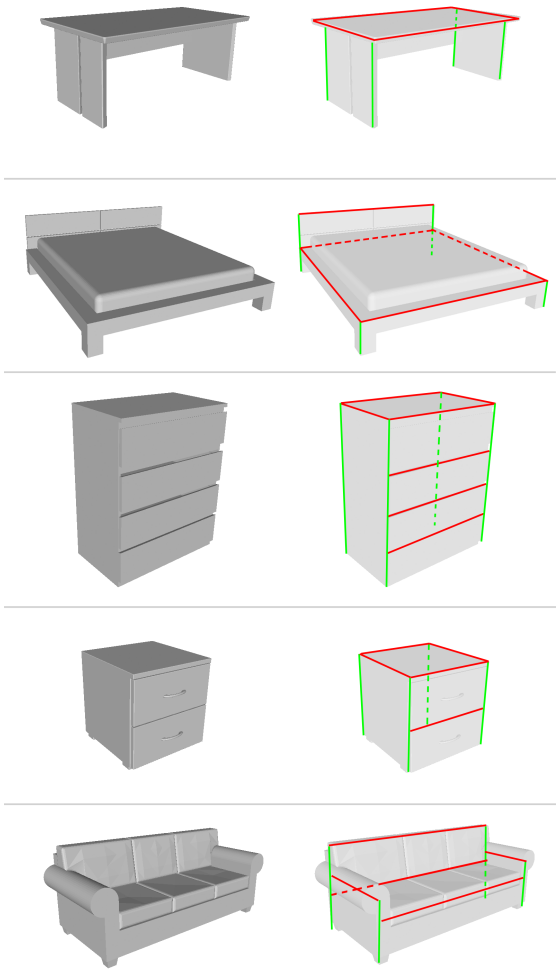


Figure 8: Furniture models in our database with corresponding line representations.

For each model in the database: to obtain a candidate transformation, we pick three orthogonal line segments on the model and associate them with three orthogonal line segments on the user sketch. A least squares solver is used to determine unknown parameters in the candidate transformation matrix. Once a candidate transformation has been obtained we apply the transformation to the database model lines and analyze how well the user sketch conforms to the transformed database model.

We use an exhaustive variant of 3D RANSAC [5] to find the best match. Since our models consist of only a handful of lines, we can afford to exhaustively consider every candidate transformation in order to find optimal consensus. Consensus is determined by measuring the euclidean distance between the end points of the 3D sketch lines (sketch points) and the end points of the transformed database model lines (model points). An inlier is defined as a model point that lies within

a threshold distance of a sketch point. The inlier threshold distance is set to 15cm.

The score for a particular candidate transformation is calculated as the ratio of the number of inliers to the total number of model points. This ensures that model transformations that more completely encompass the sketch points score higher and thus eliminates bias towards selecting simpler database models, that would otherwise easily match most sketches.

After every candidate transformation has been scored, we move on to the next model in the database. We log the maximum score for each database model and store the corresponding optimal candidate transformation. Once all models have been analyzed, the one with the maximum score is transformed and added to the 3D scene. If two or more models have an equal score, our system choses the first model that appears in the database, since based on the information provided it is not possible to break a tie. However, voice commands can be used to tiebreak.

Voice Commands

The user can optionally issue voice commands at any point by tapping the listen button (See Figure 2). The system detects silence at the end of speech and processes the preceding command according to Table 1. The most common command is a description of the object is being modeled, e.g., "Table" or "IKEA Desk". Voice commands allow for easy and unambiguous selection of objects from the database. As the size of the database increases, we have found that using voice recognition dramatically simplifies the retrieval problem, and enables more performant sketch-based retrieval. If a voice command describing an object is issued and the description is indexed in the database, the live view is frozen in place, so that the user may sketch the outline of the selected object. The sketch to 3D model algorithm is then run on only the models matching the voice query. The closest match is then placed in the scene based on the optimal transformation.

We developed a taxonomy for additional commands that may be issued in order to add, replace or modify furniture as listed in Table 1. The "Add" command appends a pre-aligned 3D model on the previously placed item, for instance the user may add a "iMac" on any desk or add "Curtains/Blinds" to windows. The "Replace" command replaces the previously placed item with the currently selected item. If the objects are from the same category, e.g., chairs, the rotation, scale and translation are transferred to the new object. The "Material" command modifies the material properties of the current object. This is achieved by replacing the material property description file of the 3D model from say wood to metal or leather to suede. We leverage the Pocketsphinx [11] library for voice command recognition. Testing the performance of the voice recognition library was outside the scope of this work.

Furniture Rearrangement and Replacement

After all the models have been placed, the user is able to view and edit the captured scene. The system provides a top-down orthographic view of the scene and an intuitive touch and drag interface that allows users to translate, rotate, scale furniture and automatically align items to the dominant directions. With this interface, the user may correct any errors in the furniture placement. Users may also rearrange furniture or replace furniture with other items from the database of a similar nature, for instance by invoking the 'replace' command, a suede couch can be swapped out for a different style couch made out of leather. Users may also delete furniture they wish to remove from the scene.

Command	*Type*	*Summary*
Furniture Item	Stand-Alone	Freezes the camera frame and prompts user to sketch the item into scene
Add	Modifier	Adds selected item on top of previously placed item
Replace	Modifier	Replaces previously placed item with currently selected item
Material	Modifier	Replaces material properties of previously placed item

Table 1: Summary of voice commands and their functions

In order to preview changes, the user holds the tablet up as though they were viewing a *window* into the scene. This gesture is automatically detected by our system and the view transitions smoothly from the overhead to a first person view based on the current position and rotation of the device. The user may now move the device and hence the viewing window around along a spherical bound from their current position. The virtual field of view roughly corresponds to the human field of view and is designed to give the user a sense of *being there*.

RESULTS

We present results and evaluation from six indoor environments. All experiments were performed using an iPad Air Retina running iOS 8.3. The captured scenes have considerable variation in layout and function, and include two bedrooms, two offices, a living room and a bathroom. The database used for the tests contains 65 3D models. For each environment we captured a 3D model that captures the salient features of the scene, such as room shape, furniture, doors, windows, textured artwork etc.

In figures 1, 9 & 10 we show 3D models captured with our system alongside a corresponding view of the real scene. The real scene was photographed with a DSLR camera in order to capture a wider field of view (and hence more objects of interest). We have attempted to closely emulate the same view with our system in order to form a suitable visual comparison for the reader. The visual similarity indicates that the captured 3D models are suitable for the purposes of furniture rearrangement, replacement and visualization, which is a goal of our system. Additionally, in a following section, we have evaluated the accuracy of the furniture placement against ground truth. Ground truth room dimensions and furniture positions were manually measured for each scene using a tape measure.

Time Comparison: Because no comparable mobile in-situ scene modeling system has been demonstrated previously in the literature, we therefore chose to compare the performance

(a) Real Scene (b) Captured Virtual Scene (c) Overhead View

(d) Real Scene (e) Captured Virtual Scene (f) Overhead View

Figure 9: Visual results from Office 1 and Living Room scenes

			Autodesk Homestyler			Our System	
Scene	Area	# Items	Measurement Time	Modeling Time	Total Time	Total Time	Furniture Error
*Bedroom*1	28.0m^2	23	18m 12s	11m 19s	29m 31s	7m 12s	42.57cm
LivingRoom	26.1m^2	12	9m 14s	8m 57s	18m 11s	6m 02s	37.19cm
*Office*1	13.9m^2	14	8m 45s	8m 33s	17m 18s	6m 32s	33.77cm
*Bedroom*2	21.1m^2	13	10m 01s	8m 34s	18m 35s	5m 58s	40.03cm
*Office*2	13.8m^2	12	8m 12s	9m 09s	17m 21s	6m 29s	36.61cm
Bathroom	15.5m^2	9	6m 20s	4m 54s	11m 14s	5m 06s	29.11cm
Avg.			10m 17s	8m 34s	18m 41s	6m 13s	36.55cm

Table 2: Quantitative results for the environments tested.

of our approach to Autodesk Homestyler [9] (an ex situ indoor scene modeling tool), since it produces the most similar end results. In Table 2 we report the time taken to capture the 3D models by an experienced user of both systems. The total time for ex situ modeling is split into time taken to make physical ground truth measurements and time taken to reproduce the scene with a traditional desktop interface. The total time for our system includes empty room reconstruction, furniture modeling and manual error correction, if needed (unrecognized voice commands, sketching errors etc.). Note that our system does not require any physical measurements. However, as a tradeoff, our furniture placement may deviate slightly from ground truth.

Our results indicate that the proposed approach is faster than using a traditional ex situ desktop interface in almost all cases. With our system, the user can capture a scene in a few minutes, in situ, without the need for manual measurement, which we believe will greatly improve the efficiency of interior design applications.

Furniture Placement Accuracy: We define a furniture error metric that quantifies the mean 2D shift of the furniture items by measuring the metric displacement of corresponding corner vertices of furniture in the model and ground truth. This is calculated by overlaying the 2D overhead views of the captured and ground truth results and for each furniture item, measuring the euclidean distance between actual and captured furniture bounding boxes. For example, if each corner of a modeled table is 5cm away from the ground truth, total error for four corners of the table is 20cm. In this way, our metric attempts to capture absolute error in scale and position of placed furniture models. (Refer to Table 2 for error measurements)

LIMITATIONS AND FUTURE WORK

We presented the design of a novel system for indoor 3d modeling that leverages the in situ and interactive aspects of the mobile platform. There are several avenues for future research in this area. First, using a larger ("internet-scale") furniture database [3] coupled with efficient search would enable the capture of a wider variety of scenes. It would be possible to index such a database, to be compatible with our technique,

(a) Real Scene (b) Captured Virtual Scene (c) Overhead View

(d) Real Scene (e) Captured Virtual Scene (f) Overhead View

(g) Real Scene (h) Captured Virtual Scene (i) Overhead View

Figure 10: Visual results from Bedroom 2, Office 2 and Bathroom scenes

with automated methods like dominant plane extraction [16]. To keep the search performant, we believe the voice input in particular has significant advantages, as queries like "black office chair" can eliminate a large number of irrelevant candidates based on textual tags in the database. Furthermore, to disambiguate between similar looking models that have been retrieved, it is possible to employ a selection UI similar to [28] and have the user pick the closest match.

The ability to model more general furniture items (that are not just horizontally and vertically faceted) and capture the appearance (texture, lighting) of the furniture and scene more accurately are also interesting research problems. Our artwork capture technique is one such attempt at image based appearance modeling. Currently objects that cannot be represented with straight lines can be easily approximated, such as the flowerpot in 9 is added by sketching a bounding box. Similarly, oddly structured objects, such as chairs without back legs are modeled by approximating the sketch of an ordinary chair. Currently, the system supports the modeling of only one room at a time. However, this can be easily extended to capture several rooms during the same session and merge them into a single 3D model of the entire home.

Our tracking currently relies on the device gyroscope, which can suffer from drift. Implementing more robust tracking with visual features would enable an accurate augmented overlay and allow the user to move freely within the scene. This would mitigate issues of occlusion and limited field of view while modeling objects.

A system like ours, that runs on widely available commodity mobile hardware, is easier to scale to several users and can enable acquisition of large datasets of real world room models and furniture relationships. A large-scale user study would be very valuable, and we intend to do so in future work. The data collected would be useful to train computer vision algorithms for automatic reconstruction and recognition tasks.

Finally, once the model has been captured, there is potential to enable compelling applications such as automatic optimal furniture rearrangement [19] and furniture removal/replacement in the original images [13].

ACKNOWLEDGEMENTS

This work was supported in part by the National Science Foundation (IIS-1250793), the University of Washington Animation Research Labs, and Google. We would also like to credit the following Sketchup [29] users Alons, SpinyMcSpleen3264, Herman Miller, Joseph D., KHITCHZ, Webric (Richard), gui, Steve K., Joey, Ryno, anonymous, Michal S., Tanjoodo, swisscows, Meco Office, Yen Ha, Sedus, Steelcase, lcr2, TinkerToy, sdmitch40, Daweeze and TurboSquid [33] users viktor7777, Fworx, AG4T, AlpArt, vandoan, CSI Renderers for the use of their 3D models in this presentation.

REFERENCES

1. 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
2. Jean-Yves Bouguet. 2013. Camera Calibration Toolbox. http://www.vision.caltech.edu/bouguetj/calib_doc/. (2013). Accessed: 2016-05-27.
3. 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].
4. Mathias Eitz, Ronald Richter, Tamy Boubekeur, Kristian Hildebrand, and Marc Alexa. 2012. Sketch-based Shape Retrieval. *ACM Trans. Graph.* 31, 4, Article 31 (July 2012), 10 pages. DOI: http://dx.doi.org/10.1145/2185520.2185527
5. 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
6. Thomas Funkhouser, Patrick Min, Michael Kazhdan, Joyce Chen, Alex Halderman, David Dobkin, and David Jacobs. 2003. A Search Engine for 3D Models. *ACM Trans. Graph.* 22, 1 (Jan. 2003), 83–105. DOI: http://dx.doi.org/10.1145/588272.588279
7. 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
8. HoloLens. 2015. Microsoft Corp. https://www.microsoft.com/microsoft-hololens/en-us. (2015). Accessed: 2016-05-27.
9. Homestyler. 2013. Autodesk. http://www.homestyler.com. (2013). Accessed: 2016-05-27.
10. 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
11. D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky. 2006. Pocketsphinx: A Free, Real-Time Continuous Speech Recognition System for Hand-Held Devices. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, Vol. 1. I–I. DOI: http://dx.doi.org/10.1109/ICASSP.2006.1659988
12. Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. 1999. Teddy: A Sketching Interface for 3D Freeform Design. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*. ACM, 409–416. DOI: http://dx.doi.org/10.1145/311535.311602
13. Natasha Kholgade, Tomas Simon, Alexei Efros, and Yaser Sheikh. 2014. 3D Object Manipulation in a Single Photograph Using Stock 3D Models. *ACM Trans. Graph.* 33, 4, Article 127 (July 2014), 12 pages. DOI: http://dx.doi.org/10.1145/2601097.2601209
14. Hyejin Kim, Gerhard Reitmayr, and Woontack Woo. 2013. IMAF: In Situ Indoor Modeling and Annotation Framework on Mobile Phones. *Personal Ubiquitous Comput.* 17, 3 (March 2013), 571–582. DOI: http://dx.doi.org/10.1007/s00779-012-0516-3
15. 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
16. Manfred Lau, Akira Ohgawara, Jun Mitani, and Takeo Igarashi. 2011. Converting 3D Furniture Models to Fabricatable Parts and Connectors. In *ACM SIGGRAPH 2011 Papers (SIGGRAPH '11)*. ACM, Article 85, 6 pages. DOI:http://dx.doi.org/10.1145/1964921.1964980
17. Manfred Lau, Greg Saul, Jun Mitani, and Takeo Igarashi. 2010. Modeling-in-context: User Design of Complementary Objects with a Single Photo. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium (SBIM '10)*. Eurographics, 17–24. http://dl.acm.org/citation.cfm?id=1923363.1923367
18. 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
19. Paul Merrell, Eric Schkufza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. 2011. Interactive Furniture Layout Using Interior Design Guidelines. *ACM Trans. Graph.* 30, 4, Article 87 (July 2011), 10 pages. DOI:http://dx.doi.org/10.1145/2010324.1964982

20. 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

21. Tom Novotny, Irma Lindt, and Wolfgang Broll. 2006. A Multi Modal Table-top 3D Modeling Tool in Augmented Environments. In *Proceedings of the 12th Eurographics Conference on Virtual Env. (EGVE'06)*. Eurographics, 45–52. DOI: http://dx.doi.org/10.2312/EGVE/EGVE06/045-052

22. 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

23. Home Planner. 2013. IKEA. http://www.ikea.com/ms/en_JP/rooms_ideas/splashplanners.html. (2013). Accessed: 2016-05-27.

24. Project Tango. 2014. Google Inc., ATAP. https://www.google.com/atap/projecttango/. (2014). Accessed: 2016-05-27.

25. 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

26. 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

27. 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

28. HyoJong Shin and Takeo Igarashi. 2007. Magic Canvas: Interactive Design of a 3-D Scene Prototype from Freehand Sketches. In *Proceedings of Graphics Interface 2007 (GI '07)*. ACM, 63–70. DOI: http://dx.doi.org/10.1145/1268517.1268530

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

30. 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

31. Mengu Sukan, Steven Feiner, Barbara Tversky, and Semih Energin. 2012. Quick Viewpoint Switching for Manipulating Virtual Objects in Hand-held Augmented Reality Using Stored Snapshots. In *Proceedings of the 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR '12)*. IEEE, 217–226. DOI: http://dx.doi.org/10.1109/ISMAR.2012.6402560

32. Michael Tsang, George W. Fitzmaurice, Gordon Kurtenbach, Azam Khan, and Bill Buxton. 2002. Boom Chameleon: Simultaneous Capture of 3D Viewpoint, Voice and Gesture Annotations on a Spatially-aware Display. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology (UIST '02)*. ACM, 111–120. DOI: http://dx.doi.org/10.1145/571985.572001

33. TurboSquid. 2016. TurboSquid. http://www.turbosquid.com/. (2016). Accessed: 2016-05-27.

34. Julien Valentin, Vibhav Vineet, Ming-Ming Cheng, David Kim, Jamie Shotton, Pushmeet Kohli, Matthias Nie, 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

35. Anton van den Hengel, Rhys Hill, Ben Ward, and Anthony Dick. 2009. In situ image-based modeling. *IEEE / ACM International Symposium on Mixed and Augmented Reality* (2009), 107–110. DOI: http://dx.doi.org/10.1109/ISMAR.2009.5336482

36. Baoxuan Xu, William Chang, Alla Sheffer, Adrien Bousseau, James McCrae, and Karan Singh. 2014. True2Form: 3D Curve Networks from 2D Sketches via Selective Regularization. *ACM Trans. Graph.* 33, 4, Article 131 (July 2014), 13 pages. DOI: http://dx.doi.org/10.1145/2601097.2601128

37. Kun Xu, Kang Chen, Hongbo Fu, Wei-Lun Sun, and Shi-Min Hu. 2013. Sketch2Scene: Sketch-based Co-retrieval and Co-placement of 3D Models. *ACM Trans. Graph.* 32, 4, Article 123 (July 2013), 15 pages. DOI:http://dx.doi.org/10.1145/2461912.2461968

38. Robert C. Zeleznik, Kenneth P. Herndon, and John F. Hughes. 1996. SKETCH: An Interface for Sketching 3D Scenes. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96)*. ACM, 163–170. DOI: http://dx.doi.org/10.1145/237170.237238