Interactive In-Situ Scene Capture on Mobile Devices

Aditya Sankar

A dissertation submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2017

Reading Committee:

Steven M. Seitz, Chair

Brian Curless

Michael Cohen

Program Authorized to Offer Degree:
UW Computer Science and Engineering

University of Washington

Abstract

Interactive In-Situ Scene Capture on Mobile Devices

Aditya Sankar

Chair of the Supervisory Committee:
Professor Steven M. Seitz
UW Computer Science and Engineering

Architectural visualizations of indoor scenes enable compelling applications in several areas, such as real estate, interior design, cultural heritage preservation, and more recently, immersive virtual reality. Computer-Aided Design (CAD) tools have been invaluable for creating such visualizations. However, creating detailed, attractive visualizations of scenes remains a challenging task, particularly for non-experts. User interfaces for CAD tools tend to be complex and require significant manual effort to operate. These tools are also designed to be used ex-situ, or off-site, making it difficult to record and reproduce details faithfully.

In this thesis, I propose novel techniques and systems for interactive in-situ scene capture on mobile devices that let non-expert users quickly and easily capture useful architectural visualizations of indoor scenes. These systems are built upon two key insights; 1) sensors on mobile devices can be leveraged to capture important aspects of the scene such as dimensions, room shape, furniture placement, etc., and 2) an in-situ user can assist in the modeling task by acting as a guide for reconstruction and object recognition algorithms. Based on these insights, the semi-automatic systems that I propose combine the strengths of the user, who is good at high-level semantic reasoning, and the computer, which excels at combinatorics and numerical optimization.

I present three systems in this thesis. First, I present a smartphone application designed to visually capture homes, offices and other indoor scenes. The application leverages data

from smartphone sensors such as the camera, accelerometer, gyroscope and magnetometer to help reconstruct the indoor scene. The output of the system is two-fold; first, an interactive visual tour of the scene is generated in real time that allows the user to explore each room and transition between connected rooms. Second, by marking distinct room features such as corners and doors, the system generates a 2D floor plan and accompanying 3D model of the scene, under a Manhattan-world assumption. This approach does not require any specialized equipment or training, and is able to produce accurate floor plans.

I then describe an interactive system to capture CAD-like 3D models of indoor scenes, on a tablet device. 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. The 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, a framework for sketch based model retrieval, and the underlying algorithms that process the input to produce useful reconstructions.

Finally, I extend the in-situ modeling approach to 3D-aware mobile devices, an emerging class of devices that can more richly sense the 3D nature of our world using depth sensing and self-localization. I propose a novel interactive system to further simplify the process of indoor 3D CAD room modeling on such devices. The proposed system leverages the sensing capabilities of a 3D aware mobile device, recent advances in object recognition, and a novel augmented reality user interface, to author 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 automatically retrieves and places a matching 3D model into the scene, from a large online database – a modality that proves to be faster, more accurate, and easier than using traditional desktop tools.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I wish to express my sincere appreciation and gratitude towards my advisor, Steve Seitz. His unfaltering faith in my ideas and ability have helped me immensely in becoming an effective explorer and communicator of ideas. Steve's remarkable intuition for identifying exciting problems and devising *"magical"* solutions has been a constant source of inspiration. This thesis would not be possible without his patience, guidance and support over the years.

I would also like to thank the faculty and my collaborators at UW including Brian Curless, Michael Cohen, Ira Kemelmacher-Shlizerman, Eli Shechtman, Linda Shapiro, Brian Johnson, Alex Anderson, Sandy Kaplan (who helped edit certain chapters of this dissertation) and several others. I thank my colleagues at GRAIL and UW CSE, who have made the journey of graduate school enjoyable and interesting. Also, the friends I have made in Seattle, who have had a great impact on me, and have been my family away from home. There are too many colleagues and friends to name here individually, but you know who you are. Thank you. I have learned so much from all of you.

I'm very fortunate to have had some incredible mentors on my journey towards my PhD. In particular I wish to thank P. Anandan, Joseph Joy, Kentaro Toyama, and Rick Szeliski for their guidance and encouragement, and for giving me the opportunity to work on some truly wonderful projects at Microsoft Research, that kindled my interest in computer science research. I hope to live up to their examples and serve as mentor to students in the future.

Finally, I wish to thank my parents and grandparents for their love and sacrifices that have allowed me to pursue my dreams.

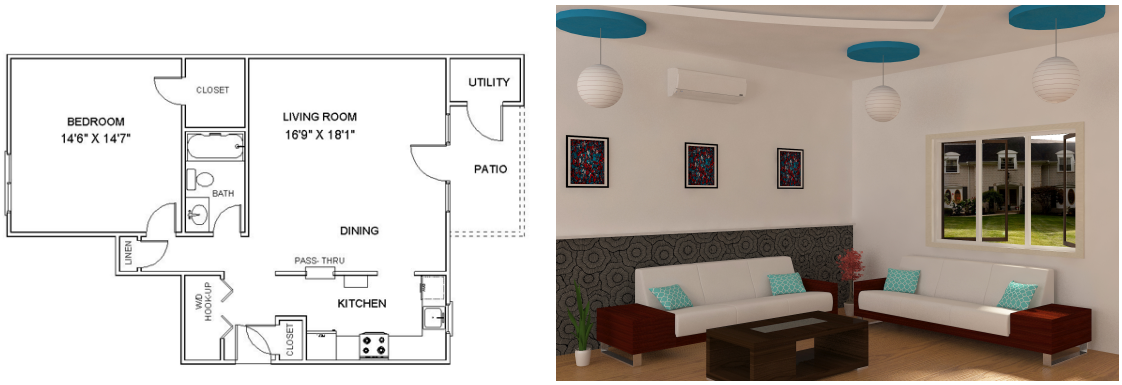
DEDICATION

To my parents, Raghu & Rohini. To all my family & friends. And in memory of Eric & Dan.

Chapter 1

INTRODUCTION

Architectural visualizations of indoor scenes depict important aspects of spaces, such as dimensions and the relationships between rooms, furniture, and other structural features. Two commonly used visualizations are 2D floor plans and 3D CAD models. A 2D floor plan, (Figure 1.1a), is similar to a conventional map and presents an to-scale, orthographic cross-section of a scene that contains symbols representing walls, doors, windows, and furniture. A 3D CAD model, (Figure 1.1b) is a more faithful rendering of a scene that conveys depth and layout of a space in the form of a virtual model, allowing viewers to literally see the design elements as if they were virtually present in the scene. Both 2D and 3D visualizations are commonly used in conjunction to provide a better overall understanding of scene layout and appearance.



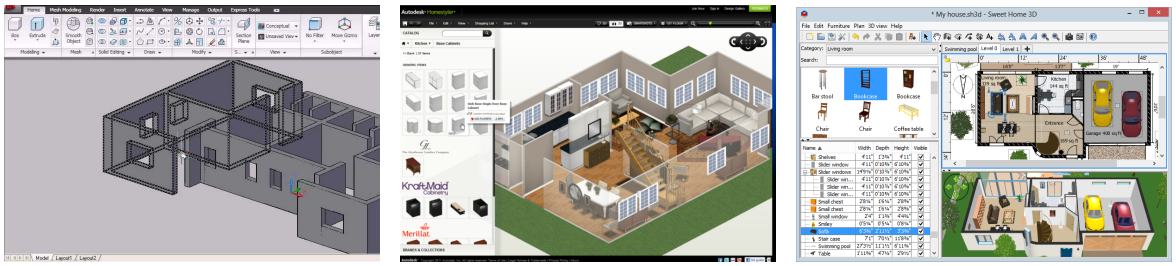
(a) *2D Floor Plan* ***(b)*** *Living Room Model*

Figure 1.1: *(1.1a) A 2D Floor Plan of a typical home and (1.1b) an indoor 3D CAD model of a living room, both created manually for real estate purposes*

These visualizations enable many compelling applications. An interior designer could use such a record to remodel, re-furnish or rearrange an existing living space, while a real-estate agent could 'virtually' introduce potential homeowners to new homes prior to their being physically present in the space. An archeologist could record and archive walkthroughs of ancient ruins or heritage sites, and a museum curator could publish walkthroughs of exhibitions/events. Further, an insurance adjuster could catalog objects of value, such as antiques, art, appliances, etc. A crime scene investigator could use such a survey to recreate a narrative and help deconstruct events in order to help solve a crime. Last but not least, casual users could capture models of their homes, enabling applications such as mobile robot navigation, home automation, and enhanced virtual and augmented reality experiences.

Traditionally, the tools used to create such a visualization were a measuring tape, camera and drafting table. The surveyor would first record dimensions of the walls, doors, and windows and note the position and extent of furniture items. He or she might then capture several photographs of the scene, from multiple angles, for later reference. Finally, after adjusting the drafting table to the correct height and angle, and preparing the drafting paper and a sharpened pencil, the surveyor would begin to convert the measurements into a detailed graphical representation of the scene. Many designers, architects and draftsmen still rely on paper and pencil to produce these graphics.

In the last couple of decades, practitioners increasingly rely on desktop Computer-Aided Design (CAD) software to record, modify, analyze or optimize their designs. Since the invention of Sketchpad [107] – a.k.a. the 'Robot Draftsman', a precursor to CAD software – computers have been an invaluable aid to modeling and designing our world. Academics, designers, and engineers have worked to improve the human-computer interface for these modeling tools to make them more powerful, easier to use, and more efficient. Despite significant advances, creating detailed, attractive 3D CAD models and floor plans of scenes remains a challenging task, particularly for non-experts. Interfaces tend to be complex (Figure 1.2) and require skill and experience to operate. Significant manual effort and hours are also required, especially when creating models from scratch. Further, most tools are



(a) Revit *(b) HomeStyler* *(c) SweetHome3D*

Figure 1.2: *While powerful and full-featured, existing indoor CAD modeling tools have complex and difficult to learn user interfaces, as shown in these three examples.*

primarily used in an ex-situ manner, i.e., used off site, away from the location being modeled, making it difficult to record and reproduce details faithfully. This thesis poses the following two research questions: First, is it possible to design user interfaces for indoor CAD modeling, that operate in-situ, with the goal of improved efficiency and accuracy? Second, is it possible to make a leap from measuring tapes, drafting tables and desktop CAD tools, to a more widely accessible digital tool, available to novice users and professionals alike?

To address these questions, I first propose replacing the canonical tape measure and drafting table with another powerful device, one that we all carry in our pockets: the mobile phone. Mobile phones, tablets and other 'smart' devices are becoming increasingly ubiquitous. With almost 80% of U.S. mobile subscribers now owning smartphones (as opposed to a basic feature phone[1]), 50% owning tablets [22, 16], and similar rising trends across the world, it is reasonable to assume that implementing a solution on a commodity mobile device can enable scene capture on a very large scale. Mobile devices also have the capability to richly sense the world around them. A majority of devices currently on the market are equipped with a camera, gyroscope and magnetometer. And an emerging class of 3D-aware devices include

[1]Feature phones are mobile phones that are limited in capabilities in contrast to modern smartphones. Feature phones typically provide voice calling and text messaging functionality, in addition to basic multimedia and Internet capabilities [123]

motion tracking and depth sensing capabilities[2]. Finally, almost all smart devices have a familiar interaction surface, the touch screen, which is generally more intuitive than desktop interfaces.

Second, I propose to involve the user in the modeling task as a guide for reconstruction and object selection algorithms. Automatic scene reconstruction and object detection is susceptible to error. However, as humans we are inherently talented at high-level recognition tasks and can easily correct a machine's mistakes. We are not easily fooled by textureless objects, reflections, occlusions, as state of the art recognition and reconstruction algorithms often are. We are also equipped with a strong semantic understanding of a scene, which may include information such as the types of objects present in the scene, e.g. 'chair,' 'table,' etc., or even awareness as specific as the exact brand and model of furniture, e.g., 'an Eames lounge chair and ottoman.' Finally, we have the ability to physically move around a scene, capturing various viewpoints and objects of interest that may not be visible from a single vantage point. Leveraging the preceding insights, I describe novel user interfaces for mobile devices

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

Figure 1.3: *One of the systems proposed in this thesis captures a 3D CAD model of an indoor scene, in a few minutes, on a commodity tablet.*

that harness the advantages of traditional desktop CAD software but present them in-situ, letting the user record aspects of the scene by annotating and sketching over a live view of

[2]More on background on 3D-aware devices available in Section 2.3

the scene, captured during the modeling process. To this end, the techniques I introduce leverage basic semantic input (annotations, sketches and voice commands) from the user, for guidance, and offload tedious reconstruction and modeling tasks to the computer, attempting to combine the strengths of both agents and have them work in synchrony. The systems I propose are designed to be operated by novice users and produce results in real time on commodity smartphones or tablets. The proposed systems do not require any additional hardware or extended user training, thereby making them easy to distribute and scale to several users. The systems are capable of producing three primary visualizations of the scene. First a visual walkthrough, that lets users interactively explore an image-based virtual tour of a scene. Second, a 2D floor plan that captures the shapes of rooms and their relative size and position within the scene. And finally an editable 3D CAD model of each room, containing accurate representations of walls, furniture, doors, windows, artwork etc. Using these proposed systems, a large audience of both casual and professional users can capture, visualize and reconstruct of a wide variety of interesting indoor scenes, from homes and offices to museums and heritage sites.

Several challenges must be overcome to realize such end-to-end systems on mobile devices. Although mobile devices contain a range of useful of sensors, they still lack the ability to convert the sensor data into semantically meaningful information about the scene, as algorithms have not yet reached the point where you can achieve results like Figure 1.3 automatically on a mobile device. Also, touch screens, while intuitive and widely understood, are small and lack many affordances of a desktop interface. Therefore, special care must be applied when designing the user interface for such form factors to maintain flexibility and power when modeling complex environments.

1.1 Contributions

There are two fundamental research challenges identified above: (1) the design of intuitive user interfaces that allow for in-situ modeling and (2) the discovery of algorithms for scene reconstruction and modeling that are fast and capable of running in real-time on commodity

mobile devices. In this thesis, I present novel interface designs and new algorithms that address each challenge. These contributions can be broadly categorized as follows:

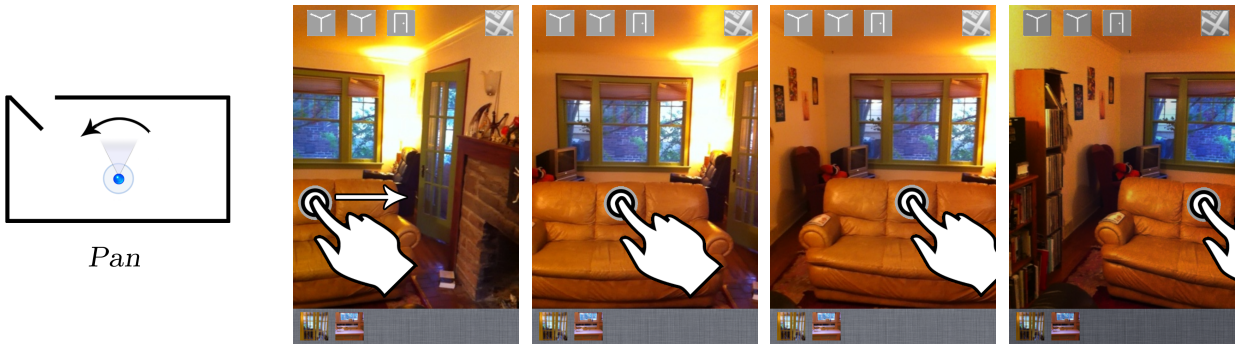


Figure 1.4: *Proposed interaction for exploring a room by viewing a video-based panorama on a mobile device*

Figure 1.5: *Proposed interaction for transitioning between connected rooms using a video walk-through.*

1. **A Framework for Visual Tour Capture on a Smartphone**: I first propose a framework for capturing and spatially indexing visual data on a smartphone, in order to create an interactive visual tour, that lets users explore an environment and presents them with a sense of *'being there'*. Prior approaches [66, 5] have required sophisticated

omnidirectional camera rigs and several hours of offline processing in order to create such tours. However, the method I propose is implemented end-to-end on a commodity smartphone and can generate tours in a matter of minutes.

To generate a tour, the user first captures video of an indoor scene by following a few simple guidelines. Alongside image data, the proposed system also records measurements from the gyroscope, accelerometer and magnetometer. This data is then fused to generate an image-based rendering of the scene that lets a viewer interactively play back a virtual tour in which he/she can explore a 360° panorama of each room (Figure 1.4) and also 'walk' between connected rooms (Figure 1.5). The generated visual tours have several uses; 1) they can provide a remote viewer with a sense of presence, 2) they can be used to annotate objects of interest such as artwork, furniture, etc., and 3) they can be used as a basis for 3D reconstruction of the scene, as described next.

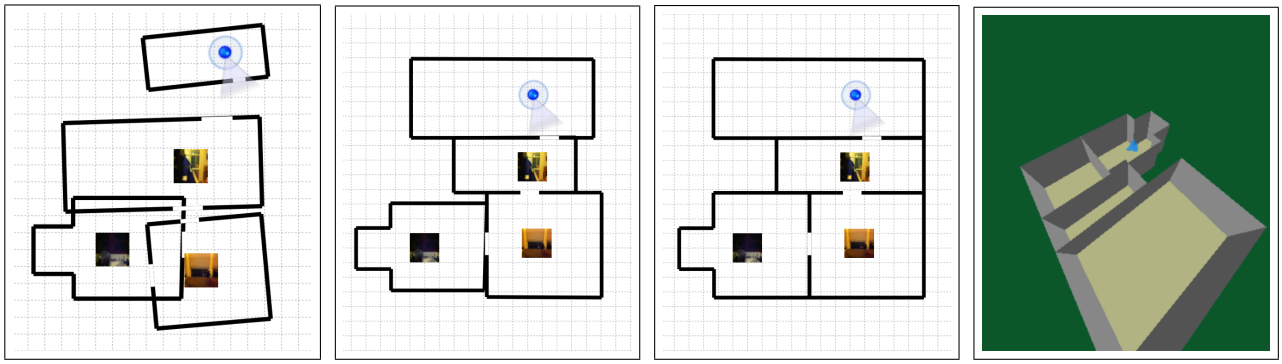
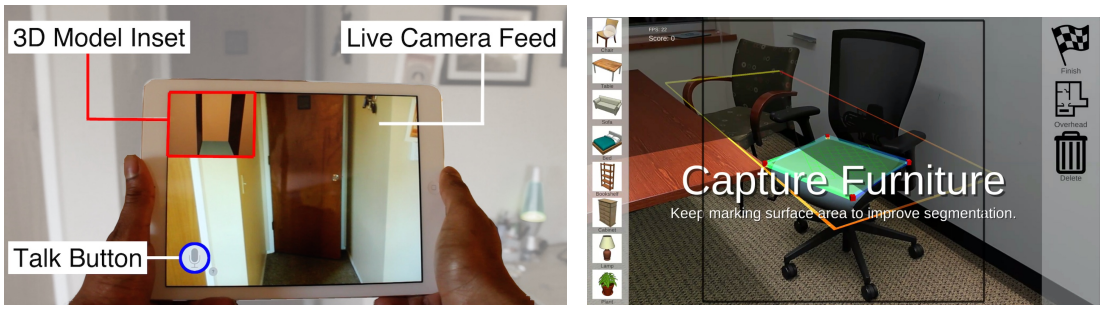


Figure 1.6: *Using a proposed interactive modeling method, a 2D floorplan of a scene can be entirely recreated in a few minutes on a mobile device.*

2. **An Interactive Technique for Floor Plan Estimation**: I propose an interactive photogrammetric modeling approach to recover room features from the visual tour (described previously), such as the location of room corners and doors. With this information, I use a novel reconstruction algorithm to estimate the position of the

camera and determine the approximate shape of each room along with the overall floor-plan representation of the environment (Figure 1.6). The method is designed to capture *Manhattan World* [26] scenes, i.e., scenes in which all surfaces are aligned with three dominant directions.

Both the interactive modeling approach and the reconstruction algorithm run in a few minutes on a commodity smartphone and can therefore potentially be used by realtors and homeowners to generate floor plans of homes, or by business owners to create maps of their establishments, all without the need for any specialized training or extra equipment.



(a) *Sketching Interface on iPad* ***(b)*** *Modeling Interface on Tango*

Figure 1.7: *Screenshots of two interfaces proposed for in-situ CAD modeling. (1.7a) utilizing only camera, accelerometer and gyroscope, and (1.7b) using additional depth and tracking data.*

3. **Interfaces for In-Situ Furniture Modeling**: Beyond floor plan estimation, it is also interesting to capture the contents within the room such as furniture, doors, windows, artwork, etc. and generate a full 3D model containing these objects. Such a 3D model has many demonstrated uses within the interior design, household furniture and entertainment/virtual reality industries.

   To capture such a model, I propose two in-situ, multi-modal user interfaces (Figure 1.7) designed to harness the user's semantic understanding of these objects in order to reproduce a faithful 3D CAD model representation of a scene. Instead of modeling

individual objects from scratch, a key insight is to leverage existing large online 3D model databases. Recent efforts such as SketchUp Warehouse [102] and Shapenet [17] have indexed a wide array of user-generated 3D models of real-world objects like aeroplanes, cars, buildings, furniture, appliances, etc. This work leverages subsets of these 3D databases, that are relevant to indoor scene modeling.

The algorithms I propose, process user input (voice, sketching, annotation) and sensor data from the mobile devices to efficiently retrieve matching 3D models from the database and and then place them at the appropriate locations in the virtual scene. The first interface (Figure 1.7a) relies on the user sketching the rough shape of the object (such as a table, chair, sofa, etc.) and providing the algorithm further guidance via the use of familiar voice commands. The second interface (Figure 1.7b), implemented on a 3D-aware mobile device (viz. one that can accurately track its 3D position and orientation in the world and estimate the depth of points in a scene), lets the user model objects with only a few taps, requiring less effort than sketching.

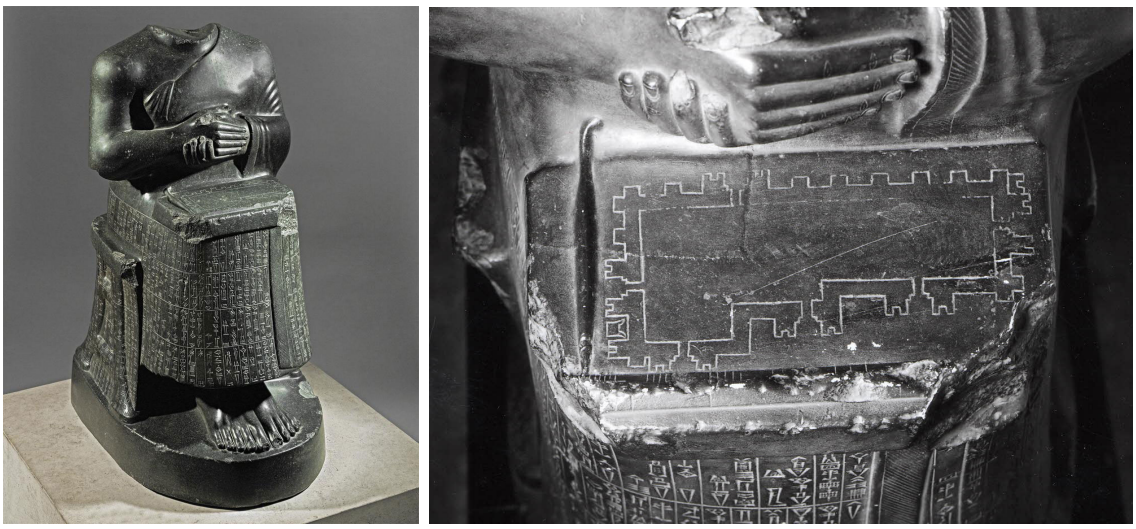
Both interfaces let users visualize the 3D model in real-time on the device, as they are constructing it. At the end of the modeling stage, the user may view and edit the constructed model directly on the device or they can upload the 3D model to their favorite desktop CAD modeling tool via a standard 3D file format.

The rest of the thesis is structured as follows. Chapter 2 provides a brief history of architectural visualizations, and also provides a background for various mobile tools used in this thesis. Chapter 3 describes the Visual Tour framework and Floor Plan estimation technique. Chapter 4 describes the first interface for In-Situ Furniture Modeling, which relies on user sketches and voice commands and is implemented on a commodity tablet device. Chapter 5 describes the second interface for In-Situ Furniture Modeling, which is implemented on a 3D-aware mobile device, along with associated user studies and feedback. Chapter 6 presents conclusions and discusses several directions for future work.

Chapter 2

BACKGROUND

2.1 Brief History of Architectural Visualization



(a) *Statue of Gudea* ***(b)*** *Close-up of tablet*

Figure 2.1: *"Architect with Plans", C. 2120 BC, Photo Credit: Musee du Louvre, Paris and British Museum, London*

The history of architectural visualization can be traced to the ancient Egyptians and Mesopotamians. One of the earliest examples of an architectural visualization, in particular a 2D floor plan, can be seen in the Headless Statue of Gudea (Figure 2.1), from Mesopotamia (present day Iraq) circa 2100 BC. In this statue, Gudea, dressed in princely attire, can be seen seated with an engraved tablet engraved laid on his knees. The tablet contains an architectural drawing which is believed to be a floor plan for a temple consecrated to Ningirsu, the great god of the state of Lagash. [30]. The plan on the tablet appears as an orthogonal projection and also contains a graduated ruler for reference.

Architectural drawings and renderings became more widespread and increasingly detailed in the second quarter of the thirteenth century, with significant developments made especially in the Gothic period. The discovery and formalization of the perspective projection by Filippo Brunelleschi, further increased the realism of architectural renderings during the renaissance.

The industrial revolution, the growth of the patent process and wartime engineering efforts transformed architectural and engineering drafting into a highly specialized and sought after skill (See examples in Figure 2.2). A traditional drafters toolset included a drawing board, pencils, pens, compasses, T-squares, protractors, triangles. A major advance in drafting tools occurred in 1901, with the invention of the Universal Drafting Machine (Figure 2.2a. This device combined the T-square, triangles, scales and protractor into a moving arm and enabled the drafter to create perpendicular lines at any orientation [121]. Despite advances in tools, the manual design process was fraught with opportunities for error, both in drawing and calculation. This set the stage perfectly for the advent of Computer Aided Design (CAD).

2.2 History of Computer-Aided Design[1]

Perhaps the most widely known and credited pre-cursor to modern Computer Aided Design, was Ivan Sutherland's Sketchpad [107], from 1963. Sketchpad belongs to a influential lineage of pioneering work in Human-Computer Interaction. Originally inspired by the Memex from "As We May Think" by Vannevar Bush [15], Sutherland's Sketchpad, in turn, went on to influence Douglas Engelbart's NLS (oN-Line System) [32], several early commercial CAD tools and, indirectly, the work proposed in this thesis.

Sketchpad ran on the Lincoln TX-2 and used a CRT display, a light pen, and a set of buttons, switches and dials. It was one of the first systems that allowed users to draw on a computer. A vocabulary of commands and interaction paradigms let users draw primitive shapes, copy/repeat previously drawn shapes (symbols), move parts of the drawing and

[1]This section summarizes the historical research contained in the book, The Engineering Design Revolution: The People, Companies and Computer Systems That Changed Forever the Practice of Engineering By David E. Weisberg. [121]

(a) (b) (c) (d) (e) (f)

Figure 2.2: Historic photographs of draftsmen and women from the 20th century.
Photo Credits: (a) History of CAD [121] (b) Eversource Electric Company, MA, USA (c) California Digital Library, Courtesy of UC Berkeley, Bancroft Library (d) Meisterklasse Factory, Carl Guderian, Flickr, 1951 CC 2.0 License (e) Donn B.A. Williams et al., City of Vancouver Archives (f) Ercole Marelli Offices, Photo by Paolo Monti, 1963.

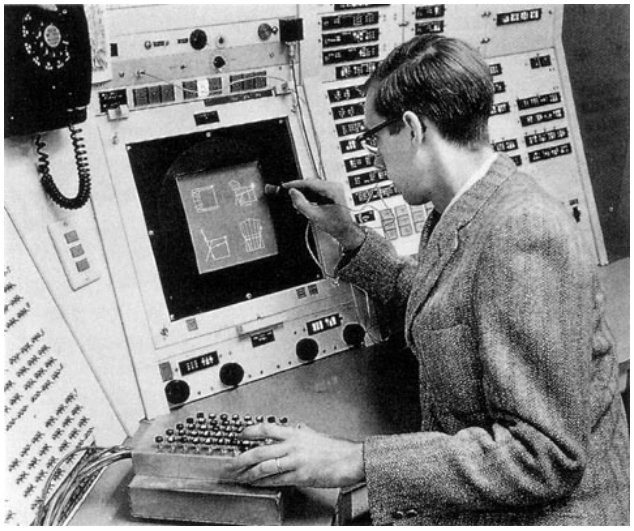


Figure 2.3: Ivan Sutherland using Sketchpad in 1962. Photo Credit: Ivan Sutherland, MIT Press and Computer History Museum.

zoom in/out of the drawing. The invention of this fast and accurate medium for graphical communication won Sutherland a Turing Award in 1988. [38].

Commercial research efforts such as Design Augmented by Computers (DAC-1) [122], by General Motors, appeared soon after. And by the 1970s, a number of companies were founded to commercialize CAD software. The invention of Non-uniform rational B-spline (NURBS) [74] allowed research into curve and surface modeling to move into 3D.

Today a number of successful commercial desktop CAD programs exist, such as AutoCAD [9], SolidWorks [108], Revit [10], FreeCAD [39], and Rhino [84]. While most of these programs are feature-rich and designed for use by professionals, programs like SketchUp offer a simplified and easier to use interface targeted towards more casual users. Tools such as Homestyler [45] and SweetHome3D [2] restrict themselves to the domain of indoor modeling, a topic of focus for this thesis.

2.3 Mobile Sensing and the Next Wave in CAD

Mobile devices that provide rich sensing capabilities are becoming increasingly ubiquitous. This thesis will explore the feasibility of indoor CAD modeling, using mobile devices as the primary medium for data acquisition and user interaction. The iPhone and iPad are two such devices that upended the mobile industry by integrating reliable and accurate touch screen technology, high quality cameras, GPS, and a variety of Micro Electro Mechanical Systems (MEMS) such as gyroscopes and magnetometers, into a compact and usable form factor. The first two systems described in this thesis are implemented on the iPhone and iPad respectively and utilize these sensors, along with the camera, in order to create 2D floor plans and 3D models of indoor scenes. We explore the capabilities of these devices in Chapters 3 and 4 of this thesis, along with a survey of parallel work in this space, such as MagicPlan [50].

This thesis also explores the use of augmented reality (AR) for indoor modeling. Augmented reality (or mixed reality) is a technology that superimposes digital content over a user's view of the real scene, while in-situ. This allows the user to more accurately capture relevant information from the scene and also visualize digital models in the context of the real world. AR has been used in the past for architectural and design applications, but only recently has it entered the mainstream, owing to the introduction of 3D aware devices such as Hololens [44] and Tango [81] (Figure 2.4) and software SDKs such as ARKit [6]. The third system, described in Chapter 5 of this thesis, is implemented on a 3D aware Tango device.

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. Tango [81] is one such effort from Google which takes the form factor of a phone. By looking at the phone screen, a user can see digital overlays over the live view of the scene, as seen from the device's camera, also referred to as indirect AR. Hololens [44] from Microsoft takes the approach of a head mounted device that overlays information directly in front of the user's eyes with see-through, stereoscopic, optical waveguide displays. Both devices use visual-inertial odometry (VIO), a combination of information from the device's motion sensing hardware with computer vision

(a) *Lenovo Phab2Pro with Tango* ***(b)*** *Microsoft Hololens*

Figure 2.4: *Emerging 3D-Aware AR Devices*

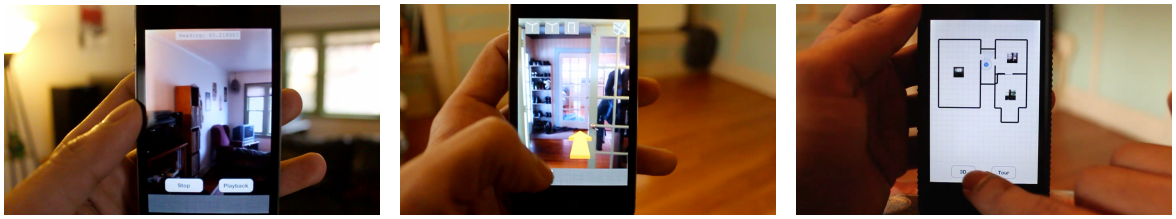
analysis of the scene visible to the device's specialized tracking cameras. By tracking the motion of fiducial features in the images, and fusion with motion sensing data, the device is able to localize its position and orientation in the world with high precision.

Recently announced AR software SDKs such as Apple's ARKit [6] have the potential to bring augmented reality to millions of existing mobile devices. By implementing a version of VIO on existing mobile hardware, ARKit eliminates the need for specialized cameras for tracking and mapping, such as those included in Hololens and Tango. While we do not explicitly explore ARKit or related software-based solutions in this thesis, the interaction techniques described herein can be directly ported to devices that support such AR SDKs.

Mobile augmented reality has the potential to provide the next wave in 3D modeling and scene reconstruction, enabling scaling to millions of casual and professional users that already own mobile devices. This thesis is one of the first attempts to explore indoor scene modeling techniques on the mobile platform. The following chapters (3, 4, and 5) describe three approaches to tackle this problem, along with a detailed survey of related research for each approach.

Chapter 3

CAPTURING INDOOR SCENES WITH SMARTPHONES



(a) *Capture data* ***(b)*** *Playback interactive tour* ***(c)*** *Generate indoor floor plan*

Figure 3.1: *Our system captures an interactive visual tour and a floor plan of an indoor scene, in real time, on a smartphone.*

In this chapter, I present a novel smartphone application designed to easily capture, visualize and reconstruct homes, offices and other indoor scenes. The application leverages data from smartphone sensors such as the camera, accelerometer, gyroscope and magnetometer to help model the indoor scene. The output of the system is two-fold; first, an interactive visual tour of the scene is generated in real time that allows the user to explore each room and transition between connected rooms. Second, with some basic interactive photogrammetric modeling, the system generates a 2D floor plan and accompanying 3D model of the scene, under a Manhattan-world assumption. The approach does not require any specialized equipment or training and is able to produce accurate floor plans.

This chapter describes work that was originally published in the Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST), 2012 [95].

3.1 Introduction

(a) *Movie Map Truck Rig* ***(b)*** *Aspen Tour Playback*

Figure 3.2: *A precursor to systems such as Google's Street View and the one described in this chapter, Aspen Movie Map was an early attempt to spatially index street imagery from the city of Aspen, CO.*

In 1979, Lippman et al. [66] used gyrostabilized cameras mounted on top of a car to create an interactive visualization of downtown Aspen. This early hypermedia system, a forerunner of Google's Street View [5], pioneered the use of spatially indexed imagery for generating interactive virtual tours (Figure 3.2). Their system allowed users to interactively explore an environment and presented the user with a sense of *'being there'*. The desire to enhance and improve this experience of virtual presence has subsequently fueled a sizable body of work around interactive visual tours. However, most of these approaches [115, 110, 5, 49] rely on specialized data acquisition equipment and have complex and time-consuming offline processing pipelines, making them inaccessible to the casual user.

I propose a system to easily create and view interactive visual tours, in real-time, entirely on a smartphone. To create a tour, the user first captures video of an indoor scene by following a few simple guidelines. This video data is then spatially and temporally indexed using sensor readings from the smartphone's gyroscope, accelerometer and magnetometer.

The proposed system then generates an image-based rendering of the scene that allows a viewer to interactively play back a virtual tour in which he/she may explore each room and 'walk' between connected rooms. Once the data capture is complete, I use an interactive photogrammetric modeling approach to recover room features such as the location of room corners and doors from within the image data. With this information, I estimate the position and orientation of the camera and compute a floor-plan representation of the environment. The method is designed to capture *Manhattan World* [26] scenes, i.e., scenes wherein all the surfaces are aligned with three dominant directions. The computed floor plans accurately capture the layout of the scene and are used to augment the interactive tour, enabling the user to understand where they are in the scene.

There are several challenges in making such a system work robustly, in real-time, on a mobile device. First, mobile phones lack accurate localization information or odometry (GPS does not work well indoors [132]), making it hard to estimate camera position. Second, the monocular video data lacks depth information about room dimensions or floor plan layout. Extracting this information from video data is a difficult task. Finally, working within the computational and memory constraints of an embedded device poses additional engineering challenges.

The main contributions of this work are a framework for capturing and spatially indexing visual data by using a combination of measurements from the gyroscope, accelerometer and magnetometer, and a novel interactive technique to estimate room geometry and floor plan layout from monocular video. At a system level, the novelty lies in the implementation of an end-to-end smartphone application capable of capturing and reconstructing indoor scenes. This application is the first of its kind to deliver an interactive tour experience coupled with a floor plan on-the-fly, entirely on a phone. I foresee the use of this system in areas such as real estate, interior design, indoor localization and mobile robot navigation.

In the remainder of this chapter I discuss related work, present an overview of the proposed system from a user's perspective, describe novel components of the system, present floor plan results and evaluate them against ground truth. I conclude with a discussion of this work.

3.2 Related Work

The idea of indoor interactive video-based tours has been explored by several authors in graphics, vision, and HCI. Brooks [14] in 1986 was one of the first to propose a system to build rapid visual prototypes of buildings for architectural use. More recently Uyttendaele et al. [115] used omnidirectional video to create indoor virtual tours. Similar approaches are used in Google's Streetview [5] and Art Project [49]. Our work is similar, in that the user can interactively look around an environment, move freely along predefined paths, and branch to different areas at decision points. These prior approaches required sophisticated omnidirectional camera rigs and several hours of offline processing, whereas our method is implemented end-to-end on a commodity smartphone and is able to generate tours in a few of minutes. Our video-based tour approach is similar to that of Quiksee [82] but differs in that they used a hand-held camcorder and an offline processing pipeline. Their method requires manual spatial registration (as camcorders lack gyroscopic sensors) and does not model the geometry of the scene. Quiksee has since been acquired by Google Inc. and no published information is available on their method.

Early mobile robot navigation systems such as those proposed by Ishiguro [51] and Yagi [128] utilized omnidirectional camera systems combined with odometry measurements to reconstruct environments for mobile robot navigation. Taylor [110] also estimated camera position and environment geometry from video data. Their approach required the user to specify several point-and-line correspondences in key-frames of the omnidirectional video, whereas our method requires users to mark each wall corner once in an interactive panorama of the room. The visual SLAM and computer vision communities have developed automatic approaches [36, 41, 103, 23, 109] to reconstruct indoor scenes from images. While computer vision-based 3D reconstruction has demonstrated potential, we avoid explicit reconstruction with computer vision as it tends to be brittle, computationally expensive and does not work well on texture-poor surfaces, e.g. painted walls, which dominate interiors. SLAM-based reconstruction has been shown to work on smartphones by Shin et al. [100] who used sensor

data to model indoor environments. Their approach is restricted to modeling only corridors while our method describes rooms, their volumes, and also presents an interactive visualization of the space. A Manhattan-world assumption was employed by Kim et al. [57], to acquire indoor floor plans in real-time. Their approach is hardware-intensive, requiring the user to carry a Kinect camera, projector, laptop and a special input device while capturing data around the house.

The recent shift of imaged-based systems to the mobile phone platform is exemplified by the mobile Photosynth application [25] that creates panoramic images in real-time on a smartphone. MagicPlan [50] is a commercial floor plan generation app available for the iPhone. By marking floor corners in the room via an augmented reality interface, their system is able to estimate dimensions of the room and generate a corresponding floor plan. While their algorithms are proprietary and unpublished, they likely estimate and use the height of the observer (in this case the phone camera) and the tilt of the camera to estimate scene depth. By estimating depth in this manner, MagicPlan is able to capture non-Manhattan world scenes. This approach is susceptible to error when floor corners are occluded by furniture, requiring the user to guess at their positions. Our approach instead uses wall edges (the lines formed where two walls meet), which typically run from floor to ceiling and are often visible in rooms regardless of furniture placement and other occlusions. MagicPlan reconstructs rooms individually and then has a user manually assemble them to form a complete floor plan, whereas in our approach, we solve for correspondences between rooms and assemble the entire floor plan automatically. Finally, the augmented reality approach employed by MagicPlan is quicker and allows users to mark floor corners and doors in a single step. In contrast our two-step approach requires users to first capture imagery of the room and thereafter mark room features over the captured imagery. However, MagicPlan focuses solely on floor plan generation, whereas our goal is to produce a sense of virtual presence, to which end we use the captured imagery to visually reconstruct an immersive tour of the scene. A comparison of floor plans generated by both the approaches is presented in the results section.

3.3 Application Overview

This section describes the function of our application at a user-level. Our application is implemented on an iPhone 4, running iOS 5.1. However, the technique is designed to work on any device that is equipped with a camera, gyroscope, accelerometer and magnetometer; features that are now standard in the latest smartphones and tablets. We encourage readers to view the accompanying supplementary video [92] in order to gain a better understanding of the application's interface. The basic steps in the approach are outlined below and in Figures 3.1, 3.3 & 3.4:

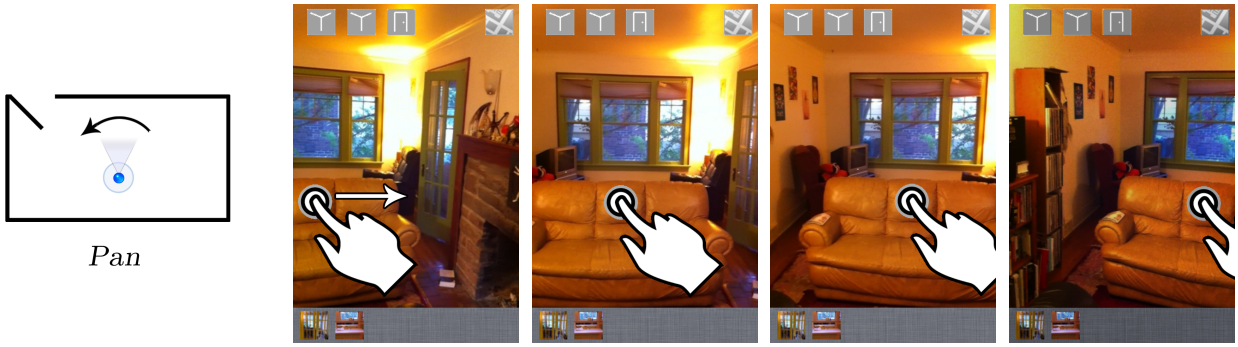


Figure 3.3: *The user may pan left or right in the panorama by dragging*

Figure 3.4: *By tapping on the motion arrow, the user may move between rooms*

1. **Data acquisition**: (Figure 3.1a) To capture a room, the user stands near the center of the room, holding the phone upright and aiming the phone camera at a wall. He/she then proceeds to rotate 360° to capture a video of the entire room. The application automatically detects when a full rotation has been completed and prompts the user to turn and face the next room. Once in position, the user indicates that he/she is ready to move to the next room by performing a *swipe up* gesture and subsequently proceeds to walk to the next room. Our system records this transition as a video. On reaching the center of the next room, the user performs a *swipe down* gesture and begins to capture the current room. The same process is repeated for every room. If a room is re-visited, the user has an option to indicate this in the interface. Captured rooms appear as thumbnails on the bottom of the screen.

2. **Interactive Tour Playback**: (Figure 3.1b) Once capture is complete, the user can instantly view a virtual tour of the scene. For each room, the system generates a 360° video panorama, in which the user may pan left or right to explore the scene (Figure 3.3). The user is afforded two modalities to interact with the panorama. They may drag left or right on the touchscreen or physically move the device in 3D space to '*look around*' the room, in a manner reminiscent of mobile virtual/augmented reality applications [83, 112]. When a door appears in the users current field of view, a motion arrow [42] appears on the screen, indicating that there is a path leading to an adjacent room. Upon tapping the arrow, a transition video is played back giving the user the effect of 'walking' into the next room (Figure 3.4). At the end of the transition, the system presents the 360° video panorama of the current room, wherein the user may once again explore interactively.

3. **Embedding Close-up Images and Text**: The user has the ability to embed close-up photographs and text in the scene, during playback. To add an embedded object, the user double taps a point of interest on the screen. This instantiates a modal dialog that prompts the user to take a high-resolution photograph of the point of interest

and optionally add some descriptive text. Upon completing this task and closing the dialog, the user is returned to the tour and the embedded object now appears as an icon overlaid on top of the panorama. The icon is positioned at the location where the user originally performed the double tap gesture and is pinned to that location as the user pans the panorama left or right. Tapping the icon spawns a modal popup that shows the hi-res photo along with the annotation text. The embedded objects are saved along with the virtual tour data for future viewing.

4. **Marking Room Features**: In playback mode, the user can mark room features such as wall edges (where two walls meet) and doors via an intuitive touch interface. To mark a wall edge the user first brings the edge into view by panning left or right in the panorama. Once the edge is visible, the user drags a marker to align with the room corner in the image. Similarly, doors are indicated by placing two markers on the extremities of the door. Note that each door is visible from both the rooms that it connects. This door correspondence is also specified by the user, (i.e., which room does a specific door lead to), by tapping the appropriate room thumbnail.

5. **2D Floor Plan**: (Figure 3.6c) An optimization algorithm uses the corner and door information and other room constraints to generate a 2D floor plan of the space. Alignment errors in the reconstructed floor plan can be manually corrected via a touch interface. To fix an alignment error, the user first taps a wall that requires re-alignment and next taps a wall to which the previously selected wall should be aligned. The walls can be in the same room or different rooms, but they must be parallel. The floor plan is rendered as a top-down map of the environment and indicates the users current position in the tour.

6. **3D Floor Plan**: (Figure 3.6d) We extrude the 2D floor plan vertically and render a 3D model of the scene. The 3D rendering can be manipulated via the touch interface and users can explore novel views of the environment (e.g., a 45° bird's-eye view).

3.4 System Components

3.4.1 Spatial Video Indexing

Indexing video spatially with camera pose information is a simple yet effective method of creating an interactive viewing experience [66]. During playback, by interactively selecting a desired viewpoint and viewing direction, we can retrieve the closest view from the previously captured video sequence. Interactive video playback has advantages over a traditional composited panorama in that it captures dynamic exposure changes and motion in the environment, which are essential to the richness and realism of the visualization [115]. However, the disadvantage of using a video sequence is the inability to zoom-out. While the idea of spatial video indexing has been explored before [110, 115], we present a few novel developments that help adapt it to a smartphone.

Calculating Camera Pose:

The on-board gyroscope on the iPhone 4 provides fine-grained information about the angular velocity of the device; however integrating this data to calculate relative orientation results in drift and requires calibration. The accelerometer and magnetometer conversely provide more reliable orientation information, but at a lower resolution and refresh rate. We use a simple sensor fusion approach to calculate pose by combining data from the gyroscope, accelerometer and magnetometer as follows. The rotation rate of the device around three axes is obtained by querying the gyroscope at regular intervals. The major axis of rotation is determined by querying the accelerometer to determine the direction of gravitational acceleration and hence the device orientation (portrait v/s landscape). Once the major axis is known, we bootstrap the camera pose with the actual orientation of the device relative to true north, as obtained from the magnetometer. Thereafter current orientation is determined as follows:

$$\theta_t^G = \theta_{t-1}^G + \Delta t \times \omega^{majoraxis} \tag{3.1}$$

Where θ_t^G represents current device orientation, as calculated by adding the delta change in orientation $\Delta t \times \omega^{majoraxis}$ to the last known orientation θ_{t-1}^G. This value is updated at approximately 100Hz, but suffers from gyroscopic drift of as much as $\pm 10^\circ$ in a full 360° rotation. We then spatially index the current image with the calculated device orientation value.

Countering Gyroscopic Drift:

In order to counter gyroscopic drift, we rely on the magnetometer to inform us when a full 360° rotation has been completed. While the intermediate magnetometer readings are noisy and not useful for updating device orientation, we have found that the resolution is sufficient to indicate when a rotation has been completed. Once we know that the rotation is complete, we can close the "gap" caused due to drift. For this we use the initial and current gyroscopically calculated orientation values and eliminate the drift by uniformly distributing it across the entire rotation.

3.4.2 User Interaction to Mark Room Elements

Once the video and orientation data have been captured, the tour is played back interactively. The playback interface is similar to previous work such as Street View [5] and VideoPlus [110] which allow the user to interactively pan within a 'viewing bubble' and branch to different bubbles at decision points.

During playback, we also present an interface to mark room elements such as corners and doors. In this interface the user is presented with a draggable marker which they can place over the current image to indicate the presence of a corner or door. Marking a corner requires only one marker, whereas marking the extremities of a door requires two. The relative heading of each marker is calculated from the indicated x-coordinate of the marker within an image, by the following equation:

$$\theta_{marker} = \theta_t^G + fov/w_{img} \times (P_{marker} - w_{img}/2) \tag{3.2}$$

The spatial index, θ_t^G, of the image on-screen corresponds to the heading of the center of the image. The one-dimensional pixel-offset of the marker from the center of the image of width w_{img} is calculated by subtracting the position of the center pixel, $w_{img}/2$, from the position of the marker, P_{marker}. The pixel-offset is then converted into a heading-offset by multiplying by the field of view of the image, fov, and dividing by the total pixel width, w_{img}. Finally, the estimated heading of the marker, θ_{marker}, is determined by adding the heading offset to the current spatial index, θ_t^G.

During tour playback, the following equation (3.2) is used to calculate position, $P_{feature}$, of a feature with a known heading $\theta_{feature}$ within an image.

$$P_{feature} = w_{img}/2 + w_{img}/fov \times (\theta_{feature} - \theta_t^G) \tag{3.3}$$

This allows us to embed UI elements such as motion arrows and annotations into the scene.

3.4.3 Floor Plan Generation

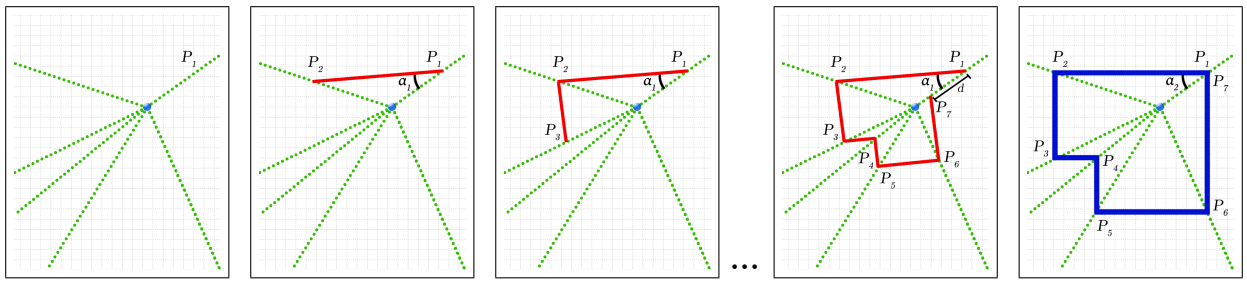


Figure 3.5: *Floor plan reconstruction algorithm: (a) Top-down view of the scene showing the corner rays (dotted green) emanating from the camera. P_1 is unit distance from camera along first ray. (b) Hypothesized wall at angle α_1 intersects second ray at P_2. (c) The next wall is rotated 90 degrees, and intersects at P_3. (d) Incorrect hypothesis results in a distance error d between first and last points. (e) Optimal wall configuration with starting wall angle α_2, minimizes d.*

Computing Room Geometry:

We treat the marked corners and doors as rays emanating from the camera center and reconstruct the room geometry that fits the given set of rays (as shown in Figure 3.5), based on the following assumptions. First, we assume that the scene satisfies the Manhattan world assumption, i.e., the walls are planar and the room corners are right-angled. Many architectural scenes approximately fit the Manhattan World assumption, and it has previously enabled very high quality reconstruction results across a wide range of scenes [41]. We also assume that the camera is located in a position within the room from which all corners are visible, or if this is not possible, that the user is able to estimate and mark the location of occluded room features. Finally, we assume that the corner ray directions are determined from the spatial indexing data, as described in the previous section.

Search Algorithm:

Our search algorithm over the possible room configurations is described below and also summarized in Algorithm 1.

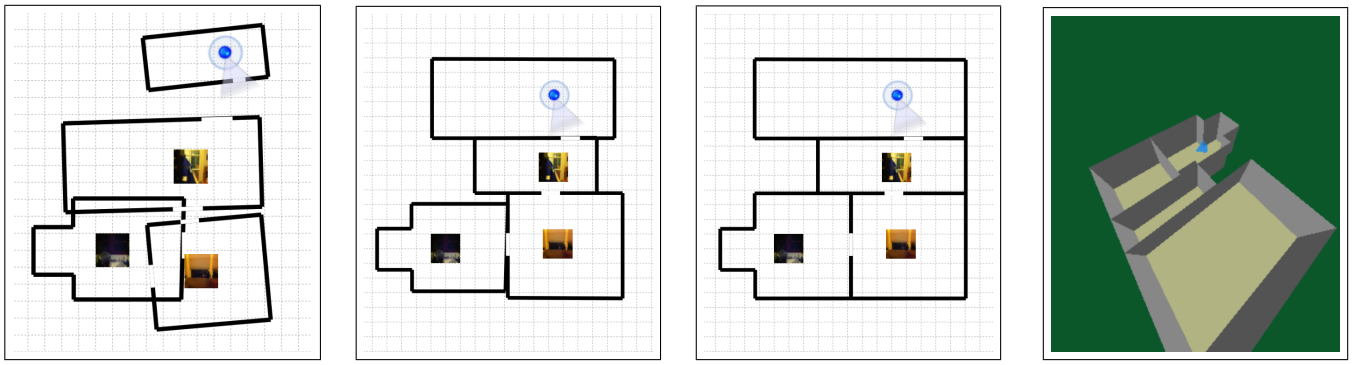
Without loss of generality, assume the camera is located at the scene origin, and the first room corner is at unit distance from the origin (we solve for the room only up to scale). Now suppose we also specify the orientation α (relative to the first corner ray) of the first wall. Under the Manhattan World Assumption, the orientations of all other walls are determined (since they meet at right angles). It also turns out that their lengths can also be inferred from the known corner ray directions (as we will describe shortly). The approach works as follows. Let P_1 be the point at unit distance from the origin along the ray direction of the first corner (Figure 3.5a). The intersection of the ray from P_1 at angle α with the ray from the origin along the ray direction of the second corner defines the position P_2 of the second corner in the room (Figure 3.5b). Rotate 90 degrees and repeat this procedure to intersect the ray from P_2 with the ray from the origin along the direction of the third corner to define P_3(Figure 3.5c), and so on. This procedure will place all of the corners in the plane. The key

Algorithm 1 Calculating optimal wall configuration

P_0 = Origin
P_1 = Unit distance from P_0 along the first ray
N = Number of corners
$minDistance = MAX\_FLOAT$
for $\alpha = 0 \rightarrow 360$ step 0.5 **do**
    for $i = 2 \rightarrow N$ **do**
        θ_i = direction of i^{th} corner ▷ from (3.2)
        $P_i \leftarrow intersect(P_{i-1},\ \alpha + (i-1) \times 90,\ P_0,\ \theta_i)$
    end for
    $distance = getDistance(P_1, P_N)$
    if $distance < minDistance$ **then**
        $minDistance \leftarrow distance$
        $minAngle \leftarrow angle$
    end if
end for
Return $minAngle$

insight is that for the correct value of α, the final ray from P_N should intersect P_1 (Figure 3.5e), assuming N corners. An incorrect value of α will generally not satisfy this criterion (Figure 3.5d). Our approach is therefore to do an exhaustive search over all angles (0 - 360, sampled at 0.5 degree increments), to solve for an α that yields the closest intersection with P_1 (see Algorithm 1). Our algorithm finds an optimal solution in $O(Nd)$ running time where N is the number of corners in the room and d is the number of angles sampled in our search.

Note that in Algorithm 1, $intersect(P_A,\ \theta_A,\ P_B,\ \theta_B)$ finds the intersection point of two lines A and B defined in point-slope form and $getDistance(P_A, P_B)$ calculates the distance between points P_A and P_B in a euclidean space.



(a) *Floor layout as estimated by odometry, with current camera position indicated by the camera frumstum*

(b) *Solving for the door constraint translates, scales, and rotates rooms to align properly*

(c) *Layout after manual edits have been made via a touch interface*

(d) *Reconstructed 3D model of home.*

Figure 3.6: *Stages of solving floor layout.*

Aligning Rooms:

Since our goal is to generate a complete floor plan of an indoor scene, we must calculate relative positions and orientations of rooms to each other. We make an initial best guess with the help of some approximate odometry. If we assume that the user walks at a constant pace between rooms during capture, we can estimate relative positions of rooms, using the user's movement direction and length of the transition video sequence. However this method is prone to error (as is seen in Figure 3.6a) due to the fact that users may move with a non-uniform velocity and change direction mid-flight.

We can more accurately calculate relative room positioning by using the supplied door rays. Once the walls have been calculated via Algorithm 1, we can find the projected co-ordinates of the door by intersecting the door rays with the walls. Since each door is visible from exactly two rooms (as indicated in the interface by the user), we can proceed to calculate a

2D rigid body transformation, to align them. The transform is written as:

$$\begin{pmatrix} x_i' \\ y_i' \end{pmatrix} = \begin{pmatrix} c & -s \\ s & c \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{3.4}$$

Here $c,\ s,\ t_x$ & t_y are unknowns; by substituting the values of corresponding doors in two rooms, we can solve for the transformation that correctly aligns the doors, and hence also aligns both rooms. The matrix equation can be solved by standard least squares fitting methods.

We solve for the transformations between every pair of connected rooms and are able to generate a complete floor plan as shown in Figure 3.6b. This intermediate solution still contains errors, such as misaligned walls and incorrectly scaled room dimensions. The errors are caused due to several factors; gyroscopic drift not being eliminated completely, human error in marking room features (typically off by a few pixels), error due to discretization of the search angle α and the fact that the physical walls may not be perfectly axis aligned. There may also be error caused due to parallax as the user may slightly move the camera center while capturing the scene.

To alleviate errors, we allow users to manually correct wall alignment using a simple touch interface, as described in the Application Overview section. The end result of this process is depicted in Figure 3.6c, which more accurately describes the room layout in the house. In this particular example, five corrections were required to produce the final result.

3.4.4 3D Rendering

3D reconstructions of a scene allow for generating novel views and a more immersive spatial visualization. Once we have a suitable 2D floor plan rendering, we extrude it vertically and render a floor plane to obtain a representative 3D model of the environment as shown in Figure 3.6d. To provide the user with additional spatial context, we render a small camera frustum that represents their current position and orientation in the tour.

3.5 Results and Discussion

We tested our application in four indoor environments. For each environment we generated an interactive visual tour and floor plan. The resulting tour experience has been described previously in section 3.3 and a demonstration of the interface is available in the supplementary video [92]. In this section we will present the results of floor plan generation.

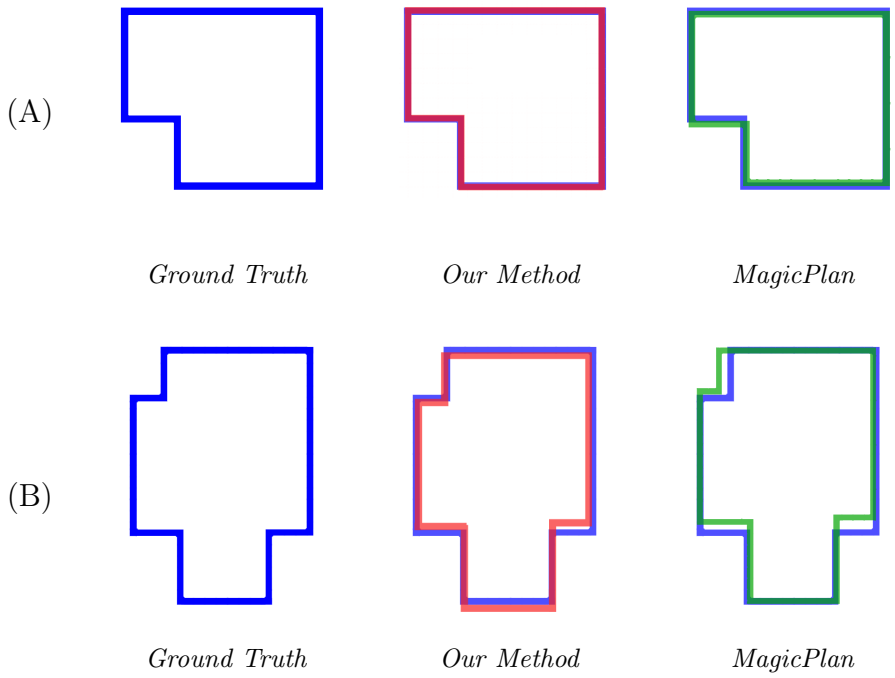


Figure 3.7: *Room geometry recovered by our approach (red) and by MagicPlan (green), when overlaid on ground truth (blue)*

3.5.1 Recovering Individual Room Geometry

In Figure 3.7 we present the results of individual room reconstruction by our approach and that of MagicPlan. Since our floor plan reconstruction is not to scale, for the overlay we manually scale and register our result over ground truth. Room 1 took 1 minute and 12

seconds to capture with our approach (44s capture + 28s marking room features) and 53 seconds to capture with MagicPlan. Room 2 took 1 minute and 34 seconds with our approach (46s capture + 48s marking room features) and 1 minute and 4 seconds with MagicPlan. The running time for both reconstruction algorithms was negligible. Our image capture time remains roughly the same regardless of room configuration, but more complex rooms require longer to mark room features. While the total time taken by MagicPlan is less, our approach also captures imagery of the room for creating an immersive visual tour experience, whereas MagicPlan outputs only a floor plan. The examples in Figure 3.7 show that our approach is able to capture the shape of the room with similar accuracy to MagicPlan. Note, however that our reconstruction does not provide metric scale, while MagicPlan does.

3.5.2 Generating Floor Plan Layout

In Table 3.1 and Figure 3.8 we present results from four indoor environments:

		Features			No. of	Total Time	Error
Env.	*Type*	R	C	D	Frames	Taken	(%)
A	House	5	26	4	1356	7m16s	5.66
B	House	6	42	5	2107	9m42s	13.38
C	House	5	26	5	1571	8m40s	8.76
D	Office	7	28	6	3422	12m35s	13.98
Avg.						9m33s	10.45

Table 3.1: *Numerical results for four environments tested. (Legend: R = Rooms, C = Corners, D = Doorways)*

The reconstructed floor plans for each environment are shown in Figure 3.8, along with manually measured ground truth. Some reconstruction errors can be observed in environments (A) and (D), where an individual room has been misaligned. In environment (C) our system approximates a bay window to be flat, in accordance with the Manhattan-world assumption.

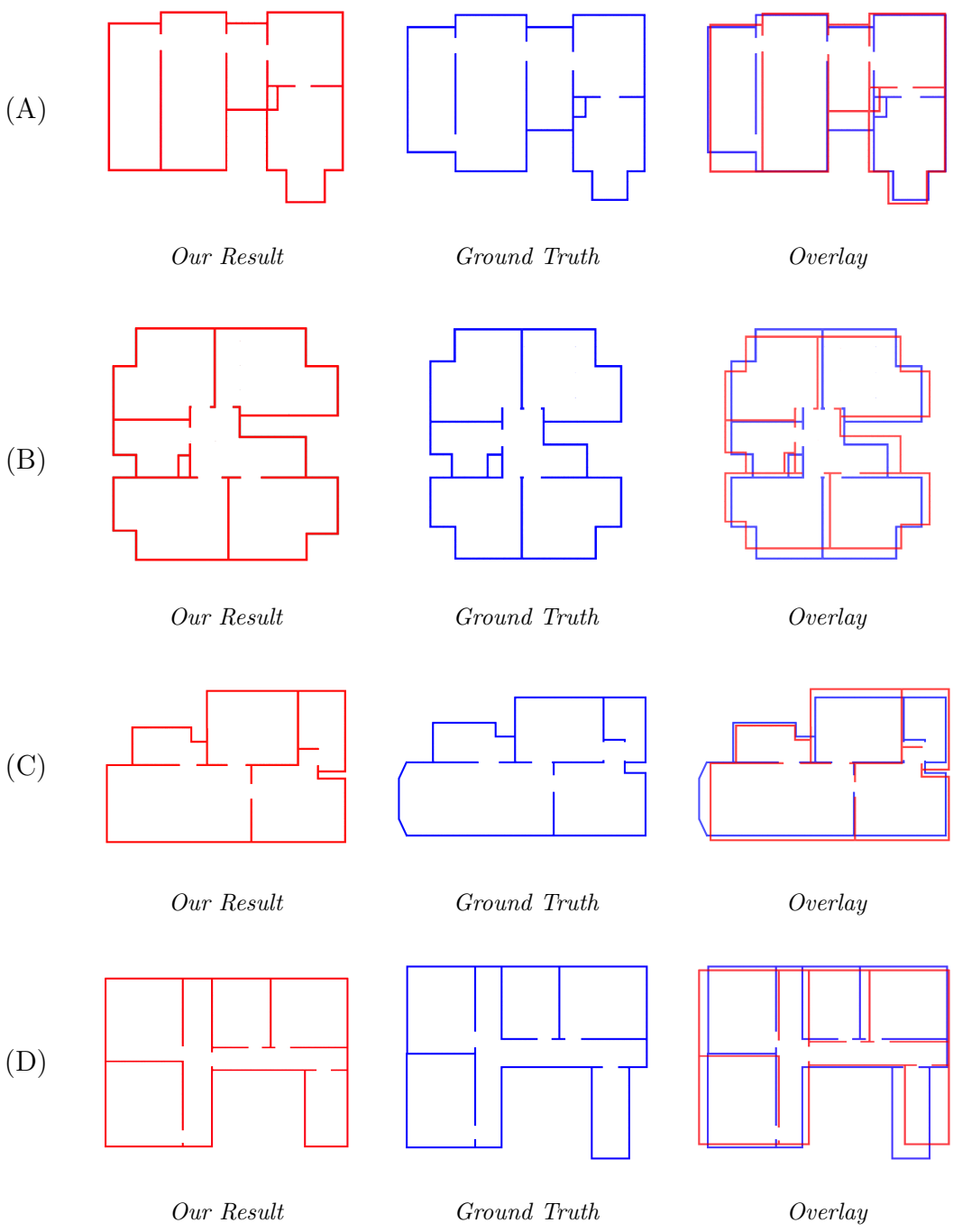


Figure 3.8: *Side-by-side comparisons of floor plans generated by our system to ground truth*

We calculate a measure of reconstruction error from the overlay, as the ratio of area incorrectly reconstructed (sum of both overestimated and underestimated areas) to the total ground truth area. The results indicate that our system is able to reconstruct the dimensions and overall floor plan of an indoor scene with an average error of 10.45% for the environments tested. Aside from the previously mentioned failure cases, the floor plan correctly captures the shape, relative position and orientation of most rooms. This meets our goal of capturing an accurate layout that is useful for the purposes of visualization and navigation in the interactive tour.

3.6 Discussion

We presented a novel end-to-end system to capture and reconstruct indoor scenes for the purpose of creating an interactive visual tour. While there is a large body of prior work in this area, our system is the first to allow casual users to quickly and easily create immersive tours on a smartphone. Our method also enables the creation of indoor floor plans and 3D renderings of the scene, without the need to remove furniture or physically measure the environment. In order to achieve our design goals, we have devised a data capture framework, an interactive photogrammetric modeling approach, and an indoor floor plan generation algorithm, designed to run independently on a commodity smartphone in real-time.

While the floor plan generation algorithm is currently unable to model curved walls or non right-angled corners, we believe that the system is sufficient to capture a qualitative sense of most indoor environments.

Several aspects of this system may be improved upon in future work. A hand-held device is susceptible to shake and the resulting imagery is sometimes motion blurred. The gyroscope data from the non-major axes can be used to select frames with minimal shake, thus stabilizing the image data. Another improvement would be to better model complex indoor scenes that contain non-Manhattan world elements. One approach would be to use computer vision techniques to estimate the depth of room features to assist the floor plan generation algorithm. The creation of more detailed 3D renderings of the scene can be enabled by adding textures

to the walls and modeling furniture and other items within a room. Finally, to enhance the effect of immersion, it should be possible to capture aspects of physical interaction in the scene, for example operating appliances or turning on/off light switches. These can be captured with our video-based approach but require explicit modeling to incorporate them into the interactive tour.

We believe that real estate is perhaps the most compelling application for our system. Realtors and homeowners can use our smartphone application to generate tours and floor plans of houses without the need for any extra equipment. Mapping is another compelling application area. Commercial mapping services such as StreetView [5] and Bing Maps [67] are increasingly incorporating indoor maps into their products. Small-business owners or other operators can use our system to create immersive tours of their establishments (cafés, restaurants, hotels, etc.) and share them with customers through such mapping services. Aspects of our approach may also be useful for other applications, in particular, the floor plans can be used to bootstrap systems for indoor localization (e.g., in a mall) and mobile robot navigation. Finally, the interactive tours give a sense of virtual presence, allowing individual users to share their experience of a space with friends and family.

With almost 80% [16] of U.S. mobile subscribers now owning smartphones and similar rising trends across the world, we believe that our method will enable casual users to capture, visualize, and reconstruct a wide variety of interesting indoor scenes, from homes and offices to museums and heritage sites.

Chapter 4

IN-SITU CAD CAPTURE

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

Figure 4.1: *Our system captures a CAD-like 3D model of an indoor scene, in a few minutes, on a commodity tablet.*

In this chapter, I present an interactive system to capture CAD-like 3D models of indoor scenes, on a tablet device. To overcome sensory and computational limitations of the mobile platform, I 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. The system recognizes the sketches and adds corresponding 3D models into the scene at the appropriate locations. 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.

This chapter describes work that was originally published in the Proceedings of the International Conference on Human-computer Interaction with Mobile Devices (MobileHCI), 2016 [96].

4.1 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.

In Chapter 3, I described how to capture the shape and layout of rooms. However, populating these models with furniture, doors, windows, and artwork is still a challenging task. A common approach is to manually augment the room models using CAD software, typically 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 chapter, I 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. The 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. The 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.

The 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. Note, that the 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 chapter, I discuss related work, describe the proposed system, present results and conclude with the description of two potential applications of the system.

4.2 Related Work

CAD: A variety of efforts have attempted to simplify the process of computer aided design of scenes and objects. Trimble SketchUp [102] (formerly Google SketchUp) is one such commercial product that simplifies the 3D modeling process by providing a limited set of controls. AutoDesk Homestyler [45] and IKEA Home Planner [78] 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 [133, 48, 126] have demonstrated the value of quick, stylus-driven interactions to create 3D models.

Several others [101, 127, 31, 40] 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. [62] 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 built upon the idea of smartphone scene capture explored in Chapter 3. Other related approaches such as [73, 59, 117], 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. [56], 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. [106] have used marker-based AR to visualize virtual furniture in a scene, but don't address capture. SemanticPaint [116] uses additional depth hardware and in-situ interaction to segment and label fused point clouds.

Multi-modal Modeling UI: Novotny et al. [71] 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. [112] 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 [47], a free and open source speech recognition system for mobile devices.

Furniture Model Alignment: Recent work [65, 8, 46] 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 [103, 41] 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 Chapter 3 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 [69, 99, 89], overcome the issue of feature matching by utilizing depth information provided by a Kinect camera. While this approach has demonstrated exciting results, the method described in this chapter does not rely on external hardware and can work on widely available commodity hardware. However, in Chapter 5 I have also explored in-situ modeling solutions on emerging depth sensing platforms such as Google's Project Tango [81].

Once a editable 3D model has been constructed, it can enable exciting applications such as furniture replacement in images [55] and automatic rearrangement [68].

4.3 User's view of the system

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

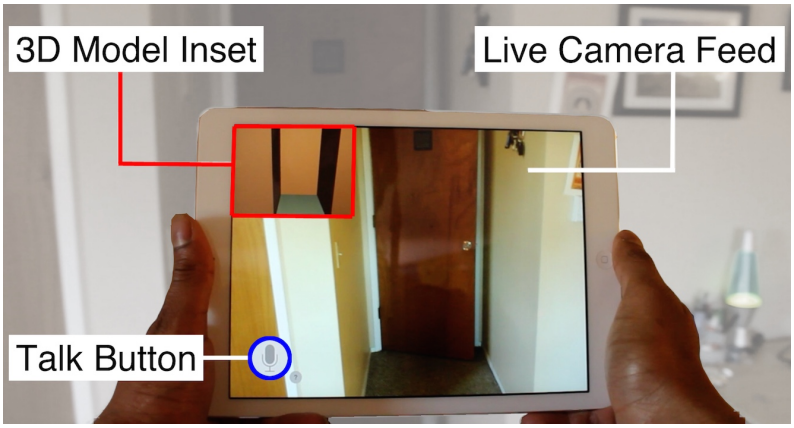


Figure 4.2: *Our mixed-reality modeling interface.*

commodity tablet equipped with a camera and inertial sensors. We strongly encourage readers to view the accompanying supplementary video [93] 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 4.2). 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 under 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 the user to visualize their space, rearrange furniture and preview the results instantly. The user 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.

4.4 System Components

4.4.1 Empty Room Reconstruction

We first reconstruct the shape and scale of the room from a panning video. Building upon the method described in Chapter 3, 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. 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, relative image movements of objects at distances of 1.2m and 2.4m from the camera differs by about 200 pixels ($\sim$15%). If we did not account for this parallax, it would cause misalignment of the

3D model that would adversely affect furniture placement.

Figure 4.3: *Top-down view of the simplified camera motion model, based on the arc radius r and orientation θ.*

As illustrated in Figure 4.3, we assume a circular camera trajectory and known camera height based on how users capture panoramas with mobile devices. The camera height is set manually, per user, as it is required to determine ground plane distance accurately. 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 [13] to determine the intrinsic parameters for the camera. This step is only 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 (4.1) to (4.4).

$$Camera = Calibration \times Projection \times Extrinsics \tag{4.1}$$

$$M = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = K \begin{bmatrix} I & 0 \end{bmatrix} T \tag{4.2}$$

$$M = \begin{bmatrix} \alpha & s & u_0 \\ & \beta & v_0 \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{4.3}$$

In this case:

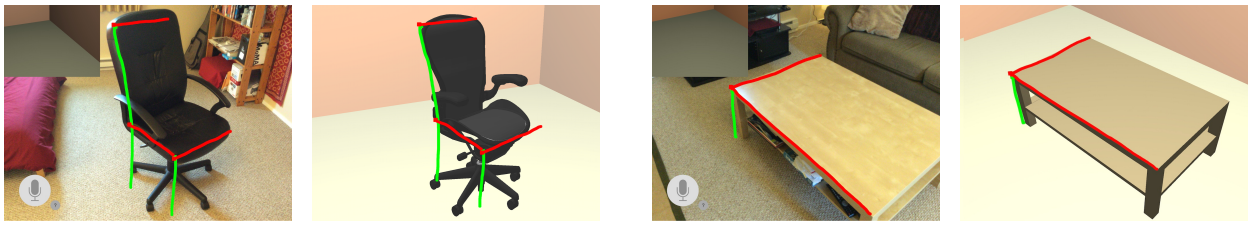
$$R = \begin{bmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad t = \begin{bmatrix} 0 \\ 0 \\ -r \end{bmatrix} \tag{4.4}$$

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 [103]. 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.



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

Figure 4.4: *Sketch based retrieval results. Figures 4.4a & 4.4c show user sketches. 4.4b & 4.4d show corresponding 3D model placements.*

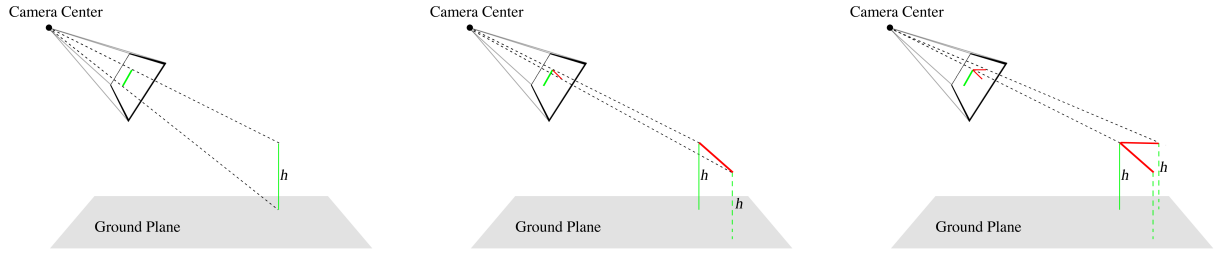
4.4.2 Furniture Placement Interface

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 4.2, 4.4) 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 system described in Chapter 5, we 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 4.4. 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.



(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 4.5: *Illustrating the 2D to 3D conversion of the user's sketch*

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

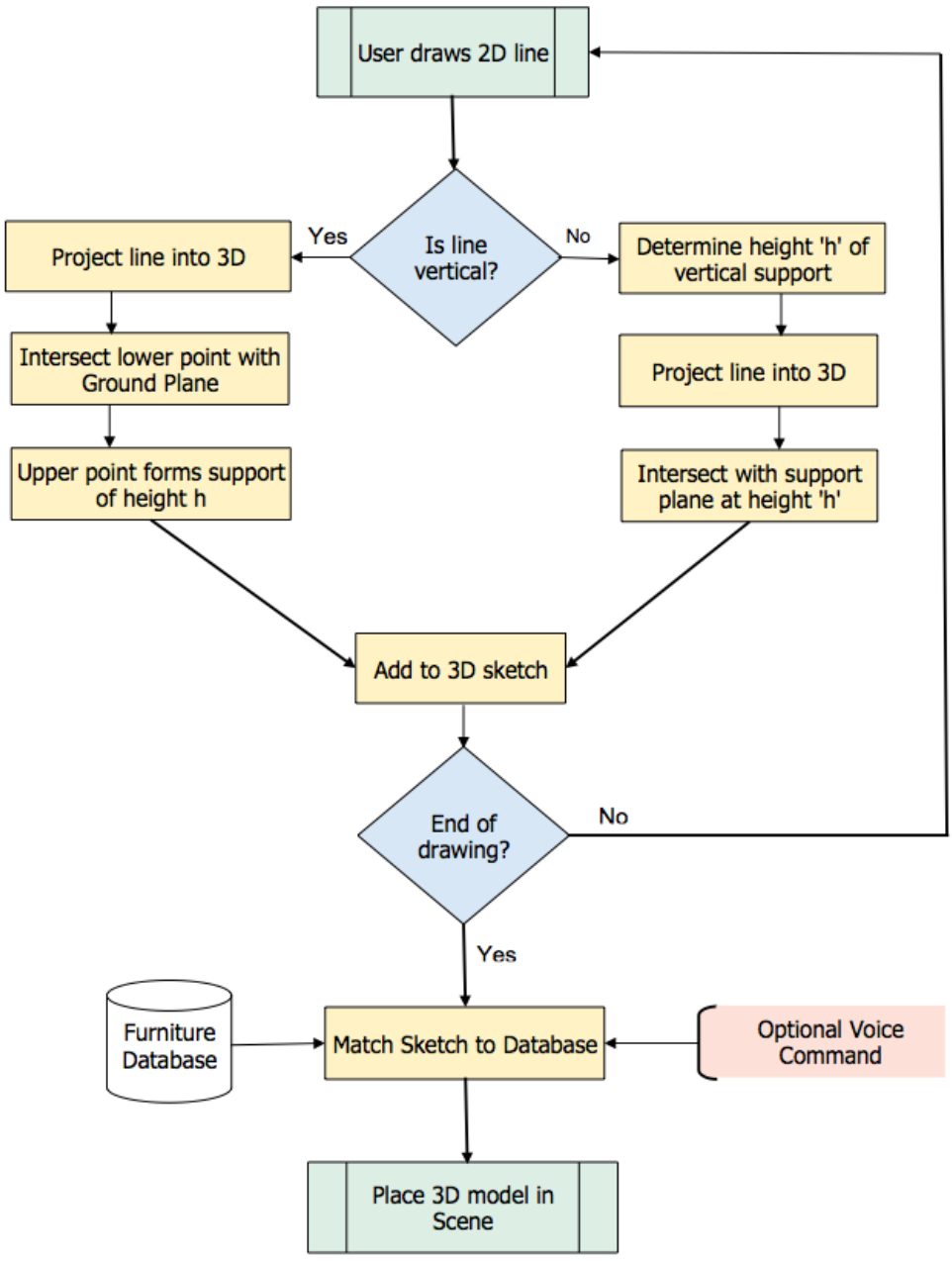


Figure 4.6: *Algorithm to convert a 2D user sketch in to a plausible 3D object sketch.*

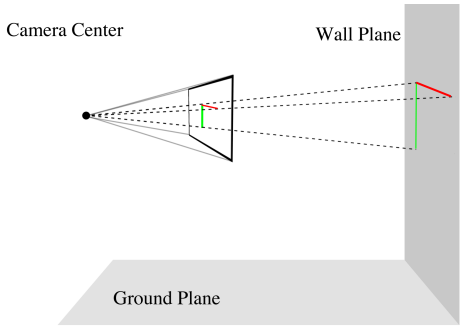


Figure 4.7: *Example of an artwork sketch projected onto a wall.*

(Figure 4.8), we can efficiently perform search and retrieval over this compact representation, based on a partial input 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 4.4a). For objects that are very thin or are placed on a wall (like artwork, windows etc.), two lines describing 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 4.6. 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 4.5a.

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 4.5a and 4.5b. 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 4.7. 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

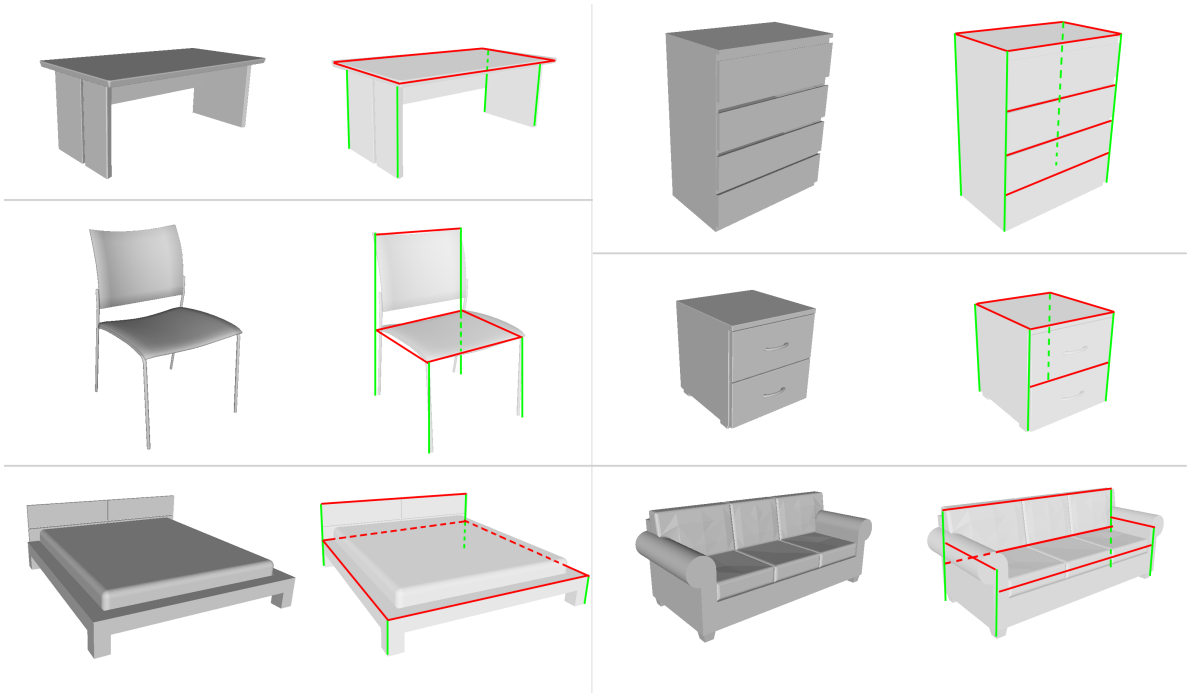


Figure 4.8: *Furniture models in our database with corresponding line representations.*

commonly occurring furniture pieces, that have been manually preprocessed to be represented as set of 3D lines, as shown in Figure 4.8. 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 [102, 114] 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 (4.5). We use an iterative search algorithm,

described next, to consider all plausible transformations of the 3D model to fit the partial sketch.

$$M = \begin{pmatrix} s_x \,.\, cos\theta & s_y \,.\, sin\theta & 0 & t_x \\ s_x \,.\, -sin\theta & s_y \,.\, cos\theta & 0 & t_y \\ 0 & 0 & s_z & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.5}$$

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 [33] 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.

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 4.1: *Summary of voice commands and their functions*

Voice Commands

The user can optionally issue voice commands at any point by tapping the listen button (See Figure 4.2). The system detects silence at the end of speech and processes the preceding command according to Table 4.1. 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.

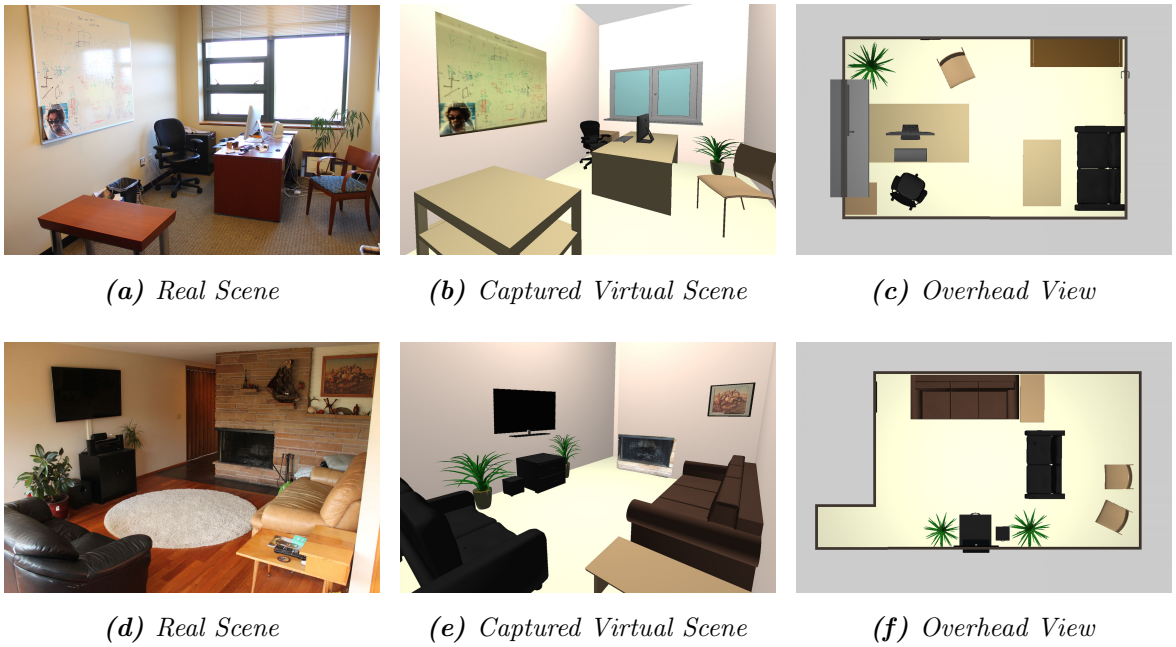
We developed a taxonomy for additional commands that may be issued in order to add, replace or modify furniture as listed in Table 4.1. The "Add" command appends a pre-aligned 3D model on the previously placed item, for instance the user may add a "iMac" on any desk or add "Curtains/Blinds" to windows. 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 [47] library for voice command recognition. Testing the performance of the voice recognition library was outside the scope of this work.

4.4.3 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.

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*.



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

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

Figure 4.9: *Visual results from Office 1 and Living Room scenes*

4.5 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.

Figures 4.1, 4.9 and 4.10 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.

			Autodesk Homestyler		
Scene	Area	# Items	Measurement Time	Modeling Time	Total Time
Bedroom 1	28.0m^2	23	18m 12s	11m 19s	29m 31s
Living Room	26.1m^2	12	9m 14s	8m 57s	18m 11s
Office 1	13.9m^2	14	8m 45s	8m 33s	17m 18s
Bedroom 2	21.1m^2	13	10m 01s	8m 34s	18m 35s
Office 2	13.8m^2	12	8m 12s	9m 09s	17m 21s
Bathroom	15.5m^2	9	6m 20s	4m 54s	11m 14s
Avg.			10m 17s	8m 34s	18m 41s

Table 4.2: *Quantitative results for the environments tested using Homestyler.*

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 of our approach to Autodesk Homestyler [45] (an ex situ indoor scene modeling tool), since it produces the most similar end results. Tables 4.2 and 4.3 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

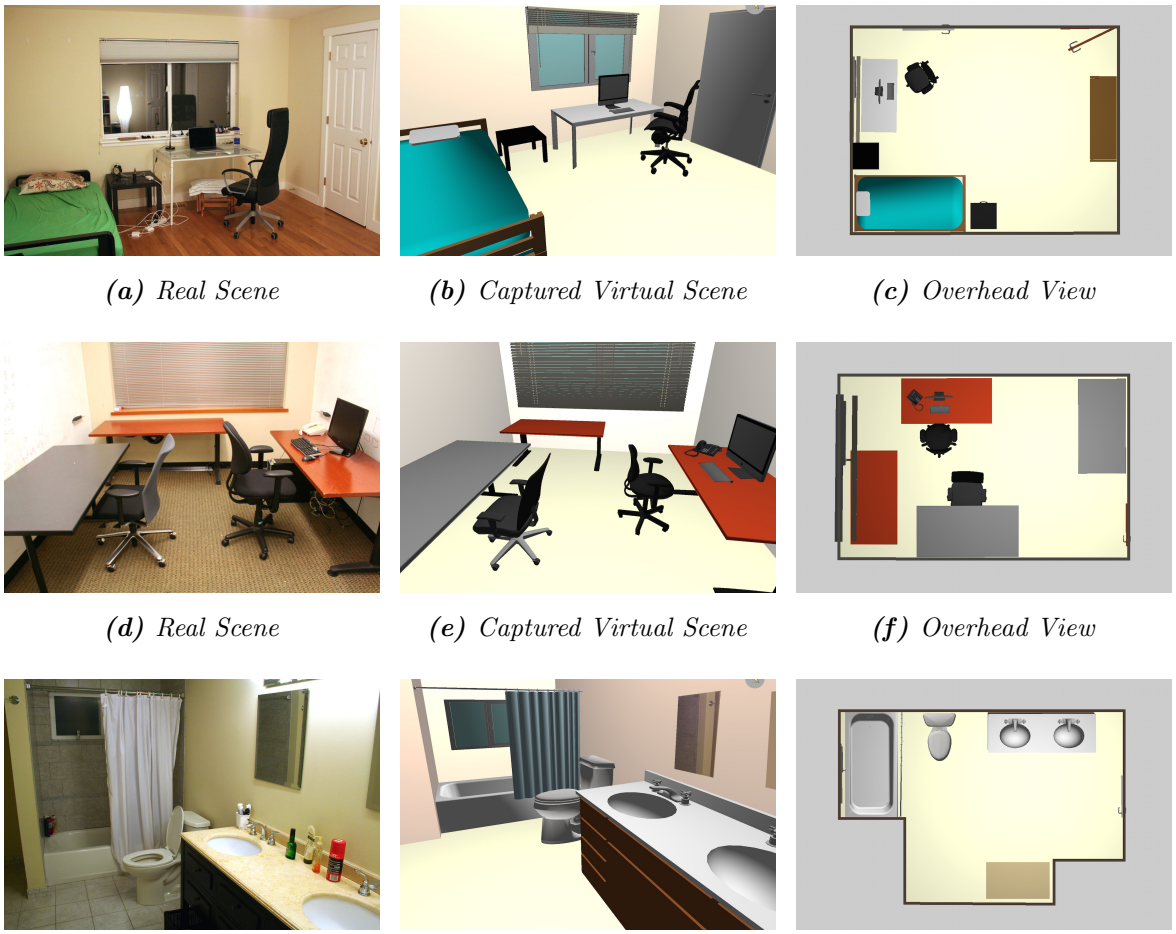
			Our System	
Scene	Area	# Items	Total Time	Furniture Error
Bedroom 1	28.0m^2	23	7m 12s	42.57cm
Living Room	26.1m^2	12	6m 02s	37.19cm
Office 1	13.9m^2	14	6m 32s	33.77cm
Bedroom 2	21.1m^2	13	5m 58s	40.03cm
Office 2	13.8m^2	12	6m 29s	36.61cm
Bathroom	15.5m^2	9	5m 06s	29.11cm
Avg.			6m 13s	36.55cm

Table 4.3: *Quantitative results for the environments tested using the proposed method.*

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 4.3 for error measurements)



(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 4.10: *Visual results from Bedroom 2, Office 2 and Bathroom scenes*

4.6 Discussion

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 [17] 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, with automated methods like

dominant plane extraction [61]. 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 [101] 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 Figure 4.9 is added by sketching a bounding box. Similarly, 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.

The tracking of the system described in this chapter currently relies on the device gyroscope, which can suffer from drift. In Chapter 5, I explore the possibilities of using more robust tracking with visual features, that enables a more 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 [68] and furniture removal/replacement in the original images [55].

Chapter 5

INTERACTIVE ROOM CAPTURE ON 3D-AWARE MOBILE DEVICES

(a) *Real Scene*

(b) *User interacting with our system (inset)*

(c) *Generated Model*

Figure 5.1: *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.*

In Chapters 3 and 4, I have explored the processes of in-situ scene capture on widely available commodity hardware such as the iPhone and iPad. These devices typically do not have accurate indoor tracking mechanisms or depth sensors and therefore are limited in the fidelity in the models they can produce. In particular, Chapter 4 explores in-situ 3D CAD modeling with an iPad. One of the major limitations of that system was that the user was unable to move freely within the scene, limiting the extent of furniture that could be modeled (both due to distance and occlusions). However, this limitation can be mitigated with the use of an emerging class of mobile devices known as 3D-aware mobile devices. 3D-aware

---

This chapter describes work that was originally published in the Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST), 2017 [91].

device manufacturers have begun to integrate depth sensors and specialized cameras to enable accurate indoor tracking, localization and depth estimation. In this chapter, I explore the possibilities of using such a 3D-aware device to create in-situ CAD models of indoor scenes. Studies conducted indicated that having free movement and additional sensor data further simplifies the process of modeling and improves its generalizability.

Traditional ex-situ 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, the proposed 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 faster, more accurate, and requires less effort than traditional desktop tools.

5.1 Introduction

The advent of '3D aware' mobile devices (phones, tablets, headsets, etc.) such as Google Tango [81], Microsoft Hololens [44], and Meta2 [1] opens up interesting new opportunities to map and understand our world. I 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 [120, 17] 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 [45, 2, 77] 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. Chapter 4 explored a novel user interface for in-situ CAD modeling on an iPad, which was a first step in mitigating the drawbacks of traditional desktop CAD modeling workflows.

In this chapter, I present another follow-up 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, I 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 this chapter 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 the system, I 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, I present results from a within-subjects user study that compares the usability of this 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.

5.2 Related Work

3D Aware Devices and Applications: Tango-enabled [81] smartphones like Lenovo Phab 2 Pro [80] and AsusZenFone AR [7], and head-mounted devices such as HoloLens [44] 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 system is the first to leverage Tango for CAD-based furniture recognition and placement.

MagicPlan [50] 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 [58] and AR Home Designer [28] are spatially aware apps for placing new virtual items into a scene. They do not, however, capture existing objects in the room. SnapToReality [72] 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 [102, 2] and web-based tools [45, 77, 79, 3, 78] 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 [52] demonstrate interactive plane based 3D modeling on images but require that all objects be approximated by planes. Others [73, 59, 60] enable building simple polyhedral shapes using a sketching interface. Others have explored free form in-situ sketching of virtual [130] 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. The system described in Chapter 4 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. [99] 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 [17], 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 [96] (65 objects), and [99] (145 objects). Finally, we note that neither [96] nor [99] have been user tested and compared to existing commercial systems, one of the contributions of this chapter.

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. [64] in order to select matching furniture models from ShapeNet. A variety of other

techniques [65, 8, 46] 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 [103, 41, 119] and depth-based [69, 19, 29, 97] methods exist for "scanning in" mesh models of objects and scenes. Acquiring 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. [89] 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 [116] uses interaction to segment and label fused point clouds, but their models are not as clean and easy to edit as CAD models.

5.3 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 [45, 2, 77], 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

Figure 5.2: *A sample of chair 3D models, indicating the variety of furniture models available in the ShapeNet [17] database.*

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 [17] database (Figure 5.2). By aligning the dominant horizontal surfaces of the model and the real object (Figure 5.3), we can therefore place the model at

(a) *Dominant horizontal surface of table in scene* ***(b)*** *Corresponding 3D model surface*

Figure 5.3: *The dominant horizontal surfaces of a real table and corresponding 3D model visualized with a green plane.*

the appropriate scale and location within the scene.

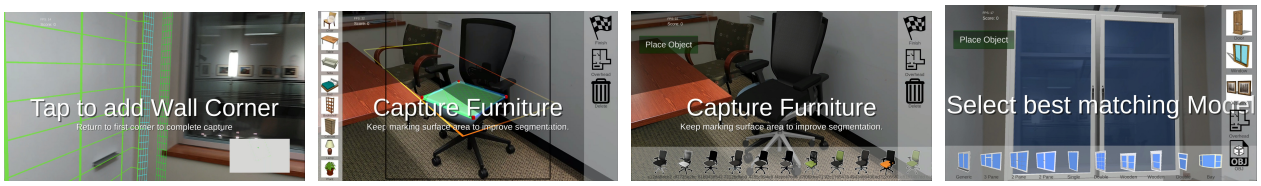
Third, we assume that the room interior is bounded by a connected set of flat, vertical walls (i.e., there is no opening other than the doors).

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.

5.4 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 [94]. Our system is implemented on a Tango-enabled [81] Lenovo Phab 2 Pro, but the



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

Figure 5.4: *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 5.4a shows a top down view of the walls as they are captured. The thin square frame in Fig 5.4b provides a guide for image capture. The thumbnails in Figs 5.4c & 5.4d show alternate model choices.*

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 5.4a) 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 shape and the captured walls are rendered in a solid color, to indicate that wall scanning is complete.

2) Furniture Capture: (Figures 5.4b and 5.4c) 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 5.4d) 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

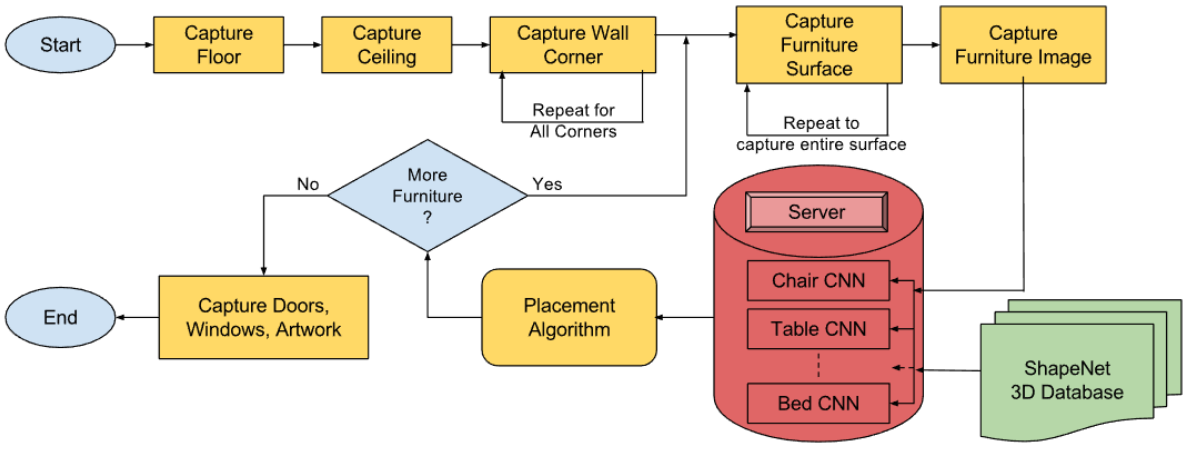


Figure 5.5: *System overview and workflow. Elements in yellow are implemented as part of our system and run entirely on the mobile device hardware. Elements in red and green are components from related work that reside on the server.*

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 5.1c and 5.13, column 2), or C) they can view a top-down rendering of the scene to reveal the floor plan and 2D furniture placement (Figure 5.13, column 3).

5.5 System Components

Our application is built on the Tango platform [81] 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.

5.5.1 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 [33] 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 [96, 95], of soliciting user input to mark wall edges. However, unlike [96, 95], 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 5.4a) 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.

5.5.2 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 [98] 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.

5.5.3 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 [64]. The Networks are identical in architecture but are individually trained on separate classes of objects from the ShapeNet database [17], 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.

5.5.4 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 (Figure 5.6) 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 5.7a). 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

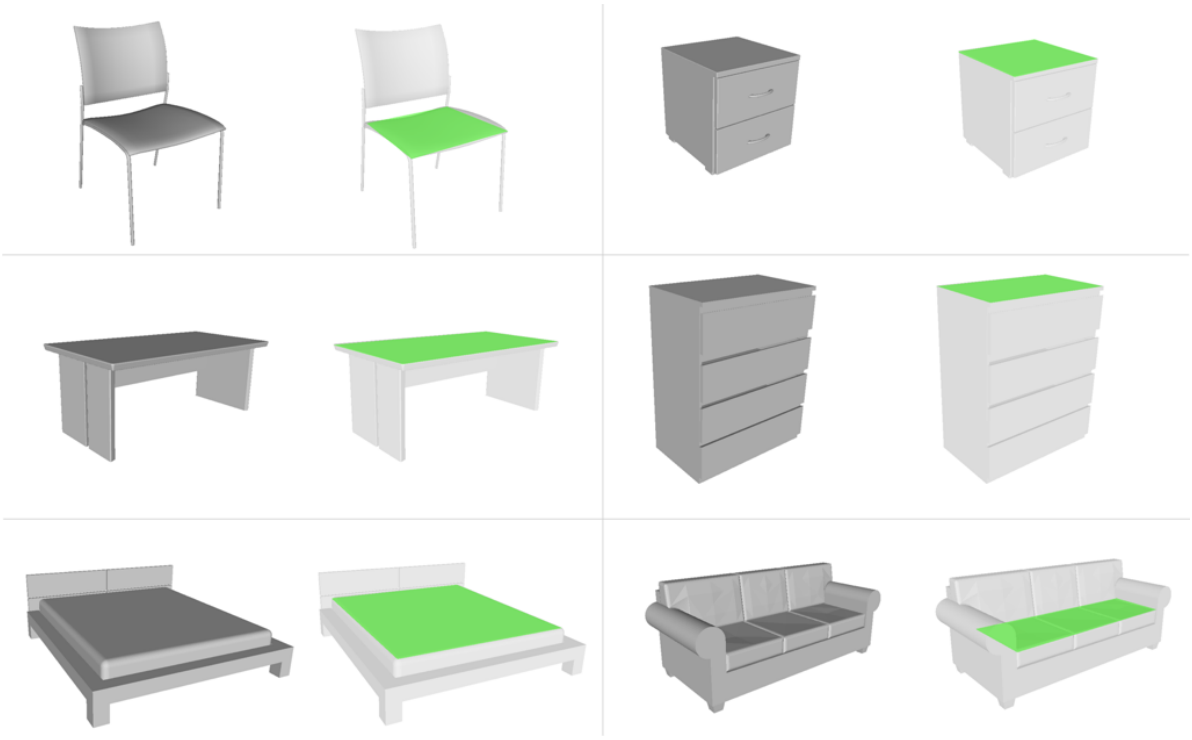
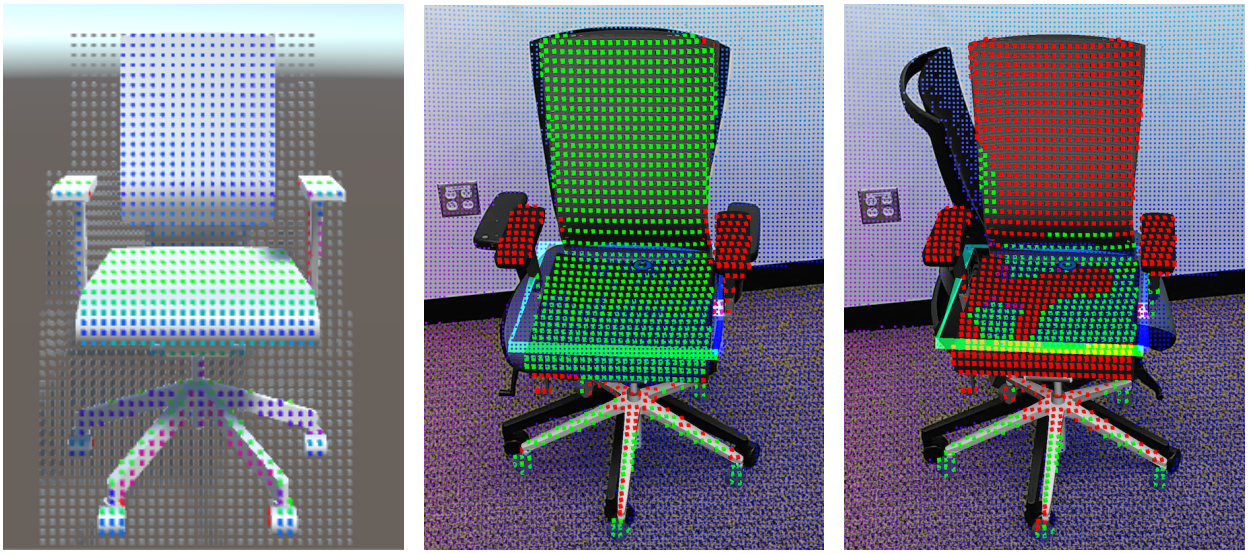


Figure 5.6: *The dominant horizontal surfaces of various furniture CAD models, visualized with a green plane.*

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*.

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



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

Figure 5.7: *Visualization of CAD model sampling (5.7a), along with a good alignment to the depth map (5.7b) and a poor alignment to the depth map (5.7c). Inliers are in light green and outliers in dark red.*

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 5.7b & 5.7c. 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 [34]. 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.

5.5.5 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 [88].

5.6 User Study

To evaluate our system we compare it to Autodesk Homestyler [45] an existing, commercially available, web-based CAD modeling tool, specifically designed for modeling indoor scenes. We picked Homestyler since it is a full-featured modeling tool that is popular amongst casual interior designers. It contains 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. However, 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 the 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 are all graduate students recruited from Mechanical Engineering or Computer Science departments at the University. All of them were familiar with desktop computers and touch-based mobile devices. When asked how much experience they have had with 3D model creation tools (such as Sketchup, Blender, AutoCAD) they rated themselves as 7 novices, and 3 intermediates. 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.

Study Design: We conducted a 10x2 within-subjects study, where each user was tasked with creating a CAD model of a room, once each with our tool and Homestyler. In order to eliminate any carryover effects caused by practice or fatigue, we randomly assigned half the users our method first. We first conducted a short pilot study with two users (results not reported), in order to eliminate hidden issues with the user interface and to verify the study

(a) *Study Room* ***(b)*** *Generated Model*

Figure 5.8: *Study Room Setup. The room contained 4 distinct chairs, 1 desk, 1 window and 1 door.*

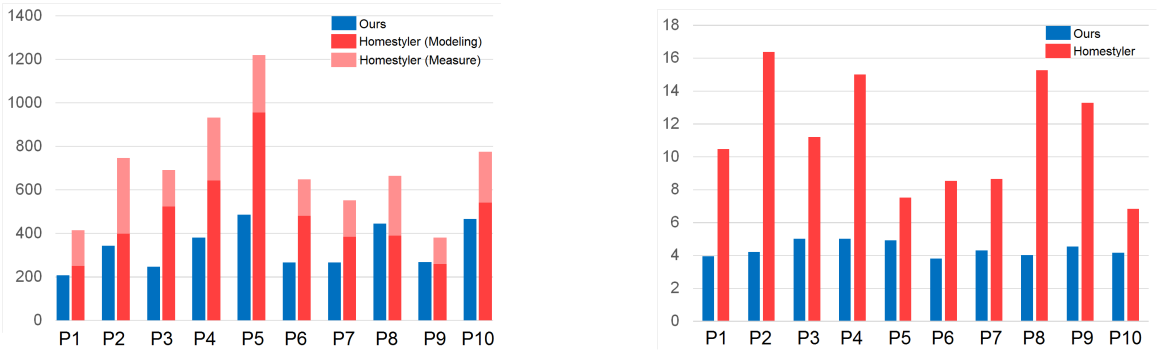
methods, time taken etc.

Task Setup: The task consisted of modeling an office scene consisting of one room of dimensions 2.9m x 4.7m. The room, shown in Figure 5.8, contained four distinct chairs, one desk, one large window and a door. Users were instructed to capture all of these elements in their models. They were to first capture the walls, then the furniture (in any order) and finally the window and door.

Study Procedure: The study began with a short introduction to the task. The users were then given a demonstration of the first assigned system, by the researcher, who performed the task of wall capture, and demonstrated the capture of one chair and table. The users were then given time to familiarize themselves with the system and repeat the demonstration tasks. This was repeated for the second assigned system.

The users were then instructed to start over and capture a full 3D model of the room with each system. They were encouraged to think out loud and ask the researcher questions if they were stuck. During the assessment the users 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 sub-tasks of floor/ceiling/wall



(a) *Total Modeling Time (in seconds)*

(b) *Metric Chamfer Distance (in cm)*

Figure 5.9: *Quantitative Comparisons: 5.9a shows total time taken by users for the modeling task with either tool. Time for Homestyler is further broken down into measurement and modeling time. 5.9b shows furniture placement error our system and Homestyler when compared to ground truth based and average symmetric Chamfer distance.*

capture, furniture capture and door/window capture, and the resulting 3D model was saved. In the case of the desktop modeling method, users were asked to record the dimensions of the walls, doorway and window, using a standard tape measure (ignoring height in all cases), so that they may enter it manually into the modeling tool. The rationale behind asking them to do this is that if a home user were to create such a model on the desktop, they would have to measure walls to ensure accuracy. 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, the user was asked a series of qualitative questions that compared our system to Homestyler.

5.7 Results

5.7.1 Task Time

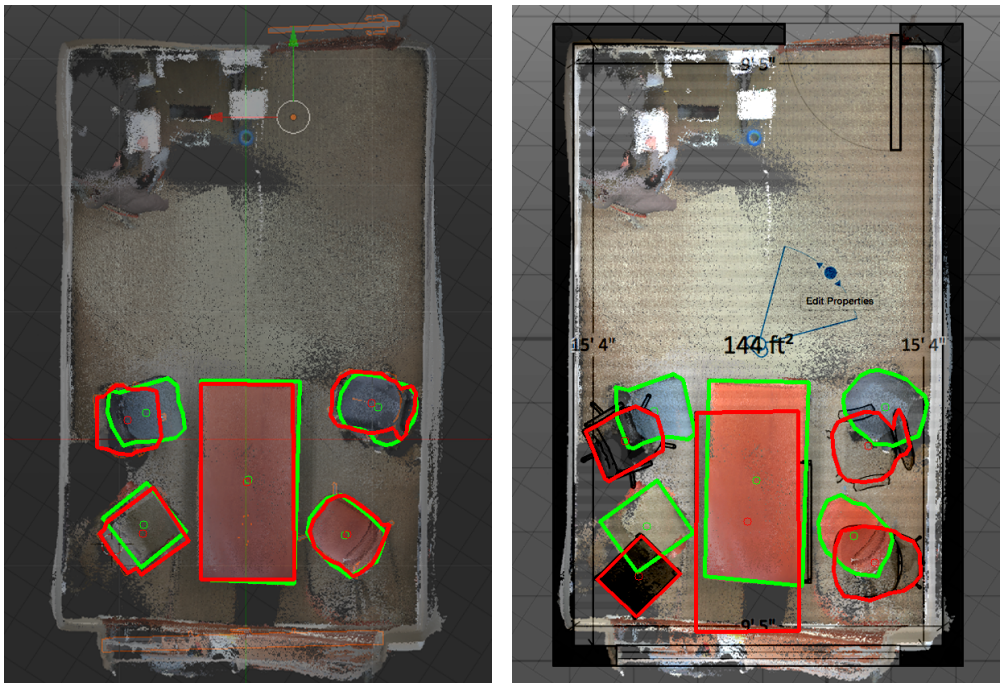
Figure 5.9 shows total time taken by each user to model the room using our system v/s Homestyler. The total time for Homestyler includes the time taken to manually measure the scene, as it is an integral step to create an accurate CAD model on the desktop. In contrast our system does not require the user to make any physical measurements, instead relying on scene measurements from the device.

We conducted paired-samples t-tests to compare metrics from both systems. Results indicate that in all participants, our system took significantly less time ($t(9) = 6.34$, $p < 0.001$) to complete the task than Homestyler for creating an equivalent model. On average with our system users took (M=336.9, SD = 100.9s) seconds, to complete the task, compared to (M=702s, SD=245.34s) with Homestyler. On average our system was 49.8% faster than Homestyler with a 95% confidence interval of a 41-58% speedup. If we do not wish to consider measurement time, our system was still faster ($t(9) = 2.88$, $p = 0.02$) than Homestyler for 8 out of 10 users. If we only considered furniture modeling time (excluding measurement, walls, windows, doors), our system was 53 seconds quicker on average, however the improvement was less significant ($t(9) = 1.49$, $p = 0.1701$) in this case.

5.7.2 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 5.10.

We then compare manually annotated contours around corresponding furniture items. For each furniture item, we compute Jaccard Index 5.1, metric centroid distance (in cm) and Chamfer Distance. Chamfer distance (5.2), which is defined as the sum of closest point



(a) *Our System Overlay* ***(b)*** *Homestyler Overlay*

Figure 5.10: *Results from P5 using our system 5.10a and Homestyler 5.10b 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.*

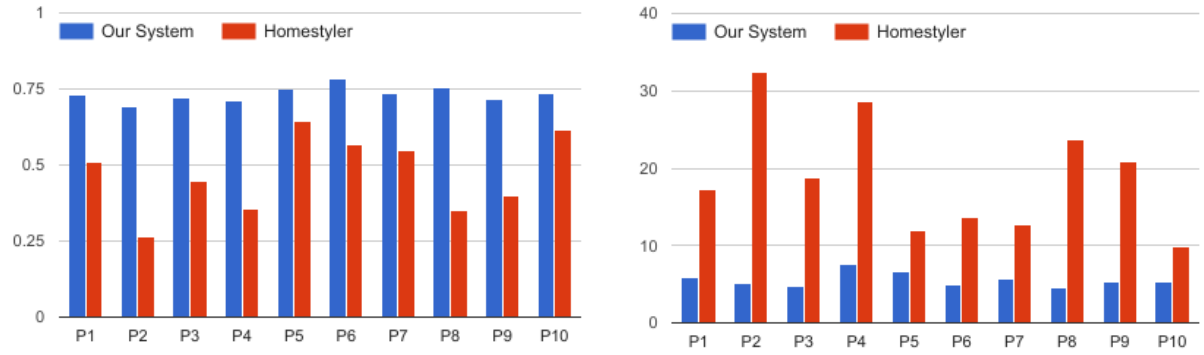
distances between the contours in the model and ground truth. This was symmaterized by (5.3). We adopt the symmetric Chamfer distance as our primary error metric, since it accounts for difference in contour shapes as well as the displacement of their centroids. However, errors calculated with other metrics are tabulated in Figure 5.11.

$$JaccardIndex = (A \cap B)/(A \cup B) \tag{5.1}$$

$$Ch(A, B) = \Sigma_{a \in A} min_{b \in B} \|a - b\| \tag{5.2}$$

$$SymmetricCh(A, B) = Ch(A, B) + Ch(B, A) \tag{5.3}$$

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



(a) *Jaccard Index (Overlap ratio)* ***(b)*** *Metric Centroid Distance (cm)*

Figure 5.11: *Other Furniture placement error measures of our system and Homestyler when compared to ground truth based on Jaccard Index (5.11a), average metric centroid distance (5.11b)*

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.

5.7.3 Qualitative Feedback

After completing both tasks, we ask 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. These axes were inspired by NASA Task Load Index [111]. For each factor, we ask the users if our system was 'Much Lower', 'Slightly Lower', 'Same as Homestyler', 'Slightly Higher', or 'Much Higher'. Responses are shown in Figure 5.12 and indicate that our system is generally less demanding to use than Homestyler. It is interesting to note that users *qualitatively perceived* our system as being quicker and less error (position) prone than Homestyler, an insight that is in agreement with our quantitative analysis.

We also asked participants several open-ended questions. The questions, along with

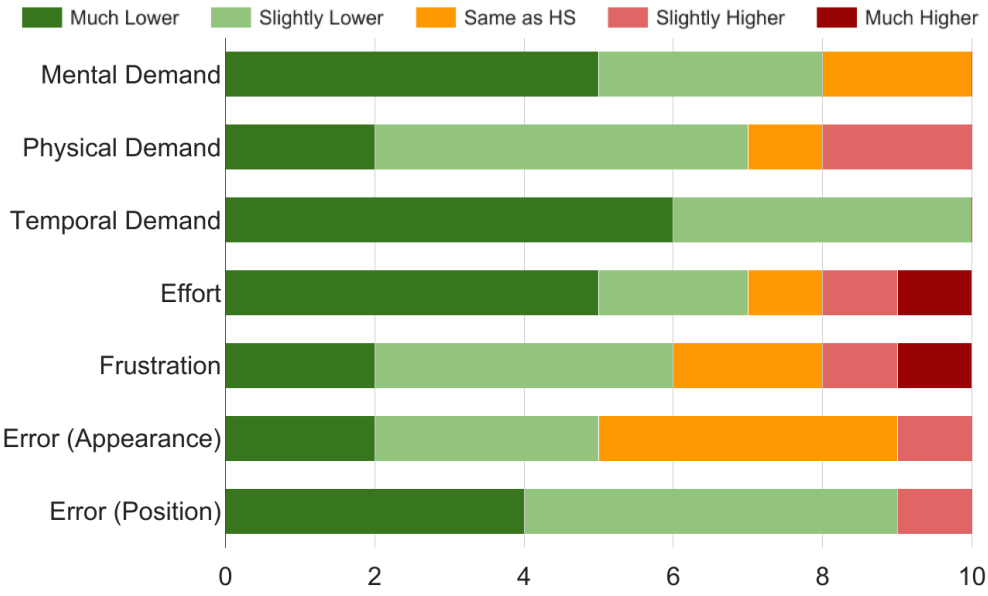


Figure 5.12: *Qualitative feedback. (Best viewed in color)*

summaries and some representative quotes are below:

"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 a much smoother process and I was more confident in the result", "AR was much easier overall, being able to quickly preview models in-place was a huge benefit", "AR. Finding matching furniture was easier and no measurement was required".* 2 users preferred the desktop system stating *"More control that the output was correct"* and *"capturing surfaces on the AR system was frustrating".* 1 user preferred a hybrid of the two systems, capturing data with AR and editing on the desktop.

"How confident are you in the assessment of Q1?" All users were confident in their response to Q1. 8 users stated that further practice would not change affect their opinion. 2 said maybe.

"What did you like or found easy about the AR system?" *"Basically no thinking*

required. Easier to find matching furniture and no frustration with sizes",“Very intuitive", “does not take much time".

“What did you not like, found difficult or frustrating about the AR system?" *“Capturing large surfaces like a big desk that takes multiple scans was frustrating/inconvenient", “Holding the Tango for several minutes can be tiring", “reverting changes was difficult", “stepping back to take pictures was difficult", “unsatisfying to see small misalignments"*

“Did the capability to model in augmented reality assist you in the modeling tasks? Please elaborate." *“Yes. The real time preview helped", “Yes. Didn't have to glance back at references for picking furniture", “Yes, co-location does make the modeling process easier".*

“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.

“If you had complete creative control over this system, how would you improved it?" *“More automation for plane detection", “manual adjustment mode"* (several users said this), *“automatic detection of object class", “easier way to capture large surfaces", “integrate with crowdsourcing".*

5.7.4 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 5.1 & 5.13 depicting the real scenes, corresponding captured CAD model view, and overhead views from 4 Living Rooms, 2 Dining Rooms and a Bedroom scene.

5.8 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

Real Scene *Corresponding CAD View* *Overhead View*

Figure 5.13: *Visual results from 4 Living Rooms, 1 Dining Room and 1 Bedroom scene*

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 5.13).

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 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 door 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 [11, 134] 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 5.13, 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 with bright 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 would diminish if our system were to be implemented on a head-mounted device.

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.

Chapter 6

CONCLUSION

In this thesis, I have proposed several methods to capture indoor scenes interactively, using mobile devices as the primary data acquisition tool. The systems I proposed are designed to be operated by casual users and generate semantically rich architectural visualizations, suitable for the purposes of interior design, remodeling, and real estate. These models also enable AR/VR experiences and a variety of other interesting interactive applications. The ability to create such models using a commodity mobile device enables scene capture on a very large user scale, by both casual and professional users. A key insight that drives the proposed systems is the involvement of an in-situ user, who can provide high level semantic guidance to ensure success of the modeling tasks. The specific aspects of indoor scenes captured by these methods are visual appearance and layout, floor plans, and 3D CAD model representations. Below, I summarize each contribution.

1. **A framework for visual tour capture on a smartphone**: Prior approaches to creating interactive visual tours of indoor scenes usually required sophisticated camera rigs and involved several hours of offline processing, whereas the proposed method lets casual users create and view tours on a commodity mobile phone in a matter of minutes. To capture a tour, a user walks around a scene in a structured manner, while capturing video. This video is processed alongside motion data from the phone's IMU to recreate an interactive version of the walkthrough, that lets the viewer look around a panorama of each room and naturally transition between adjacent rooms. These tours provide viewers with a sense of 'being there' and form a basis for my next contribution, a photogrammetric modeling tool to recover home floor plans. This system enables casual users, business owners, or curators to create interactive visual tours of their

homes, business or museums that are easily shareable and can be viewed on either a desktop or mobile device.

2. **An interactive technique for floor plan estimation**: I proposed an interactive photogrammetric modeling approach to recover room features (corners and doors) from the visual tour. A simple touch interface allows users to mark these points of interest over the visual tour. I introduced a reconstruction algorithm that uses these features to recover an overall floor-plan representation of the environment. The feature extraction interface and reconstruction algorithm run entirely end-to-end on the mobile device. In a few seconds, an estimated reconstruction is presented to the user, who can then refine results via an editing interface. I also compared results to ground truth and competing floor plan reconstruction approaches. Such a system allows homeowners and realtors to create floorplans of homes in real-time using only a commodity mobile phone, without the need for manually measuring room dimensions.

3. **Interfaces for in-situ scene modeling**: I presented two interfaces for in-situ, interactive, 3D CAD model room capture. I proposed leveraging existing large online 3D model databases and state of the art object detection algorithms, to recover representative CAD models from these databases and place them at appropriate positions, orientations, and scales in the scene. The first interface, implemented on an iPad, uses a fixed viewpoint and invites the user to sketch a rough outline of the object, to estimate its position, orientation, and scale. The second interface, implemented on a Tango 3D aware mobile device, allows users to move freely within a scene and processes depth information to extract planes and thereby estimate the pose and scale of furniture items. Both interfaces allow the capture of doors, windows, and textured artwork. The mobile augmented reality approach that I proposed allows users to capture 3D CAD models of rooms in a faster and more accurate manner. These systems can enable interior designers and contractors to easily preview remodels and rearrangements of furniture. Having a CAD model of a room can also be useful for providing semantic cues for

virtual and augmented reality applications.

6.1 Future Work

The work presented in this thesis is a first step towards creating rich semantic models and visualizations of architectural scenes using mobile devices. With the increasing popularity of augmented reality and deep learning, there is an unprecedented opportunity to explore both user interfaces and reconstruction algorithms that enable in-situ scene reconstruction, that will enable thousands or perhaps millions of casual users to create rich, semantic, useful models of the real world. In this section, I present some open problems and possible solutions to achieve this vision:

6.1.1 Improved Image Capture from Mobile Cameras

One of the primary issues with data capture on mobile devices is the smaller size and lower quality of the image sensor, when compared to professional SLR camera systems. Being handheld, a mobile camera is also susceptible to unintentional shake, causing motion blur. Reducing motion blur artifacts by discarding affected frames or applying deblurring techniques [105] could significantly improve the quality of indoor visual tours and also enable more accurate extraction of semantic information from images.

6.1.2 Relaxing Manhattan-World Constraints for Floor Plans

The floor plan reconstruction algorithm proposed in Chapter 3 is unable to model curved walls, non right-angled corners, or other non-Manhattan scene elements. An interesting avenue of future work is to relax the Manhattan-World constraints and enable more general scene capture. The work proposed in Chapter 3 has been extended by other authors [76, 75] to work with non-right angled corners, however curved walls are still difficult to model with a monocular approach. Stereo or depth based reconstruction or interactive annotation are possible solutions to the curved wall problem.

6.1.3 Relaxing Planar Surface Constraints for Furniture

While several important furniture types can be captured and modeled using the planar facet approach described in Chapters 4 and 5, other types of furniture do not fit into this framework, like lamps, curtains, flowerpots, appliances, telephones, etc. Since these items do not have easily identifiable planar facets, it may be possible to fall back on techniques such as ICP [12, 34] to find optimal placement of these objects, given a point cloud of the scene. If initialized correctly, ICP converges to the best possible registration of generalized point clouds. Drawbacks of ICP include increased computation time and potential of erroneous registration if there is only partial correspondence in point data or the initialization is poor.

Another possible approach to capturing more general furniture types is to feed depth information along with the RGB image into a new class of Volumetric Convolutional Neural Nets [125]. Deep nets have also shown promise in extracting structure and position of objects from single images [53, 124, 104].

6.1.4 Full Floor Reconstruction

The proposed room modeling approaches are currently designed to work for one room at a time. However, for indoor models to be useful for applications such as real-estate, a full-floor reconstruction is usually required (similar to the model in Figure 6.1). An avenue for future work would like to enable seamless capture of an entire floor and render it in a stylized manner. An key feature of 3D-aware devices, such as Tango [81] and Hololens [44], is that they enable robust tracking through an entire house, including in-between rooms. It would therefore be possible to extend the in-situ modeling system to work seamlessly across rooms and output a full floor reconstruction at the end of the workflow.

6.1.5 Modeling Scene Appearance

My work has primarily focused on the structural composition of scenes. However, beyond structural fidelity, the visual appearance of scene elements goes a long way in making a

Figure 6.1: *An Indoor 3D CAD model, created manually for real estate purposes, depicting rooms, furniture and texture, with some walls shortened for better visibility.*

model more recognizable. This includes capturing material properties, textures, and scene lighting. I explore some of these concepts by estimating average wall color in scenes and also transferring textures, such as artwork and tapestries, from the scene into the model. However, forming a more visually accurate model will involve capturing intrinsic material properties of scene elements [11], modeling the effect of scene lighting [134], and transferring these into the captured model. Not only would this improve the visual fidelity of the models, but it would also enable interesting applications such as relighting and previewing furniture items in-situ.

6.1.6 Modeling Physical Interactions with the Scene

My modeling techniques assume that scenes are static. However, real world objects can be physically manipulated by moving them, removing them, etc. It would be interesting to explore how to model these physical interactions and reproduce them digitally. One direction is to model behaviors for individual objects. For instance, when pushed, a rolling chair will move differently than a chair with fixed legs. Another direction is to use the proposed CAD capture tools to digitally rearrange furniture. Our tools already allow the user to make manual edits to the furniture layout, however, an improvement would be to incorporate automatic furniture rearrangement algorithms [68, 131, 129] in order to produce machine optimize arrangements. It is my belief that simulating such furniture rearrangements will increase the utility of these models, particularly in virtual reality applications, where realism is key.

6.1.7 Modeling Internal Structure

While the work proposed in this this thesis focuses on the objects contained within the room, it would also be interesting to determine the invisible objects contained within walls. Examples of this include support structures, electric and heating ducts, plumbing etc. The location of these structural elements is crucial for applications such as home remodeling, home energy audits, etc. We could conceivably use alternate sensing modalities on the mobile device to determine their location. Some possibilities are to use the magnetometer to detect metallic structures within walls, or use ultrasonic mapping to determine placement of dense objects such as beams. This could be an interesting and practical direction for future research.

6.1.8 Implementation on other Platforms

Finally, it would very interesting to explore how in-situ modeling interactions will differ when used with emerging head-mounted devices. Microsoft HoloLens [44], is one such device that does away with the touch screen and is able to superimpose 3D content over the real world

using a see-through augmented reality display. As a proof-of-concept, a preliminary prototype of the in-situ modeling system was implemented on the Hololens. This system was able to detect and place a particular kind of office chair in a given scene. A video demonstration captured from the point of view of the user (using Hololens' mixed reality capture) is available here [90]. While this is a very early prototype, it demonstrates the feasibility and ease of in-situ modeling with head-mounted devices. Virtual reality displays such as the Oculus Rift [87] can be used to visualize captured 3D models and can provide a strong sense of presence, that may prove useful for visualization and interior design applications.

It will also be interesting to port the systems to use software-based augmented reality SDKs such as ARKit [6], in order to enable in-situ movements on commodity mobile devices that lack specific hardware for 3D tracking.

6.2 Summary

In this thesis, I introduced the problem of 'Interactive In-Situ Scene Capture on Mobile Devices' along with three systems that allow casual users to capture and visualize various aspects of indoor scenes. I strongly believe that low-cost, easy-to-use, and widely accessible mobile solutions, such as the ones proposed in this thesis, will empower a large number of users to capture a variety of interesting places on earth, ranging from homes and offices to museums and heritage sites.

It is my hope that the principles in this thesis – such as involving an in-situ user, and replacing conventional drafting tools with a mobile device – can inspire future research in this exciting area.

BIBLIOGRAPHY

[1] Meta 2. Metavision. https://www.metavision.com/, 2017. Accessed: 2017-04-04.

[2] Sweet Home 3D. eteks. http://www.sweethome3d.com/, 2016. Accessed: 2017-04-04.

[3] Planner 5D. 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. Protopiper: Physically sketching room-sized objects at actual scale. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology*, UIST '15, pages 427–436, New York, NY, USA, 2015. ACM.

[5] D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver. Google street view: Capturing the world at street level. *Computer*, 43(6):32 –38, june 2010.

[6] Apple. Arkit. https://developer.apple.com/arkit/, 2017. Accessed: 2017-04-04.

[7] Zen Phone AR. Asus. https://www.asus.com/Phone/ZenFone-AR-ZS571KL/, 2017. Accessed: 2017-04-04.

[8] Mathieu Aubry, Daniel Maturana, Alexei A. Efros, Bryan C. Russell, and Josef Sivic. 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, pages 3762–3769. IEEE, 2014.

[9] Autodesk. Autocad. https://www.autodesk.com/products/autocad/overview, 2017. Accessed: 2017-04-04.

[10] Autodesk. Revit family. https://www.autodesk.com/products/revit-family/overview, 2017. Accessed: 2017-04-04.

[11] Sean Bell, Kavita Bala, and Noah Snavely. Intrinsic images in the wild. *ACM Trans. on Graphics (SIGGRAPH)*, 33(4), 2014.

[12] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, February 1992.

[13] Jean-Yves Bouguet. Camera calibration toolbox. http://www.vision.caltech.edu/bouguetj/calib_doc/, 2013. Accessed: 2016-05-27.

[14] F. Brooks. Walkthrough: a dynamic graphics system for simulating virtual buildings. In *Proceedings of the 1986 workshop on Interactive 3D graphics*, I3D '86, pages 9–21, 1987.

[15] Vannevar Bush. As we may think. *interactions*, 3(2):35–46, March 1996.

[16] Pew Research Center. Fact tank: Record shares of americans now own smartphones, have home broadband. http://www.pewresearch.org/fact-tank/2017/01/12/evolution-of-technology/, 2017. Accessed: 2017-04-04.

[17] 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. ShapeNet: An Information-Rich 3D Model Repository. Technical Report http://arxiv.org/abs/1512.03012 [cs.GR], 2015.

[18] Shenchang Eric Chen. Quicktime vr: an image-based approach to virtual environment navigation. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '95, pages 29–38, New York, NY, USA, 1995. ACM.

[19] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 00:5556–5565, 2015.

[20] Jonathan M. Cohen, John F. Hughes, and Robert C. Zeleznik. Harold: A world made of drawings. In *Proceedings of the 1st International Symposium on Non-photorealistic Animation and Rendering*, NPAR '00, pages 83–90. ACM, 2000.

[21] The Neilsen Company. Neilsen mobile insights 2012. http://www.nielsen.com/us/en/insights.html, 2012. Accessed: 2017-04-04.

[22] comScore MobiLens Reports. February 2016 u.s. smartphone subscriber market share. https://www.comscore.com/Insights/Rankings/comScore-Reports-February-2016-US-Smartphone-Subscriber-Market-Share, 2016. Accessed: 2017-04-04.

[23] S. Coorg and S. Teller. Automatic extraction of textured vertical facades from pose imagery. Technical report, Cambridge, MA, USA, 1998.

[24] Satyan R. Coorg and Seth J. Teller. Extracting textured vertical facades from controlled close-range imagery. In *CVPR*, pages 1625–. IEEE, 1999.

[25] Microsoft Corporation. Photosynth (now shut down). `http://photosynth.net/`, 2011. Accessed: 2017-01-01.

[26] J.M. Coughlan and A.L. Yuille. Manhattan world: Compass direction from a single image by bayesian inference. In *Proceedings of the International Conference on Computer Vision*, ICCV '99, pages 941–947, 1999.

[27] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 303–312. ACM, 1996.

[28] AR Home Designer. Elementals. `http://www.elementalsweb.com/home_augmented_reality_designer/`, 2017. Accessed: 2017-04-04.

[29] Hao Du, Peter Henry, Xiaofeng Ren, Marvin Cheng, Dan B. Goldman, Steven M. Seitz, and Dieter Fox. Interactive 3d modeling of indoor environments with a consumer depth camera. In *Proceedings of the 13th International Conference on Ubiquitous Computing*, UbiComp '11, pages 75–84, New York, NY, USA, 2011. ACM.

[30] Musée du Louvre. Headless statue of gudea, prince of lagash. `http://www.louvre.fr/en/oeuvre-notices/headless-statue-gudea-prince-lagash`, 2017. Accessed: 2017-04-04.

[31] Mathias Eitz, Ronald Richter, Tamy Boubekeur, Kristian Hildebrand, and Marc Alexa. Sketch-based shape retrieval. *ACM Trans. Graph.*, 31(4):31:1–31:10, July 2012.

[32] Douglas C. Engelbart and William K. English. A research center for augmenting human intellect. In *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I*, AFIPS '68 (Fall, part I), pages 395–410, New York, NY, USA, 1968. ACM.

[33] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[34] Robert B. Fisher. Projective icp and stabilizing architectural augmented reality overlays. In *Proc. Int. Symp. on Virtual and Augmented Architecture (VAA01*, pages 69–80, 2001.

[35] A. Flint, C. Mei, I. Reid, and D. Murray. Growing semantically meaningful models for visual slam. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 467 –474, june 2010.

[36] A. Flint, D. Murray, and I. Reid. Manhattan scene understanding using monocular, stereo, and 3d features. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2228 –2235, nov. 2011.

[37] Sean Follmer, David Carr, Emily Lovell, and Hiroshi Ishii. Copycad: Remixing physical objects with copy and paste from the real world. In *Adjunct Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 381–382, New York, NY, USA, 2010. ACM.

[38] Association for Computing Machinery. Ivan sutherland, acm turing award winner, 1988. `http://amturing.acm.org/award_winners/sutherland_3467412.cfm`, 1988. Accessed: 2017-04-04.

[39] FreeCAD. `https://www.freecadweb.org/`, 2017. Accessed: 2017-04-04.

[40] Thomas Funkhouser, Patrick Min, Michael Kazhdan, Joyce Chen, Alex Halderman, David Dobkin, and David Jacobs. A search engine for 3d models. *ACM Trans. Graph.*, 22(1):83–105, January 2003.

[41] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Reconstructing building interiors from images. In *In Proc. of the International Conference on Computer Vision (ICCV '09)*, 2009.

[42] Dan B Goldman, Brian Curless, David Salesin, and Steven M. Seitz. Schematic storyboarding for video visualization and editing. In *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, pages 862–871, New York, NY, USA, 2006. ACM.

[43] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[44] HoloLens. Microsoft Corp. `https://www.microsoft.com/microsoft-hololens/en-us`, 2015. Accessed: 2016-05-27.

[45] Homestyler. Autodesk. `http://www.homestyler.com`, 2013. Accessed: 2016-05-27.

[46] Qixing Huang, Hai Wang, and Vladlen Koltun. Single-view reconstruction via joint analysis of image and shape collections. *ACM Trans. Graph.*, 34(4):87:1–87:10, July 2015.

[47] D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky. 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*, volume 1, pages I–I, May 2006.

[48] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: A sketching interface for 3d freeform design. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 409–416. ACM, 1999.

[49] Google Inc. Google art project. `http://www.googleartproject.com/`, 2011. Accessed: 2017-04-04.

[50] Sensopia Inc. Magicplan. `http://www.sensopia.com/`, 2011. Accessed: 2017-04-04.

[51] H. Ishiguro, K. Ueda, and S. Tsuji. Omnidirectional visual information for navigating a mobile robot. In *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 799 –804 vol.1, may 1993.

[52] Tomoya Ishikawa, Kalaivani Thangamani, Masakatsu Kourogi, AndrewP. Gee, Walterio Mayol-Cuevas, Keechul Jung, and Takeshi Kurata. In-situ 3d indoor modeler with a camera and self-contained sensors. In Randall Shumaker, editor, *Virtual and Mixed Reality*, volume 5622 of *Lecture Notes in Computer Science*, pages 454–464. Springer Berlin Heidelberg, 2009.

[53] Hamid Izadinia, Qi Shan, and Steven M. Seitz. IM2CAD. *CoRR*, abs/1608.05137, 2016.

[54] Shunichi Kasahara, Valentin Heun, Austin S. Lee, and Hiroshi Ishii. Second surface: Multi-user spatial collaboration system based o= n augmented reality. In *SIGGRAPH Asia 2012 Emerging Technologies*, SA '12, pages 20:1–20:4. ACM, 2012.

[55] Natasha Kholgade, Tomas Simon, Alexei Efros, and Yaser Sheikh. 3d object manipulation in a single photograph using stock 3d models. *ACM Trans. Graph.*, 33(4):127:1–127:12, July 2014.

[56] Hyejin Kim, Gerhard Reitmayr, and Woontack Woo. Imaf: In situ indoor modeling and annotation framework on mobile phones. *Personal Ubiquitous Comput.*, 17(3):571–582, March 2013.

[57] Young Min Kim, Jennifer Dolson, Michael Sokolsky, Vladlen Koltun, and Sebastian Thrun. Interactive acquisition of residential floor plans. In *ICRA*, pages 3055–3062. IEEE, 2012.

[58] Vision Innovation Labs. Lowes. `http://www.lowesinnovationlabs.com/tango/`, 2017. Accessed: 2017-04-04.

[59] Tobias Langlotz, Stefan Mooslechner, Stefanie Zollmann, Claus Degendorfer, Gerhard Reitmayr, and Dieter Schmalstieg. Sketching up the world: In situ authoring for mobile augmented reality. *Personal Ubiquitous Comput.*, 16(6):623–630, August 2012.

[60] Manfred Lau, Masaki Hirose, Akira Ohgawara, Jun Mitani, and Takeo Igarashi. Situated modeling: A shape-stamping interface with tangible primitives. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*, TEI '12, pages 275–282, New York, NY, USA, 2012. ACM.

[61] Manfred Lau, Akira Ohgawara, Jun Mitani, and Takeo Igarashi. Converting 3d furniture models to fabricatable parts and connectors. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, pages 85:1–85:6. ACM, 2011.

[62] Manfred Lau, Greg Saul, Jun Mitani, and Takeo Igarashi. 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, pages 17–24. Eurographics, 2010.

[63] David C. Lee, Martial Hebert, and Takeo Kanade. Geometric reasoning for single image structure recovery. In *In proc. CVPR*, 2009.

[64] Yangyan Li, Hao Su, Charles Ruizhongtai Qi, Noa Fish, Daniel Cohen-Or, and Leonidas J. Guibas. Joint embeddings of shapes and images via cnn image purification. *ACM Trans. Graph.*, 34(6):234:1–234:12, October 2015.

[65] Joseph J. Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing ikea objects: Fine pose estimation. In *Proceedings of the 2013 IEEE International Conference on Computer Vision*, ICCV '13, pages 2992–2999. IEEE, 2013.

[66] Andrew Lippman. Movie-maps: An application of the optical videodisc to computer graphics. In *Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '80, pages 32–42, New York, NY, USA, 1980. ACM.

[67] Bing Maps. Microsoft. `http://www.bing.com/maps`, 2017. Accessed: 2017-04-12.

[68] Paul Merrell, Eric Schkufza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. Interactive furniture layout using interior design guidelines. *ACM Trans. Graph.*, 30(4):87:1–87:10, July 2011.

[69] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, pages 127–136. IEEE, 2011.

[70] Thanh Nguyen, Raphael Grasset, Dieter Schmalstieg, and Gerhard Reitmayr. Interactive syntactic modeling with a single-point laser range finder and camera. In *IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2013, Adelaide, Australia, October 1-4, 2013*, pages 107–116. IEEE Computer Society, 2013.

[71] Tom Novotny, Irma Lindt, and Wolfgang Broll. A multi modal table-top 3d modeling tool in augmented environments. In *Proceedings of the 12th Eurographics Conference on Virtual Env.*, EGVE'06, pages 45–52. Eurographics, 2006.

[72] Benjamin Nuernberger, Eyal Ofek, Hrvoje Benko, and Andrew D. Wilson. Snaptoreality: Aligning augmented reality to the real world. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 1233–1244, New York, NY, USA, 2016. ACM.

[73] Patrick Paczkowski, Min H. Kim, Yann Morvan, Julie Dorsey, Holly Rushmeier, and Carol O'Sullivan. Insitu: Sketching architectural designs in context. *ACM Trans. Graph.*, 30(6):182:1–182:10, December 2011.

[74] Les Piegl and Wayne Tiller. *The NURBS Book (2Nd Ed.)*. Springer-Verlag New York, Inc., New York, NY, USA, 1997.

[75] Giovanni Pintore, Fabio Ganovelli, Enrico Gobbetti, and Roberto Scopigno. Mobile reconstruction and exploration of indoor structures exploiting omnidirectional images. In *SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications*, SA '16, pages 1:1–1:4, New York, NY, USA, 2016. ACM.

[76] Giovanni Pintore and Enrico Gobbetti. Effective mobile mapping of multi-room indoor structures. *Vis. Comput.*, 30(6-8):707–716, June 2014.

[77] Floor Planner. `http://www.floorplanner.com`, 2016. Accessed: 2017-04-04.

[78] Home Planner. Ikea. `http://www.ikea.com/ms/en_JP/rooms_ideas/splashplanners.html`, 2013. Accessed: 2016-05-27.

[79] Planoplan. `http://planoplan.com/en/`, 2017. Accessed: 2017-04-04.

[80] Phab 2 Pro. Lenovo. `http://shop.lenovo.com/us/en/tango/`, 2017. Accessed: 2017-04-04.

[81] Project Tango. Google inc., ATAP. `https://www.google.com/atap/projecttango/`, 2014. Accessed: 2016-05-27.

[82] Quiksee. http://www.quiksee.com/, 2011. Accessed: 2017-04-04.

[83] Gerhard Reitmayr and Tom Drummond. Going out: robust model-based tracking for outdoor augmented reality. In *Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality*, ISMAR '06, pages 109–118, Washington, DC, USA, 2006. IEEE Computer Society.

[84] Robert McNeel & Associates. Rhinoceros. https://www.rhino3d.com/, 2017. Accessed: 2017-04-04.

[85] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[86] Point Grey Research. Ladybug omni-directional camera. http://www.ptgrey.com, 2011. Accessed: 2017-04-04.

[87] Rift. Oculus. https://www.oculus.com/, 2017. Accessed: 2017-04-04.

[88] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "grabcut": Interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3):309–314, August 2004.

[89] Renato F. Salas-Moreno, Richard A. Newcombe, Hauke Strasdat, Paul H.J. Kelly, and Andrew J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1352–1359, 2013.

[90] Aditya Sankar. Hololens in-situ chair modeling. https://youtu.be/YeUB34zDn6g, 2017. Accessed: 2017-10-03.

[91] Aditya Sankar and Steve M. Seitz. Interactive room capture on 3d-aware mobile devices. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, UIST '17, pages 415–426, New York, NY, USA, 2017. ACM.

[92] Aditya Sankar and Steven Seitz. Capturing indoor scenes with smarthphones, project page. http://grail.cs.washington.edu/videotour/, 2012. Accessed: 2017-04-12.

[93] Aditya Sankar and Steven Seitz. In-situ cad capture, project page. http://grail.cs.washington.edu/projects/insitucad/, 2016. Accessed: 2017-04-12.

[94] Aditya Sankar and Steven Seitz. Interactive room capture on 3d-aware mobile devices, project page. http://grail.cs.washington.edu/projects/armodeling/, 2017. Accessed: 2017-04-12.

[95] Aditya Sankar and Steven M. Seitz. Capturing indoor scenes with smartphones. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 403–412. ACM, 2012.

[96] Aditya Sankar and Steven M. Seitz. In situ cad capture. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '16, pages 233–243, New York, NY, USA, 2016. ACM.

[97] T. Schöps, T. Sattler, C. Häne, and M. Pollefeys. 3d modeling on the go: Interactive 3d reconstruction of large-scale scenes on mobile devices. In *2015 International Conference on 3D Vision*, pages 291–299, Oct 2015.

[98] Michael Ian Shamos. Computational geometry. `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.366.8570`, 1978. Ph.D. Thesis, Accessed: 2017-04-04.

[99] Tianjia Shao, Weiwei Xu, Kun Zhou, Jingdong Wang, Dongping Li, and Baining Guo. An interactive approach to semantic modeling of indoor scenes with an rgbd camera. *ACM Trans. Graph.*, 31(6):136:1–136:11, November 2012.

[100] H. Shin, Y. Chon, and H. Cha. Unsupervised construction of an indoor floor plan using a smartphone. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):889–898, Nov 2012.

[101] HyoJong Shin and Takeo Igarashi. Magic canvas: Interactive design of a 3-d scene prototype from freehand sketches. In *Proceedings of Graphics Interface 2007*, GI '07, pages 63–70. ACM, 2007.

[102] Sketchup. Trimble Navigation Limited. `http://www.sketchup.com/`, 2016. Accessed: 2016-05-27.

[103] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, pages 835–846, New York, NY, USA, 2006. ACM.

[104] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 945–953, Washington, DC, USA, 2015. IEEE Computer Society.

[105] Shuochen Su, Mauricio Delbracio, Jue Wang, Guillermo Sapiro, Wolfgang Heidrich, and Oliver Wang. Deep video deblurring. *CoRR*, abs/1611.08387, 2016.

[106] Mengu Sukan, Steven Feiner, Barbara Tversky, and Semih Energin. 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, pages 217–226. IEEE, 2012.

[107] Ivan E. Sutherland. Sketchpad: A man-machine graphical communication system. In *Proceedings of the May 21-23, 1963, Spring Joint Computer Conference*, AFIPS '63 (Spring), pages 329–346. ACM, 1963.

[108] Dassault Systèmes. Solidworks. `www.solidworks.com/`, 2017. Accessed: 2017-04-04.

[109] Richard Szeliski and Heung-Yeung Shum. Creating full view panoramic image mosaics and environment maps. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 251–258, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[110] C. J. Taylor. Videoplus: A method for capturing the structure and appearance of immersive environments. *IEEE Transactions on Visualization and Computer Graphics*, 8(2):171–182, April 2002.

[111] TLX. Nasa. `http://www.nasatlx.com/`, 2009. Accessed: 2016-05-27.

[112] Michael Tsang, George W. Fitzmaurice, Gordon Kurtenbach, Azam Khan, and Bill Buxton. 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, pages 111–120. ACM, 2002.

[113] Steve Tsang, Ravin Balakrishnan, Karan Singh, and Abhishek Ranjan. A suggestive interface for image guided 3d sketching. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 591–598. ACM, 2004.

[114] TurboSquid. Turbosquid. `http://www.turbosquid.com/`, 2016. Accessed: 2016-05-27.

[115] Matthew Uyttendaele, Antonio Criminisi, Sing Bing Kang, Simon Winder, Richard Szeliski, and Richard Hartley. Image-based interactive exploration of real-world environments. *IEEE Computer Graphics and Applications*, 24:52–63, 2004.

[116] Julien Valentin, Vibhav Vineet, Ming-Ming Cheng, David Kim, Jamie Shotton, Pushmeet Kohli, Matthias Nießner, Antonio Criminisi, Shahram Izadi, and Philip Torr. Semanticpaint: Interactive 3d labeling and learning at your fingertips. *ACM Trans. Graph.*, 34(5):154:1–154:17, November 2015.

[117] Anton van den Hengel, Rhys Hill, Ben Ward, and Anthony Dick. In situ image-based modeling. *IEEE / ACM International Symposium on Mixed and Augmented Reality*, pages 107–110, 2009.

[118] Vive. Htc. `https://www.vive.com/us/`, 2017. Accessed: 2017-04-04.

[119] Michael Waechter, Mate Beljan, Simon Fuhrmann, Nils Moehrle, Johannes Kopf, and Michael Goesele. Virtual rephotography: Novel view prediction error for 3d reconstruction. *ACM Trans. Graph.*, 36(1):8:1–8:11, January 2017.

[120] Sketchup 3D Warehouse. Trimble Navigation Limited. `https://3dwarehouse.sketchup.com/`, 2016. Accessed: 2017-04-04.

[121] David E. Weisberg. The engineering design revolution: The people, companies and computer systems that changed forever the practice of engineering. `www.cadhistory.net/`, 2008. Accessed: 2017-04-04.

[122] Wikipedia. Dac-1, by general motors. `https://en.wikipedia.org/wiki/DAC-1`, 2017. Accessed: 2017-04-04.

[123] Wikipedia. Feature phone. `https://en.wikipedia.org/wiki/Feature_phone`, 2017. Accessed: 2017-04-04.

[124] Jiajun Wu, Tianfan Xue, Joseph J. Lim, Yuandong Tian, Joshua B. Tenenbaum, Antonio Torralba, and William T. Freeman. Single image 3d interpreter network. *CoRR*, abs/1604.08685, 2016.

[125] Zhirong Wu, Shuran Song, Aditya Khosla, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets for 2.5d object recognition and next-best-view prediction. *CoRR*, abs/1406.5670, 2014.

[126] Baoxuan Xu, William Chang, Alla Sheffer, Adrien Bousseau, James McCrae, and Karan Singh. True2form: 3d curve networks from 2d sketches via selective regularization. *ACM Trans. Graph.*, 33(4):131:1–131:13, July 2014.

[127] Kun Xu, Kang Chen, Hongbo Fu, Wei-Lun Sun, and Shi-Min Hu. Sketch2scene: Sketch-based co-retrieval and co-placement of 3d models. *ACM Trans. Graph.*, 32(4):123:1–123:15, July 2013.

[128] Y. Yagi, S. Kawato, and S. Tsuji. Real-time omnidirectional image sensor (copis) for vision-guided navigation. *Robotics and Automation, IEEE Transactions on*, 10(1):11 –22, feb 1994.

[129] Meng Yan, Xuejin Chen, and Jie Zhou. An interactive system for efficient 3d furniture arrangement. In *Proceedings of the Computer Graphics International Conference*, CGI '17, pages 29:1–29:6, New York, NY, USA, 2017. ACM.

[130] Brandon Yee, Yuan Ning, and Hod Lipson. Augmented reality in-situ 3d sketching of physical objects, 2009.

[131] Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F. Chan, and Stanley J. Osher. Make it home: Automatic optimization of furniture arrangement. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, pages 86:1–86:12, New York, NY, USA, 2011. ACM.

[132] Paul A Zandbergen. Accuracy of iphone locations: A comparison of assisted gps, wifi and cellular positioning. *Transactions in GIS*, 13:5–25, 2009.

[133] Robert C. Zeleznik, Kenneth P. Herndon, and John F. Hughes. Sketch: An interface for sketching 3d scenes. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 163–170. ACM, 1996.

[134] Edward Zhang, Michael F. Cohen, and Brian Curless. Emptying, refurnishing, and relighting indoor spaces. *ACM Trans. Graph.*, 35(6):174:1–174:14, November 2016.

Appendix A

APPENDIX A

A.1 Funding/Grant Acknowledgements

This work was supported in part by National Science Foundation grants IIS-0811878, IIS-1250793 and IIS-0963657, SPAWAR, the University of Washington Animation Research Labs, and Google.

A.2 3D Model and Photo Credits

We would also like to credit the following Sketchup [102] 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 [114] users viktor7777, Fworx, AG4T, AlpArt, vandoan, and CSI Renderers for the use of their 3D models in this presentation.

VITA

Aditya Sankar grew up in New Delhi, India, and received his B. Tech. in Information and Communication Technology from the Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT), Gandhinagar, Gujarat, India in 2008. For two years thereafter he worked, first as an Intern and then as a Research Software Developer at Microsoft Research India, Bangalore, before pursuing graduate studies in the United States at the University of Washington - Seattle. For his Ph.D., he was advised by Professor Steven M. Seitz. During his graduate studies Aditya also interned at Microsoft Research, in Bangalore, India, Microsoft Research in Redmond, Washington, and Floored Inc. in New York City, New York. In 2012, he received his Masters in Computer Science and in 2017, he received his Doctor of Philosophy degree.