Indoor Scene Augmentation via Scene Graph Priors

Mohammad Keshavarzi

Electrical Engineering and Computer Sciences
University of California, Berkeley

May 8, 2022

Indoor Scene Augmentation via Scene Graph Priors

by Mohammad Keshavarzi

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

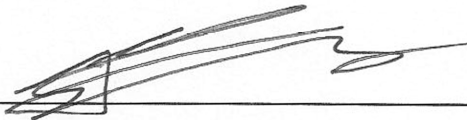Approval for the Report and Comprehensive Examination:

Committee:

Associate Professor Bjoern Hartmann
Research Advisor

5/7/2022

Date:

* * * * * * *

Dr. Allen Y. Yang
Second Reader

5-2-2022

Date:

Abstract

Indoor Scene Augmentation via Scene Graph Priors

by

Mohammad Keshavarzi

Master of Science, Plan II in Electrical Engineering and Computer Science

University of California, Berkeley

Associate Professor Bjoern Hartmann, Chair

In spatial computing experiences augmenting virtual objects to existing scenes requires a contextual approach, where geometrical conflicts are avoided, and functional relationships to other objects are maintained. Yet, due to the complexity and diversity of user environments, automatically predicting contextual and adaptive placements of virtual content is considered a challenging task. Motivated by this problem, in this paper we introduce SceneGen, a generative contextual scene augmentation framework that predicts virtual object placement within existing scenes. SceneGen takes a scene as input, and outputs positional and orientational probability maps for placing virtual content. We formulate a novel spatial Scene Graph representation, which encapsulates explicit topological properties between objects, object groups, and rooms. We use kernel density estimation to build a multivariate conditional knowledge model trained using prior spatial Scene Graphs extracted from real-world 3D scanned data as prior. To further capture orientational properties, we develop a fast pose annotation tool to extend current real-world datasets with orientational labels. Furthermore, we conduct comparative and user experiments to demonstrate the performance of our system in various indoor scene augmentation scenarios. Finally, to demonstrate our system in action, we develop an Augmented Reality application, in which objects can be contextually augmented in real-time.

To my dear Family ...

Contents

List of Figures

List of Tables

Acknowledgements

This project is derived from the following paper:

Chapter 1

Introduction

1.1 Spatial Limitations in Spatial Computing

Spatial Computing experiences such as *augmented reality* (AR) and *virtual reality* (VR) have formed a newly exciting market in today's technological space. New applications and experiences are being launched daily across the categories of gaming, healthcare, design, education, and more. However, for all of the countless applications available, they are physically constrained by the geometry and semantics of the 3D user environment where existing furniture and building elements are present [35, 40] . Contrary to traditional 2D graphical user interface, where a flat rectangular region hosts digital content, 3D spatial computing environments are often occupied by physical obstacles that are diverse and often times non-convex. Therefore, how one can assess content placement in spatial computing experiences is highly dependent on the user's target scene.

However, since different users may reside in different spatial environments, which differ in dimensions, functions (rooms, workplace, garden, etc.), and open usable spaces, existing furniture and their arrangements are often unknown to the developers, making it very challenging to design a virtual experience that would adapt to all users' environments. Currently, contextual placement is addressed by asking users themselves to identify the usable spaces in their surrounding environments or manually positioning the augmented object(s) within the scene. Then, virtual object placement in most AR experiences is limited to specific surfaces and locations, e.g., placing objects naively in front of the user with no scene understanding, or only using basic horizontal or vertical surface detection. These simplistic strategies may work to some extent for small virtual objects, but the methods break down for larger objects or complex scenes with multiple object augmentation requirements. This limitation is further elevated in remote multi-user interaction scenarios, where finding a common virtual ground physically accessible to all participants to augment their content becomes challenging [19]. Hence, such experiences automatically become less immersive once the users encounter implausible virtual object augmentation in their environments.

1.2   Contrained Scene Synthesis

The task of adding objects to existing constructed scenes falls under the problem of *constrained scene synthesis*. The work of [18, 31, 24, 38, 42, 57] are examples of such approach. However, there are two major challenges in the general literature that create bottlenecks for virtual content augmentation in spatial computing experiences. First, current scanned 3D datasets publicly available are limited in size and diversity, and may not offer all the data required to capture topological properties of the rooms. For instance, *pose*, the direction in which the object is facing, is a critical feature for understanding the orientational property of an object, and yet, such a property is not clearly annotated for any objects in many large-scale real-world datasets such as SUN-RGBD and Matterport3D. Therefore, more recent research has adapted synthetic datasets, which do not necessarily need to be manually annotated for pose prior information.

However, a critical drawback of using synthetic datasets is that it cannot capture the natural transformation and topological properties of objects in real-world settings. Indeed, topological relationships between objects in real-world scenes typically exceed theoretical design assumptions of an architect, and instead capture contextual relationships from a living environment. Moreover, the limitations of the modeling software for synthetic datasets can also introduce unwanted biases to the generated scenes. The SUNCG [51] dataset, for instance, was built with Planner5D platform, an online tool which any user around the world can use. However, it comes with modeling limitations for generating rooms and furniture. Orientations are also snapped to right angles by default, which makes most scenes in the dataset Manhattan-like. More importantly, there is no indication whether the design is complete or not, namely, a user may just start playing with the software and then leave at a random time, while the resulting arrangement is still captured as a legitimate human-modeled arrangement in the dataset.

Second, recent models take advantage of implicit deep learning models and have shown promising results in synthesizing indoor scenes. Yet, these approaches fall short for content developers to parameterize customized placement in relation to standard objects in the scene, and to generate custom spatial functionalities. One major limitation of these studies is that they do not have direct control over objects in the generated scene. For example, authors of [24] reported they could not specify object counts or constrain the scene to contain a subset of objects. Such limitations come from the implicit nature of such neural networks. Implicit models produce a black-box tool, which is difficult to comprehend should a end-user wishes to tweak its functions. In cases where a new object type (which has not been previously seen in prior datasets) need to be placed, implicit structures may not provide abilities to take into account manually defined topological properties by a user. Moreover, training deep neural networks requires large datasets, a bottleneck that we have discussed above.

1.3   Research Objectives and Contributions

Motivated by these challenges, in this paper we introduce SceneGen, a generative contextual augmentation framework that leverages explicit scene graph models to predict the functional

placements of new virtual objects in a target indoor scene. Contrary to the implicit models, SceneGen is based on clear, logical object attributes and their architectural relationships with other objects and the room. In light of the existing body of literature on semantic scene graphs, we leverage this approach to encapsulate the relevant object relationships for scene augmentation. Scene graphs have already been in use for general scene generation tasks; they can also inform the intelligent placement of virtual objects in physical scenes. We use kernel density estimation (KDE) to build a multivariate conditional model to encapsulate explicit positioning and clustering information for objects in various room types. This information will allow our algorithm to calculate a probability distribution to place and orient the new object in a scene while satisfying their physical and functional requirements. From the calculated probabilities, we generate a score for each potential placement of the new object, visualized in a heat map over the room. Our system is designed for both fully automated scene augmentation and also user-in-the-loop scenarios, allowing the user to understand the influence of the relationship features and their impact on the results.

Our contributions can be summarized as follows:

1. We introduce a spatial Scene Graph representation which encapsulates positional and ori-entational relationships of a scene. Our proposed Scene Graph captures pairwise topology between objects, object groups, and the room.

2. We develop a prediction model for object contextual augmentation in existing scenes. We construct an explicit Knowledge Model which is trained from Scene Graph representations captured from real-world 3D scanned data.

3. To learn orientational relationships from real-world 3D scanned data, we have manually labeled the Matterport3D dataset with pose directions using an open-source labeling tool for fast pose labeling.

4. We develop an Augmented Reality (AR) application that scans a user's room and generates a Scene Graph based on the existing objects. Using our model, we sample poses across the room to determine a probabilistic heat map of where the object can be placed. By placing objects in poses where the spatial relationships are likely, we are able to augment scenes that are realistic.

The proposed framework developed in this thesis can play a role in increasing the adoption of Spatial Computing interfaces in everyday environments. The research would allow developers to design content without knowing the target scene of the user itself. Content itself can adapt to the target scene while addressing topological goals with the real and virtual objects. Such approach can pave the way for developing Responsive Spatial Computing frameworks. Similar to responsive web design [33], responsive spatial computing can be as a cost-effective alternative to hard-coded applications due to its ability to house all of the code in a single program.

Moreover the proposed system can facilitate a wide variety of spatial computing applications. Augmenting virtual objects to scenes has been explored in online-shopping settings, and collabora-tive environments require placing one user's objects into another user's surroundings. In addition,

content creation of augmented and virtual reality experiences requires long hours of cross platform development on current applications, so our system will allow faster scene generation and content generation in AR/VR experiences.

Chapter 2

Background

2.1 Scene Understanding

Semantic Scene Graphs form one part of the overall task of scene understanding. On this topic, a progression of papers attempted to encapsulate human "common-sense" knowledge in various ways: physical constraints and statistical priors [50], physical constraints and stability reasoning [16], physics-based stability modeling [69], language priors [30], and statistical modeling with deep learning [10]. A similar approach was detailed in [21] for 3D reconstruction, taking advantage of the regularity and repetition of furniture arrangements in certain indoor spaces, e.g., office buildings. In [59], the authors proposed a technique that potentially could be well suited to AR applications, as it builds a 3D reconstruction of the scene through consecutive depth acquisitions, which could be taken incrementally as a user moves within their environment. Some recent work has addressed problems such as retrieving 3D layouts from 2D panoramic input [52, 22] or floorplan sketches [20], building scenes from 3D point clouds [36, 49], and 3D plane reconstruction from a single image [65, 28]. One can consult a recent overview of the topic in [29]. Our approach leverages these works on scene understanding, because our model operates on the assumption that we already have locations and bounding boxes of the existing objects in scene.

2.2 Semantic Scene Graphs

Semantic Scene Graphs have been applied to various tasks in the past, including image retrieval [17], visual question answering [54], image caption generation [62], and more. The past research can be divided into two approaches: (1) separate stages of object detection and graph inference, and (2) joint inference of object classes and graph relationships. Papers that followed the first approach often leverage existing object detection networks [41, 25, 66, 62, 9]. Similarly to other scene understanding tasks, many methods also involved learning prior knowledge of common scene structures in order to apply them to new scenes, such as physical constraints from stability reasoning [61] or frequency priors represented as recurring scene motifs [66]. Most methods were benchmarked based on the Visual Genome dataset [23]. However, recent studies found this dataset

has an uneven distribution of examples across its data space. In response, researchers in [15] and [9] proposed new networks to draw from an external knowledge base and to utilize statistical correlations between objects and relationships, respectively. Our work focuses on the task of construction and utilization of the semantic Scene Graph. Work of [55] utilized PointNet [37] and Graph Convolutional Networks to regress a scene graph from the point cloud of a scene. Similar to [55] and [66, 9], we also use statistical relationships and dataset priors; but unlike these papers, we use a explicit graph representation combined with a KDE model, rather than deep convolution networks to address user in the loop scenarios or semi-autonomous scene augmentation tasks.

2.3 Scene Synthesis

The general goal of indoor scene synthesis is to produce a feasible furniture layout of various object classes which satisfy both functional and aesthetic criteria [67]. Early work of synthetic generation focused on hard-coded rules, guideline and grammars, resembling a procedural approach for this problem [3, 60, 14]. The work of [34] is a successful example of hard-coded design guidelines as priors for the scene generation process. They extracted these guidelines through consulting manuals on furniture layout [47, 58, 53] and interviewing professional designers who specialize in arranging furniture. A similar approach is also seen in [64], while [63] attempted synthesizing open world layouts with hard-coded factor graphs.

The work of [12] can be seen as one of the early adapters of example-based scene synthesis. They synthesized scenes by training to build a probabilistic model based on Bayesian networks and Gaussian mixtures. Their problem, however, was one of generating the entire scene, and they utilized a more limited set of input example scenes. In the work of [18], a full 3D scene was synthesized iteratively by adding a single object at a time. This system learned some priors similar to ours, including pairwise and higher-order object relations. Compared to this work, we incorporate additional priors, including objects' relative position within the room bounds. The work of [27, 26] and [13] also took room functions into account. While object topologies differ in various room function, a major challenge in this approach is that not all spaces can be classified with a certain room function. For instance, in a small studio apartment, the living room might serve additional functions such as dining room and a study space. [44] also proposed a similar approach, involving a Gaussian mixture model and kernel density estimation. However, their system targeted an inverse problem of ours, namely, their problem received a selected object location as input and was asked to predict an object type. We find our problem to be more relevant to the needs of a content creator who knows what object they wish to place in scene, but does not have prior knowledge about a user's surroundings.

Another data-driven approach to scene generation involves modeling human activities and interactions with the scene [11, 31, 13, 39]. Research following this approach generally seeks to model and adjust the entire scene according to human actions or presence. There have also been a number of interesting studies that take advantage of logical structures modeled for natural language processing (NLP) scenarios. The work of [5, 4, 7, 32] are examples of such approach. More specifically, [32] bears a minor resemblance to our approach, in training on object relations and the

ability to augment an initial input scene. But unlike our work, it augments scenes by merging in subscenes retrieved from a database. In contrast, we seek to add in individual objects, which is more aligned with the needs of creators of augmented reality experiences. A series of papers (including [46, 8, 1]) proposed generating a 3D scene representation by recreating the scene from RGB-D image input, using retrieved and aligned 3D models. This research, however, involves recreating an existing physical scene, and does not handle adding new objects.

More recent work endeavored to improve learning-based methods, using deep convolutional priors [56], scene-autoencoding [24], and new representations of object semantics [2], to name just a few. [68] addressed a related but distinct problem of synthesizing a scene by arranging and grouping an input set of objects. The work of [42] is another example of using deep generative models for scene synthesis. Their method sampled each object attribute with a single inference step to allow constrained scene synthesis. This work was extended in PlanIt [57], where the authors proposed a combination of two separate convolutional networks to address constrained scene synthesis problems. They argued that object-level relationships can facilitate high-level planning of how a room should be laid out, while room-level relationships perform well at placing objects in precise spatial configurations. Similar to our scene graph approach is the work of [70], which utilizes a dense scene graph for passing neural messages to augment an input 3D indoor scene with new objects matching their surroundings. However, its scene graph representation does not cover orientational relationships and only covers limited positional relationships between objects.

Our method differs from these studies in utilizing an explicit model rather than an implicit structure and taking advantage of alternative discrete relationships with the room itself. Moreover, our model can be trained on datasets that are significantly smaller in size, with faster training and inference time. Furthermore, While work of [57, 70] extend pairwise object to object relations to object to wall relations, we consider the room as a separate entity and simply evaluate whether the object is on edge of the room, the corner of the room, or in the middle of the room. From an architectural perspective, while the walls of indoor spaces are elements that create the encasement of the room, the general location in which the object sits within the room plays a critical role in the collective functionality of the overall space. Therefore, we show modeling the room relationship as a separate explicit entity in the scene graph would benefit the scene augmentation process in both fully automated and user in the loop scenarios.

Moreover, while our work maintains wide range of overlap with studies in the field of scene synthesis, the main goal of our work is to facilitate AR/VR content generation in single-blind scenarios, where content developers are not aware of target scenes. Therefore, in this paper we aim to emphasize our explicit structure, which allows AR/VR developer to define new object categories in which may not be available in large scale public datasets.

Chapter 3

Methodology

3.1 SceneGen Overview

SceneGen is a framework to augment scenes with virtual objects using a generative model to maximize the likelihood of the relationships captured in a spatial Scene Graph. Specifically, if given a partially filled room, SceneGen augments it with one or more new virtual objects in a realistic manner using an explicit model trained on relationships between objects in the real world. The SceneGen workflow is shown in Figure 3.1.

In this paper, we first introduce a novel Scene Graph that connects the objects and the room (both represented as nodes) using spatial relationships (represented as edges) in Section 3.2. For each object, these relationships are determined by positional and orientational features between itself and other objects, object groups, and the room.

In Section 3.3 we show how from a dataset of rooms, we can extract these Scene Graphs to construct a Knowledge Model that is used to train explicit models that approximate the probability density functions of position and orientation relationships for a given object using kernel density estimation. In order to augment a scene with a virtual object, SceneGen samples possible positions and orientations in a scene, building updated Scene Graphs for each sample. We estimate the probability of each sample and place an object at the most likely pose. SceneGen also shares a heat map of the likelihood of each sample to suggest alternate high probability placements. This can be repeated to augment multiple virtual objects.

Our implementation of SceneGen is built using data extracted from the Matterport3D dataset as priors and is detailed in Section 3.4. As using object scans results in unoriented bounding boxes in Matterport3D, we develop an application to facilitate the labeling of the facing direction of each object.

We assess the effectiveness of SceneGen in Sections **??** and 4.4 for eight categories of objects across several types of room including bedroom, living room, hallway, and kitchen. In order to understand the effectiveness of each relationship on predicting where and how a new object should be placed, we run a series of ablation tests on each feature. We use *K*-fold cross validation to partition the Matterport3D dataset, building the Knowledge Model on a training set and assessing
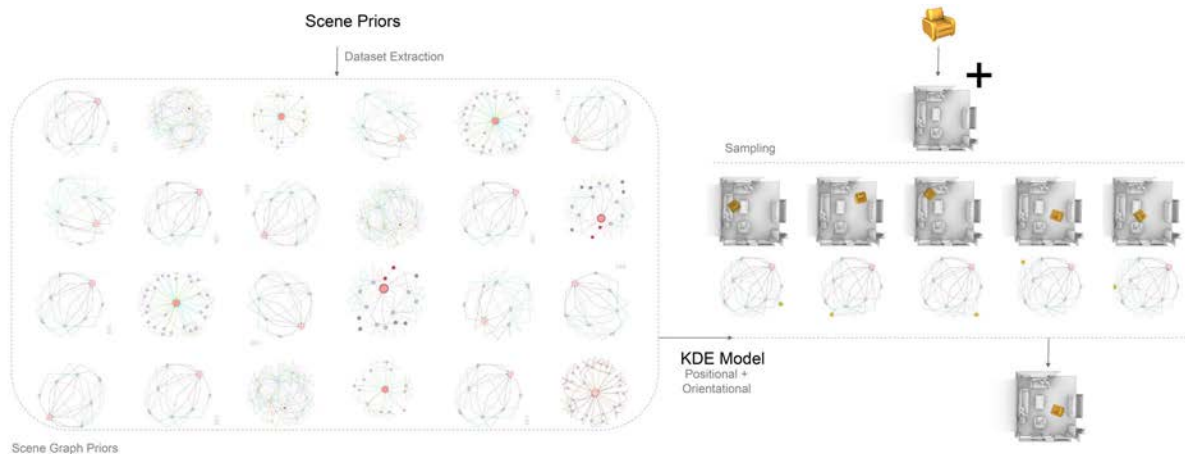
Figure 3.1: End-to-end workflow of SceneGen shows the main modules of our framework to augment rooms with virtual objects. Left: the training procedure including scene prior processing for the Knowledge Model creation. Right: the test time procedure of sampling and prediction.

how well the model can replace removed objects from a validation set. Additionally, we carry out a user study to analyze how SceneGen compares with a random placement and the reference scene in placing new objects into virtual rooms and to evaluate the value of a heat map showing the probability of all samples.

Finally, Section 4.6 details an Augmented Reality mobile application that we have developed to demonstrate the user experience when employing SceneGen to add new virtual objects. This application locally computes the semantic segmentation and generates a Scene Graph before estimating sample probabilities on an external server, and then parses and visualizes the prediction results.

3.2   Scene Representation

Graph Representation based on Extracted Features

In this section, we introduce a novel spatial Scene Graph that prepresents a room and objects in it as a graph using extracted spatial features. A Scene Graph $\mathscr{G}$ is defined by nodes representing objects, object groups, and the room, and by its edges representing the spatial relationships between the nodes. While various objects hold different individual functions (eg. a chair to sit, a table to dine, etc), their combinations and topological relationships tend to generate the main functional purpose of the space. In other words, spatial functions are created by the pairwise topologies of objects and their relationships with the room. In our proposed Scene Graph representation, we intend to explicitly extract a wide variety of positional and orientational relationships that can be
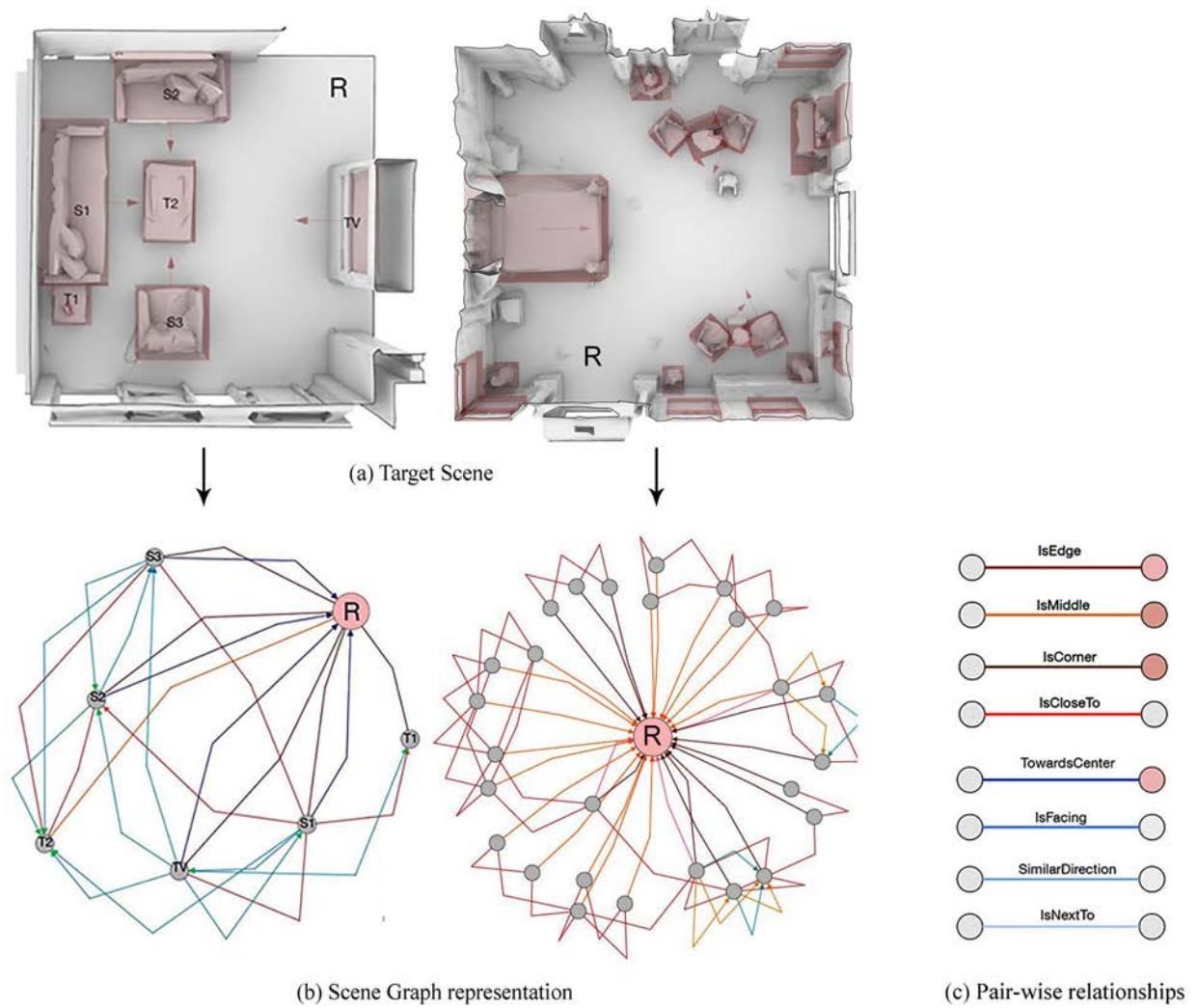
Figure 3.2: Our proposed Scene Graph representation is extracted from each scene capturing orientation and position based relationships between objects in a scene (pairwise) and between objects and the room itself. Visualization shows only a subset of features for clarity.

present between objects. We model descriptive topologies that are commonly utilized by architects and interior designers to generate spatial functionalities in a given space. Therefore, our Scene Graph representation can also be described as a function map, where objects (nodes) and their relationships (edges) correspond to a single or multiple spatial functionalities in a scene. Figure 3.2 illustrates two examples of our Scene Graph representation, where a subset of topological features are visualized in the graph.

Definitions for Room and Objects

We consider a room or a scene in 3D space where its floor is on the flat (x,y)-plane and the z-axis is orthogonal to the (x,y)-plane. In this orientation, we denote the room space in a floor-plan representation as R, namely, an orthographic projection of its 3D geometry plus a possible adjacency relationship that objects in R may overlap on the (x,y)-plane but on top of one another along the z-axis. Specifically, the "support" relationship is defined in Section 3.2. This can also be viewed as a 2.5-D representation of the space.

Further denote the k-th object (e.g., a bed or a table) in R as O_k. The collection of all n objects in R is denoted as $\mathcal{O} = \{O_1, O_2, ...O_n\}$. $B(O_k)$ represents the bounding box of the object O_k. $\dot{O}_k$ represents the center of the object O_k. Every object O_k has a label to classify its type. Related to the same R, we also have a set of groups $G = \{g_1, ..., g_m\}$, where each group g_i contains all objects of the same type within R.

Furthermore, each O_k has a primary axis a_k and a secondary axis b_k. For Frontal Facing objects, a_k represents the orientation of the object. a_k and b_k are both unit vectors such that b_k is a $\frac{\pi}{2}$ radian counter clockwise rotation of a_k. We define θ_{a_k} and θ_{b_k} to be the angle in radians represented by a_k and b_k respectively.

For each room R, we define $\mathcal{W} = \{W_1, W_2, ..., W_l\}$ where each W_k is a wall of the l-sided room. In the floor plan representation, W_k is represented by a 1D line segment. We also introduce a distance function $\delta(a,b)$ as the shortest distance between a and b objects. For example, $\delta(B(O_k), \dot{R})$ is the shortest distance between the bounding box of O_k and the center of the room R.

Positional Relationships

We first introduce features for objects based on their spatial positions in a scene. We include both pairwise relationships between objects (eg. between a chair and a desk), object groups (eg. between a dining table and dining chairs), and relationships between an object and the room.

Object to Room Relationships

RoomPosition: The room position feature of an object denotes whether an object is at the middle, edge, or corner of a room. This is based on how many walls an object is less than ρ distance from:

$$\text{RoomPosition}(O_k, R) = \sum_{W_i \in (W)} \mathbb{1}(\delta(B(O_k), W_i) < \rho). \tag{3.1}$$

In other words, if $\text{RoomPosition}(O_k, R) \geq 2$, the object is near at least 2 walls of a room, and hence is near a *corner* of the room; if $\text{RoomPosition}(O_k, R) = 1$, the object is near only one wall of the room and is at the *edge* of the room; otherwise, the object is not near any wall and is in the *middle* of the room.

Object to Object Group Relationships

AverageDistance: For each object, and each group of objects we calculate the average distance between that object and all objects within that group. For cases where the object is a member of the group, we do not count the distance between the object in question and itself in the average.

$$\text{AverageDistance}(O_k, g_i) = \sum_{\substack{O_j \in g_i \\ j \neq k}} \delta(B(O_k), B(O_j)) / \sum_{\substack{O_j \in g_i \\ j \neq k}} 1. \tag{3.2}$$

SurroundedBy: For each object, and each group of objects, we compute how many objects in the group are within a distance ε of the object. For cases where the object is a member of the group, we do not count the object in question.

$$\text{SurroundedBy}(O_k, g_i) = \sum_{\substack{O_j \in g_i \\ j \neq k}} \mathbb{1}\left(\delta\left(B(O_j), B(O_k)\right) < \varepsilon\right). \tag{3.3}$$

Object Support Relationships

Support: An object is considered to be supported by a group if is directly on top of an object from the group, or supports a group if it is directly underneath an object from the group.

$$\text{Support}(O_k, g_i) = \begin{cases} 1 & \exists O_j \in g_i \text{ where } O_k \text{ is on top of } O_j; \\ -1 & \exists O_j \in g_i \text{ where } O_k \text{ is under } O_j; \\ 0 & \text{otherwise.} \end{cases} \tag{3.4}$$

Orientation Relationships

We categorize the objects in our scenes into three main groups:

1. G_{omn}: Omnidirectional objects such as coffee tables and house plants that have no clear front-facing direction;

2. G_{frf}: Frontal Facing objects such as beds and chairs that can be oriented to face in a specific direction;

3. G_{in}: Inside Facing objects such as paintings and storage that are always facing opposite to the wall of the room where they are situated.

In this section we discuss features applicable to objects with a defined facing decisions, and not for Omnidirectional objects.

Object to Room Relationships

We first define an indicator equation that is 1 if a ray extending from the center in the direction d_k of an object intersects a wall W_i:

$$f(\dot{O}_k, d_k, W_i) = \mathbb{1}(\exists \gamma \geq 0 | \dot{O}_k + \gamma d_k \in W_i).\tag{3.5}$$

TowardsCenter: An object is considered to be facing towards the center of the room, if an ray extending from the center of the object intersects one of the furthest $\frac{l}{2}$ walls from the object:

$$
\begin{aligned}
c_1 &=_{W_i \in (W)} \delta(\dot{O}_k, W_i);\\
c_2 &=_{W_i \in (W \setminus c_1)} \delta(\dot{O}_k, W_i);\\
&\ldots\\
c_{\frac{l}{2}} &=_{W_i \in (W \setminus c_1 \ldots c_{\frac{l}{2}-1})} \delta(\dot{O}_k, W_i).
\end{aligned}
\tag{3.6}
$$

$$\text{TowardsCenter}(O_k) = f(\dot{O}_k, a_k, c_1) \vee \ldots \vee f(\dot{O}_k, a_k, c_{\frac{l}{2}-1}).\tag{3.7}$$

AwayFromWall: An object is considered facing away from a wall if it is oriented away from and is normal to the closest wall to the object:

$$
\begin{aligned}
c_1 &=_{W_i \in (W)} \delta(B(O_k), W_i);\\
\text{AwayFromWall}(O_k) &= f(\dot{O}_k, -a_k, c_1) \wedge (a_k \perp c_i).
\end{aligned}
\tag{3.8}
$$

DirectionSimilarity: An object has a similar direction as one or more objects within a constant ε distance from the object if the other objects are facing in the same direction or in the opposite direction (π radians apart) from the first object subject to some small angular error φ:

$$
\begin{aligned}
\text{Same}(O_k) &= \sum_{\substack{O_j \in \mathscr{O}, j \neq k \\ \delta(B(O_k), B(O_j)) \leq \varepsilon}} \mathbb{1}(|\theta_{a_k} - \theta_{a_j}| \leq \varphi),\\
\text{Opp}(O_k) &= \sum_{\substack{O_j \in \mathscr{O}, j \neq k \\ \delta(B(O_k), B(O_j)) \leq \varepsilon}} \mathbb{1}(|\pi - |\theta_{a_k} - \theta_{a_j}|| \leq \varphi),\\
\text{DirectionSimilarity}(O_k) &= [\text{Same}(O_k), \text{Opp}(O_k)] \in \mathbb{R}^2.
\end{aligned}
\tag{3.9}
$$

Object to Object Group Relationships

We first define an indicator function that is 1 if a ray extending from the center of the object in direction d_k intersects the bounding box of a second object:

$$h(\dot{O}_k, d_k, B(O_j)) = \mathbb{1}(\exists \gamma \geq 0 | \dot{O}_k + \gamma d_k \in B(O_j)).\tag{3.10}$$

Facing: Between an object and a group of objects we count how many objects of the group are within a distance ε of the object and are in the direction of the primary axis of the first object:

$$\text{Facing}(O_k, g_i) = \sum_{\substack{O_j \in g_i, j \neq k \\ \delta(B(O_k), B(O_j)) \leq \varepsilon}} h(\dot{O}_k, a_k, B(O_j)). \tag{3.11}$$

NextTo: Between an object and a group of object we count how many objects of the group are within a distance ε of the object and are in the direction of the positive or negative secondary axis of the first object:

$$\text{NextTo}(O_k, g_i) = \sum_{\substack{O_j \in g_i, j \neq k \\ \delta(B(O_k), B(O_j)) \leq \varepsilon}} h\left(\dot{O}_k, \pm b_k, B(O_j)\right). \tag{3.12}$$

3.3 Knowledge Model

Feature Vectors for Position and Orientation

To evaluate the plausibility of a new arrangement, we compare its corresponding Scene Graph with a population of viable Scene Graphs priors. By extracting Scene Graphs from a corpus of rooms, we construct a Knowledge Model which serves as our spatial priors for the position and orientation relationships of each object group. For each object instance, we assemble a data vector for positional features from $\mathscr{G}$. For Frontal Facing objects, we similarly create a data vector for orientational features. First we define the following that represent an object's relationships with all groups $G = \{g_1, ..., g_m\}$:

$$
\begin{aligned}
\text{AD}(O_k) &= [\text{AverageDistance}(O_k, g_i)|i = 1, \cdots, m] \in \mathbb{R}^m, \\
\text{S}(O_k) &= [\text{SurroundedBy}(O_k, g_i)|i = 1, \cdots, m] \in \mathbb{R}^m, \\
\text{F}(O_k) &= [\text{Facing}(O_k, g_i)|i = 1, \cdots, m] \in \mathbb{R}^m, \\
\text{NT}(O_k) &= [\text{NextTo}(O_k, g_i)|i = 1, \cdots, m] \in \mathbb{R}^m, \\
\text{SP}(O_k) &= [\text{Support}(O_k, g_i)|i = 1, \cdots, m] \in \mathbb{R}^m.
\end{aligned}
\tag{3.13}
$$

This allows us to construct data arrays, $d_p(O_k)$ and $d_o(O_k)$, containing features that relate to the position and orientation of an objects respectively. RoomPosition is also included in the data array for orientational features, d_o, since the other features of d_o are strongly correlated with an object's position in the room. This is abbreviated as RP. We also abbreviate TowardsCenter as TC and DirectionSimilarity as DS. For succinctness, when using these abbreviations for our features, the parameter O_k is dropped:

$$
\begin{aligned}
d_p(O_k) &= [\text{RP} \in \mathbb{R}, \text{AD} \in \mathbb{R}^m, \text{SP} \in \mathbb{R}^m, \text{S} \in \mathbb{R}^m] \in \mathbb{R}^{3m+1}, \\
d_o(O_k) &= [\text{RP} \in \mathbb{R}, \text{TC} \in \mathbb{R}, \text{DS} \in \mathbb{R}^2, \text{F} \in \mathbb{R}^m, \text{NT} \in \mathbb{R}^m] \in \mathbb{R}^{2m+4}.
\end{aligned}
\tag{3.14}
$$

Finally, given one feature vector per object for position and orientation, respectively, we can collect more samples from a database, which we will discuss in Section 3.4, to form our Knowledge

Model. The model collects feature vectors separately with respect to different object types in multiple room spaces. To do so, we introduce $g_{i,j}$ to collect all of the i-th type objects in room $R_j, j = 1, \cdots, r$. Without loss of generality, we assume that the i-th object type is the same across all rooms. Therefore, we can collect all the objects of the same i-th type from a database as

$$g_{i,*} = \bigcup_{j=1}^{r} g_{i,j}.$$

Then $D_p(g_{i,*})$ and $D_o(g_{i,*})$ represent the collections of all feature vectors in (3.14) from objects in $g_{i,*}$:

$$\begin{aligned} \mathscr{D}_p(g_{i,*}) &= \{d_p(O_k) \,|\, \forall O_k \in g_{i,*}\}, \\ \mathscr{D}_o(g_{i,*}) &= \{d_o(O_k) \,|\, \forall O_k \in g_{i,*}\}. \end{aligned} \tag{3.15}$$

Scene Augmentation

Given the feature samples for the same type of object in (3.15), now we can estimate their likelihood distribution. In particular, given an object placement O of the i-th type, we seek to estimate the likelihood function for its position features:

$$P(d_p(O)|\mathscr{D}_p(g_{i,*})). \tag{3.16}$$

If O is Frontal Facing, we also seek to estimate the likelihood function for its orientation features:

$$P(d_o(O)|\mathscr{D}_o(g_{i,*})). \tag{3.17}$$

However, if O is an Inside Facing object, then with certainty its orientation will be determined by that of its adjacent wall. Similarly, an Omnidirectional object O has no clear orientation. Therefore, for these categories of objects, estimation of their orientation likelihood is not needed. In this section, we discuss how to estimate (3.16) and (3.17)

We can approximate the shape of these distributions using multivariate kernel density estimation (KDE). Kernel density estimation is a non-parametric way to create a smooth function approximating the true distribution by summing kernel functions, K, placed at each observation $X_i...X_n$ [48]:

$$\hat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right). \tag{3.18}$$

This allows us to estimate the probability distribution function (PDF) of the position and orientation relationships from the spatial priors in our Knowledge Model, $\mathscr{D}_p(g_{i,*}), \mathscr{D}_o(g_{i,*})$ for each group g_i.

SceneGen Algorithm

Algorithm 1 describes the SceneGen algorithm. Given a room model R and a set of existing objects $\mathscr{O} = \{O_1, O_2, ...O_n\}$, the algorithm evaluates the position and orientation likelihood of augmenting a new object O' and recommends its most likely poses.

Figure 3.1 shows how potential scene graphs are created for sampled placements. For scenes where multiple objects need to be added, we repeat Algorithm 1 for each additional object.

Algorithm 1: SceneGen Algorithm

Given a training database, calculate $\mathscr{D}_p(g_{i,*})$ and $\mathscr{D}_o(g_{i,*})$ as prior.

For a given room R, construct the Scene Graph $\mathscr{G}$ of its objects $\mathscr{O}$.

while *Sample the position of O' of type i in R* **do**

    Calculate $P(d_p(O')|\mathscr{D}_p(g_{i,*}))$.

    while *Sample the orientation of $O' \in [0, 2\pi)$* **do**

        Calculate $P(d_o(O')|\mathscr{D}_o(g_{i,*}))$

    end

end

Generate a heat map displaying the likelihood distributions.

Make recommendation to place O' at the highest probability pose.

3.4 Implementation

In this section, we discuss the implementation detail of SceneGen based on the relationship data learned from the Matterport3D dataset.

Dataset

Matterport3D [6] is a large-scale RGB-D dataset containing 90 building-scale scenes. The dataset consists of various building types with diverse architecture styles, including numerous spatial functionalities and furniture layouts. Annotations of building elements and furniture have been provided with surface reconstruction as well as 2D and 3D semantic segmentation.

Pose Standardization

In order to use the Matterport3D dataset as prior for SceneGen, we must make a few modifications to standardize object orientation using an annotation tool that we have also developed. In particular, different from Section 3.2, our annotation tool interacting with the dataset is fully in 3D environment (i.e., through Unity 3D). After the annotation, the relationship data then are consolidated back to the 2.5-D representation conforming to the computation of the SceneGen models.

For each object O_k, the Matterport3D dataset supplies labeled oriented 3D bounding boxes $B(O)$ aligned to the (x, y)-plane. This is defined by a center position $\dot{O}$, primary axis a, secondary axis b, an implicit tertiary axis c, and $r \in \mathbb{R}^3$ denotes the radius vector of O. However, the Matterport3D dataset does not provide information about which labeled direction the object is facing or aligns with the z-axis. Hence, it will rely on our labeling tool to resolve the ambiguities.

To provide a consistent definition, we describe a scheme to label these axes such that the primary axis a points in the direction the object is facing as a^*. Since we know that only one of these three axes has a z component, we shall store this in the third axis c^* and define b^* to be orthogonal to a^* on the x, y plane. The box size r will also be updated to correspond to the correct axes. By
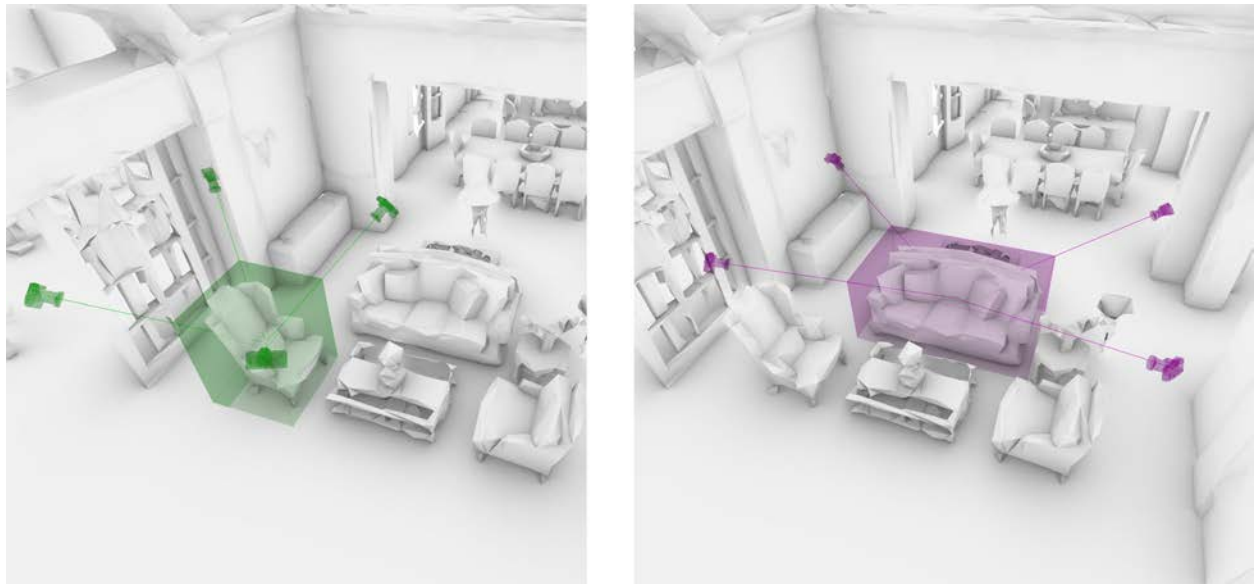
Figure 3.3: In our annotation tool, a camera is orbited around each object to facilitate labeling of object orientations.

constraining these aligned axes to be right handed, for a given a^* we have:

$$c^* \doteq [0,0,1], \quad b^* \doteq c^* \times a^*. \tag{3.19}$$

In order to correctly relabel each object, we have developed an application to facilitate the identification of the correct primary axis for all the Frontal Facing objects and supplemented this to the updated data set. For each object, we view the house model mesh at different camera positions around the bounding box in order to determine the primary axis of the object as displayed in Figure 3.3. Our annotation tool shown in Figure 3.4 allows a labeler to select from two possible directions at each camera position or can move the camera clockwise or counter clockwise to get a better view. Once a selection is made, the orienting axis a^* can be determined. We then use (3.19) to standardize the axes. Using the annotation tool, the average time for each object to be labeled is 2.6 seconds. For example, if a MatterPort3D house had 80 Frontal Facing objects that needs to be labeled, it would take an estimate of only 3.5 minutes for the annotation task of the house.

Category Reduction

For this study, we have reduced the categories of object types considered for building our model and placing new objects. Though the Matterport3D dataset includes many different types of furniture, organized with room labels to describe furniture function (e.g. "dining chair" versus "office chair"), we found that the dataset has a limited amount of instances for many object categories.
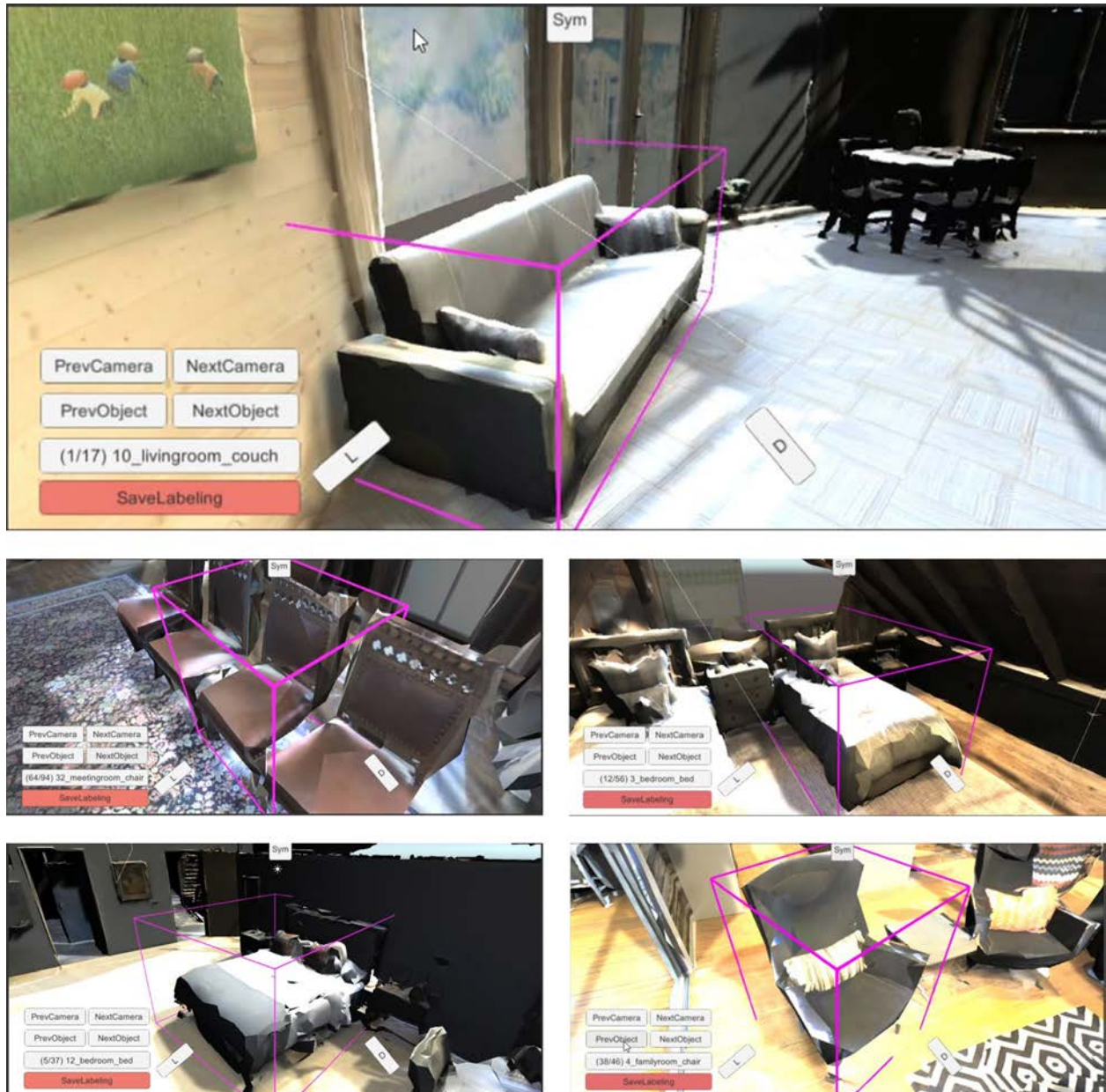
Figure 3.4: A labeler using our annotation tool can select which direction the object is facing or move to the next camera to get a better view. The selection is used to automatically standardize the axes of each object's bounding box.

Because we build statistical models for each object category, we require an adequate representation of each category. Thus, we reduce the categories to a sufficiently represented subset, and filter objects that do not fall in these categories for the purposes of this study.

We group the objects into 8 coarse categories: $G = \{$Bed, Chair, Decor, Picture, Sofa, Storage, Table, TV$\}$. Each of these categories has a specific type of orientation, as described in Section 3.2. Of these categories, Frontal Facing objects are $G_{\text{frf}} = \{$Bed, Chair, Sofa, TV$\}$, Omnidirectional objects are $G_{\text{omn}} = \{$Decor, Table$\}$, and Inside Facing objects are $G_{\text{in}} = \{$Picture, Storage$\}$.

For room types, we consider the set $\{$ library, living room, meeting room, TV room, bedroom, rec room, office, dining room, family room, kitchen, lounge$\}$ to avoid overly specialized rooms such as balconies, garages and stairs. We also filter rooms which hold more than 95% unoccupied areas to avoid unusual empty rooms that come without any spatial arrangements. After the data reduction, a total of 1,326 rooms and 7,017 objects are in our training and validation sets. The object and room categories used can be expanded if sufficient data are available.

Knowledge Model

We use the processed dataset as prior to train the SceneGen Knowledge Model. The procedure first estimates each object O_k according to (3.14), and subsequently constructs $\mathscr{D}_p(g_{i,*})$ and $\mathscr{D}_o(g_{i,*})$ in (3.15) for categories in G and G_{frf} respectively. Given our priors, we estimate the likelihood functions $P(d_p(O)|\mathscr{D}_p(g_{i,*}))$ and $P(d_o(O)|\mathscr{D}_p(g_{i,*}))$ from (3.16) and (3.17) using Kernel Density Estimation.

We utilize a KDE library developed by [45] with a normal reference rule of thumb bandwidth with ordered, discrete variable types. We make an exception for AverageDistance, which is continuous. When there are no objects of a certain group, g_i in a room, the value of AverageDistance(O_k, g_i) is set to a large constant (1000), and we use a manually tuned bandwidth (0.1) to reduce the impact of this on the rest of the distribution.

Furthermore, we found for this particular dataset, a subset of features, Facing, TowardsCenter and RoomPosition, are most impactful in predicting orientation as detailed in Section 4.4. Therefore, while we model all of the orientational features, we only use the Facing, TowardsCenter and RoomPosition features for our implementation of SceneGen and in the User Studies. Finally, due to overlapping bounding boxes in the dataset, calculating object support relationships (SP) precisely is not possible. Thus in our implementation, we allow the certain natural overlaps defined heuristically instead of using these features. A visualization of our priors from the Matterport3D dataset can be seen in Figure 3.5.

We use Algorithm 1 to augment a room R with an object of type i and generate a probability heat map. This can be repeated in order to add multiple objects. To speed up computation in this implementation, we first sample positions, and then sample orientations at the most probable position, instead of sampling orientations at every possible position.

Figure 3.6 shows how our implementation of SceneGen adds a new object to a scene and examples of scenes are augmented with multiple objects iteratively is shown in Figure 3.7. The illustrated heatmaps are color-coded based on their normalized probability rank. In positional visualizations, the top k of valid samples are rendered from zero opacity to a dark solid color (high
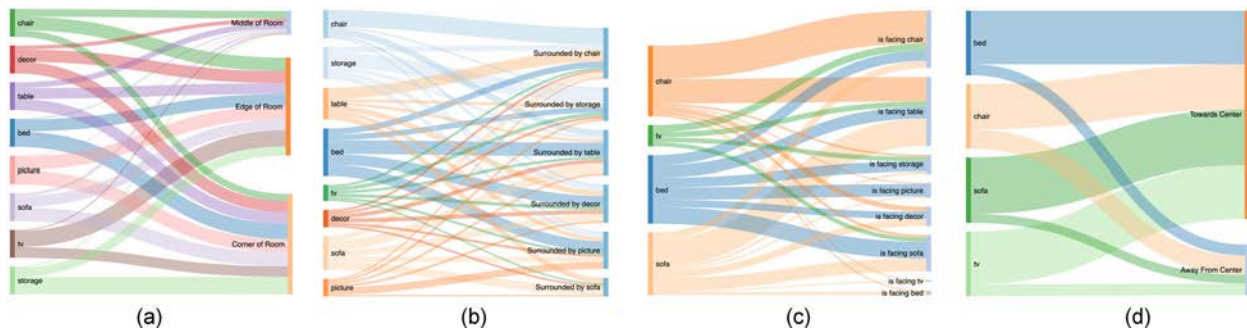
Figure 3.5: Visualization of the Knowledge Model built from Scene Graphs extracted from the Matterport3D Dataset shows for each group of objects: (a) frequency of each Room Position, (b) frequency the object is surrounded by multiple objects from another group, (c) frequency the object is facing an object from another group, (d) frequency the object is facing towards the center of the room or not.

probability), while in orientational visualizations angular samples are color-coded from red (low probability) to red (high probability).

Computation Time We train and evaluate our model using a machine with an 4-core Intel i7-4770HQ CPU and 16GB of RAM. In training, creating our Knowledge Model and estimating distributions for 8 categories of objects takes approximately 12 seconds. In testing, it takes ≈2 seconds to extract a scene graph and generate a heat map indicating the probabilities of 250 sampled poses.

Figure 3.6: Scene Gen places objects into scenes by extracting a Scene Graph from each room, sampling positions and orientations to create probability maps, and then placing an object in the most probable pose. (a) A sofa placed in a living room, (b) a bed placed in a bedroom, (c) a chair placed in an office, (d) A table placed in a family room, (e) a storage placed in a bedroom.

(a) Target scene   (b) Objects to be augmented   (c) SceneGen heatmaps   (d) SceneGen placement   (e) Dataset ground truth
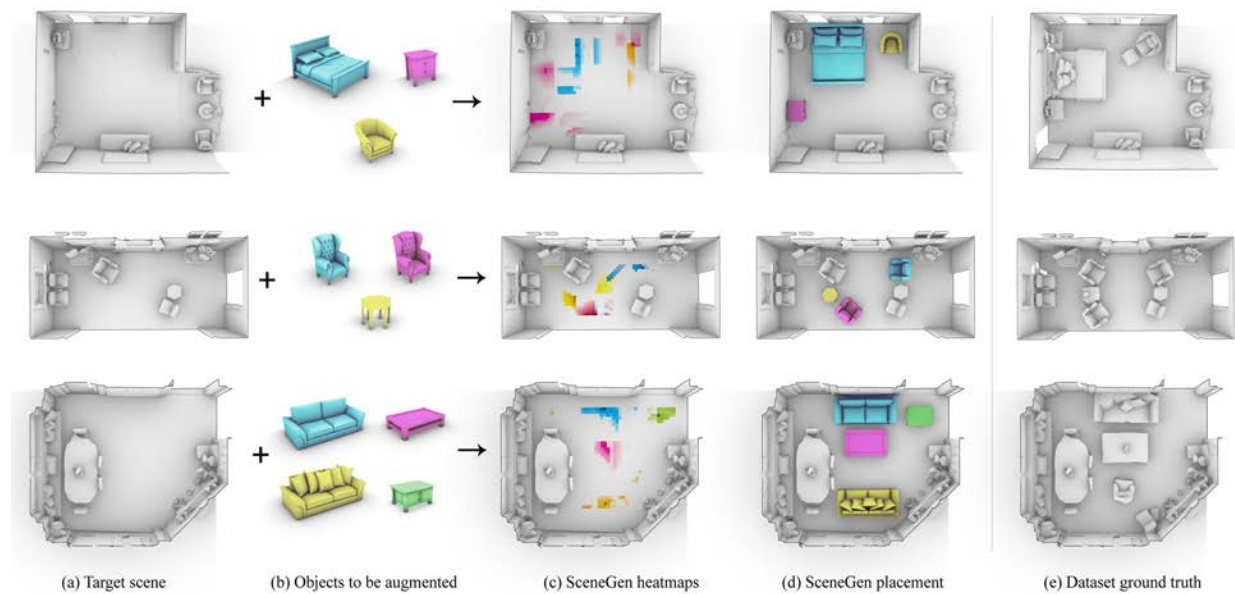
Figure 3.7: Examples of adding multiple virtual objects to a scene using SceneGen. Each object is placed in the most likely position and orientation iteratively into a partially decorated room. Top: A bed, storage, and sofa are first extracted from the room model, then reorganized in a viable alternative to the dataset ground truth; Middle: Two sofas and a table are reorganized by SceneGen to a living room in an arrangement similar to ground truth; Bottom: A sofa, a table are reorganized, and another sofa and a table are added to a family room, showing an augmented scene with new virtual objects comparing to the ground truth.

Chapter 4

Experiments

4.1   Ablation Studies

To evaluate our prediction system, we run ablation studies, examining how the presence or absence of particular features affects our object position and orientation prediction results. We use a $K = 4$-fold cross validation method on our ablation studies, with 100 rooms in each validation set and the remaining rooms in our training set.

Position Features Evaluation

The full position prediction model, SceneGen, trains three features: AverageDistance (AD), SurroundedBy (S), RoomPosition (RP). The combination is denoted as AD+S+RP. We further consider three reduced versions of our system: AD+RP, using only AverageDistance and RoomPosition features; S+RP, using only Surrounding and RoomPosition features; and RP, solely using the RoomPosition feature.

We evaluate each system using the K-fold method described above. In this study, we remove each object in the validation set, one at a time, and use our model to predict where the removed object should be positioned. The orientation of the replaced object will be the same as the original. We compute the distance between the original object location and our system's prediction.

However, as inhabitants of actual rooms, we are aware that there is often more than one plausible placement of an object, though some may be more optimal than others. Thus, we raise the question of whether there is more than one ground truth or correct answer for our object placement problem. Hence, in addition to validating our model's features, our first ablation study validates them in relation to the simple approach of taking the single highest-scored location from our system. Meanwhile, our second ablation study uses the top 5 highest-scored locations, opening up examination to multiple potential "right answers. "

Orientation Features Evaluation

We run a similar experiment to evaluate our orientation prediction models for Frontal Facing objects. Our Scene Graphs capture five relationships based on the orientation of the objects: Facing (F), TowardsCenter (C), NextTo (NT), DirectionSimilarity (DS), and RoomPosition (RP). We assess models based on several combinations of these relationships.

We evaluate each of these models using the same *K*-fold approach, removing the orientation information of each object in the validation set, and then using our system to predict the best orientation, keeping the object's position constant. We compare the angular difference between the predicted and the original orientations.

4.2   Comparative Studies

We compare the performance system of our system with SceneGraphNet [70] in both quantitative and qualitative experiments. The experiments are similar to the *K*-fold ablation study trained on the MatterPort3D dataset mentioned in the previous section, where we remove each object in the validation set dataset, and compare the model's ability to predict where the removed object should be positioned. However, due to the fact that SceneGraphNet does not predict orientations of the object placement, we only limit the experiment to positional calculations only. In our qualitative results, we used the original orientation of the ground truth as a basis for SceneGraphNet's augmentation.

SceneGraphNet utilizes a neural message passing approach to the scene synthesis problem. However, the system is primarily designed to predict the probability distribution over the type of objects given a query position in a scene. To get the probability distribution for placement of a specific object across a scene, their code was augmented with a sampling mechanism similar to SceneGen to evaluate all possible coordinates which fit in the room. Next, each coordinate was fed as an input to the SceneGraphNet procedure to predicts the probability distribution for all categories at that position. Using this exhaustive search approach, we are able to calculate the most probable location to place an specific object category in the scene.

4.3   User Evaluation

We conduct user studies with a designed 3D application based on our prediction system to evaluate the plausibility of our predicted positions and the usefulness of our heat map system. We recruited 40 participants, of which 8 were trained architects. To ensure unbiased results, the participants were randomly divided into 4 groups. Each group of users were shown 5 scenes from each of the 5 levels for a total of 25 scenes. The order these scenes were presented in was randomized for each user and they were not told which level a scene was at.

We reconstructed 34 3D scenes from our dataset test split, where each scene had one object randomly removed. In this reconstruction process, we performed some simplification and regularized the furniture designs using prefabricated libraries, so that users would evaluate the layout of the

room, rather than the design of the object itself, while matching the placement and size of each object. An example of this scene reconstruction and simplification can be seen in Figure 4.5(a-b).

The five defined levels test different object placement methods as shown in Figure 4.5(c-g) to replace the removed object. Levels I and II are both random placements, generated at run time for each user. The Level I system initially places the object in a random position and orientation in the scene. The Level II system places the object in an open random position and orientation, where the placement does not overlap with the room walls or other objects. Levels III and IV use SceneGen predictions. The Level III system places the object in the position and orientation predicted by SceneGen. Level IV also places the object in the predicted position and orientation, but also overlays a probability map. The Level V system places the object at the position it appears in the Matterport3D dataset, i.e., the ground truth.

We recorded the users' Likert rating of the plausibility of the initial object placement on a scale of 1 to 5 (1 = implausible/random,3 = somewhat plausible, 5 = very plausible). We also recorded whether the user chose to adjust the initial placement, the Euclidean distance between the initial placement and the final user-chosen placement, the orientation change between the initial orientation and the final user-chosen orientation. No comparison was made between the scenes by the participants and the scores were to be given to each scene independent to another. We expect higher initial Likert ratings and smaller adjustments to position and orientation for levels initialized by our system than for levels initialized to random positions.

Each participant used an executable application on a desktop computer. The goal of the study was explained to the user and they were shown a demonstration of how to use the interface. For each scene, the user was shown a 3D room and an object that was removed. After inspecting the initial scene and clicking "place object," the object was placed in the scene using the method corresponding to the level of the scene. In Level IV Scenes, the probability heat map was also visualized. Note that participants were not aware whether Level IV was generating results from the our system, random, or ground truth. The user was shown multiple camera angles and was able to pan, zoom and orbit around the 3D room to evaluate the placement.

For each scene, the user was first asked to rate the plausibility of placement on a Likert Scale from 1-5. Following this, the user was asked if they wanted to move the object to a new location. If they answered "no," the user would progress to the next scene. If they answered "yes," the UI displayed transformation control handles (position axis arrows, rotation axis circles) to object position and orientation. After moving the object to the desired location, the user could save the placement and progress to the next scene. An IRB approval was maintained ahead of the experiment.

4.4 Ablation & Comparative Study Results

Position Features

In Figure 4.1, we plot the cumulative distance between the ground truth position and the top position prediction, and in Figure 4.2, we plot the cumulative distance between the ground truth
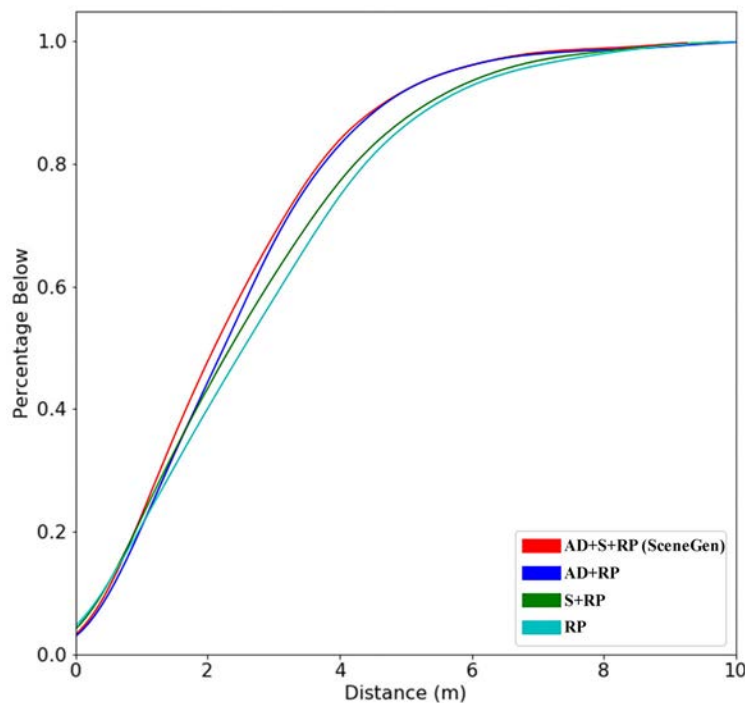
Figure 4.1: Distance between a ground truth object's position and where SceneGen and other ablated versions of our system predict the object should be re-positioned is shown in a cumulative density plot.

position and the nearest out of the the top 5 position predictions, using our full system and three ablated versions.

We find that the full SceneGen system predicts a placement most similar to ground truth than any of the ablated versions, followed by the models using AverageDist and RoomPosition features (AD+RP), and SurroundedBy and RoomPosition (S+RP). The predictions furthest from the ground truth are generated by only using the RoomPosition (RP) feature. These curves are consistent between the best and the closest of the top 5 predicted positions, and the fact indicates that each of our features for position prediction contributes to the accuracy of the final result.

In addition, when the top 5 predictions are considered, we see that each system we assess is able to identify high probability zones closer to the ground truth compared to only using the best prediction. This is supported by the slope of the curves in Figure 4.2, which rise much more sharply than in Figure 4.1. This difference provides support for the importance of predicting multiple locations instead of simply returning the highest-scored locations. A room can contain multiple plausible locations for a new object, so the system's predicted location with the highest score may
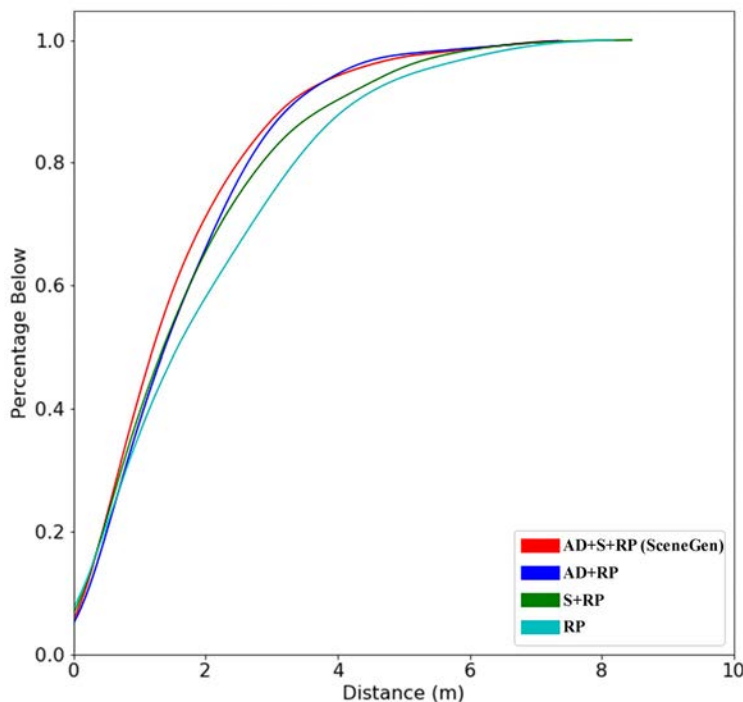
Figure 4.2: Distance between the ground truth object's position and the nearest of the 5 highest probability positions predicted by SceneGen and other ablated versions of our system is shown in a cumulative density plot.

not necessarily be same as the ground truth. For this reason, our system returns probabilities across sampled positions using a heat map to show multiple viable predictions for any placement query.

Table 4.1 shows the mean distance of the position prediction to ground truth position separated by object categories in all SceneGen ablation and also SceneGraphNet. We find that the object categories where the full SceneGen system outperforms its ablations are chairs, storage, and decor. For beds and TVs, SceneGen only produces the closest placements out of the system versions when considering the top five predictions. For pictures and tables, SceneGen's top prediction is closest to ground truth, and is only slightly further when comparing the nearest of the top 5 predictions.

Furthermore, our results indicate SceneGen outperforming SceneGraphNet in all categories in the positional placement experiment. Figure 4.3 illustrates a qualitative comparison for the object augmentation tasks between the two systems on MatterPort3D scenes. While both systems show their ability to predict plausible placement in relation to other objects in the target scene, we observe a slightly better performance in SceneGen when taking into account the object's position in relation to the room.

Table 4.1: Distance between ground truth and predicted position for different models, with smallest distances for each object type in bold (ablation study). Topology features are abbreviated as follows: AverageDistance as AD, SurroundedBy as S, and RoomPosition as RP.

System	*Bed*		*Chair*		*Storage*		*Decor*		*Picture*		*Table*		*Sofa*		*TV*		*Overall*	
	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5
AD+S+RP (SceneGen)	1.58	**0.87**	**2.26**	**1.35**	**2.27**	**1.45**	**2.71**	**1.71**	**2.80**	1.99	**2.15**	1.47	2.56	1.58	2.49	**1.52**	**2.40**	**1.54**
AD + RP	**1.40**	0.95	2.40	1.47	2.55	1.67	2.79	1.96	2.95	2.03	2.26	**1.46**	2.58	**1.58**	**2.39**	1.731	2.49	1.65
S + RP	1.85	1.32	2.46	1.56	2.46	1.64	3.38	2.14	2.82	**1.92**	2.67	1.72	**2.53**	1.64	2.51	1.55	2.67	1.73
RP	1.99	1.31	2.95	2.31	2.75	1.53	3.12	2.56	2.95	2.21	2.70	1.57	2.55	1.72	2.95	2.32	2.80	1.96
SceneGraphNet [70]	1.91	1.56	3.01	2.49	2.37	1.95	3.14	2.70	3.36	2.94	3.80	3.31	3.57	3.12	3.97	3.40	3.25	2.80

Table 4.2: Angular difference in radians between ground truth and predicted orientation for different model architectures (ablation study). Topology features are abbreviated as follows: Facing as F, TowardsCenter as C, RoomPosition as (RP), NextTo as NT, DirectionSimilarity as DS.

System	*Bed*	*Chair*	*Sofa*	*TV*	*Overall*
F+C+RP (SceneGen)	**0.65**	**0.98**	**0.67**	0.66	**0.85**
F only	1.13	1.66	1.51	0.91	1.54
F+C	1.13	1.55	1.18	0.49	1.35
F+C+NT	1.18	1.53	1.23	**0.46**	1.35
F+C+DS	1.54	1.55	1.21	0.59	1.39
F+C+DS+NT	1.22	1.50	1.23	0.63	1.35

Orientation Features Results

In our orientational ablation studies, we assess the ability of various versions of our model to reorient Frontal Facing objects from test scenes. In Figure 4.4, we plot the angular difference between the ground truth orientation and the top orientation prediction from various versions of our system. The base model includes only Facing (F), and is the lowest performing. We find that the system that also includes TowardsCenter and RoomPosition features performs best overall. We use this system (F+C+RP) in our implementation of SceneGen. The other four versions of our system perform similarly to each other overall.

Table 4.2 shows the results of the orientation ablation study separated by object category. In this case, the system with Facing, TowardsCenter and RoomPosition features (F+C+RP) outperforms all other versions across on all categories except for TVs, where the system that includes Facing, TowardsCenter and NextTo (F+C+NT) produces the least deviation. In fact, all three of the systems that included either DirectionSimilarity or NextTo, predict the orientation of TVs more closely than the overall best performing system, but perform more poorly on other objects such as beds when compared with systems without those features. This suggests that for other datasets, these features could be more effective in prediction orientations.

(a) SceneGraphNet   b) SceneGen (Ours)
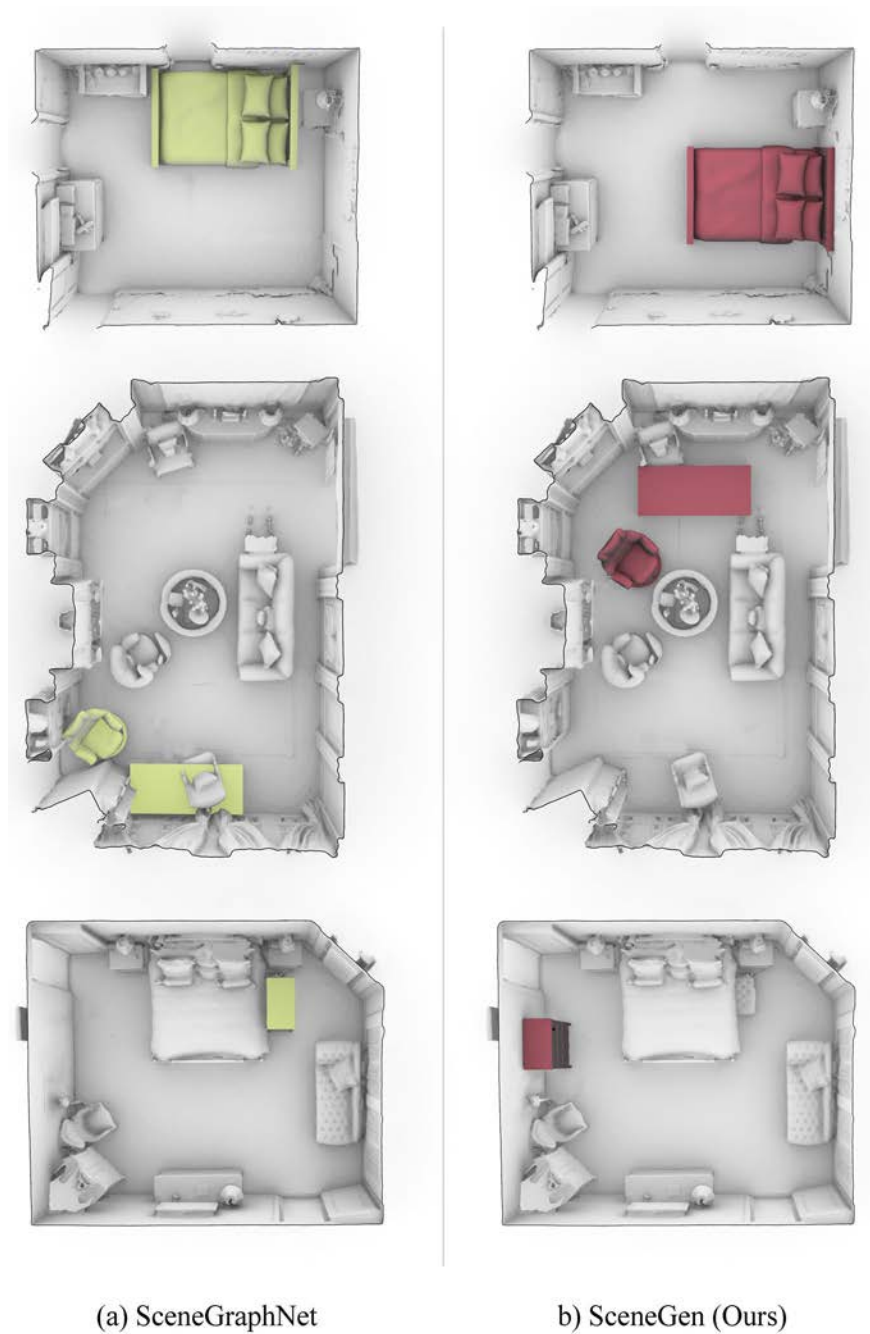
Figure 4.3: Comparison between SceneGraphNet [70] (left/yellow) and our proposed system (right/red) for the scene augmentation task on example MatterPort3D scenes. Objects are removed and augmented back into the scene via the constrained scene augmentation models. Illustration includes augmentation comparison of a bed (top), sofa+ table (middle) in an office, and a storage (bottom).
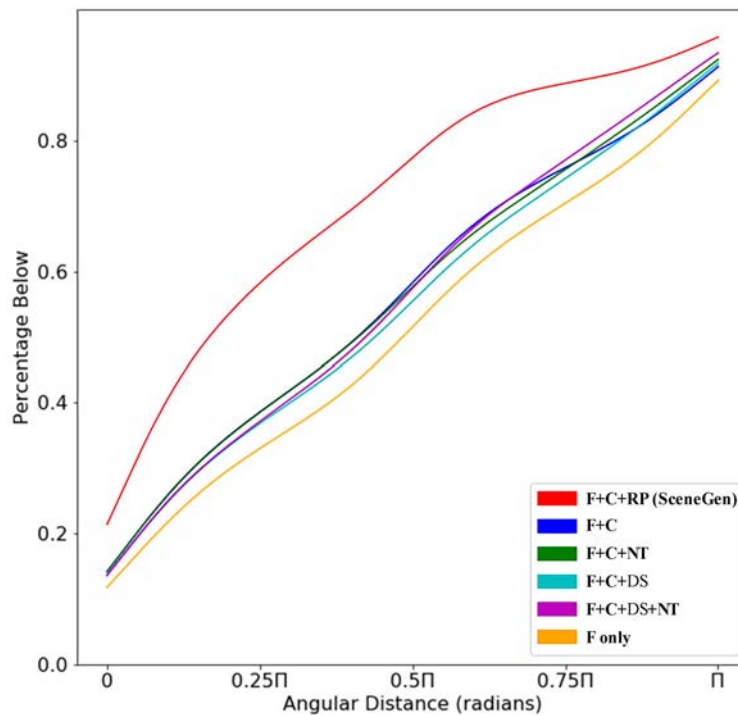
Figure 4.4: Cumulative density plot indicates angular difference between ground truth orientation and our system's predicted orientation for SceneGen and other subsets of orientation features. The range is $[0, \pi)$.

4.5   User Study Results

Plausibility of Placement Results

We show the distributions of Likert ratings by levels in Figure 4.6. We also run a one-way ANOVA test on the Likert ratings of initial placements, finding significant differences between all pairs of levels except for Levels IV and V. In other words, the ratings for Level IV's representation of our prediction system are not significantly different from ground truth placements. Across multiple tests, we see that Level IV result ratings are significantly different from levels based on randomization, while those from Level III are not as significant. The difference between Levels III and IV could support our conjecture that accounting for multiple "right answer" placements improves the predictions.

(a) Original MatterPort3D scene

USER STUDY

(b) Reconstructed simplified scene

(c) Level 1 (Random)

(c) Level 2 (Open Random)

(e) Level 3 (SceneGen)

(f) Level 4 (SceneGen + heatmap)

(g) Level 5 (Ground Truth)
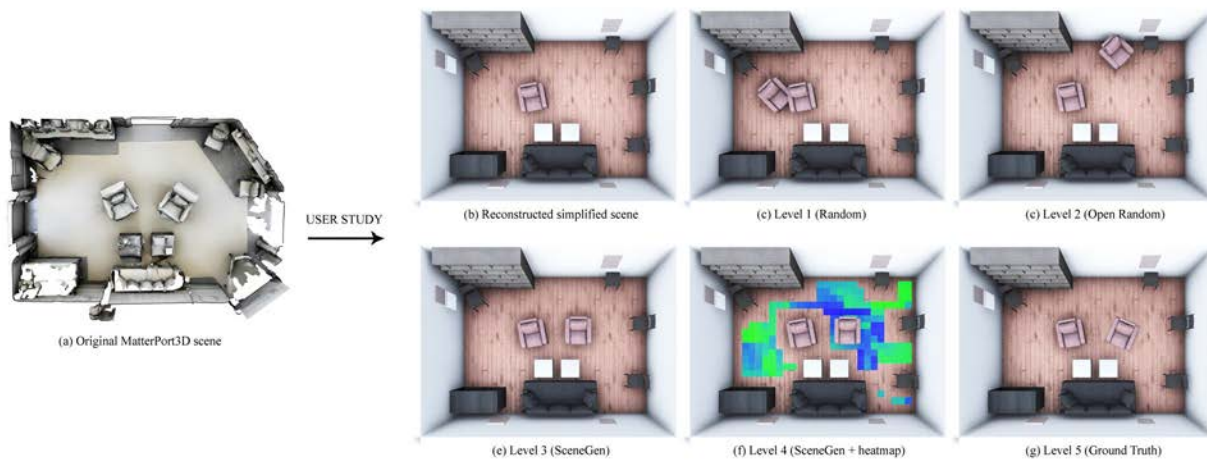
Figure 4.5: Users are shown scene models that are simplified based on original Matterport3D rooms. An object is reorganized using each of five levels of the systems. Level I places the object randomly in the room. Level II replaces the object randomly at an open space. Levels III and IV use SceneGen to predict the most likely placement and orientation, and Level IV further shows a heat map visualizing the underlying probability score at each sampled position. In Level V, the user sees the original placement in the ground truth . When providing scores during experiment, the user has multiple camera angles available and is able to pan, zoom, and orbit around the room to evaluate the placement.

Position Prediction Results

We analyze how participants' choices to adjust placement vary across different scene levels. Results of this can be seen in Figure 4.7. A one-way ANOVA test of the distance users moving objects from its placements finds a significant difference ($p = 1.8622e^{38}$) between two groupings of levels: 1) Levels I and II (with higher means), and 2) Levels III, IV, and V (with lower means). The differentiation in groupings supports the plausibility of our system's position prediction over random placements.

Orientation Prediction Results

A one-way ANOVA test is also performed on the change in object orientation from the participants' manual adjustment, and finds a significant difference ($p = 1.8112e^{16}$) between a different pair of level groupings: 1) Levels I, II, and III, and 2) Levels IV and V. In Figure 4.8, we show the distributions of angular difference between the initial object orientation and the final user-chosen orientation, for each level. The levels IV and V have distributions are most concentrated at no rotation by the user. In Levels I and II, the users rotate objects more than half of the time, with
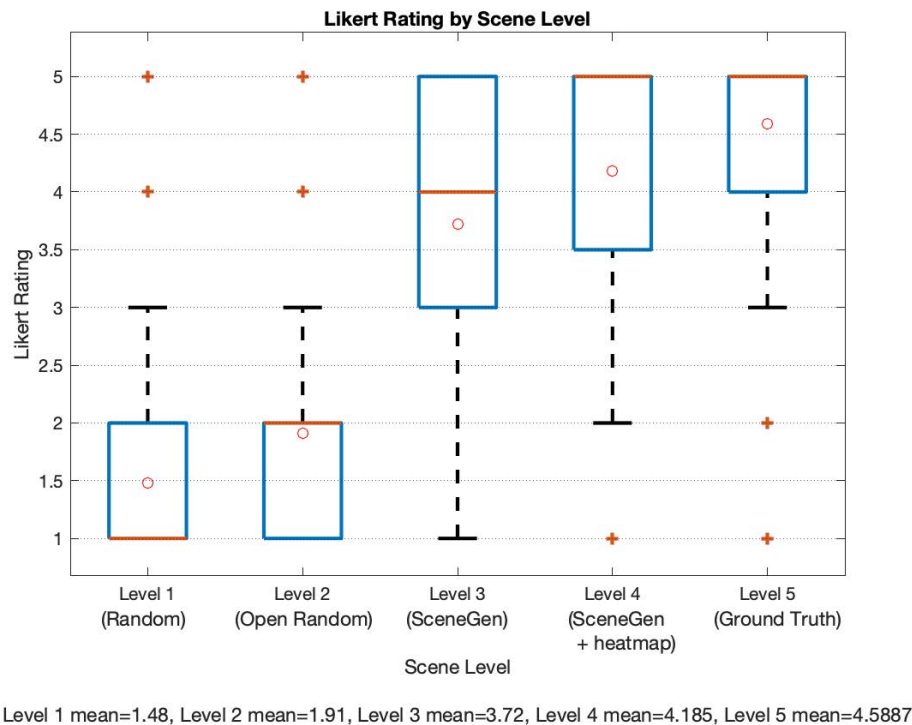
Figure 4.6: Users rank the plausibility of object placement averaged on the Likert Scale from 1 to 5. (1= Implausible/ Random, 3= Somewhat Plausible, 5 = Very Plausible). Scores are displayed in a box plot separated by the user study level.

an average rotation greater than $\frac{\pi}{6}$ radians. A vast majority of objects placed by Levels III, IV, V systems are not rotated by the user, lending support to the validity of our prediction system.

4.6 Augmented Reality Application

To demonstrate a way to integrate our prediction system in action, we have implemented an augmented reality application that augments a scene using SceneGen. Users can overlay bounding boxes over the existing furniture to see the object bounds used in our predictions. On inserting a new object to the scene, the user can visualize a portability map to observe potential positions. Our Augmented Reality application consists of five main modules: (i) local semantic segmentation of the room; (ii) local Scene Graph generation (iii) heat map generation which is developed on an external server (iv) local data parsing and visualization; and finally (v) the user interface. A brief demonstration of the AR application's interface and workflow is shown in Figure 4.9.
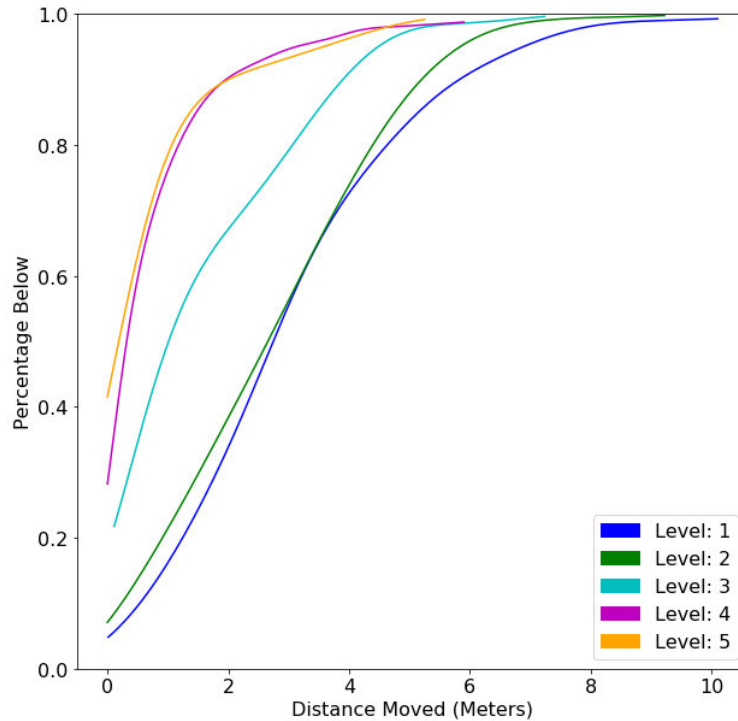
Figure 4.7: Cumulative density plot indicates the distance an object is moved from its predicted placement in each level by users.

Semantic segmentation of the room can be done either manually or automatically, using integrated tools available on augmented reality devices. However, as not all current AR devices are equipped with depth-sensing capturing hardware, we use techniques previously introduced by [43], allowing the user themselves to manually generate and annotate semantic bounding boxes of objects of the target scene. The data acquired are then converted to our proposed spatial Scene Graph, resulting in an abstract representation of the scene. Both semantic segmentation and graph generation modules are performed locally on the AR devices, ensuring the privacy of the raw spatial data of the user.

Once the Scene Graph is generated, it is sent to a remote server where SceneGen engine can calculate positional and orientation augmentation probability maps for all object categories for the target scene. Such approach would allow faster computation time, since current AR devices come with limited computational and memory resources. The results are sent back to the local device, in which can be parsed and visualized using the Augmented Reality GUI. For simplicity, we limit the positional sampling to be only on the floor. However, based on the object category, if an object is already present on that location, the probability map will render on top of the object to maintain a

Level 1
Random

N = 1 , avgAng = 46.71°±9.16°, r = 0.5654
p_{Rayl} = 0.000, Z_{Rayl} = 63.9259, $p_{C. an.}$ = 0.161,

Level 2
Open Random

N = 1 , avgAng = 38.51°±8.67°, r = 0.5919
p_{Rayl} = 0.000, Z_{Rayl} = 70.0664, $p_{C. an.}$ = 0.209,

Level 3
SceneGen

N = 1 , avgAng = 12.75°±6.86°, r = 0.7109
p_{Rayl} = 0.000, Z_{Rayl} = 101.0624, $p_{C. an.}$ = 0.395

Level 4
SceneGen + heatmap

N = 1 , avgAng = 7.57°±3.70°, r = 0.9072
p_{Rayl} = 0.000, Z_{Rayl} = 164.5951, $p_{C. an.}$ = 0.275

Level 5
Ground truth

N = 1 , avgAng = 2.24°±2.74°, r = 0.9381
p_{Rayl} = 0.000, Z_{Rayl} = 203.2781, $p_{C. an.}$ = 0.350,

Figure 4.8: Radial histograms display distribution of how much a user rotates an object from its orientation in each level of the user study.
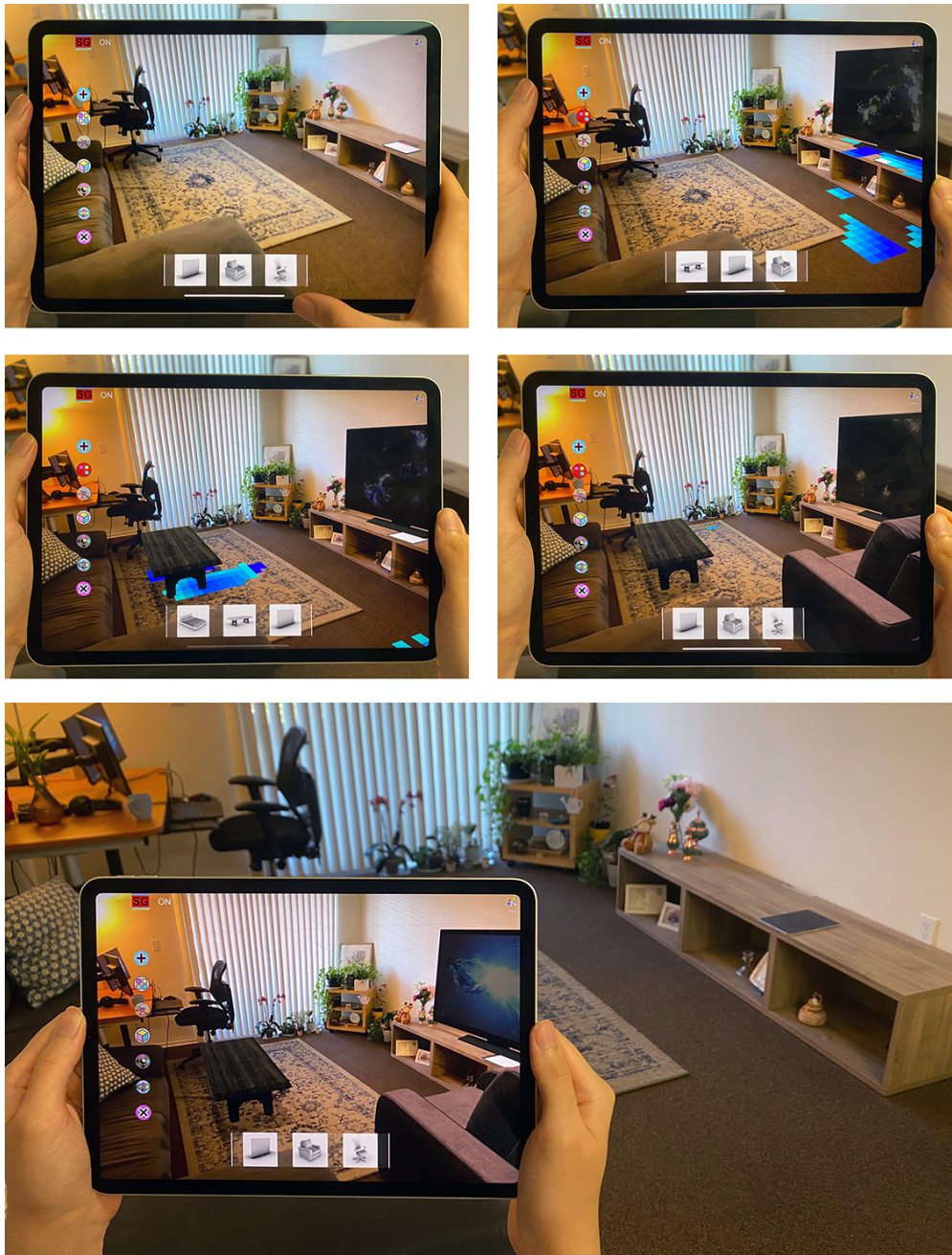
Figure 4.9: Augmented Reality application demonstrates how SceneGen can be used to add virtual objects to a scene. Top Left: the target scene, Top Right: adding a TV, Middle Left: adding a table, Middle Right: adding a sofa. A probability map displays how likely each position is. Bottom: the AR application with virtual objects is compared to the original scene.

clearer visualization of the probability map.

The instantiation system can toggle between two modes: *Manual* and *SceneGen*. In Manual mode, the object is placed in front of the user, on the intersection of the camera front-facing vector direction with the floor. This would normally result in augmenting the object in the middle of the screen. While such conventional approach allows the user control the initial placement by determining the pose of the AR camera, in many cases additional movements are necessary to place the object in a plausible final location. In such cases, the user can then further move and rotate the objects to its desirable location. In SceneGen mode, the virtual object is augmented using the prediction of our system, resulting in faster and contextual placements.

Chapter 5

Discussion and Conclusions

5.1   Discussion

Features and Predictions

The Scene Graph we introduce in this paper is designed to capture spatial relationships between objects, object categories, and the room. Overall, we have found that each of the relationships we have presented improves the SceneGen algorithm's ability to augment virtual objects in realistic placements in a scene. These relationships are important to understand the functional purposes of the space in addition to the individual objects. Our approach can also be extended to predicting the best category of new object type to augment by running an exhaustive search on all the categories for a given input room coordinate.

In SceneGen, RoomPosition is used as a feature in predicting both orientation and position of an object. While this is a feature based solely on the position of the object, it also has a strong impact on how it should be oriented. For example, a chair in a corner of the room is very likely to face towards the center of the room, while a chair in the middle of the room is more likely to face towards a table or a sofa. When analyzing our placement predictions probability maps and our user study results, we have observed that the best orientation is not only affected by the nearby objects but also by the sampled position within the room.

Explicit Knowledge Model

In our evaluation of SceneGen, we have found a number of benefits in using an explicit model to predict object placements. One benefit is that if we want to define a non-standard object to be placed in relation with standard objects by specifying your own relationship distributions, it is feasible with our system but would not be possible for implicit models. For example, in a collaborative virtual environment, where special markers are desired to be placed near each user, one could directly specify distributions for relationships such as NextTo chair and Facing table without retraining an implicit model such as neural networks.

Another benefit is that explicit models can be easily examined directly to understand why objects are being placed where they are. For example, the Bed orientation feature distribution, based on the Matterport3D priors in Figure 3.5, marginalized with respect to all other variables except TowardsCenter show that beds are nearly 5 times as likely to face the center of the room, while marginalizing features except position of the Storage show that a storage is found in a corner of a room 63% of the time, along an edge 33% of the time, and only in the middle of the room in 4% of occurrences.

Dataset

One important consideration in our choice of dataset is that we aim to learn spatial relationships for real world scenes. One can imagine idiosyncrasies of lived-in rooms, such as an office chair that is not always tucked into a desk but often left rotated away from it or a dining table pushed into a wall to create more space in a family room. Using personal living spaces, from the Matterport3D dataset, as our priors, we can capture these relationships that exist only in real world, lived-in scenes.

One drawback of using the Matterport3D dataset is that it is not as large as some synthetic datasets. For instance, the SUNCG [51] synthetic dataset, which was unavailable during the course of our study holds more than 45,000 environments, while our model was trained on only 1,326 rooms from the Matterport3D dataset. In our implementation, we were forced to group objects into broader groups to ensure adequate representation to ensure that all object categories are represented well enough to approximate the distribution of a large feature space. A larger dataset would have allowed us to model more diverse object categories in a data-driven approach.

Another downside of using a real-world dataset is its accuracy in labeling where many human errors occur in this labour intensive process. Such mismatches are unlikely to happen in synthetic datasets as the geometry is already assigned in a digital format. To mitigate some of these concerns, we have developed a labeling application that allows us to determine the correct orientation of each objects, and also filter out rooms with corrupted scans and inaccurate labeling.

Subjectivity of Placements

Where and how an object is placed in a scene is often very subjective and preferences can differ between users. This is demonstrated by the Likert scale plausibility ratings in Level V reference scenes in the user studies. Figures 4.6 and 5.2 show that some users would only give scores of *somewhat plausible* to scenes that are modelled from real world ground truth Matterport3D rooms. This supports providing a heat map of probabilities for each sampled placement, as alternate high probability positions may be more preferable to different users. Our results also indicate that most users prefer level IV scenes, with the heat map, compared to level III scenes, even though the placements use the same SceneGen models. This suggests that the inclusion of the heat map guides the users towards the system's placement and may help in convincing them of the viability and reasoning for such a choice.
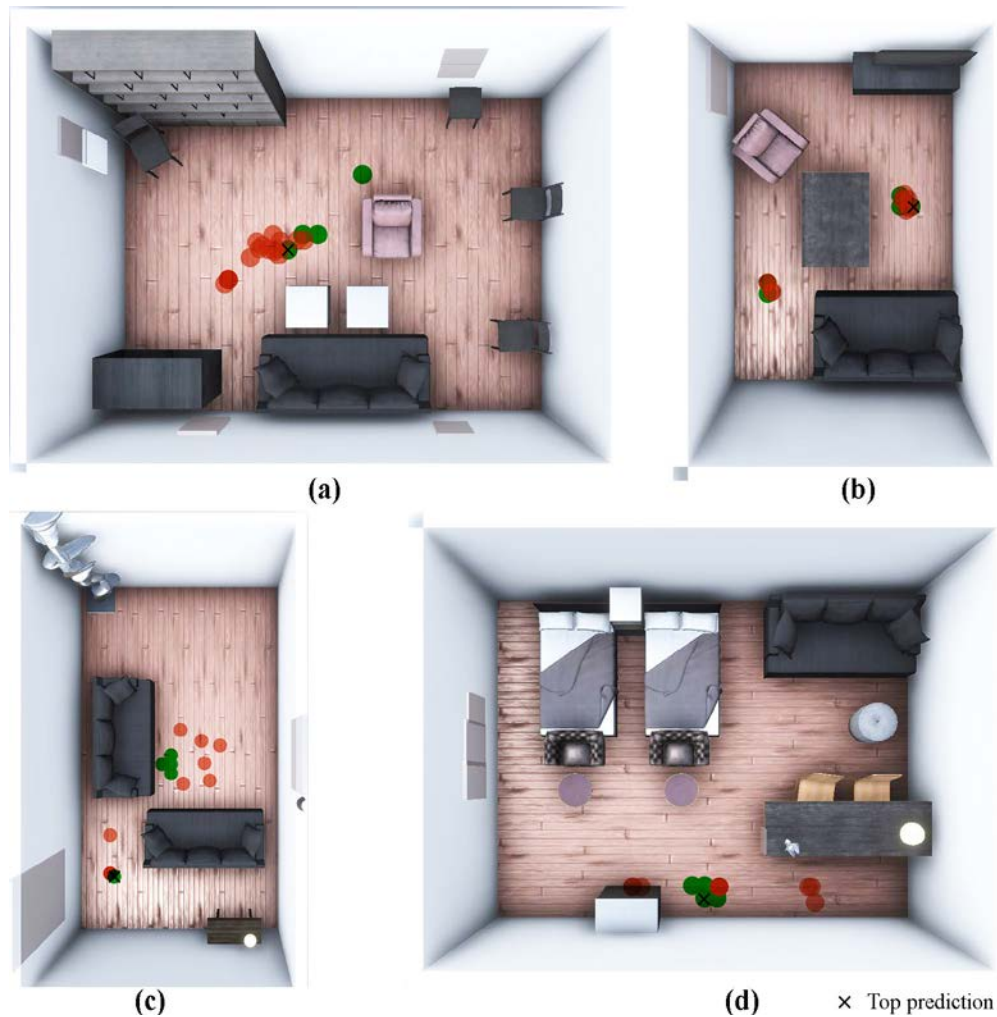
Figure 5.1: Top 5 highest probability positions for placing sofa (a,b), table (c) and TV (d) predicted by SceneGen (green) are compared to the user placements (red) showing that different users's preferences do vary and SceneGen find the clusters as the users' best consensus.

We also see that some users move objects to other high probability alternatives as seen in Figure 5.1. This is a similar result to the position prediction experiment, which compares the ground truth position to the closest of SceneGen's top 5 predictions and shows that while the reference position may not always be the top prediction, it was often one of the top predictions. Moreover, results in Figure 5.2 show the subjectivity of an object placement is highly dependent on the size and type of object itself. In any room, there are very few natural places to put a bed. Hence the results for placing beds cluster in one or two high probability locations. Other objects such as decor are more

likely to be subject to user preferences.
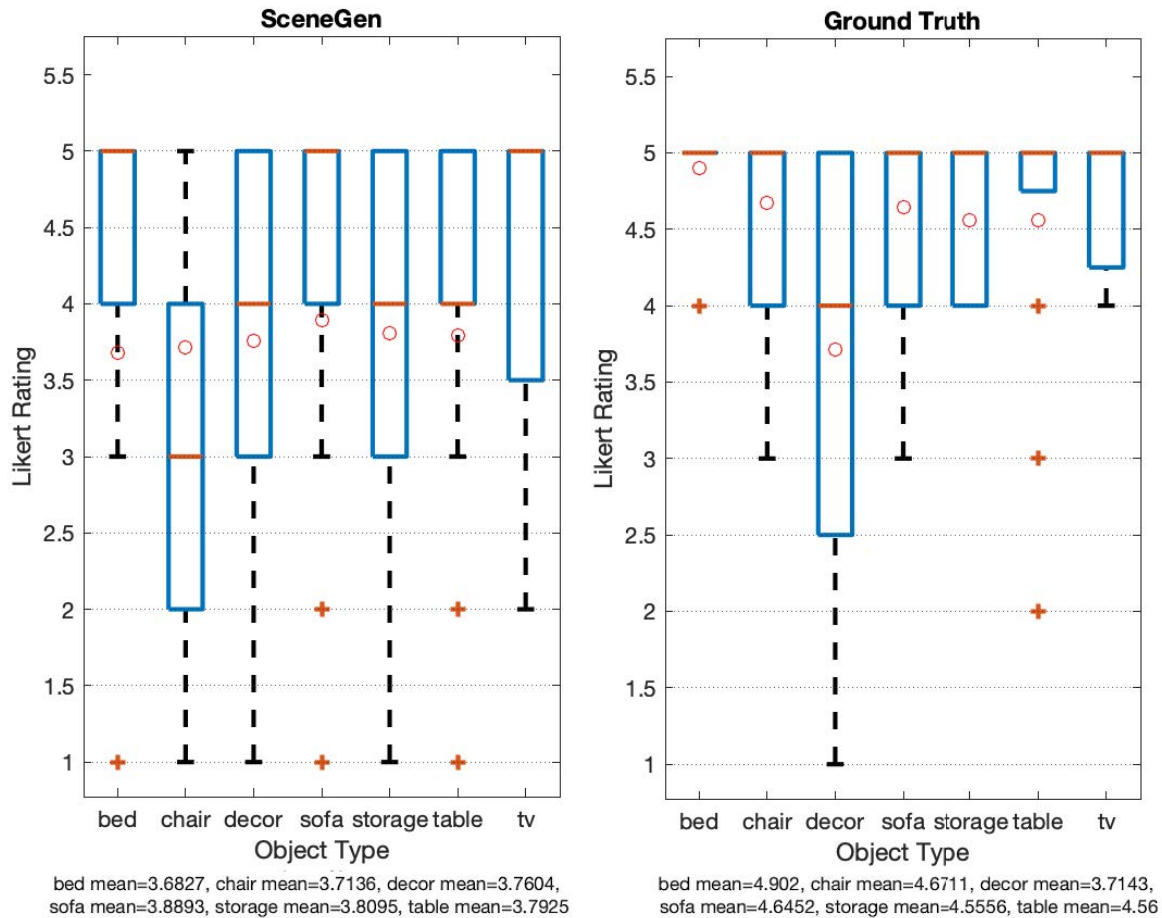
Figure 5.2: The plausibility score for each object category on the Likert Scale given by users is compared between the average scores from SceneGen Levels III and IV (left) and the ground truth Level V (right).

5.2 Conclusion

In this paper we introduce a framework to augment scenes with one or more virtual objects using an explicit generative model trained on spatial relationship priors. Scene Graphs from a dataset of scenes are aggregated into a Knowledge Model and used to train a probabilistic model. This explicit model allows for direct analysis of the learned priors and allows for users to input custom

relationships to place non-standard objects alongside traditional objects. SceneGen places the object in the highest probability pose and also offers alternate highly likely placements.

We implement SceneGen using the Matterport3D, a dataset composed of 3D scans of lived-in rooms, in order to understand object relationships in a real world setting. The features that SceneGen extracts to build our Scene Graph are assessed through an ablation study, identifying how each feature contributes to our model's ability to predict realistic object placements. User Studies also demonstrate that SceneGen is able to augment scenes in a much more plausible way than system that places objects randomly or in open spaces. We also found that different users have their own preferences for where an object should be placed. Suggesting multiple high probability possibilities through a heat map allows users an intuitive visualization of the augmentation process.

There are of course, limitations to our work. While SceneGen is able to iteratively add objects to a scene, the resulting layout is dependent on the order in which objects are placed. Such approach does not consider all possible permutations of the possible arrangements. In addition, it can narrow down the open possible spaces for later objects, forcing placements that are far from optimal. Moreover, in scenarios where a large number of objects are to be augment, the current approach may not have the ability to *fit* all the objects within the usable space, as initial placements are not aware of upcoming objects. Future work can comprise of incorporating floorplanning methodologies with the current sampling mechanism allowing a robust search in the solution space, while addressing combinatorial arrangement.

Moreover, SceneGen is a framework that naturally fits into spatial computing applications. We demonstrate this in a augmented reality application that augments a scene with a virtual object using SceneGen. Contextual scene augmentation can be useful in augmenting collaborative mixed reality environments or in other design applications, and using this framework allows for fast and realistic scene and content generation. We plan on improving our framework in providing the option to contextually augment non-standard objects by parameterizing topological relationships, a feature that would facilitate content creation for future spatial computing workflows.

Bibliography

[1] Armen Avetisyan et al. "Scan2cad: Learning cad model alignment in rgb-d scans". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2614–2623.

[2] J Timothy Balint and Rafael Bidarra. "A generalized semantic representation for procedural generation of rooms". In: *Proceedings of the 14th International Conference on the Foundations of Digital Games*. ACM. 2019, p. 85.

[3] Richard W Bukowski and Carlo H Séquin. "Object associations: a simple and practical approach to virtual 3D manipulation". In: *Proceedings of the 1995 symposium on Interactive 3D graphics*. 1995, 131–ff.

[4] Angel Chang, Manolis Savva, and Christopher D Manning. "Interactive learning of spatial knowledge for text to 3D scene generation". In: *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*. 2014, pp. 14–21.

[5] Angel Chang, Manolis Savva, and Christopher D Manning. "Learning spatial knowledge for text to 3D scene generation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 2028–2038.

[6] Angel Chang et al. "Matterport3D: Learning from RGB-D data in indoor environments". In: *Proceedings - 2017 International Conference on 3D Vision, 3DV 2017*. 2018, pp. 667–676. ISBN: 9781538626108. DOI: 10.1109/3DV.2017.00081. arXiv: 1709.06158.

[7] Angel X Chang et al. "SceneSeer: 3D scene design with natural language". In: *arXiv preprint arXiv:1703.00050* (2017).

[8] Kang Chen et al. "Automatic semantic modeling of indoor scenes from low-quality RGB-D data using contextual information". In: *ACM Transactions on Graphics* 33.6 (2014).

[9] Tianshui Chen et al. "Knowledge-Embedded Routing Network for Scene Graph Generation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 6163–6171.

[10] Bo Dai, Yuqi Zhang, and Dahua Lin. "Detecting visual relationships with deep relational networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 3076–3086.

[11] Matthew Fisher et al. "Activity-centric scene synthesis for functional 3D scene modeling". In: *ACM Transactions on Graphics (TOG)* 34.6 (2015), pp. 1–13.

[12] Matthew Fisher et al. "Example-based synthesis of 3D object arrangements". In: *ACM Transactions on Graphics* 31.6 (2012), p. 1. ISSN: 07300301. DOI: 10.1145/2366145.2366154.

[13] Qiang Fu et al. "Adaptive synthesis of indoor scenes via activity-associated object relation graphs". In: *ACM Transactions on Graphics (TOG)* 36.6 (2017), pp. 1–13.

[14] Tobias Germer and Martin Schwarz. "Procedural Arrangement of Furniture for Real-Time Walkthroughs". In: *Computer Graphics Forum*. Vol. 28. 8. Wiley Online Library. 2009, pp. 2068–2078.

[15] Jiuxiang Gu et al. "Scene graph generation with external knowledge and image reconstruction". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1969–1978.

[16] Zhaoyin Jia et al. "3d-based reasoning with blocks, support, and stability". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 1–8.

[17] Justin Johnson et al. "Image retrieval using scene graphs". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3668–3678.

[18] Z Sadeghipour Kermani et al. "Learning 3D Scene Synthesis from Annotated RGB-D Images". In: *Computer Graphics Forum*. Vol. 35. 5. Wiley Online Library. 2016, pp. 197–206.

[19] Mohammad Keshavarzi et al. "Optimization and Manipulation of Contextual Mutual Spaces for Multi-User Virtual and Augmented Reality Interaction". In: *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE. 2020, pp. 353–362.

[20] Mohammad Keshavarzi et al. "SketchOpt: Sketch-based Parametric Model Retrieval for Generative Design". In: *arXiv preprint arXiv:2009.00261* (2020).

[21] Young Min Kim et al. "Acquiring 3D indoor environments with variability and repetition". In: *ACM Transactions on Graphics (TOG)* 31.6 (2012), p. 138.

[22] Yash Kotadia et al. "IndoorNet: Generating Indoor Layouts from a Single Panorama Image". In: *Advanced Computing Technologies and Applications: Proceedings of 2nd International Conference on Advanced Computing Technologies and Applications—ICACTA 2020*. Springer. 2020, pp. 57–66.

[23] Ranjay Krishna et al. "Visual genome: Connecting language and vision using crowdsourced dense image annotations". In: *International Journal of Computer Vision* 123.1 (2017), pp. 32–73.

[24] Manyi Li et al. "GRAINS: Generative recursive autoencoders for indoor scenes". In: *ACM Transactions on Graphics (TOG)* 38.2 (2019), p. 12.

[25] Yikang Li et al. "Scene graph generation from objects, phrases and region captions". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 1261–1270.

[26] Yuan Liang, Song-Hai Zhang, and Ralph Robert Martin. "Automatic data-driven room design generation". In: *International Workshop on Next Generation Computer Animation Techniques*. Springer, 2017, pp. 133–148.

[27] Yuan Liang et al. "Knowledge graph construction with structure and parameter learning for indoor scene design". In: *Computational Visual Media* 4.2 (2018), pp. 123–137. ISSN: 20960662. DOI: 10.1007/s41095-018-0110-3.

[28] Chen Liu et al. "Planercnn: 3d plane detection and reconstruction from a single image". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4450–4459.

[29] Daqi Liu, Miroslaw Bober, and Josef Kittler. "Visual Semantic Information Pursuit: A Survey". In: *arXiv preprint arXiv:1903.05434* (2019).

[30] Cewu Lu et al. "Visual relationship detection with language priors". In: *European Conference on Computer Vision*. Springer. 2016, pp. 852–869.

[31] Rui Ma et al. "Action-driven 3D indoor scene evolution." In: *ACM Trans. Graph.* 35.6 (2016), pp. 173–1.

[32] Rui Ma et al. "Language-driven synthesis of 3D scenes from scene databases". In: *SIGGRAPH Asia 2018 Technical Papers*. ACM. 2018, p. 212.

[33] Ethan Marcotte. *Responsive web design: A book apart n 4*. Editions Eyrolles, 2017.

[34] Paul Merrell et al. "Interactive furniture layout using interior design guidelines". In: *ACM transactions on graphics (TOG)* 30.4 (2011), pp. 1–10.

[35] Sahil Narang, Andrew Best, and Dinesh Manocha. "Simulating movement interactions between avatars & agents in virtual worlds using human motion constraints". In: *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE. 2018, pp. 9–16.

[36] Francesco Pittaluga et al. "Revealing scenes by inverting structure from motion reconstructions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 145–154.

[37] Charles R Qi et al. "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[38] Charles R. Qi et al. "Volumetric and Multi-View CNNs for Object Classification on 3D Data". In: (2016). ISSN: 10636919. DOI: 10.1109/CVPR.2016.609. arXiv: 1604.03265. URL: http://arxiv.org/abs/1604.03265.

[39] Siyuan Qi et al. "Human-centric indoor scene synthesis using stochastic grammar". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 5899–5908.

[40] Sharif Razzaque, Zachariah Kohn, and Mary C Whitton. "Redirected Walking". In: *Proceedings of EUROGRAPHICS* (2001), pp. 289–294. ISSN: 00044172.

[41] Shaoqing Ren et al. "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[42] Daniel Ritchie, Kai Wang, and Yu-an Lin. "Fast and flexible indoor scene synthesis via deep convolutional generative models". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 6182–6190.

[43] Vedant Saran, James Lin, and Avideh Zakhor. "Augmented Annotations: Indoor Dataset Generation with Augmented Reality". In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* (2019).

[44] Manolis Savva, Angel X Chang, and Maneesh Agrawala. "Scenesuggest: Context-driven 3D scene design". In: *arXiv preprint arXiv:1703.00061* (2017).

[45] Skipper Seabold and Josef Perktold. "statsmodels: Econometric and statistical modeling with python". In: *9th Python in Science Conference*. 2010.

[46] Tianjia Shao et al. "An interactive approach to semantic modeling of indoor scenes with an rgbd camera". In: *ACM Transactions on Graphics (TOG)* 31.6 (2012), p. 136.

[47] V. Sharp. *The Art of Redesign*. Sharp Publishing, 2008. ISBN: 9780980883503. URL: https://books.google.com/books?id=2kxqIfFC1EcC.

[48] Simon J. Sheather. "Density Estimation". In: *Statistical Science* 19.4 (2004), pp. 588–597. ISSN: 08834237. URL: http://www.jstor.org/stable/4144429.

[49] Yifei Shi et al. "Hierarchy Denoising Recursive Autoencoders for 3D Scene Layout Prediction". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1771–1780.

[50] Nathan Silberman et al. "Indoor segmentation and support inference from rgbd images". In: *European conference on computer vision*. Springer. 2012, pp. 746–760.

[51] Shuran Song et al. "Semantic Scene Completion from a Single Depth Image". In: *Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition* (2017).

[52] Cheng Sun et al. "Horizonnet: Learning room layout with 1d representation and pano stretch data augmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1047–1056.

[53] Carole Talbott and Maggie Matthews. *Decorating for Good: A Step-by-step Guide to Rearranging What You Already Own*. Clarkson Potter, 1999.

[54] Damien Teney, Lingqiao Liu, and Anton van den Hengel. "Graph-structured representations for visual question answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1–9.

[55] Johanna Wald et al. "Learning 3D Semantic Scene Graphs from 3D Indoor Reconstructions". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 3961–3970.

[56] Kai Wang et al. "Deep convolutional priors for indoor scene synthesis". In: *ACM Transactions on Graphics (TOG)* 37.4 (2018), p. 70.

[57] Kai Wang et al. "Planit: Planning and instantiating indoor scenes with relation graph and spatial prior networks". In: *ACM Transactions on Graphics (TOG)* 38.4 (2019), p. 132.

[58] Lauri Ward. *Use what You Have Decorating: Transform Your Home in One Hour with Ten Simple Design Principles Using...* Penguin, 1999.

[59] Kai Xu et al. "3d attention-driven depth acquisition for object identification". In: *ACM Transactions on Graphics (TOG)* 35.6 (2016), p. 238.

[60] Ken Xu, James Stewart, and Eugene Fiume. "Constraint-based automatic placement for scene composition". In: *Graphics Interface*. Vol. 2. 2002, pp. 25–34.

[61] Michael Ying Yang et al. "On support relations and semantic scene graphs". In: *ISPRS journal of photogrammetry and remote sensing* 131 (2017), pp. 15–25.

[62] Ting Yao et al. "Exploring visual relationship for image captioning". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 684–699.

[63] Yi-Ting Yeh et al. "Synthesizing open worlds with constraints using locally annealed reversible jump mcmc". In: *ACM Transactions on Graphics (TOG)* 31.4 (2012), pp. 1–11.

[64] Lap-Fai Yu et al. "Make it Home: Automatic Optimization of Furniture Arrangement Lap-Fai". In: *ACM Transactions on Graphics* 30.4 (July 2011), p. 1. ISSN: 07300301. DOI: 10.1145/2010324.1964981. URL: http://portal.acm.org/citation.cfm?doid=2010324.1964981.

[65] Zehao Yu et al. "Single-image piece-wise planar 3d reconstruction via associative embedding". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1029–1037.

[66] Rowan Zellers et al. "Neural motifs: Scene graph parsing with global context". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 5831–5840.

[67] Song-Hai Zhang et al. "A Survey of 3D Indoor Scene Synthesis". In: *Journal of Computer Science and Technology* 34.3 (2019), pp. 594–608.

[68] Song-Hai Zhang et al. "Fast 3D Indoor Scene Synthesis with Discrete and Exact Layout Pattern Extraction". In: *arXiv preprint arXiv:2002.00328* (2020).

[69] Bo Zheng et al. "Scene understanding by reasoning stability and safety". In: *International Journal of Computer Vision* 112.2 (2015), pp. 221–238.

[70] Yang Zhou, Zachary While, and Evangelos Kalogerakis. "SceneGraphNet: Neural Message Passing for 3D Indoor Scene Augmentation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 7384–7392.