# Interactive Learning of Spatial Knowledge for Text to 3D Scene Generation

**Angel X. Chang, Manolis Savva** and **Christopher D. Manning**
Computer Science Department, Stanford University
{angelx,msavva,manning}@cs.stanford.edu

## Abstract

We present an interactive text to 3D scene generation system that learns the expected spatial layout of objects from data. A user provides input natural language text from which we extract explicit constraints on the objects that should appear in the scene. Given these explicit constraints, the system then uses prior observations of spatial arrangements in a database of scenes to infer the most likely layout of the objects in the scene. Through further user interaction, the system gradually adjusts and improves its estimates of where objects should be placed. We present example generated scenes and user interaction scenarios.

## 1 Introduction

People possess the power of visual imagination that allows them to turn descriptions of scenes into imagery. The conceptual simplicity of generating pictures from descriptions has spurred the desire to make systems capable of this task. However, research into computational systems for creating imagery from textual descriptions has seen only limited success.

Most current 3D scene design systems require the user to learn complex manipulation interfaces through which objects are constructed and precisely positioned within scenes. However, arranging objects in scenes can much more easily be achieved using natural language. For instance, it is much easier to say “Put a cup on the table’, rather than having to search for a 3D model of a cup, insert it into the scene, scale it to the correct size, orient it, and position it on a table ensuring it maintains contact with the table. By making 3D scene design more accessible to novice users we empower a broader demographic to create 3D scenes for use cases such as interior design, virtual storyboarding and personalized augmented reality.

Unfortunately, several key technical challenges restrict our ability to create text to 3D scene systems. Natural language is difficult to map to formal representations of spatial knowledge and constraints. Furthermore, language rarely mentions common sense facts about the world, that contain critically important spatial knowledge. For example, people do not usually mention the presence of the ground or that most objects are supported by it. As a consequence, spatial knowledge is severely lacking in current computational systems.

Pioneering work in mapping text to 3D scene representations has taken two approaches to address these challenges. First, by restricting the discourse domain to a micro-world with simple geometric shapes, the SHRDLU system demonstrated parsing of natural language input for manipulating the scene, and learning of procedural knowledge through interaction (Winograd, 1972). However, generalization to scenes with more complex objects and spatial relations is very hard to attain.

More recently, the WordsEye system has focused on the general text to 3D scene generation task (Coyne and Sproat, 2001), allowing a user to generate a 3D scene directly from a textual description of the objects present, their properties and their spatial arrangement. The authors of WordsEye demonstrated the promise of text to scene generation systems but also pointed out some fundamental issues which restrict the success of their system: a lot of spatial knowledge is required which is hard to obtain. As a result, the user has to use unnatural language (e.g. “the stool is 1 feet to the south of the table”) to express their intent.

For a text to scene system to understand more natural text, it must be able to infer implicit information not explicitly stated in the text. For instance, given the sentence “there is an office with a red chair”, the system should be able to infer that the office also has a desk in front of the chair. This sort of inference requires a source of prior spatial knowledge. We propose learning this spatial knowledge from existing 3D scene data. However, since the number of available scenes is small, it is difficult to have broad coverage. Therefore, we also rely on user interaction to augment and grow the spatial knowledge. Luckily, user interaction is also natural for scene design since it is an inherently interactive process where user input is needed for refinement.

Our contributions address the fundamental challenges of establishing and interactively expanding a spatial knowledge base. We build on prior work in data-driven scene synthesis (Fisher et al., 2012) to automatically extract general spatial knowledge from data: knowledge of what objects occur in scenes, and their expected spatial relations. Our system then uses this knowledge to generate scenes from natural text inferring implicit constraints. It then leverages user interaction to allow refinement of the scene, and improve the spatial knowledge base. We demonstrate that user interaction is critical in expanding and improving spatial knowledge learned from data.

## 2 Background

A key insight for enabling text to scene generation is that linguistic and non-linguistic spatial knowledge is critical for this task and can be learned directly from data representing the physical world and from interactions of people with such data. User feedback allows us to interactively update spatial knowledge, an idea that we illustrate here in the domain of spatial relations. Early work on the PUT system (Clay and Wilhelms, 1996) and the SHRDLU system (Winograd, 1972) gives a good formalization of the interactive linguistic manipulation of objects in 3D scenes. Recently, there has been promising work on generating 2D clipart for sentences using probabilistic models with placement priors learned from data (Zitnick et al., 2013).

### 2.1 Text to Scene Systems

Prior work on text to 3D scene generation has resulted in systems such as WordsEye (Coyne and Sproat, 2001) and other similar approaches (Seversky and Yin, 2006). These systems are typically not designed to be fully interactive and do not leverage user interaction to improve their results. Furthermore, they mostly rely on manual annotation of 3D models and on hand crafted rules to map text to object placement decisions, which makes them hard to extend and generalize. More recent work has used crowdsourcing platforms, such as Amazon Mechanical Turk, to collect necessary annotations (Coyne et al., 2012). However, this data collection is treated as a separate pre-process and the user still has no influence on the system's knowledge base. We address one part of this issue: learning simple spatial knowledge from data and interactively updating it through user feedback. We also infer unstated implicit constraints thus allowing for more natural text input.

### 2.2 Automatic Scene Layout

Prior work on scene layout has focused largely on room interiors and determining good furniture layouts by optimizing energy functions that capture the quality of a proposed layout. These energy functions are encoded from interior design guidelines (Merrell et al., 2011) or learned from input scene data (Fisher et al., 2012). Knowledge of object co-occurrences and spatial relations is represented by simple models such as mixtures of Gaussians on pairwise object positions and orientations. Methods to learn scene structure have been demonstrated using various data sources including simulation of human agents in 3D scenes (Jiang et al., 2012; Jiang and Saxena, 2013), and analysis of supporting contact points in scanned environments (Rosman and Ramamoorthy, 2011).

However, prior work has not explored methods for enabling users of scene generation algorithms to interactively refine and improve an underlying spatial knowledge model – a capability which is critically important. Our work focuses on demonstrating an interactive system which allows a user to manipulate and refine such spatial knowledge. Such a system is useful regardless of the algorithm used to get the input spatial knowledge.

### 2.3 Interactive Learning

In many tasks, user interaction can provide feedback to an automated system and guide it towards a desired goal. There is much prior work in various domains including interactive systems for refining image search algorithms (Fogarty et al., 2008) and for manipulating social network group creation (Amershi et al., 2012). We focus on the domain of text to 3D scene generation where despite the success of data-driven methods there has been little work on interactive learning systems.

## 3 Approach Overview

What should an interactive text to scene system look like from the perspective of a user? The user should be able to provide a brief scene description in natural language as input. The system parses this text to a set of explicitly provided constraints on what objects should be present, and how they are arranged. This set of constraints should be automatically expanded by using prior knowledge so that "common sense" facts are reflected in the general scene – an example is the static support hierarchy for objects in the scene (i.e. plate goes on table, table goes on ground). The system generates a candidate scene and then the user is free to interact with it by direct control or through textual commands. The system can then leverage user interaction to update its spatial knowledge and integrate newly learned constraints or relations. The final output is a 3D scene that can be viewed from any position and rendered by a graphics engine. In this paper we select an initial viewpoint such that objects are in the frame and view-based spatial relations are satisfied.

How might we create such a system? Spatial knowledge is critical for this task. We need it to understand spatial language, to plausibly position objects within scenes and to allow users to manipulate them. We learn spatial knowledge from example scene data to ensure that our approach can be generalized to different scenarios. We also learn from user interaction to refine and expand existing spatial knowledge. In §5 we describe the spatial knowledge used by our system.

We define our problem as the task of taking text describing a scene as input, and generating a plausible 3D scene described by that text as output. More concretely, based on the input text, we select objects from a dataset of 3D models (§4) and arrange them to generate output scenes. See Figure 1 for an illustration of the system architecture. We break the system down into several subtasks:

**Constraint Parsing** (§6): Parse the input textual description of a concrete scene into a set of constraints on the objects present and spatial relations between them. Automatically expand this set of constraints to account for implicit constraints not specified in the text.

**Scene Generation** (§7): Using above constraints and prior knowledge on the spatial arrangement of objects, construct a scene template. Next, sample

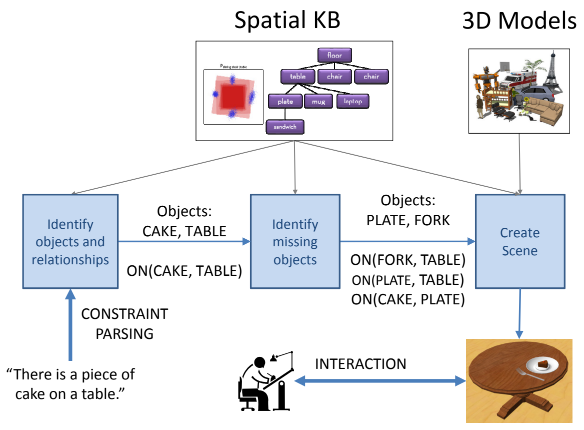

Figure 1: Diagram illustrating the architecture of our system.

the template and select a set of objects to be instantiated. Finally, optimize the placement of the objects to finalize the arrangement of the scene.

**Interaction and Learning** (§8): Provide means for a user to interactively adjust the scene through direct manipulation and textual commands. Use any such interaction to update the system's spatial knowledge so it better captures the user's intent.

## 4 Object Knowledge from 3D Models

To generate scenes we need to have a collection of 3D models for representing physical objects. We use a 3D model dataset collected from Google 3D Warehouse by prior work in scene synthesis and containing about 12490 mostly indoor objects (Fisher et al., 2012). These models have text associated with them in the form of names and tags. In addition, we semi-automatically annotated models with object category labels (roughly 270 classes). We used model tags to set these labels, and verified and augmented them manually.

In addition, we automatically rescale models so that they have physically plausible sizes and orient them so that they have a consistent up and front direction (Savva et al., 2014). Due to the number of models in the database, not all models were rescaled and re-oriented. We then indexed all models in a database that we query at run-time for retrieval based on category and tag labels.

## 5 Spatial Knowledge

Here we describe how we learn spatial knowledge from existing scene data. We base our approach on that of (Fisher et al., 2012) and use their dataset of 133 small indoor scenes created with 1723 3D Warehouse models. Relative object-to-object position and orientation priors can also be learned from the scene data but we have not yet incorporated them in the results for this paper.

### 5.1 Support Hierarchy

We observe the static support relations of objects in existing scenes to establish a prior over what objects go on top of what other objects. As an example, by observing plates and forks on tables most of the time, we establish that tables are more likely to support plates and forks than chairs. We estimate the probability of a parent category $C_p$ supporting a given child category $C_c$ as a simple conditional probability based on normalized observation counts.

$$P_{support}(C_p|C_c) = \frac{count(C_c \text{ on } C_p)}{count(C_c)}$$

### 5.2 Supporting surfaces

To identify which surfaces on parent objects support child objects, we first segment parent models into planar surfaces using a simple region-growing algorithm based on (Kalvin and Taylor, 1996). We characterize support surfaces by the direction of their normal vector limited to the six canonical directions: *up, down, left, right, front, back*. We then learn a probability of supporting surface normal direction $S_n$ given child object category $C_c$. For example, posters are typically found on walls so their support normal vectors are in the horizontal directions. Any unobserved child categories are assumed to have $P_{surf}(S_n = up|C_c) = 1$ since most things rest on a horizontal surface (e.g. floor).

$$P_{surf}(S_n|C_c) = \frac{count(C_c \text{ on surface with } S_n)}{count(C_c)}$$

### 5.3 Spatial Relations

For spatial relations we use a set of predefined relations: *left, right, above, below, front, back, on top of, next to, near, inside, and outside.* These are measured using axis-aligned bounding boxes from the viewer's perspective. More concretely, the bounding boxes of the two objects involved in a spatial relation are compared to determine volume overlap or closest distance (for proximity relations). Table 1 gives a few examples of the definitions of these spatial relations.

Since these spatial relations are resolved with respect to the current view of the scene, they correspond to view-centric definitions of these spatial concepts. An interesting line of future work would be to explore when ego-centric and object-centric spatial reference models are more likely in a given utterance, and resolve the spatial term accordingly.

| *Relation* | $P(relation)$ |
|---|---|
| inside(A,B) | $\frac{Vol(A \cap B)}{Vol(A)}$ |
| outside(A,B) | 1 - $\frac{Vol(A \cap B)}{Vol(A)}$ |
| left(A,B) | $\frac{Vol(A \cap \text{ left } (B))}{Vol(A)}$ |
| right(A,B) | $\frac{Vol(A \cap \text{ right } (B))}{Vol(A)}$ |
| near(A,B) | $\mathbb{1}(dist(A, B) < t_{near})$ |

Table 1: Definitions of spatial relation using object bounding box computations. Note that $dist(A, B)$ is normalized with respect to the maximum extent of the bounding box of $B$.

## 6 Constraint Parsing

During constraint parsing we take the input text and identify the objects and the relations between them. For each object, we also identify properties associated with it such as category label, basic attributes such as color and material, and number of occurrences in the scene. Based on the object category and attributes, and other words in the noun phrase mentioning the object, we identify a set of associated keywords to be used later for querying the 3D model database. Spatial relations between objects are extracted as predicates of the form *on(A,B)* or *left(A,B)* where *A* and *B* are recognized objects.

As an example, given the input "There is a room with a desk and a red chair. The chair is to the left of the desk." we extract the following objects and spatial relations:

*Objects:*

| index | category | attributes | keywords |
|---|---|---|---|
| 0 | room | | room |
| 1 | desk | | desk |
| 2 | chair | *color*:red | chair, red |

*Relations: left*(chair, desk)

The input text is processed using the Stanford CoreNLP pipeline[1]. We use the Stanford coreference system to determine when the same object is being referred to. To identify objects, we look for noun phrases and use the head word as the category, filtering with WordNet (Miller, 1995) to determine which objects are visualizable (under the

[1] http://nlp.stanford.edu/software/corenlp.shtml

| *Dependency Pattern* | *Example Text* |
|---|---|
| `tag:VBN=verb >nsubjpass =nsubj >prep (=prep >pobj =pobj)` | The chair$_{\texttt{[nsubj]}}$ is made$_{\texttt{[verb]}}$ of$_{\texttt{[prep]}}$ wood$_{\texttt{[pobj]}}$ |
| `tag:VB=verb >dobj =dobj >prep (=prep >pobj =pobj)` | Put$_{\texttt{[verb]}}$ the cup$_{\texttt{[dobj]}}$ on$_{\texttt{[prep]}}$ the table$_{\texttt{[pobj]}}$ |

Table 2: Example dependency patterns for extracting spatial relations.

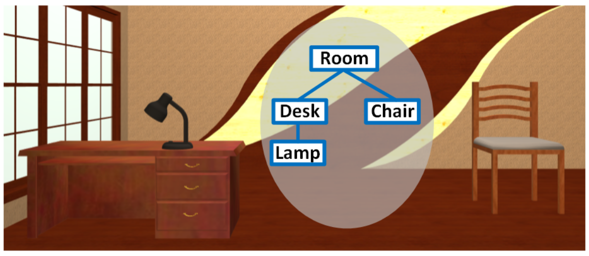

Figure 2: Generated scene for "There is a room with a desk and a lamp. There is a chair to the right of the desk." The inferred scene hierarchy is overlayed in the center.

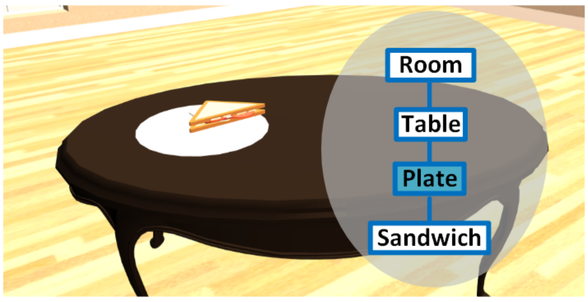

Figure 4: Generated scene for "There is a room with a table and a sandwich." Note that the plate is not explicitly stated, but is inferred by the system.

physical object synset, excluding locations). To identify properties of the objects, we extract other adjectives and nouns in the noun phrase. We also match dependency patterns such as "X is made of Y" to extract more attributes and keywords. Finally, we use dependency patterns to extract spatial relations between objects (see Table 2 for some example patterns).

We used a fairly simple deterministic approach to map text to the scene template and user actions on the scene. An interesting avenue for future research is to automatically learn how to map text using more advanced semantic parsing methods.

## 7 Scene Generation

During scene generation we aim to find the most likely scene given the input utterance, and prior knowledge. Once we have determined from the input text what objects exist and their spatial relations in the scene, we select 3D models matching the objects and their associated properties. We sample the support hierarchy prior $P_{support}$ to obtain the support hierarchy for the scene.

We then initialize the positions of objects within the scene by traversing the support hierarchy in depth-first order, positioning the largest available child node and recursing. Child nodes are positioned by selecting a supporting surface on a candidate parent object through sampling of $P_{surf}$ and ensuring no collisions exist with other objects. If there are any spatial constraints that are not satisfied, we remove and randomly reposition the objects violating the constraints, and iterate to improve the layout. The resulting scene is rendered and presented to the user.

Figure 2 shows a rendering of a generated scene along with the support hierarchy and input text. Even though the spatial relation between lamp and desk was not mentioned explicitly, we infer that the lamp is supported by the top surface of the desk. In Figure 3 we show another example of a generated scene for the input "There is a room with a poster bed and a poster". Note that the system differentiates between a "poster" and a "poster bed" – it correctly selects and places the bed on the floor, while the poster is placed on the wall.

Figure 4 shows an example of inferring missing objects. Even though the plate was not explicitly mentioned in the input, we infer that the sandwich is more likely to be supported by a plate rather than directly placed on the table. Without this infer-

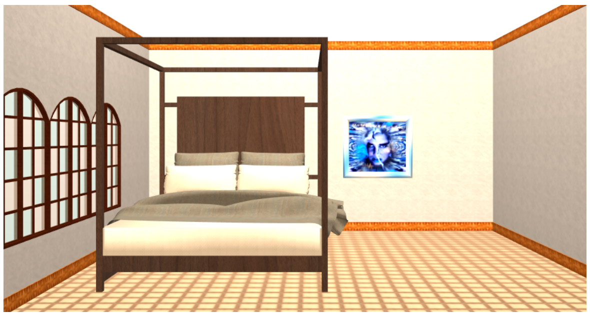

Figure 3: Generated scene for "There is a room with a poster bed and a poster."

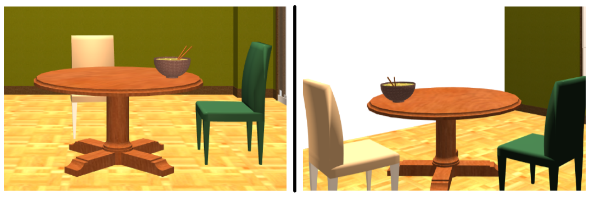
Figure 5: **Left**: chair is selected using "the chair to the right of the table" or "the object to the right of the table". Chair is not selected for "the cup to the right of the table". **Right**: Different view results in different chair being selected for the input "the chair to the right of the table".

ence, the user would need to be much more verbose with text such as "There is a room with a table, a plate and a sandwich. The sandwich is on the plate, and the plate is on the table."

## 8 Interactive System

Once a scene is generated, the user can view the scene and manipulate it using both simple action phrases and mouse interaction. The system supports traditional 3D scene interaction mechanisms such as navigating the viewpoint with mouse and keyboard, selection and movement of object models by clicking. In addition, a user can give simple textual commands to select and modify objects, or to refine the scene. For example, a user can request to "remove the chair" or "put a pot on the table" which requires the system to resolve referents to objects in the scene (see §8.1). The system tracks user interactions throughout this process and can adjust its spatial knowledge accordingly. In the following sections, we give some examples of how the user can interact with the system and how the system learns from this interaction.

### 8.1 View centric spatial relations

During interaction, the user can refer to objects with their categories and with spatial relations between them. Objects are disambiguated by both category and view-centric spatial relations. We use the WordNet hierarchy to resolve hyponym or hypernym referents to objects in the scene. In the left screenshot in Figure 5, the user can select a chair to the right of the table using the phrase "chair to the right of the table" or "object to the right of the table". The user can then change their viewpoint by rotating and moving around. Since spatial relations are resolved with respect to the current viewpoint, we see that a different chair is selected for the same phrase from the different viewpoint in the right screenshot.

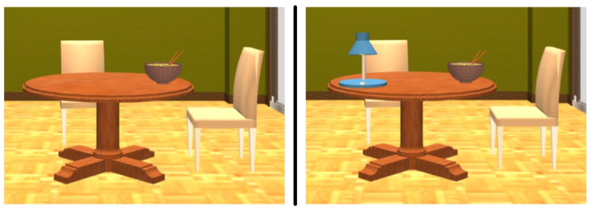
Figure 6: **Left**: initial scene. **Right**: after input "Put a lamp on the table".

### 8.2 Scene Editing with Text

By using simple textual commands the user can edit the scene. For example, given the initial scene on the left in Figure 6, the user can then issue the command "put a lamp on the table" which results in the scene on the right. The system currently allows for adding objects to new positions and removing existing objects. Currently, repositioning of objects is performed only with direct control, but in the future we also plan to support repositioning of objects by using textual commands.

### 8.3 Learning Support Hierarchy

After a user requests that a lamp be placed on a table, the system updates its prior on the likelihood of a lamp being supported by a table. Based on prior observations the likelihood of lamps being placed on tables was very low (4%) since very few lamps were observed on tables in the scene dataset. However, after the user interaction, we recompute the prior including the scene that the user has created and the probability of lamp on table increases to 12% (see Figure 7).

### 8.4 Learning Object Names

Often, objects or parts may not have associated labels that the user would use to refer to the objects. In those cases, the system can inform the user that it cannot resolve a given name, and the user can then select the object or part of the object they were referring to and annotate it with a label. For instance, in Figure 8, the user annotated the different parts of the room as "floor", "wall", "window", and "door". Before annotation, the system did not know any labels for these parts of the room. After annotation, the user can select these parts using the associated names. In addition, the system updates its spatial knowledge base and can now predict that the probability of a poster being placed on a wall

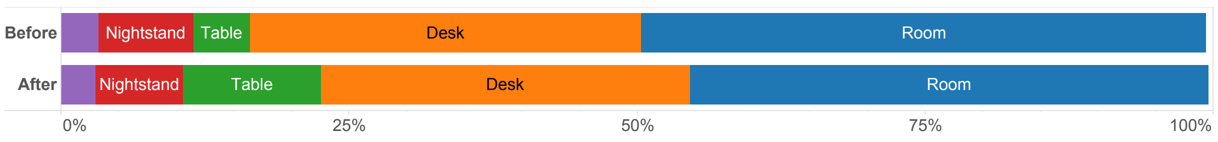

Figure 7: Probability of supporting parent categories for lamps before and after the user explicitly requests a lamp on a table.

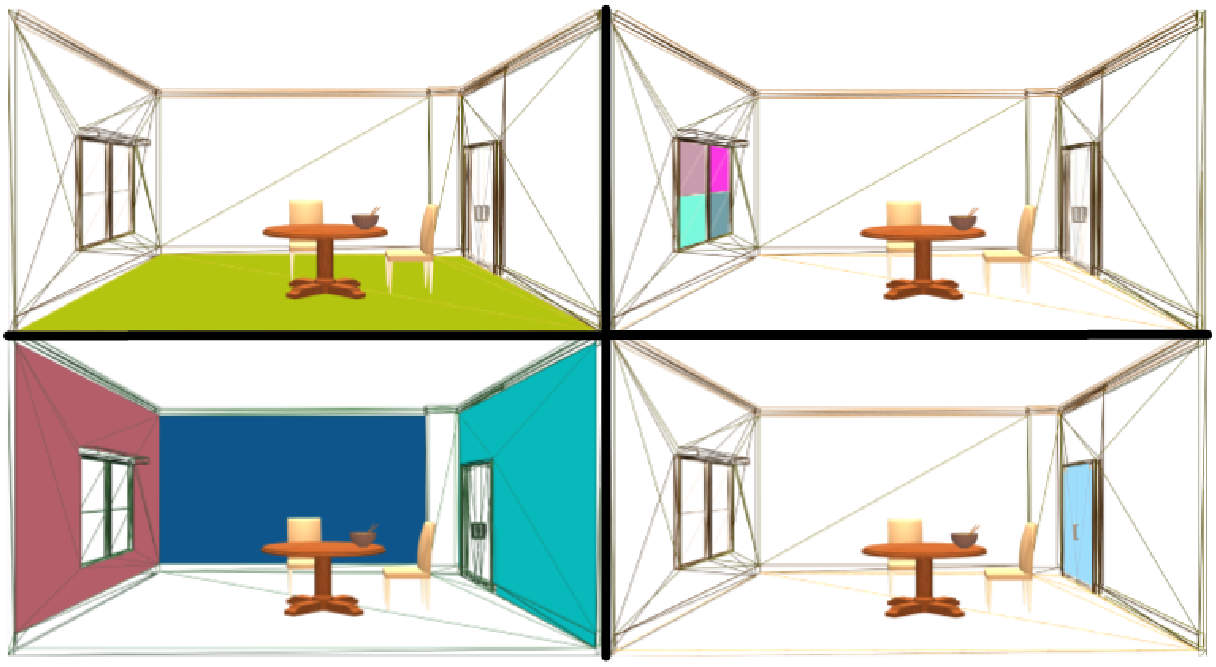

Figure 8: The user clicks and selects parts of the scene, annotating them as “floor”, “wall”, “window”, “door”. After annotation, the user can also refer to these parts with the associated names. The system spatial knowledge base is updated accordingly.

is 40%, and that the probability of a table being placed on the floor is 23%. Note that these probabilities are based on multiple observations of the annotated room. Accumulating annotations such as these and propagating labels to new models is an effective way to expand spatial knowledge.

## 9 Future Work

We described a preliminary interactive text to 3D scene generation system that can learn from prior data and user interaction. We hope to improve the system by incorporating more feedback mechanisms for the user, and the learning algorithm.

If the user requests a particular object be selected but the system gets the referent wrong, the user could then indicate the error and provide a correction. We can then use this feedback as a source of training data to improve the interpretation of text to the desired user action. For example, if the user asks to “select the red bowl” and the system could not resolve “red bowl” to the correct object, the user could intervene by clicking on the correct referent object. Simple interactions such as this are incredibly powerful for providing additional data for learning. Though we did not focus on this aspect, a dialogue-based interaction pattern is natural for our system. The user can converse with the system to iteratively refine the scene and the system can ask for clarifications at any point – when and how the system should inquire for more information is interesting future research.

To evaluate whether the generated scenes are satisfactory, we can ask people to rate them against input text descriptions. We can also study usage of the system in concrete tasks to see how often users need to provide corrections and manually manipulate the scene. A useful baseline to compare against would be a traditional scene manipulation system. By doing these studies at a large scale, for instance by making the interface available on the web, we can crowdsource the accumulation of user interactions and gathering of spatial knowledge. Simultaneously, running formal user studies to better understand preference for text-based versus direct interactions during different actions would be very beneficial for more informed design of text-to-scene generation systems.

## 10 Conclusion

We have demonstrated the usefulness of an interactive text to 3D scene generation system. Spatial knowledge is essential for text to 3D scene generation. While it is possible to learn spatial knowledge purely from data, it is hard to have complete coverage of all possible scenarios. Interaction and user feedback is a good way to improve coverage and to refine spatial knowledge. In addition, interaction is a natural mode of user involvement in scene generation and creative tasks.

Little prior work has addressed the need for interaction or the need for recovering implicit spatial constraints. We propose that the resolution of unmentioned spatial constraints, and leveraging user interaction to acquire spatial knowledge are critical for enabling natural text to scene generation.

User interaction is essential for text to scene generation since the process is fundamentally under-constrained. Most natural textual descriptions of scenes will not mention many visual aspects of a physical scene. However, it is still possible to automatically generate a plausible starting scene for refinement.

Our work focused on showing that user interaction is both natural and useful for a text to scene generation system. Furthermore, refining spatial knowledge through interaction is a promising way of acquiring more implicit knowledge. Finally, any practically useful text to scene generation will by necessity involve interaction with users who have particular goals and tasks in mind.

## References

Saleema Amershi, James Fogarty, and Daniel Weld. 2012. Regroup: interactive machine learning for on-demand group creation in social networks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

Sharon Rose Clay and Jane Wilhelms. 1996. Put: Language-based interactive manipulation of objects. *Computer Graphics and Applications, IEEE*.

Bob Coyne and Richard Sproat. 2001. WordsEye: an automatic text-to-scene conversion system. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*.

Bob Coyne, Alexander Klapheke, Masoud Rouhizadeh, Richard Sproat, and Daniel Bauer. 2012. Annotation tools and knowledge representation for a text-to-scene system. *Proceedings of COLING 2012: Technical Papers*.

Matthew Fisher, Daniel Ritchie, Manolis Savva, Thomas Funkhouser, and Pat Hanrahan. 2012. Example-based synthesis of 3D object arrangements. *ACM Transactions on Graphics (TOG)*.

James Fogarty, Desney Tan, Ashish Kapoor, and Simon Winder. 2008. CueFlik: interactive concept learning in image search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

Yun Jiang and Ashutosh Saxena. 2013. Infinite latent conditional random fields for modeling environments through humans.

Yun Jiang, Marcus Lim, and Ashutosh Saxena. 2012. Learning object arrangements in 3D scenes using human context. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*.

Alan D Kalvin and Russell H Taylor. 1996. Superfaces: Polygonal mesh simplification with bounded error. *Computer Graphics and Applications, IEEE*.

Paul Merrell, Eric Schkufza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. 2011. Interactive furniture layout using interior design guidelines. In *ACM Transactions on Graphics (TOG)*.

G.A. Miller. 1995. WordNet: a lexical database for english. *CACM*.

Benjamin Rosman and Subramanian Ramamoorthy. 2011. Learning spatial relationships between objects. *The International Journal of Robotics Research*.

Manolis Savva, Angel X. Chang, Gilbert Bernstein, Christopher D. Manning, and Pat Hanrahan. 2014. On being the right scale: Sizing large collections of 3D models. *Stanford University Technical Report CSTR 2014-03*.

Lee M Seversky and Lijun Yin. 2006. Real-time automatic 3D scene generation from natural language voice and text descriptions. In *Proceedings of the 14th annual ACM international conference on Multimedia*.

Terry Winograd. 1972. Understanding natural language. *Cognitive psychology*.

C Lawrence Zitnick, Devi Parikh, and Lucy Vanderwende. 2013. Learning the visual interpretation of sentences. In *IEEE Intenational Conference on Computer Vision (ICCV)*.