# Active Arrangement of Small Objects in 3D Indoor Scenes

Suiyun Zhang, Zhizhong Han, Yu-Kun Lai, Matthias Zwicker, Hui Zhang

**Abstract**—Small object arrangement is very important for creating detailed and realistic 3D indoor scenes. In this paper, we present an interactive framework based on active learning to help users create customized arrangements for small objects according to their preferences. To achieve this with minimal user effort, we first learn the prior knowledge about small object arrangement from a 3D indoor scene dataset through a probability mining method, which forms the initial guidance for arranging small objects. Then, users are able to express their preferences on a few small object categories, which are automatically propagated to all the other categories via a novel active learning approach. In the propagation process, we introduce a novel metric to obtain the propagation weights, which measures the degree of interchangeability between two small object categories, and is calculated based on a spatial embedding model learned from the small object neighborhood information extracted from the 3D indoor scene dataset. Experiments show that our framework is able to help users effectively create customized small object arrangements with little effort.

**Index Terms**—3D object layout, Active learning, Scene enrichment, Computer-aided aesthetic design, Human computer interaction.

## 1 Introduction

Small objects are key components for increasing the realism of 3D indoor scenes in various applications, such as virtual reality, interior design and 3D video games [1]. However, it is time-consuming and tedious to manually arrange small objects in 3D indoor scenes, especially when the number of scenes becomes large. Although small object arrangements can be simply achieved by applying basic heuristic rules in an automatic manner, e.g., using collision tests [2], the arrangement results are not always plausible. Moreover, users have their own preferences, and such *customized* small object arrangements are more tedious to achieve. Therefore, how to efficiently generate both plausible and customized small object arrangements remains a challenge in the field of 3D scene modeling and understanding.

This problem has been little explored. The previous work on small object arrangement can be divided into data-driven methods and user-guided methods. In data-driven methods, the arrangements are generated by applying the occurrence statistics of small objects learned from either a few example images [3] or 3D indoor scenes [4]. Although these methods are capable of generating plausible arrangements, users cannot easily express their intentions about small object arrangements. Hence the results are not easily customizable. In contrast, the user-guided methods can generate customized arrangements thanks to the detailed information specified by users, such as the absolute priority of each small object category [5] and the targeted location of each small object [1]. However, too much human labor is required in these cases. First, users need to exhaustively provide information for each single category. Second, new input is required when arranging a new set of small objects. Our work fills the gap as being both data-driven and user-guided. Human effort is dramatically reduced as our framework maximizes the benefits of user input preferences through a propagation process. Thus, we address the demanding problem of combining knowledge learned from a dataset with a small set of user preferences to efficiently produce both plausible and customized small object arrangements.

In this paper, we propose an active arrangement framework to address this challenge. The proposed framework pays attention to the pairwise spatial relationship between any two small object categories, which we define as a relative priority probability. In addition, our key contribution to reduce user effort is to automatically propagate user preferences for a small set of object categories across all categories in a meaningful manner. To achieve this, we present a novel active learning method that inherits the advantages of traditional active learning methods [6], while also absorbing the prior knowledge learned from a dataset, which remedies the disadvantages of traditional methods [7].

Our framework consists of an offline knowledge learning stage and an online arrangement generation stage. The offline stage learns the initial priority probabilities about relative placements of pairs of object categories from a 3D scene dataset. Due to the limited information provided by the current dataset, a probability mining method is proposed to infer the priority probability for any two small object categories that do not co-occur in the dataset by indirectly bridging them through a third category. These priority probabilities serve as the initial guidance for arranging small objects in the online stage. Meanwhile, to model the user preference propagation process, we need to calculate a meaningful degree of interchangeability between

- *S. Zhang, H. Zhang are with the School of Software, Tsinghua University, Beijing 100084, China and also with Beijing National Research Center for Information Science and Technology (BNRist). E-mail: zhangsuiyun13@mails.tsinghua.edu.cn, huizhang@tsinghua.edu.cn.*
- *Z. Han and M. Zwicker are with the Department of Computer Science, University of Maryland, Maryland, College Park, MD 20742. E-mail: h312h@umd.edu, zwicker@cs.umd.edu.*
- *Y.-K Lai is with School of Computer Science & Informatics, Cardiff University, Wales, United Kingdom. E-mail: Yukun.Lai@cs.cardiff.ac.uk.*

*(Corresponding author: Hui Zhang)*

any two categories. The intuition is that if two categories are interchangeable, then user preferences can be propagated from one to the other. We achieve this using a neural network model to learn the spatial embedding of small object categories from the context information of categories in the 3D scene dataset. The idea is that two categories with similar neighborhoods tend to be interchangeable, and the spatial embedding can be leveraged to quantify the degree of interchangeability.

In the online stage, an initial arrangement is first generated by applying the initial priority probabilities learned from the dataset. Users can then specify their preferences for the spatial relations of any pair of categories on the arrangement result. The preferences are propagated across all categories according to the degree of interchangeability between two categories, which we calculate from the spatial embedding space. Afterwards, we apply the propagated priority probabilities to update the original arrangements into customized ones as specified by user preferences. In summary, our main contributions are as follows:

- We present an active arrangement framework for small object arrangement in 3D indoor scenes, which simultaneously exploits both the prior knowledge learned from data and user specified preferences to efficiently produce plausible, customized small object arrangement results.
- We introduce a probability mining approach that infers the unknown priority probabilities of two 3D small object categories which never co-occur in the dataset, using the information of a third category which co-occurs with both categories.
- We propose a neural network model that learns the spatial embedding of small object categories, based on which we introduce a novel metric to measure the degree of interchangeability between categories.
- We provide a user friendly graphical interface that allows users to interactively express their preferences for small object arrangement. Our overall system significantly reduces interaction time required to produce desired results.

## 2 Related Work

According to the techniques involved in our framework, we review three classes of related work.

### 2.1 Small object arrangement

Many researchers seek to automatically arrange small objects in 3D scenes by applying the knowledge learned from datasets. Majerowicz et al. [3] populated the arrangement of an example shelf to empty shelves in a style-preserving manner, where the style denoted the relative positions between any two small objects and the global properties including density, grouping and symmetry. Fisher et al. [4] learned Gaussian mixture models from example 3D scenes to represent pairwise spatial information between object categories, such as locations and rotations. Kermani et al. [8] learned an arrangement model from an RGB-D image dataset based on a factor graph. Jiang et al. [9] modeled the spatial relationship between small objects and human poses using Dirichlet process from a given 3D scene. Wang et al. [10] learned the spatial distribution of 3D objects in the scenes from a large 3D scene dataset with three different Convolutional Neural Networks (CNNs). Although these methods could generate quite plausible arrangements based on the knowledge learned from datasets, it is hard for users to express their preferences in order to create customized arrangements.

To involve users into the process of arranging small objects, a few works allowed users to define their preferences by specifying detailed information. Yu et al. [1] proposed an interactive indoor scene detailing system which arranged small objects according to user specified locations. Savva et al. [11] presented an interactive system for 3D scene assembling, using the prior knowledge of object support, position and orientation learned from a 3D scene dataset. Zhang et al. [5] generated small object arrangements using the priority of each small object category which needed to be specified as absolute values by users. In these methods, although customized arrangements can be obtained, they are at the cost of involving substantial human labor, because the relationship between the spatial properties of small object categories is not effectively modeled, and the involved human effort is not sufficiently exploited.

Different from these works, we base our framework on the prior knowledge learned from the dataset, and meanwhile maximize the use of information provided by the limited number of user preferences through an active learning method. In this way, we manage to find a balance between the learned knowledge and user preferences to generate both plausible and customized small object arrangements.

Ma et al. [12] simulated the scene evolution process by a group of human actions, which are sampled from an action graph learned from a scene dataset. However, the choice of human actions is limited to the nodes of the learned action graph. Comparatively, our approach mainly focuses on incorporating user preferences for small object arrangements by active learning. This allows users to manipulate the arrangement results at a finer level. Moreover, in their method, the spatial distribution of 3D objects is relative to the human poses. Thus, unlike our approach, their method cannot model the relation between the supporting heights of small objects to support the small object arrangement on multi-layer furniture objects.

### 2.2 Active learning in 3D

Active learning has been used in various 3D applications for its potential of propagating the information provided by few labeled instances to other unlabeled ones [6]. Gao et al. [13] proposed an active learning based approach to explore large 3D model repositories, where the user preferences for the shapes were propagated to other shapes in order to display those of interest. Top et al. [14] applied active learning to segment 3D medical images by querying users with the most uncertain segments. Yi et al. [15] enriched the semantic region annotations of 3D shape datasets, which was achieved by propagating the manual annotations of a few shapes across the whole shape set under an active learning framework. Song et al. [16], [17] learned a customized classifier for 3D shapes in an interactive way, where the

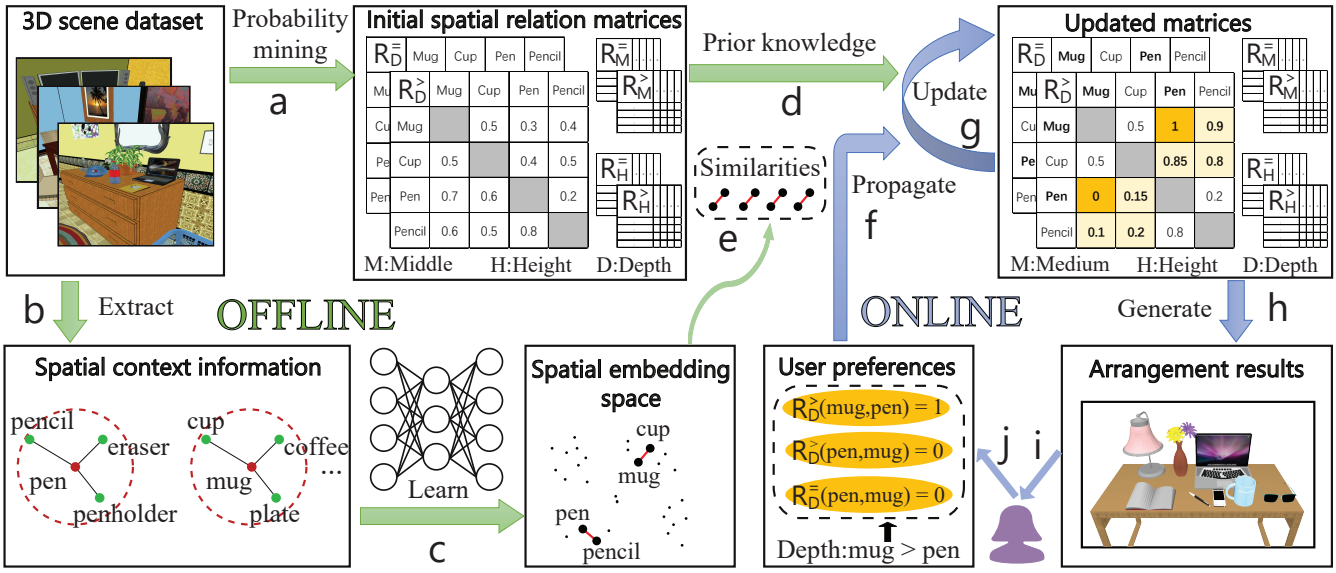



Figure 1. The pipeline of our framework. In the offline stage, the initial spatial relation matrices are learned (a) to provide the prior knowledge of arrangement (d). The spatial embedding space for small object categories is learned by training a neural network (c) using the spatial context information extracted (b) from a 3D scene dataset. The similarity of spatial embeddings (e) provides the degree of interchangeability between categories, which is further used for propagating (f) user preferences. In the online stage, the relation matrices are updated (g) according to both prior knowledge and user preferences. Then, small object arrangements are generated (h) using the matrices and are presented (i) to users, where they can specify (j) further preferences.

classifier was refined according to user provided feedback. Wang et al. [18] proposed an active learning method to carry out shape segmentation and labeling, where users were asked to assign must-link or cannot-link constraints between segments.

Our framework shares the spirit of propagating user preferences across the whole set of small object categories with these methods. However, the prior knowledge learned from the dataset is also considered in our framework in the optimization process to improve the propagation results.

### 2.3 Embedding methods in 3D

Creating an embedding space for unstructured data in order to measure their similarity with continuous representations is fundamental for various applications in computer vision [19], [20], [21], [22], [23]. Recently, to facilitate the analysis across different modalities such as 3D shapes, depth images, sketches, descriptions, etc., embedding methods are applied in 3D applications as well [24]. Tesse et al. proposed Shape2Vec [25], which introduced word embedding space of shape categories as a common space to link the features of various modalities in order to perform cross-modal retrieval. Li et al. [26] learned a common embedding space between 3D shapes and 2D images. They first constructed another embedding space from 3D descriptors, then trained a CNN to purify images so that each image could be mapped to a feature point in the common embedding space.

Different from these methods, rather than embedding the semantic meaning of 3D shapes, we embed the spatial context of small object categories into a vector space through a neural network model, which facilitates the propagation of user preferences about small object arrangements.

## 3 Overview

Given a set of small object categories $\mathbf{C} = \{c_i | i = 1, \ldots, N_\mathbf{C}\}$, where $N_\mathbf{C}$ is the number of categories, our framework aims to arrange small objects of categories $c_i$ in $\mathbf{C}$ on a supporting furniture object, conforming to both prior knowledge learned from the dataset and user specified preferences for small object arrangements.

It is observed that "viewers care about the relative rather than absolute difference between small objects' properties" [3]. Regarding the property of 3D position, we tackle the arrangement problem by considering three types of relative spatial relations between small object categories, including *middleness, height and depth*. They denote the relative spatial information along the $x$, $y$ and $z$ axes, respectively, in the right-handed coordinate system. It should be noted that *middleness* denotes the quantity of being close to the middle axis of a supporting surface for a small object. The definition of the spatial relations will be detailed in Sec. 4.2. For each type of relation, the probability of priority of any two categories $c_i$ and $c_j$ is given by two matrices, representing the inequality and equality probabilities respectively, denoted as $\boldsymbol{R}^{>}_{\mathrm{X}}$ and $\boldsymbol{R}^{=}_{\mathrm{X}}$, where X is one of the three types of relations, including H for height, M for middleness and D for depth. For example, $\boldsymbol{R}^{>}_{\mathrm{H}}(c_i, c_j)$ is the probability that $c_i$ is placed higher than $c_j$, and $\boldsymbol{R}^{=}_{\mathrm{H}}(c_i, c_j)$ is the probability that $c_i$ is placed at the same height as $c_j$. The equality relations are necessary because e.g. objects are often placed at the same height on a shelf. The corresponding entries in a pair of two matrices $\boldsymbol{R}^{>}_{\mathrm{X}}, \boldsymbol{R}^{=}_{\mathrm{X}}$ should meet the condition

$$\boldsymbol{R}^{>}_{\mathrm{X}}(c_i, c_j) + \boldsymbol{R}^{>}_{\mathrm{X}}(c_j, c_i) + \boldsymbol{R}^{=}_{\mathrm{X}}(c_i, c_j) = 1. \quad (1)$$

Thus, our goal is to obtain a set of 6 matrices $\mathbf{R} = \{\boldsymbol{R}^{>}_{\mathrm{X}}, \boldsymbol{R}^{=}_{\mathrm{X}} | \mathrm{X} \in \{\mathrm{D}, \mathrm{M}, \mathrm{H}\}\}$, as shown in Fig. 1, which rep-

resents both the prior knowledge learned from the 3D scene dataset and the user specified preferences.

Fig. 1 shows the pipeline of our framework, which is composed of an offline knowledge learning stage and an online active arrangement generation stage.

In the offline stage, two aspects of knowledge are learned to feed the online stage. First, the initial spatial relation matrix set, denoted as $\tilde{\mathbf{R}}$, is learned by analyzing the co-occurrence information of small object categories from a 3D scene dataset (Sec. 4). $\tilde{\mathbf{R}}$ is fixed once learned. As not every pair of small object categories co-occur in the same scene from the dataset, a probability mining method (Fig. 1a) is proposed to infer the priority probabilities of these pairs, where a third common category that co-occurs separately with each category from the pair is employed to bridge the two categories. $\tilde{\mathbf{R}}$ is used to provide prior knowledge (Fig. 1d) for the matrix updating in the online stage. Second, knowledge for propagating user preferences is learned. In the online stage, users will provide a small set of preferences for arrangements of pairs of object categories. To reduce user effort, we propagate these preferences to all other semantically similar, interchangeable categories. The propagation weights are calculated by the degree of interchangeability between categories, which is measured based on the similarity (Fig. 1e) of the spatial embeddings of categories. We train a neural network model (Fig. 1c) in the offline stage to learn the spatial embedding (Sec. 6) of small object categories using the spatial context information extracted (Fig. 1b) from the 3D scene dataset.

In the online stage, users are interactively involved in the process of arranging small objects (Sec. 5). An initial arrangement is generated (Sec. 4.4) and presented (Fig. 1h,i) to users using $\tilde{\mathbf{R}}$ (Fig. 1d) obtained from the offline stage. Then, starting from $\mathbf{R} = \tilde{\mathbf{R}}$, $\mathbf{R}$ is iteratively updated (Fig. 1g) through an active learning based optimization process according to each new user preference. The updated $\mathbf{R}$ is in return applied to arrange the set of small objects on the furniture object. With the current arrangement results, users can specify (Fig. 1j) their preferences, each about any type of spatial relations for two categories shown in the results. For example, a possible preference could be "the depth of the mug should be greater than the pen's". The preferences are firstly converted to constraints on the priority probability between the specified categories, which are further propagated to the whole set of categories (Fig. 1f) through the optimization process, using the degree of interchangeability calculated from the spatial embedding learned in the offline stage. This process will iterate until users get satisfied with the arrangement results.

## 4 Priority Probability Mining

We learn the priority probabilities of three relative spatial relations (Sec. 4.2) for each pair of small object categories in $\mathbf{C}$ from a 3D indoor scene dataset (Sec. 4.1). Sparse initial spatial relation matrices are first extracted from the co-occurrence information of small object categories shown in the dataset. Then, these matrices are completed by a priority probability inference method to form the initial probability matrices $\tilde{\mathbf{R}}$, which provide the prior knowledge about small object arrangement (Sec. 4.3). Small object arrangements are generated using the priority probabilities provided by the matrices in $\mathbf{R}$ (initialized with $\tilde{\mathbf{R}}$ and updated with active learning) (Sec. 4.4).

### 4.1 Dataset

The 3D indoor scene dataset constructed by Fisher et al. [4] provides rich information about small object occurrence and arrangement. Thus, we use it as the source for learning both the priority probabilities between small object categories and the spatial embedding of small object categories. The original dataset contains 133 3D scenes, which were composed by human modelers using 1,741 unique 3D objects. Although each 3D object was assigned with a list of names and tags, the basic categories of these objects were not provided consistently. To facilitate the analysis between categories, each 3D object is manually assigned to a category by us. Besides furniture categories, 203 different small object categories are observed in this dataset, i.e., $N_{\mathbf{C}} = 203$, of which the top 30 categories with most 3D objects are shown in Fig. 2.

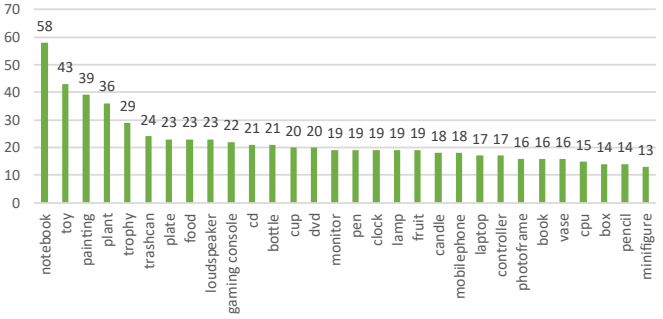



Figure 2. The 30 most common categories and their numbers of objects.

### 4.2 Relative spatial relations

Unlike [27], which computed a few total orderings of all the items under certain user preferences, we instead use probabilities to define the relation between any two small object categories. Specifically, we consider three types of relative spatial relations between two categories $c_i$ and $c_j$, as defined below.

**Height priority.** There are certain rules for placing small objects on different layers of a multi-layer furniture object, e.g., *shoes* are more likely to be placed on a lower layer than a *perfume* on a shelf. We use $y_{\mathrm{H}}(c_i)$ to denote the supporting height of $c_i$ when it is placed on a multi-layer furniture, and $\boldsymbol{R}_{\mathrm{H}}^{>}(c_i, c_j)$ to denote the probability that $c_i$ is placed at a higher position than $c_j$, i.e., $p(y_{\mathrm{H}}(c_i) > y_{\mathrm{H}}(c_j))$.

**Depth priority.** Some categories are more likely to be placed in front of others, e.g., placing a *cup* in front of a *lamp* is more reasonable than the opposite way. We use $y_{\mathrm{D}}(c_i)$ to denote the depth of $c_i$, and $\boldsymbol{R}_{\mathrm{D}}^{>}(c_i, c_j)$ to denote the probability of $c_i$ being placed in front of $c_j$, i.e., $p(y_{\mathrm{D}}(c_i) > y_{\mathrm{D}}(c_j))$.

**Middleness priority.** Similarly, some small objects tend to be placed nearer to the middle axis of a supporting surface than others, e.g., placing a *laptop* to the middle of a desk is beneficial for accessibility. We use $y_{\mathrm{M}}(c_i)$ to denote

the distance from $c_i$ to the middle axis of the supporting furniture, and $\boldsymbol{R}^{>}_{\mathrm{M}}(c_i, c_j)$ to denote the probability of $c_i$ being placed farther to the axis than $c_j$, i.e., $p(y_{\mathrm{M}}(c_i) > y_{\mathrm{M}}(c_j))$.

In addition to the inequality probabilities, we also consider equality probabilities, which are denoted as $\boldsymbol{R}^{=}_{\mathrm{D}}(c_i, c_j)$, $\boldsymbol{R}^{=}_{\mathrm{M}}(c_i, c_j)$ and $\boldsymbol{R}^{=}_{\mathrm{H}}(c_i, c_j)$ for depth, middleness and height priorities, respectively. For example, $\boldsymbol{R}^{=}_{\mathrm{H}}(c_i, c_j)$ denotes the probability that $c_i$ and $c_j$ are placed at the same height on a furniture object.

### 4.3 Priority probability inference

Let X be any type of the spatial relations, according to the probability definition, the probabilities of inequality and equality relations between two categories $c_i$ and $c_j$ are estimated by enumerating the occurrence of unequal/equal relations of the two categories in the dataset and computing their proportion w.r.t. the total number of co-occurrence between $c_i$ and $c_j$:

$$\boldsymbol{R}^{>}_{\mathrm{X}}(c_i, c_j) = p(y_{\mathrm{X}}(c_i) > y_{\mathrm{X}}(c_j)) = \frac{n(y_{\mathrm{X}}(c_i) > y_{\mathrm{X}}(c_j))}{n(c_i, c_j)}, \tag{2}$$

$$\boldsymbol{R}^{=}_{\mathrm{X}}(c_i, c_j) = p(y_{\mathrm{X}}(c_i) = y_{\mathrm{X}}(c_j)) = \frac{n(y_{\mathrm{X}}(c_i) = y_{\mathrm{X}}(c_j))}{n(c_i, c_j)}. \tag{3}$$

$n(y_{\mathrm{X}}(c_i) > y_{\mathrm{X}}(c_j))$ and $n(y_{\mathrm{X}}(c_i) = y_{\mathrm{X}}(c_j))$ denote the number of times that the corresponding property regarding relation X of $c_i$, $y_{\mathrm{X}}(c_i)$, is larger than and equal to that of $c_j$, $y_{\mathrm{X}}(c_j)$, in the dataset, respectively. For example, $n(y_{\mathrm{H}}(book) > y_{\mathrm{H}}(cup))$ is the number of times that *book* is placed at a higher position than *cup* on the same furniture object. $n(c_i, c_j)$ is the number of times that $c_i$ and $c_j$ co-occur on the same furniture. When calculating middleness and depth relations, the equality condition is relaxed to be within a threshold $\epsilon$, which is set to 5 $cm$ for both relations. The diagonal elements of $\boldsymbol{R}^{>}_{\mathrm{X}}$ and $\boldsymbol{R}^{=}_{\mathrm{X}}$ are the probabilities of self relations for a category $c$, where the equality probability $\boldsymbol{R}^{=}_{\mathrm{X}}(c, c)$ is computed similarly using Eq. 3 and the inequality probability $\boldsymbol{R}^{>}_{\mathrm{X}}(c, c)$ is computed by $1 - \boldsymbol{R}^{=}_{\mathrm{X}}(c, c)$.

The priority probabilities are calculated for each co-occurring pair of categories that has ever been observed on some furniture in the dataset. The six matrices recording these priority probabilities are denoted as $\mathbf{R^0}$.

#### 4.3.1 Matrix completion

Due to the inadequate 3D scenes in the dataset, not all pairs of categories co-occur on the same furniture object, so the priority probabilities for the three types of spatial relations of these pairs could not be directly calculated using Eq. 2 or Eq. 3. As a result, many entries of the matrices in $\mathbf{R^0}$ remain uninitialized. To resolve this issue, the information for a pair of categories $c_i$ and $c_j$, which do not co-occur with each other, is predicted through a third category $c_k$, which co-occurs with $c_i$ and $c_j$ separately on the same furniture.

Specifically, we wish to use known probabilities of $(c_i, c_k)$ and $(c_k, c_j)$ to predict the unknown probabilities of $(c_i, c_j)$. We assume that the distribution of the property value of each category is independent in our dataset. For example, the distribution of the depth of a *lamp* is irrelevant to that of a *cup*, as a *lamp* is usually placed at the back while a *cup* is placed in the front. Therefore, we further assume that the relation between the corresponding property values of two small object categories, e.g., $(c_i, c_k)$, is independent from another pair of categories, e.g., $(c_k, c_j)$. As derived in Appendix A, under the above independence assumptions, by applying the *sum rule* of probability, $\boldsymbol{R}^{>}_{\mathrm{X}}(c_i, c_j)$, $\boldsymbol{R}^{>}_{\mathrm{X}}(c_j, c_i)$ and $\boldsymbol{R}^{=}_{\mathrm{X}}(c_i, c_j)$ can be computed as follows, which are denoted by $\boldsymbol{R}^{>}_{i,j}$, $\boldsymbol{R}^{>}_{j,i}$ and $\boldsymbol{R}^{=}_{i,j}$, respectively:

$$\boldsymbol{R}^{>}_{i,j} = \frac{\sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)(\, \boldsymbol{R}^{>}_{i,k}\boldsymbol{R}^{=}_{k,j} + \boldsymbol{R}^{>}_{i,k}\boldsymbol{R}^{>}_{k,j} + \boldsymbol{R}^{=}_{i,k}\boldsymbol{R}^{>}_{k,j})}{\sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)(\, 1 - \boldsymbol{R}^{>}_{j,k} - \boldsymbol{R}^{>}_{k,i}\,)}, \tag{4}$$

$$\boldsymbol{R}^{>}_{j,i} = \frac{\sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)(\, \boldsymbol{R}^{>}_{j,k}\boldsymbol{R}^{=}_{k,i} + \boldsymbol{R}^{>}_{j,k}\boldsymbol{R}^{>}_{k,i} + \boldsymbol{R}^{=}_{j,k}\boldsymbol{R}^{>}_{k,i})}{\sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)(\, 1 - \boldsymbol{R}^{>}_{i,k} - \boldsymbol{R}^{>}_{k,j}\,)}, \tag{5}$$

$$\boldsymbol{R}^{=}_{i,j} = \frac{\sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)\boldsymbol{R}^{=}_{i,k}\boldsymbol{R}^{=}_{k,j}}{\sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)(1 - \boldsymbol{R}^{>}_{i,k} - \boldsymbol{R}^{>}_{k,j})}, \tag{6}$$

where $\boldsymbol{R}^{>}_{i,j}$, $\boldsymbol{R}^{>}_{j,i}$ and $\boldsymbol{R}^{=}_{i,j}$ are normalized by their sum to meet Eq. 1. $\mathbf{K}_{c_i,c_j}$ is the set of co-occurring categories with both $c_i$ and $c_j$. $p(c_k)$ is the prior occurrence probability of a category $c_k$, which is the frequency of $c_k$ appearing in the dataset. The numerators of Eqs. 4, 5 and 6 calculates the priority probabilities of $c_i$ and $c_j$ that can be inferred when the property of $c_k$, i.e., $y_{\mathrm{X}}(c_k)$, is in-between those of $c_i$ and $c_j$. Take Eq. 4 for an example, $\boldsymbol{R}^{>}_{i,k}\boldsymbol{R}^{>}_{k,j}$ calculates the probability of $y_{\mathrm{X}}(c_i) > y_{\mathrm{X}}(c_k) > y_{\mathrm{X}}(c_j)$ while $\boldsymbol{R}^{=}_{i,k}\boldsymbol{R}^{>}_{k,j}$ calculates that of $y_{\mathrm{X}}(c_i) = y_{\mathrm{X}}(c_k) > y_{\mathrm{X}}(c_j)$. $y_{\mathrm{X}}(c_i) > y_{\mathrm{X}}(c_j)$ can be inferred from both relations. Summing the probabilities from all possible $c_k$ gives the priority probability for $c_i$ and $c_j$.

We apply a greedy strategy to update the matrix set $\mathbf{R}$. Starting from $\mathbf{R} = \mathbf{R^0}$, in each iteration, for each uninitialized entry indexed by $c_i$ and $c_j$ in each matrix from $\mathbf{R}$, we find the co-occurring category set $\mathbf{K}_{c_i,c_j}$, and then update corresponding uninitialized entries by Eqs. 4, 5 and 6. The algorithm ends when no more co-occurring categories can be found for $c_i$ and $c_j$. In our experiments, this process converges after 3 iterations. Table 1 shows the completion rate after each iteration, which is the proportion of the entries that have been calculated in matrices. It should be noted that, not all kinds of matrices are necessarily completely filled, such as $\boldsymbol{R}^{=}_{\mathrm{D}}$ or $\boldsymbol{R}^{>}_{\mathrm{M}}$. It is because for some pairs of categories, there is no category co-occurring with them, thus they can never reach each other even after exhaustive augmentation. In case we need to arrange small objects from such pairs of categories, say, $c_i$ and $c_j$, we set by default $\boldsymbol{R}^{=}_{\mathrm{X}}(c_i, c_j) = \boldsymbol{R}^{>}_{\mathrm{X}}(c_i, c_j) = \boldsymbol{R}^{>}_{\mathrm{X}}(c_j, c_i) = 1/3$ in our experiments for completeness. These should be rare, assuming the 3D scene dataset is reasonably extensive. The resulting matrix set is denoted as $\tilde{\mathbf{R}}$, which provides the prior knowledge about spatial relations of small object categories learned from the dataset.

### 4.4 Arrangement generation

The goal is to arrange a set of small objects on a furniture object $f$ according to the spatial relation matrices in $\mathbf{R}$. We follow a similar process of arranging small objects as detailed in [5], in which a cost function was defined to

Table 1
The completion rate of each matrix through each iteration. The inference process converged after 3 iterations.

| **Compl. Rate** | Initial | Iter. 1 | Iter. 2 | Iter. 3 |
|---|---|---|---|---|
| $\boldsymbol{R}_{\mathrm{D}}^{>}, \boldsymbol{R}_{\mathrm{D}}^{=}$ | 9.67% | 60.88% | 94.19% | 94.23% |
| $\boldsymbol{R}_{\mathrm{M}}^{>}, \boldsymbol{R}_{\mathrm{M}}^{=}$ | 9.34% | 57.92% | 91.92% | 91.96% |
| $\boldsymbol{R}_{\mathrm{H}}^{>}, \boldsymbol{R}_{\mathrm{H}}^{=}$ | 13.11% | 74.84% | 99.81% | 100.00% |

measure how an arrangement on $f$ conforms to the user-specified absolute priorities of each small object category.

#### 4.4.1 Cost function

Our cost function differs from theirs in two ways. First, instead of absolute priorities, it is composed using the relative priority probabilities from $\mathbf{R}$, which are more flexible to apply. Second, in addition to inequality relations, our cost function considers equality relations as well. Specifically, for a pair of small object categories $c_i$ and $c_j$ that are placed on $f$, their cost w.r.t. the spatial relation $\mathrm{X} \in \{\mathrm{D}, \mathrm{M}, \mathrm{H}\}$ is defined as follows:

$$\begin{aligned} F_{\mathrm{X}}(c_i, c_j) =& \mathbb{1}(y_i < y_j)\boldsymbol{R}_{\mathrm{X}}^{>}(c_i, c_j)g(|y_i - y_j|) \\ &+\mathbb{1}(y_i \neq y_j)\boldsymbol{R}_{\mathrm{X}}^{=}(c_i, c_j)g(|y_i - y_j|), \end{aligned} \tag{7}$$

where $y_i$ is short for $y_{\mathrm{X}}(c_i)$. $\mathbb{1}(\cdot)$ is an indicator function, which returns 1 if the condition meets and 0 otherwise. $g(z) = 1/(1 + e^{-z})$ is the sigmoid function which maps the value of $z$ to $0 \sim 1$. The two terms on the right side of Eq. 7 represent the cost for inequality and equality of two categories $c_i$ and $c_j$. The general idea is to penalize the degree of misplacement of two categories according to their priority probabilities. For inequality case, as $\boldsymbol{R}_{\mathrm{X}}^{>}(c_i, c_j)$ is the probability of $y_{\mathrm{X}}(c_i) > y_{\mathrm{X}}(c_j)$, thus a cost will be introduced and weighted by $\boldsymbol{R}_{\mathrm{X}}^{>}(c_i, c_j)$ only if $c_i$ and $c_j$ are placed in the opposite way, which makes $\mathbb{1}(y_{\mathrm{X}}(c_i) < y_{\mathrm{X}}(c_j))$ return 1. The cost is defined by the difference of the properties of $c_i$ and $c_j$, which is mapped through a sigmoid function and weighted by the priority probability $\boldsymbol{R}_{\mathrm{X}}^{>}(c_i, c_j)$. For equality case, the cost is only introduced when $y_{\mathrm{X}}(c_i) \neq y_{\mathrm{X}}(c_j)$ if X is H, or when $|y_{\mathrm{X}}(c_i) - y_{\mathrm{X}}(c_j)| > \epsilon$ if X is M or D.

Thus, the cost for spatial relations between small object categories is defined as the sum of the costs from all pairs of small object categories on $f$ for all three spatial relations:

$$F_{Rel}(A) = \sum_{c_i, c_j \in \mathbf{C}_f, i \neq j} F_{\mathrm{D}}(c_i, c_j) + F_{\mathrm{M}}(c_i, c_j) + F_{\mathrm{H}}(c_i, c_j), \tag{8}$$

where $\mathbf{C}_f$ is the set of categories on furniture $f$. $F_{\mathrm{D}}$, $F_{\mathrm{M}}$ and $F_{\mathrm{H}}$ are specialized forms of Eq. 7. Then, the overall cost function for an arrangement $A$ is composed of two parts:

$$F_{arr}(A) = F_{Rel}(A) + F_{Hard}(A), \tag{9}$$

where $F_{Rel}(A)$ is the cost for spatial relations and $F_{Hard}$ is the cost for hard constraints such as collision cost or out-of-boundary cost, which are defined the same as in [5].

#### 4.4.2 Optimization

Since $F_{arr}(A)$ is multi-modal, we apply a stochastic sampling method to obtain the optimized arrangement, which is similar to the method in [28], [29].

Specifically, Eq. 9 is first converted to a Boltzmann-like density function:

$$q(A) = 1/J \cdot e^{-\beta_1 F_{arr}(A)},$$

where $\beta_1$ is a constant temperature parameter and is set to 5 empirically in our experiments. Afterwards, we use the Metropolis-Hasting algorithm [30], [31] to explore the density function $q(A)$ to avoid the computation of the partition function $J$.

The algorithm starts with an initial arrangement as the current arrangement $A$, and keeps obtaining the next arrangement $A^*$ by applying a randomly picked proposal move from the following on $A$:

- Swap the positions of two randomly selected small objects.
- Perturb the position of a random small object by a Gaussian term.

The acceptance probability of each proposal move from $A$ to $A^*$ is computed from $q$:

$$\alpha(A \rightarrow A^*) = \min(1, \frac{q(A^*)}{q(A)})$$

This process stops when the iteration number reaches the budget. All the accepted arrangements are retained and sorted by their cost. The lowest-cost one is returned as the final arrangement.

## 5 Active Small Object Arrangement

Based on the initial arrangement on furniture $f$ generated by $\tilde{\mathbf{R}}$, users can provide their preferences about small object arrangement by specifying the spatial relation for any two categories in $\mathbf{C}_f$. These preferences are then converted to entries in the relation matrices in $\mathbf{R}$ (Sec. 5.1), and propagated to other entries by updating $\mathbf{R}$ through optimizing an energy function (Sec. 5.2). During the optimization, the three groups of matrices in $\mathbf{R}$, including $(\boldsymbol{R}_{\mathrm{D}}^{>}, \boldsymbol{R}_{\mathrm{D}}^{=})$, $(\boldsymbol{R}_{\mathrm{M}}^{>}, \boldsymbol{R}_{\mathrm{M}}^{=})$ and $(\boldsymbol{R}_{\mathrm{H}}^{>}, \boldsymbol{R}_{\mathrm{H}}^{=})$, are updated separately according to user specified preferences on different spatial relations. For each spatial relation X, $\boldsymbol{R}_{\mathrm{X}}^{=}$ and $\boldsymbol{R}_{\mathrm{X}}^{>}$ are updated in sequence to meet the probability constraint defined in Eq. 1 (Sec. 5.3).

### 5.1 User preferences

Given an arrangement result on furniture $f$, users are queried to specify the spatial relation preferences for any two categories in $\mathbf{C}_f$ using our graphical user interface, as shown in Fig. 3. Different from the typical active learning methods, our query process is implied by the arrangement result in each iteration. Specifically, two categories $c_i$ and $c_j$ are first selected by clicking on the corresponding 3D objects in the arrangement result, which will be highlighted with red bounding boxes, as shown in the left display panel in Fig. 3. Then, from the option panel on the right side, one of the spatial relations X, i.e., depth, middleness and height priority, is chosen. Afterwards, users can specify the relation of the priority preference between the property of X for $c_i$ and $c_j$, including $y_{\mathrm{X}}(c_i) > y_{\mathrm{X}}(c_j)$, $y_{\mathrm{X}}(c_i) = y_{\mathrm{X}}(c_j)$ or $y_{\mathrm{X}}(c_i) < y_{\mathrm{X}}(c_j)$.

As $\boldsymbol{R}_{\mathrm{X}}^{>}$ and $\boldsymbol{R}_{\mathrm{X}}^{=}$ are used for generating small object arrangements, user preferences are embedded through

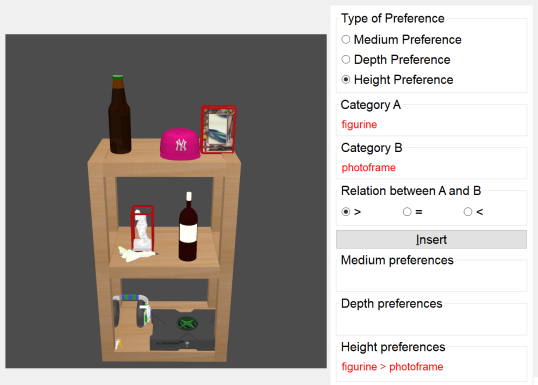



Figure 3. Graphical user interface of our framework.

changing certain entries in these two matrices for spatial relation X. Specifically, the priority preference between two categories specified by users is converted to priority probabilities of 1 or 0 in $\boldsymbol{R}_{\mathrm{X}}^{>}$ and $\boldsymbol{R}_{\mathrm{X}}^{=}$ as detailed in Table 2. For simplicity, we consider one of the matrices, either $\boldsymbol{R}_{\mathrm{X}}^{>}$ or $\boldsymbol{R}_{\mathrm{X}}^{=}$, and the same process applies to all the 6 matrices. Let $r_p$ be the priority probability for a pair of categories $p$ in the matrix, and $\boldsymbol{r}$ be the vector containing all $r_p$ values. As there are $N_{\mathbf{C}}$ categories in total, $\boldsymbol{r}$ is of size ${N_{\mathbf{C}}}^2$. Then we denote the set of indices in $\boldsymbol{r}$ that the user preferences are applied on as $\mathbf{U}_{\mathrm{X}}$. The size of $\mathbf{U}_{\mathrm{X}}$ is proportional to the number of user preferences specified on spatial relation X.

Table 2
User preferences are embedded into $\mathbf{R}$ by fixing corresponding probabilities to 1 or 0.

| Values $v$ in $\mathbf{R}$ | $\boldsymbol{R}_{\mathrm{X}}^{>}(c_i, c_j)$ | $\boldsymbol{R}_{\mathrm{X}}^{>}(c_j, c_i)$ | $\boldsymbol{R}_{\mathrm{X}}^{=}(c_i, c_j)$ |
|---|---|---|---|
| $y_{\mathrm{X}}(c_i) > y_{\mathrm{X}}(c_j)$ | 1.0 | 0.0 | 0.0 |
| $y_{\mathrm{X}}(c_i) < y_{\mathrm{X}}(c_j)$ | 0.0 | 1.0 | 0.0 |
| $y_{\mathrm{X}}(c_i) = y_{\mathrm{X}}(c_j)$ | 0.0 | 0.0 | 1.0 |

### 5.2 Propagation energy function

In order to effectively utilize user preferences, we develop an active learning approach, where user preferences for a few small object categories can be propagated to all the categories in $\mathbf{C}$. Typical active learning methods cannot be directly applied to our problem for two reasons. First, with a typical active learning method [6], users are asked to provide labels only for those instances considered 'informative', which may not necessarily be of interest to users. In our problem, users have clear intentions about their preferences for the arrangement of small object categories, and thus it is more meaningful to let users actively label their concerned categories. Second, the unlabeled instances in typical active learning methods do not have initial values for their labels, so the propagated results are completely dependent on the labels specified by users [7]. However, in our case, we require the propagated results of small object categories to additionally conform to the prior knowledge learned from the dataset.

Our goal is to find the optimal priority probabilities, encoded in vector $\boldsymbol{r}$, for both $\boldsymbol{R}_{\mathrm{X}}^{>}$ and $\boldsymbol{R}_{\mathrm{X}}^{=}$. It should meet the following two conditions. First, similar pairs should have similar priority probabilities, e.g., the priority probability of *pen* and *cup* should be similar to that of *pencil* and *mug*. Second, the updated probability of a pair of categories should to some extent conform to the value in the previous iteration in order to balance the impact of user preference and prior knowledge. Considering these two aspects, we construct the following energy function of $\boldsymbol{r}$ regarding $\boldsymbol{R}_{\mathrm{X}}^{>}$ or $\boldsymbol{R}_{\mathrm{X}}^{=}$:

$$E(\boldsymbol{r}) = \frac{1}{2} \sum_{p,q \in [1, {N_{\mathbf{C}}}^2]} w_{pq}(r_p - r_q)^2 + \frac{1}{2}\lambda \sum_{p \in [1, {N_{\mathbf{C}}}^2]} (r_p - r'_p)^2. \tag{10}$$

The two terms in Eq. 10 correspond to the above two conditions, respectively. In the first term, to encourage two pairs of categories with great interchangeability to have similar priority probabilities in $\boldsymbol{R}_{\mathrm{X}}^{>}$ or $\boldsymbol{R}_{\mathrm{X}}^{=}$, we use the degree of interchangeability of them as the punishing coefficient heuristically. The degree of interchangeability is denoted by $w_{pq}$ and will be detailed in Sec. 6. The second term accounts for the conformance of the propagated result to the prior knowledge learned from the dataset, where $r'_p$ is the value of $r_p$ in the previous iteration during the optimization. The initial value of $r'_p$ is from $\tilde{\mathbf{R}}$. As the two terms have very different magnitudes, $\lambda = 10$ is used empirically to balance the contribution from them. Both $w_{pq}$ and $\lambda$ are fixed during the optimization process of Eq. 10.

### 5.3 Optimization

For a specific $\boldsymbol{r}$, let $\boldsymbol{W}$ be the similarity matrix which contains $w_{pq}$, $\boldsymbol{\lambda}$ be a diagonal matrix whose diagonal entries are $\lambda$, $\boldsymbol{D}$ be a diagonal matrix whose diagonal entries are the sum of the rows of $\boldsymbol{W}$. $\boldsymbol{W}$, $\boldsymbol{\lambda}$ and $\boldsymbol{D}$ are all of size ${N_{\mathbf{C}}}^2 \times {N_{\mathbf{C}}}^2$. Eq. 10 can be rewritten in the matrix form as follows:

$$\begin{aligned} E(\boldsymbol{r}) &= \frac{1}{2}[\boldsymbol{r}^T(\boldsymbol{D} - \boldsymbol{W})\boldsymbol{r} + (\boldsymbol{r} - \boldsymbol{r}')^T\boldsymbol{\lambda}(\boldsymbol{r} - \boldsymbol{r}')] \\ &= \frac{1}{2}\boldsymbol{r}^T(\boldsymbol{D} - \boldsymbol{W} + \boldsymbol{\lambda})\boldsymbol{r} - \boldsymbol{r}'^T\boldsymbol{\lambda}\boldsymbol{r} + \frac{1}{2}\boldsymbol{r}'^T\boldsymbol{\lambda}\boldsymbol{r}'. \end{aligned} \tag{11}$$

In Eq 11, since the last term is a constant which is only related to the value in the previous iteration, i.e., $\boldsymbol{r}'$, it can be ignored for optimization. Thus, the optimal $\boldsymbol{r}$ can be found by solving the following quadratic programming problem:

$$\begin{aligned} \hat{\boldsymbol{r}} = \arg\min(&\frac{1}{2}\boldsymbol{r}^T(\boldsymbol{D} - \boldsymbol{W} + \boldsymbol{\lambda})\boldsymbol{r} - \boldsymbol{r}'^T\boldsymbol{\lambda}\boldsymbol{r}), \\ &\text{subject to } 0 \le r_p \le 1, p = 1, \ldots, {N_{\mathbf{C}}}^2 \\ &r_u = v_u, u \in \mathbf{U}_{\mathrm{X}} \end{aligned} \tag{12}$$

where $u$ is an index from $\mathbf{U}_{\mathrm{X}}$, and its corresponding value is $v_u$, as given in Table 2.

It should be noted that, after the optimization, the values of entries indexed by two categories $c_i$ and $c_j$ in $\boldsymbol{R}_{\mathrm{X}}^{>}$ and $\boldsymbol{R}_{\mathrm{X}}^{=}$ should still meet the probability condition defined in Eq. 1. Thus, the optimization process is carried out first on $\boldsymbol{R}_{\mathrm{X}}^{=}$, then with $\boldsymbol{R}_{\mathrm{X}}^{=}(c_i, c_j)$ fixed, Eq. 1 is applied as extra constraints in Eq. 12 to optimize $\boldsymbol{r}$ for $\boldsymbol{R}_{\mathrm{X}}^{>}$.

The iteration starts from $\mathbf{R} = \tilde{\mathbf{R}}$, where an arrangement is generated using the priority probabilities provided by $\mathbf{R}$ with the method described in Sec. 4.4. Then, users can interactively specify preferences on the arrangement result, i.e., $\mathbf{U}_{\mathrm{X}}$, which will be applied to update $\mathbf{R}$. The updated $\mathbf{R}$

can then be used to update the current arrangement. This process continues until no more preferences are provided.

## 6 Interchangeability Metric

To reduce user effort, we propagate the user preference on a pair of small object categories to other pairs of categories according to their spatial interchangeability. Two categories are considered to be interchangeable if they have similar neighbors around them [4] and of similar sizes. While measuring size similarity is trivial, measuring the similarity of their neighborhoods or context is not. Inspired by the recent success of word embedding for various applications [25], [32], we aim to find an embedding space of small object categories to capture their spatial context information.

To achieve this goal, we first construct a spatial context corpus from the dataset by finding neighbors for each small object within a radius (Sec 6.1). Then, a neural network is trained from the corpus to obtain a spatial embedding of categories, where each category is represented by a continuous vector and the spatial context similarity between categories can be measured (Sec. 6.2). Afterwards, an interchangeability metric is proposed to measure the degree of interchangeability between categories by including both the category size similarity and the spatial context similarity provided by the spatial embedding (Sec. 6.3). Finally, the interchangeability between two pairs of categories, i.e., $w_{pq}$ in Eq. 10, is defined based on the proposed interchangeability metric, which forms the propagation weights in the active learning process (Sec. 6.4).

### 6.1 Spatial context corpus

For a small object $d$ of category $c_i$ on a supporting surface of a furniture object, the categories of all the small objects on the same supporting surface that are within a spatial context radius $l$ to the small object $d$ are considered to be the neighboring categories of $c_i$. We refer to such a spatial context relationship as a *context record*, and denote it as "$f_{\mathbf{C}}(d) : \mathcal{N}_d$", where $f_{\mathbf{C}}(d)$ is the category of small object $d$ and $\mathcal{N}_d$ is the set of neighboring categories for small object $d$. A context record is only valid if $|\mathcal{N}_d| > 0$.

For a radius $l$, denote by $\mathbf{N}_l$ the spatial context corpus, which is the set of all context records discovered using radius $l$ in all 3D scenes in the dataset. Technically, for each 3D small object in our dataset, there should always be a corresponding context record. However, when the radius $l$ is very small, there might be no neighboring small objects around a small object $d$, which makes $\mathcal{N}_d$ empty. Thus, the size of the constructed corpus will increase as the context radius $l$ grows. Fig. 4 plots the size of $\mathbf{N}_l$ ($y$-axis) w.r.t. the spatial context radius $l$ ($x$-axis). We set $l = 40\ cm$ in our experiments as it is a radius that not only enables a fair amount of context records (1,454) but also is small enough to account for a meaningful spatial context for a small object.

### 6.2 Spatial embedding model

Inspired by Word2Vec [33], we train a neural network to embed small object categories into a spatial embedding space, in which their similarity of spatial context is measured through their embedding vectors. Our model

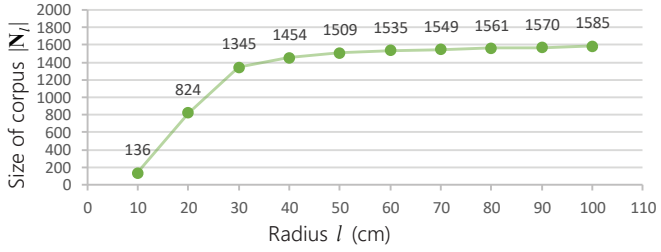



Figure 4. The size of corpus $|\mathbf{N}_l|$ w.r.t. the spatial context radius $l$.

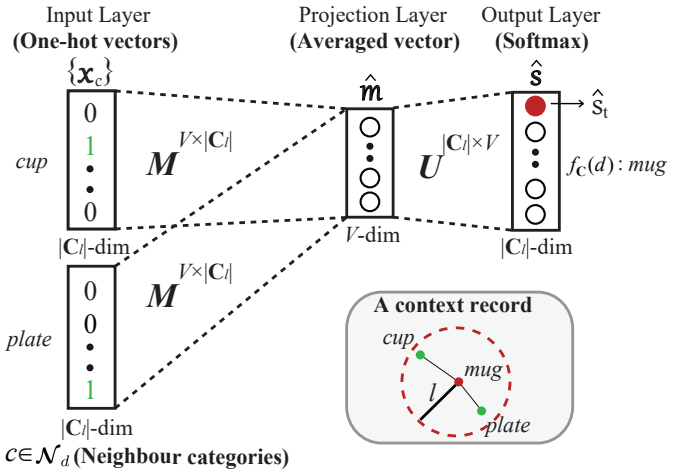



Figure 5. The structure of our embedding model. The one-hot vectors of neighboring categories $\boldsymbol{x}_c$ are fed into the input layer. They are multiplied by an embedding matrix $\boldsymbol{M}$ and averaged to form an embedding vector $\hat{\boldsymbol{m}}$ in the projection layer. The vector is then mapped through a matrix $\boldsymbol{U}$ and converted to a probability distribution $\hat{\boldsymbol{s}}$ over all categories via a softmax function in the output layer.

shares the spirit with the Continuous Bag-Of-Words model (CBOW) [33], which predicts a target word using its context words within a window size from the position of the target word. In our problem, the neighboring categories in $\mathcal{N}_d$ for a small object $d$ are used to predict the category of $d$.

Specifically, there are three layers in our neural network, including an input layer, a projection layer and an output layer. Let $\mathbf{C}_l$ be the set of unique categories in the spatial context corpus $\mathbf{N}_l$, and $V$ be the dimension of the target embedding vector. Fig. 5 illustrates the structure of the network, taking a context record as an example, where *mug* is the target category $f_{\mathbf{C}}(d)$. {*cup*, *plate*} is the set of neighboring categories $\mathcal{N}_d$. For a context record regarding a small object $d$, each neighboring category $c \in \mathcal{N}_d$ is first converted to a one-hot vector, denoted as $\boldsymbol{x}_c \in \mathbb{R}^{|\mathbf{C}_l|}$. The one-hot vectors are fed to the input layer. Then, all the one-hot vectors are multiplied by a spatial context embedding matrix $\boldsymbol{M}^{V\times|\mathbf{C}_l|} = [\boldsymbol{m}_1, \ldots, \boldsymbol{m}_{|\mathbf{C}_l|}]$, where $\boldsymbol{m}_i \in \mathbb{R}^V$ is the embedding vector for the $i^{\text{th}}$ category in $\mathbf{C}_l$. The multiplication operation extracts the corresponding embedding vectors for each category $c$ in $\mathcal{N}_d$. These extracted vectors are then averaged to form an embedding vector $\hat{\boldsymbol{m}}$ in the projection layer:

$$\hat{\boldsymbol{m}} = \frac{1}{|\mathcal{N}_d|} \sum_{c\in\mathcal{N}_d} \boldsymbol{M}\boldsymbol{x}_c. \tag{13}$$

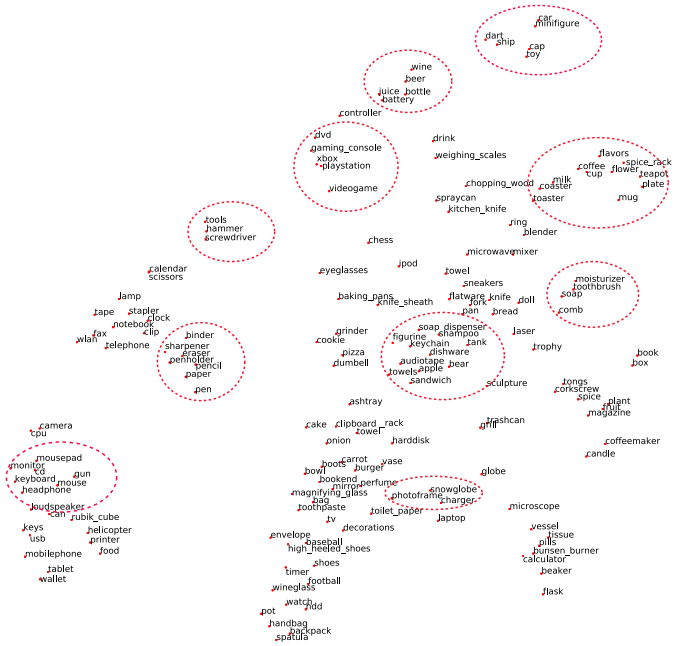



Figure 6. 2D visualization of small object category vector representations in spatial embedding space.

Afterwards, $\hat{\boldsymbol{m}}$ is converted to a probability distribution over all the categories in $\mathbf{C}_l$, denoted as $\hat{\boldsymbol{s}}$, by first being mapped with a weighting matrix $\boldsymbol{U} \in \mathbb{R}^{|\mathbf{C}_l| \times V}$, and then applied with a softmax function:

$$\hat{\boldsymbol{s}} = \text{softmax}(\boldsymbol{U}\hat{\boldsymbol{m}}). \tag{14}$$

For a target category $t \in \mathbf{C}_l$, its predicted probability is $\hat{s}_t$. The loss for $t$ is then defined as the negative logarithm of its predicted probability, i.e., $-\log(\hat{s}_t)$. Thus, the overall loss for our spatial context embedding model is:

$$\mathcal{L} = -\sum_{t \in \mathbf{C}_l} \log(\hat{s}_t), \tag{15}$$

which is back-propagated to update both $\boldsymbol{M}$ and $\boldsymbol{U}$.

The training process is accelerated by applying the hierarchical softmax scheme [33]. The resulting $\boldsymbol{M}$ contains the spatial context embedding vectors for each category in $\mathbf{C}_l$. We denote the cosine similarity between the embedding vectors of two categories $c_i$ and $c_j$ in $\boldsymbol{M}$ as $L(c_i, c_j)$. The embedding matrix $\boldsymbol{M}$ for our experimental setting $l = 40\ cm$ and $V = 20$ is projected in 2D using parametric $t$-SNE [34], as shown in Fig. 6, where several sets of small object categories with similar spatial context are highlighted.

### 6.3 Interchangeability metric

Two categories are regarded interchangeable if they are similar in both spatial context and size. For two categories such as *eraser* and *pen*, as they have similar spatial context, i.e., both are surrounded by *notebook* and *ruler*, they are likely to be interchangeable. Moreover, for a category $c_i$, among the categories of most similar spatial context with $c_i$, the ones with similar sizes are considered more interchangeable with $c_i$. For example, both *eraser* and *pencil* are suggested to have similar spatial context to *pen* by our spatial embedding model. However, *pencil* is obviously more interchangeable with *pen* than *eraser* as they have more similar sizes.

For a specific category, only a few categories are actually interchangeable with it. Thus, to achieve more meaningful interchangeability and to reduce the computation, the degree of interchangeability is only measured for those pairs of categories with similar spatial context obtained by $L(c_i, c_j)$. Denote by $\boldsymbol{knn}_{c_i}$ the set of $K$ nearest neighbor categories of $c_i$, i.e. $\boldsymbol{knn}_{c_i} = \{c_j | rank(L(c_i, c_j)) \leq K, \forall j\}$, where $rank(\cdot)$ gives the order of the number. We define the degree of interchangeability of two categories $c_i$ and $c_j$ as the weighted sum of their spatial context similarity and their size similarity:

$$S(c_i, c_j) = \mathbb{1}(c_j \in \boldsymbol{knn}_{c_i})[\alpha_1 s_o(c_i, c_j) + \alpha_2 e^{-\beta_2 |d(c_i) - d(c_j)|}], \tag{16}$$

where $\mathbb{1}(c_j \in \boldsymbol{knn}_{c_i})$ ensures that interchangeability is only measured between $c_i$ and the categories in $\boldsymbol{knn}_{c_i}$. $s_o(c_i, c_j)$ is the spatial context similarity which will be detailed shortly. The last term in Eq. 16 accounts for the size difference, where $d(c_i)$ is the average diagonal length of the bounding boxes of all 3D objects in category $c_i$, which is used as the size of $c_i$. $\beta_2 = 4$ and $K = 10$ are used in our experiments empirically. Specifically, $\beta_2$ is chosen to ensure that a small difference of $d(c_i)$ and $d(c_j)$ leads to a similarity value very close to 1 while a large difference leads to a quite small similarity value. $\alpha_1$ and $\alpha_2$ are both set to 0.5 so that the two terms are equally weighted.

As the categories in $\boldsymbol{knn}_{c_i}$ are very close to $c_i$ in the spatial embedding space, their spatial context similarities with $c_i$ are with quite similar values, e.g., all slightly larger than 0.9, which are not well differentiated. Thus, the spatial context similarity of two categories $c_i$ and $c_j \in \boldsymbol{knn}_{c_i}$ is normalized to $0 \sim 1$ as follows:

$$s_o(c_i, c_j) = 1 - \frac{rank(c_j)}{K + 1}, \tag{17}$$

where $rank(c_j) \in [1, K]$ is the order of $c_j$ in $\boldsymbol{knn}_{c_i}$.

### 6.4 Propagation weights

The interchangeability between two pairs of categories $(c_i, c_j)$ and $(c_k, c_m)$ can then be measured as the product of the degree of interchangeability of corresponding categories:

$$w((c_i, c_j), (c_k, c_m)) = S(c_i, c_k) S(c_j, c_m), \tag{18}$$

which corresponds to $w_{pq}$ in Eq. 10. Since only interchangeability between each category and their top-$K$ neighbor categories are measured, the weighting matrix $\boldsymbol{W}$ is very sparse, which makes the propagation of user preferences more effective and efficient.

## 7 Evaluation and Results

The offline prior knowledge learning is implemented in Python and the online active arrangement is implemented in C++. Our experiments were carried out on a laptop with a 2.50 GHz Intel Core i5-7300HQ CPU and 8 GB RAM.

The experiments are carried out as follows: Sec. 7.1 shows that our framework can generate small object arrangements that can at least meet all the user-specified preferences. Afterwards, in Sec. 7.2, we demonstrate the

effectiveness of our interchangeability metric between small object categories, which can thus be reliably used to propagate the user preferences to other categories, as shown in Sec. 7.3. Lastly, we compare our framework with the most related work [5] in a few different aspects in Sec. 7.4.

### 7.1 User preferences

In this experiment, we show that all the preferences specified by users can be effectively employed by our framework to generate arrangements. In this experiment, we do not focus on demonstrating propagation.

Fig. 7 demonstrates the arrangement results for a cabinet and a desk in (a)-(e) and (f)-(i), respectively. Figs. 7 (a) and (f) show the arrangements generated using the initial priority probabilities learned from the dataset, i.e., $\tilde{\mathbf{R}}$. Afterwards, one user preference w.r.t. one type of spatial relations is applied to update the previous arrangement. For example, (b) is generated by applying the user preference 'H: *book* $>$ *photo frame* ', which states that "*book* should be placed higher ($>$) than *photo frame*" on the cabinet in (a), where the two small objects are highlighted with red bounding boxes. As shown in (a) and (b), after applying this user preference, *book* has been accordingly moved above *photo frame*. Based on (b), another user preference 'H: *doll* $<$ *book*' is applied to generate (c). (d) and (e) are generated based on (c) and (d) respectively by applying another two user preferences for height priority. It should be noted that when users provide a new preference, the previous ones are also retained. As a result, the arrangement in (e) not only meets the user preference specified on (d), which is 'H: *notebook* $>$ *trophy*', but also meets the previous three preferences. Similarly, (g), (h) and (i) are the arrangements for a desk generated by applying another three different user preferences in terms of middleness (M) and depth (D) on (f), (g) and (h), respectively. (i) also meets all the three user preferences.

### 7.2 Interchangeability metric

In this experiment, we evaluate the effectiveness of our proposed interchangeability metric (Eq. 16) which considers both spatial context similarity and size similarity by comparing it with the metric merely calculated by spatial embedding (Eq. 17) or Word2Vec [35].

Specifically, for Word2Vec, we use a pre-trained word embedding model [33] which is trained from a public corpus containing 100 billion words using the negative sampling method [35]. The model produces 300-dimensional vectors for 3 million words and phrases. To allow comparison between our spatial interchangeability metric and the one calculated by applying Word2Vec, all the names of small object categories in $\mathbf{C}$ are mapped to the closest words found in the vocabulary list in the Word2Vec model.

Next, we carried out a user study to demonstrate the effectiveness of our interchangeability metric for small object categories. As shown in Fig. 8, starting from an initial set of small objects that have been arranged on a furniture object (a), three sets of small objects are generated by replacing each small object in the initial scene with a small object from the most interchangeable category suggested by different metrics (b-d). For Word2Vec metric (b), the cosine similarity between the 300-dimensional vectors of categories is used while for our "Spatial Embedding" metric (c) and "Spatial Embedding+Size" metric (d), Eq. 17 and Eq. 16 are used respectively to measure the degree of interchangeability.

Table 3
Statistical Comparison of Interchangeability Metrics

| $p$-value | S.E.+Size vs Word2Vec | S.E.+Size vs S.E. |
|---|---|---|
| Desk | $2.01 \times 10^{-5}$ | $1.21 \times 10^{-6}$ |
| Dining Table | $2.79 \times 10^{-7}$ | $1.66 \times 10^{-3}$ |
| TV Stand | $5.63 \times 10^{-9}$ | $1.69 \times 10^{-3}$ |

*S.E. is short for Spatial Embedding. p-values of t-test between the Spatial Embedding+Size metric (Eq. 16) against Word2Vec and Spatial Embedding metric (Eq. 17) are reported.*

In the user study, three sets of rendered images for each scene including Desk, Dining Table and TV Stand, were obtained by the three metrics and presented to users. In each set, users were provided with the image for the initial scene arranged with the initial set of small objects (a) and were asked to rank the three images for the same scene arranged with replaced sets of small objects (b-d) based on two criteria, including the plausibility of the small object replacement and the spatial similarity to the initial scene. The three rendered images for the same scene were displayed in a random order to avoid bias. To quantify the results, for each image, its rankings of 1st, 2nd and 3rd from a user are mapped to scores of 3, 2, 1 respectively. 60 users participated in this user study. The average ranking scores w.r.t. different scenes generated by the three metrics with $95\%$ confidence intervals are shown in Fig. 9.

In Fig. 9, our interchangeability metric involving both spatial context similarity and size similarity (Eq. 16) consistently achieves the highest score in all the three scenes. Specifically, given a category $c$, Word2Vec suggests a category with most similar semantics to $c$, but it may not necessarily be interchangeable when assembling a 3D scene. For example, in the Dining Table scene shown in Fig. 8 (b), Word2Vec suggests *coffee machine* for *coffee* in the initial scene, which is semantically similar while spatially improper. For the metric with merely spatial embedding (Eq. 17), although it can suggest categories with similar spatial context, it may generate results with less resemblance of the spatial distribution as the initial scenes. For example, in Fig. 8 (c), the result generated by "Spatial Embedding" metric for the Desk scene is too empty compared to the initial scene, while the results for Dining Table and TV Stand contain occasional collisions between small objects, e.g., *flower* and *fruit* for the Dining Table. By considering the similarity of both spatial context and size, the replaced sets of small objects in Fig. 8 (d) are not only plausibly interchangeable with the initial sets of categories but also with similar sizes to those in initial scenes to avoid emptiness or collision. As a result, the results in Fig. 8 (d) are most preferred by the participants in our user study.

A two-sample $t$-test was carried out to compare the user ranking scores of our interchangeability metric (Eq. 16) with Word2Vec and the metric with merely spatial embedding (Eq. 17). As shown in Table 3, our metric that considers both spatial context similarity and size similarity is superior to

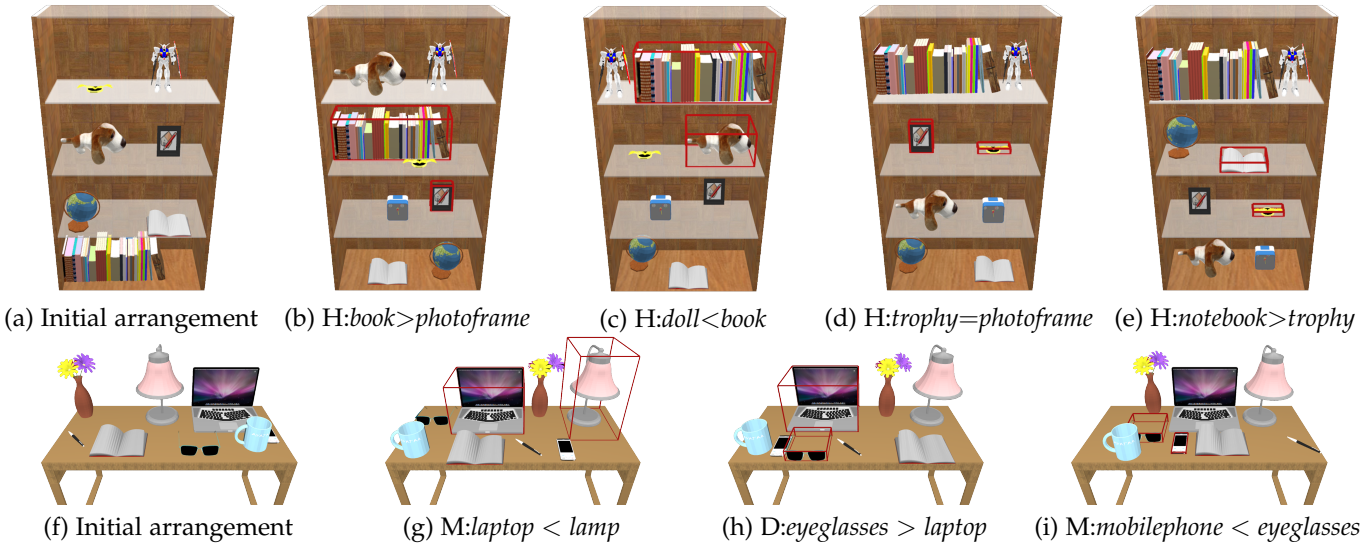



Figure 7. Effectiveness of employing user preferences. (a)-(e) are arrangement results for a cabinet, where (a) is the arrangement generated by applying priority probabilities learned from the dataset, and (b)-(e) are the arrangements generated in sequence by applying an additional user preference for height priority to the previous arrangements. The arrangement in (e) meets all the four user preferences specified. Similarly, (f)-(i) are arrangement results for a desk, where (f) is the initial arrangement and (g)-(i) are those generated by applying user preferences. (i) meets all the three user preferences. H, M and D denote preferences for height, middleness and depth priorities respectively, as detailed in Sec. 4.2.

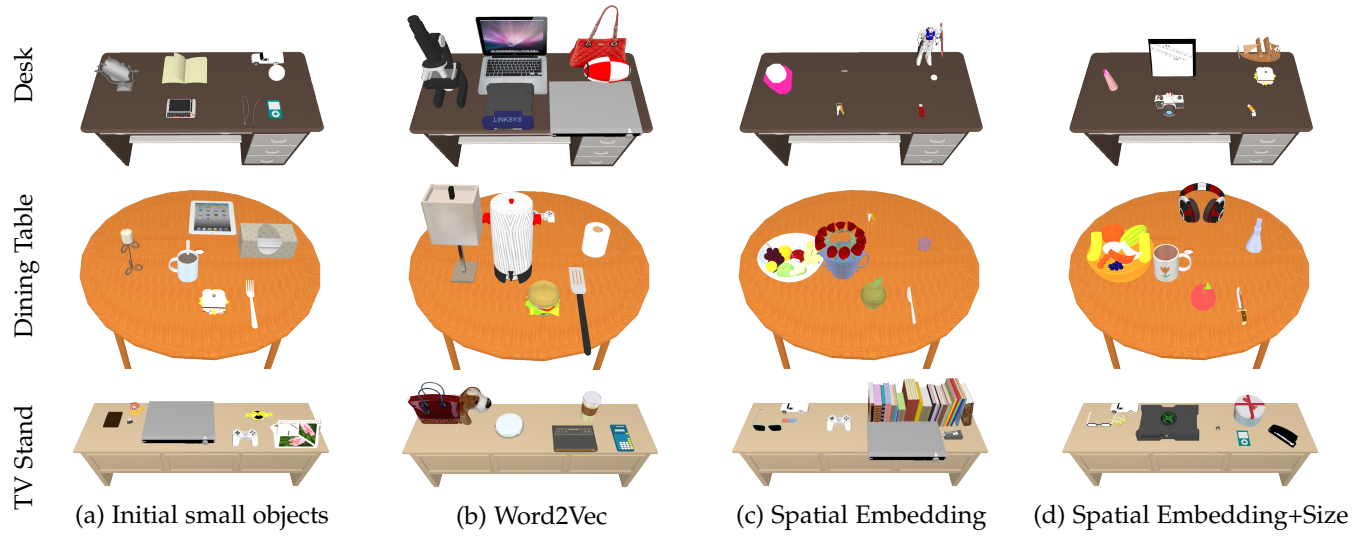



Figure 8. Comparison between different small object category interchangeability metrics.

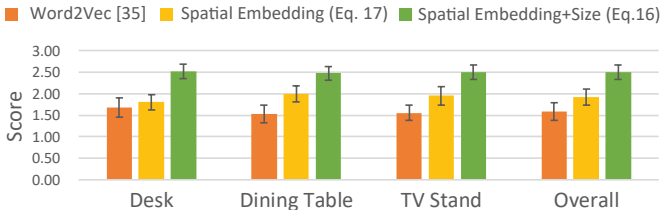
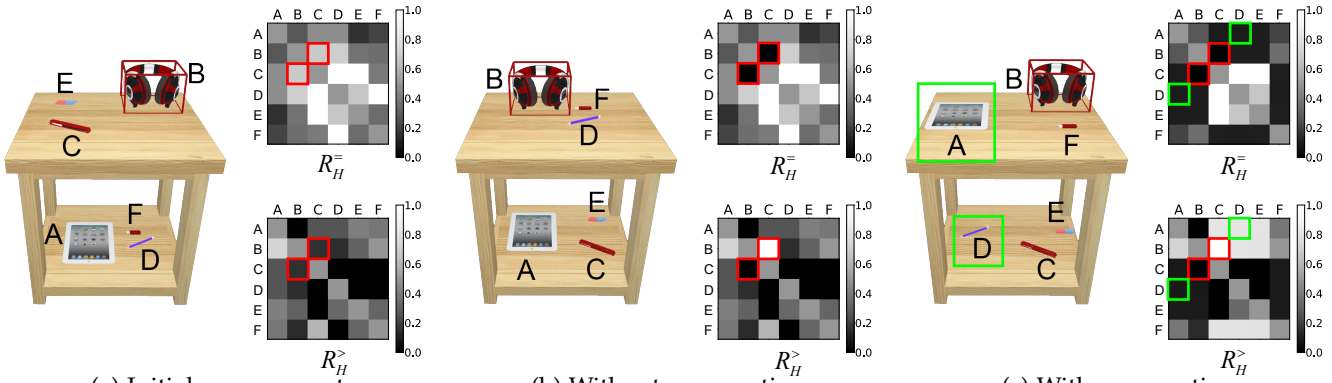



Figure 9. Average ranking scores for different scenes generated by Word2Vec metric [35], Spatial Embedding metric (Eq. 17) and Spatial Embedding+Size metric (Eq. 16) with 95% confidence intervals.

the other two metrics at $p = 0.01$ level in all cases.

### 7.3 User preference propagation

Through our active learning process, we can effectively propagate user preferences on spatial relations of a few categories to the entire set of categories as shown in Fig. 10. Specifically, (a) is the initial arrangement generated by applying the initial spatial relation matrices learned from the dataset, i.e., $\tilde{\mathbf{R}}$, and the two matrices about height priority probabilities, i.e., $\boldsymbol{R}_{\mathrm{H}}^{=}$ and $\boldsymbol{R}_{\mathrm{H}}^{>}$, are shown together with the result. From the initial arrangement result, a user specified a spatial relation preference as 'Height: *headphone* (B) > *pen* (C)'. The two categories were originally placed on the same layer in (a). (b) shows the result of directly applying user preferences without propagation, which only updated the probabilities related to the user preference in the two spatial relation matrices $\boldsymbol{R}_{\mathrm{H}}^{=}$ and $\boldsymbol{R}_{\mathrm{H}}^{>}$, including $\boldsymbol{R}_{\mathrm{H}}^{=}(\mathrm{B},\mathrm{C}) = \boldsymbol{R}_{\mathrm{H}}^{=}(\mathrm{C},\mathrm{B}) = 0$, $\boldsymbol{R}_{\mathrm{H}}^{>}(\mathrm{B},\mathrm{C}) = 1$ and $\boldsymbol{R}_{\mathrm{H}}^{>}(\mathrm{C},\mathrm{B}) = 0$, as




(a) Initial arrangement (b) Without propagation (c) With propagation

Figure 10. The effects of propagating user preferences. (a) is the arrangement result using priority probabilities learned from the dataset. Given a user preference Height:*headphone* > *pen*, (b) and (c) are the results of applying user preference without and with propagation, respectively. The two matrices $\boldsymbol{R}_{\mathrm{H}}^{=}$ and $\boldsymbol{R}_{\mathrm{H}}^{>}$ used for generating arrangements are shown correspondingly with the results on the right. A to F denote the six categories shown in the scene, including *tablet*, *headphone*, *pen*, *pencil*, *eraser* and *usb*.

highlighted in the red boxes on the two matrices in (b). As can be seen in (b), *pen* is moved to the lower layer to meet the user preference by using the updated matrices to generate arrangement. In comparison, (c) shows the result of propagating user preferences to other categories. In the two matrices shown in (c), some other probabilities are also updated due to the active learning process. For example, *tablet* (A) and *pencil* (D) are considered to be spatially interchangeable with *headphone* (B) and *pen* (C) respectively by our method, thus, the user preference towards B and C will strongly affect the probabilities relating A and D, as highlighted in the green boxes in (c). As a result, in the arrangement result in (c), not only *headphone* is placed higher than *pen*, but also *tablet* is placed higher than *pencil*.

Fig. 11 demonstrates more examples of propagating user preferences for three different scenes. (a), (e) and (i) in the first column show the arrangement results generated using the initial spatial relation matrices $\tilde{\mathbf{R}}$. The rest three columns are the arrangement results generated by applying and propagating a new user preference over the previous arrangement results. For example, (b) is the arrangement result after applying the user preference 'H:*notebook* > *fruit*' on (a). Similar to Fig. 10, in each image, red bounding boxes denote the small objects corresponding to the user specified preference and green boxes denote those small object categories that are strongly affected by the active learning process. For example, after propagating the user preference 'D: *notebook* < *usb*' on (b), the arrangement result in (c) shows that both *notebook* and *calendar* are placed behind *usb* and *ipod* respectively. It proves that the preference of *notebook* and *usb* is successfully propagated to *calendar* and *ipod* due to their high interchangeability. The same observation also applies to the height priority. For example, the result in (k) shows that after propagating preference 'H: *milk* > *headphone*', not only *milk* is placed higher than *headphone*, but also *teapot* is placed higher than *loudspeaker* at the same time.

### 7.4 Comparison with [5]

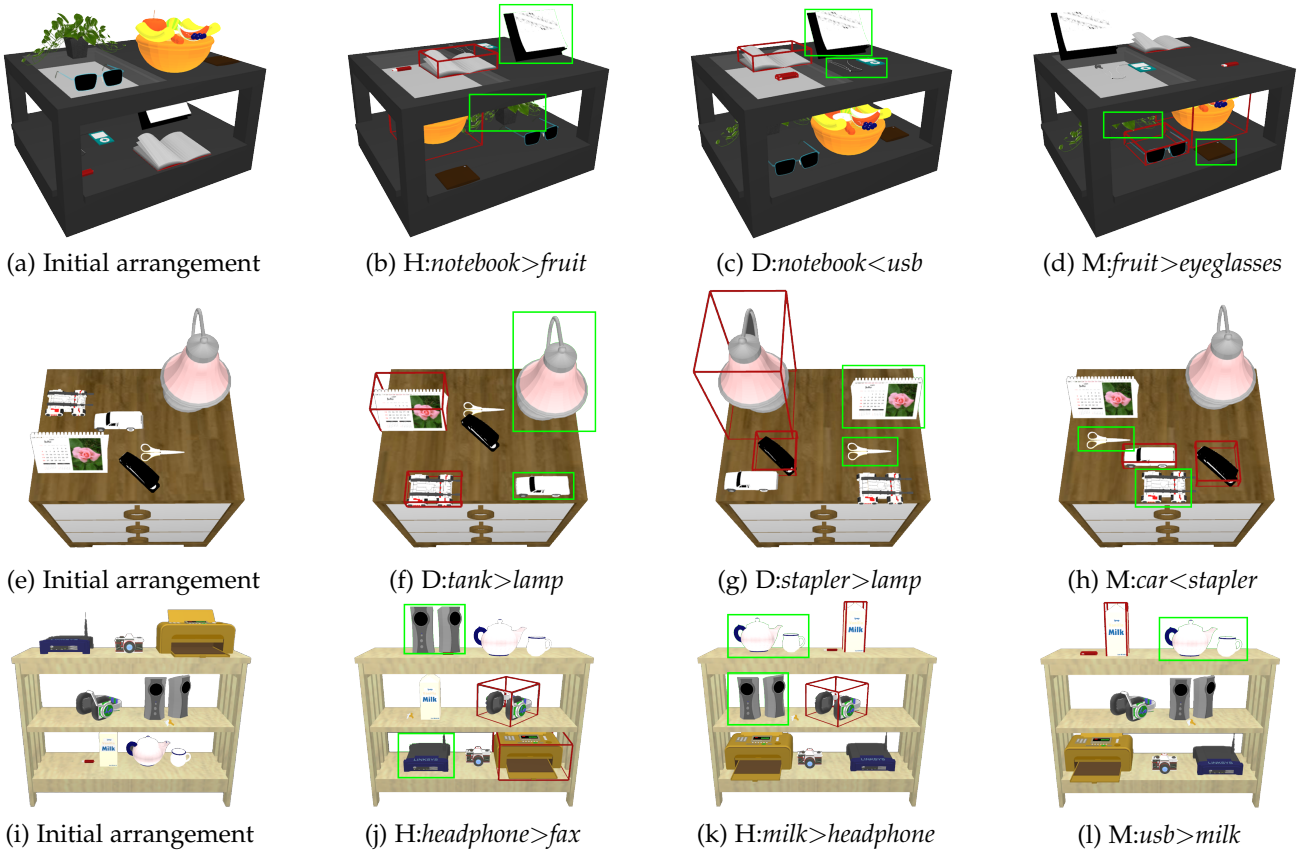
Our method differs from [5] in the following three aspects.

**Prior knowledge.** With the prior knowledge about spatial relations for all small object categories learned from the dataset, our method can generate plausible arrangements even without any user intervention, as shown in Fig. 12 (b). Comparatively, [5] can merely generate arrangement results that only meet the hard constraints to avoid collision and out-of-boundary cases, as shown in Fig. 12 (a). To achieve arrangements of similar quality in Fig. 12 (b), [5] requires users to exhaustively assign an absolute value for each small object category to denote their priorities for arrangement, which is time-consuming and laborious.

**User interface.** In the sense of interaction, although both our method and [5] allow users to specify preferences, we provide a friendly user interface to help users to employ their preferences more conveniently, as shown in Fig. 3. In [5] however, no such user interface was provided, which made it very difficult for users to specify the absolute priorities for each small object category.

**Active learning.** As our method utilizes active learning to propagate the user preferences on a few small object categories to the whole set of categories, satisfactory arrangements can usually be generated within a few user interactions, as shown in Fig. 11. However, in [5], users need to modify the priority values for all related small object categories. For example, in Fig. 10, our method can generate the arrangement shown in (c) by only providing one preference on (a), which is 'H: *headphone* > *pen*'. The other two categories, e.g. *tablet* and *pencil* are adapted accordingly as a result of propagation. In [5], from (a) to (c), users need to explicitly modify the height priorities for all the four categories, which involves much more human labor than our method. To generate similar results in Fig. 11, [5] requires users to modify the priority values for at least 4 categories for each user preference while our method only needs one user interaction each time.

**User study.** A user study was conducted to compare the effectiveness of our method and the method in [5]. 12 users were invited to arrange a same set of small objects for each furniture shown in Fig. 11 using the two methods. Specifically, a user was first provided with an initial arrangement on the furniture. Then, they were asked to figure out an ideal arrangement in their mind and to adjust the initial arrangement to meet their desired arrangement as much as possible using both methods. With our method, the



(a) Initial arrangement (b) H:*notebook>fruit* (c) D:*notebook<usb* (d) M:*fruit>eyeglasses*

(e) Initial arrangement (f) D:*tank>lamp* (g) D:*stapler>lamp* (h) M:*car<stapler*

(i) Initial arrangement (j) H:*headphone>fax* (k) H:*milk>headphone* (l) M:*usb>milk*

Figure 11. Effectiveness of propagating multiple user preferences. The first column shows the initial arrangements generated by using prior knowledge learned from the dataset. The rest three columns are the arrangement results generated by applying and propagating the user preferences shown under each image over the previous arrangement results. H, M and D denote preferences for height, middleness and depth priorities respectively.

(a) (b)

Figure 12. The comparison between the initial arrangements generated by [5] (a) and our method (b).

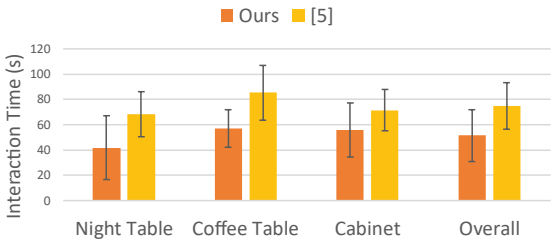



Figure 13. Average interaction time for different scenes generated by our method and method in [5] with 95% confidence intervals.

user can specify preferences and apply them to update the arrangement. With the method in [5], the user can modify the absolute priority values for each small object category to update the arrangement. Both processes are iterated until the users get their desired arrangement results. All the arrangement results from the 12 users can be found in the supplementary material.

The interaction time of specifying preferences or modifying priorities is recorded separately for the two methods. The average interaction time of all the users for the three furniture objects with 95% confidence intervals is plotted in Fig. 13. A two-sample $t$-test shows that the time spent using our method is shorter than [5] at $p = 0.01$ level for the night table ($p = 0.009$) and the coffee table ($p = 0.006$), and at $p = 0.05$ level for the cabinet ($p = 0.01$).

## 8 Discussion and Future Work

Our framework provides users with an effective way to arrange small objects according to their preferences in terms of relative spatial relations of small object categories. As our analysis is based on small object categories instead of 3D object instances [4], our framework can potentially achieve higher variety for small object arrangements than their method by using different 3D small objects from the same category. Moreover, our framework can be used to assist enhancing the output scenes of the furniture layout methods [28], [29], [36] by placing small objects to increase

the realism. Our framework can be improved in a few ways as follows.

Firstly, our framework targets the small object arrangement problem which involves the relative relations between small objects that can be converted to pairwise spatial priority probabilities, such as height, middleness and depth priorities as detailed in Sec. 4.2. Users can define their preference for small object arrangement interactively with our framework by manipulating the spatial relations between any two small objects. However, at the current stage, our framework cannot support advanced spatial relations such as "surrounded by" [37], which cannot be simply interpreted as multiple pairwise relations. It would be interesting to include these relations by proposing another effective data form, e.g. scene graph [12], to store the spatial relations as well as to enable user manipulation.

Secondly, our framework assumes that all the furniture objects for placing small objects have only one front direction, such as the Desk and TV Stand shown in Fig. 8. However, in real life, some furniture objects might have multiple front directions such as the round Dining Table in Fig. 8. In such cases, our current solution is to specify one direction as the front direction so that all the front directions of small objects will be aligned with this direction. In the future, we would like to explore how to arrange small objects according to the geometric shape of furniture objects.

Thirdly, in our framework, users need to specify preferences between two small objects manually from the right panel shown in Fig. 3, which is not the most natural way of interaction. In the future, preferences between small objects can be inferred through users' direct manipulations on the small objects. For example, swapping two small objects on different layers of a cabinet can be interpreted as one of them should be placed higher than the other one.

Fourthly, without taking into account appearance, geometry or orientation properties of small objects leads to less aesthetic results. It would be a promising and interesting direction to explore how to incorporate these factors into the small object arrangement algorithms similar to those in [38], [39], [40].

Lastly, due to the insufficiency of data, we use the same relation matrices for all the furniture categories. As a result, currently our framework cannot automatically differentiate the small object arrangements for different furniture categories which have completely different small object relations. However, our framework provides the users with the flexibility to adjust the arrangement according to their preferences through a few interactions. We are interested in mining the relations of small objects separately for each furniture category in the future.

## 9 Conclusions

We proposed an interactive active small object arrangement framework that reduces the human labor required to obtain customized small object arrangements in 3D indoor scenes. Our framework employs both the prior knowledge learned from a 3D indoor scene dataset and user specified preferences for small object arrangement. We first learn the prior knowledge by inferring the spatial priority probabilities between any two small object categories with our probability mining method. Then, to reduce user effort, we effectively propagate the user preferences on a few small object categories to all other categories using active learning. The propagation weights are determined by the degree of interchangeability between small object categories, which are calculated based on our spatial embedding model learned from the neighborhood information. Our experiments demonstrate the effectiveness of both the prior knowledge and the active learning based preference propagation for generating small object arrangements. We expect our framework to help users to arrange small objects with their preferences in a more efficient and effective way.

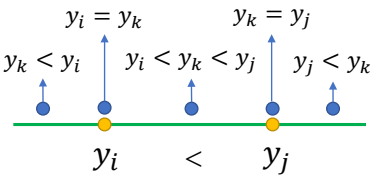



Figure 14. The relation of $y_i, y_j$ and $y_k$.

## Acknowledgements

This work was supported by the National Natural Science Foundation of China (61373070), NSF (award 1813583), and Tsinghua-Kuaishou Institute of Future Media Data.

## Appendix A
## Derivation of Equations 4, 5 and 6

For clarity and to be consistent with the symbols in Sec. 4.3.1, we use $y_X(c_i), y_X(c_j)$ and $y_X(c_k)$ to denote three random variables which represent the property value on any one of the three spatial relations for three different small object categories $c_i$, $c_j$ and $c_k$, respectively. For simplicity, we use $y_i$ in short for $y_X(c_i)$.

Assume $c_k$ co-occurs with both $c_i$ and $c_j$, the probability of the inequality relation between $y_i$ and $y_j$ can thus be calculated through $y_k$ according to the *sum rule* of probability:

$$\begin{aligned} p(y_i < y_j) &= \sum_{y_k} p(y_i < y_j, y_k) \\ &= p(y_k < y_i < y_j) \ + \ p(y_i = y_k < y_j) \ + \\ &\quad p(y_i < y_k < y_j) \ + \ p(y_i < y_k = y_j) \ + \\ &\quad p(y_i < y_j < y_k), \end{aligned} \tag{19}$$

where $p(y_i < y_j)$ can be considered as the marginal probability. The summation in Eq. 19 is expanded by enumerating all the situations that $y_k$ can have with $y_i < y_j$ fixed, as illustrated by Fig. 14.

There are more than one category that can co-occur with $c_i$ and $c_j$, the co-occurred category set is denoted by $\mathbf{K}_{c_i,c_j}$. Each $c_k \in \mathbf{K}_{c_i,c_j}$ can be used to calculate $p(y_i < y_j)$ with Eq. 19. To utilize the whole set of categories that co-occur with $c_i$ and $c_j$, $p(y_i < y_j)$ is estimated by the weighted

results of using each corresponding $y_k$ for $c_k \in \mathbf{K}_{c_i,c_j}$, as shown in below:

$$\begin{aligned} p(y_i < y_j) = & \frac{1}{\sum p(c_k)} \sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)[\, p(y_k < y_i < y_j) + \\ & p(y_i = y_k < y_j) \;+\; p(y_i < y_k < y_j) + \\ & p(y_i < y_k = y_j) \;+\; p(y_i < y_j < y_k)\,], \end{aligned} \tag{20}$$

where $\sum p(c_k)$ is short for $\sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)$, which is used to normalize $p(y_i < y_j)$.

We assume that the relation between a pair of any two variables is independent from a different pair, e.g., $y_k < y_i$ is independent from $y_i < y_j$. Therefore, each single term in the summation of Eq. 20 can be derived by applying the *product rule* of probability. For example, $p(y_k < y_i < y_j)$ can be derived as follows:

$$\begin{aligned} p(y_k < y_i < y_j) &= p(y_k < y_i, y_i < y_j) \\ &= p(y_k < y_i \mid y_i < y_j) p(y_i < y_j) \\ &= p(y_k < y_i) p(y_i < y_j) \end{aligned} \tag{21}$$

Expanding all items in Eq. 20, we can achieve:

$$\begin{aligned} p(y_i < y_j) = & \frac{1}{\sum p(c_k)} \sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)[\, p(y_k < y_i) p(y_i < y_j) + \\ & p(y_i = y_k) p(y_k < y_j) \;+\; p(y_i < y_k) p(y_k < y_j) + \\ & p(y_i < y_k) p(y_k = y_j) \;+\; p(y_i < y_j) p(y_j < y_k)\,]. \end{aligned} \tag{22}$$

In Eq. 22, $p(y_i < y_j)$ is irrelevant to the choice of $c_k$, by moving it outside of the summation, we can further derive:

$$\begin{aligned} & p(y_i < y_j) = \\ & \frac{p(y_i < y_j)}{\sum p(c_k)} \sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)[\, p(y_k < y_i) + p(y_j < y_k)] + \\ & \frac{1}{\sum p(c_k)} \sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)[\, p(y_i = y_k) p(y_k < y_j) + \\ & \quad p(y_i < y_k) p(y_k < y_j) \;+\; p(y_i < y_k) p(y_k = y_j)\,]. \end{aligned} \tag{23}$$

Moving all terms with $p(y_i < y_j)$ to the left side of the equation then multiplying each side with $\sum p(c_k)$, we can obtain:

$$\begin{aligned} & p(y_i < y_j) \sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)[\, 1 - p(y_k < y_i) - p(y_j < y_k)\,] \\ & = \sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)[\, p(y_i = y_k) p(y_k < y_j) + \\ & \qquad p(y_i < y_k) p(y_k < y_j) \;+\; p(y_i < y_k) p(y_k = y_j)\,]. \end{aligned} \tag{24}$$

In our paper, we use symbol $\boldsymbol{R}^{>}_{X}(c_j, c_i)$ to denote $p(y_i < y_j)$. For short, we use $\boldsymbol{R}^{>}_{j,i}$ to denote $\boldsymbol{R}^{>}_{X}(c_j, c_i)$ and omit the subscript of the summation. Therefore, Eq. 5 can be derived from Eq. 24 by first dividing the coefficient on the left side of Eq. 24 then substituting the symbols:

$$\boldsymbol{R}^{>}_{j,i} = \frac{\sum p(c_k)(\, \boldsymbol{R}^{>}_{j,k} \boldsymbol{R}^{=}_{i,k} + \boldsymbol{R}^{>}_{j,k} \boldsymbol{R}^{>}_{k,i} + \boldsymbol{R}^{=}_{j,k} \boldsymbol{R}^{>}_{k,i})}{\sum p(c_k)(\, 1 - \boldsymbol{R}^{>}_{i,k} - \boldsymbol{R}^{>}_{k,j}\,)}.$$

Eq. 4 is a symmetric form of Eq. 5, which can be obtained by swapping $i$ and $j$ in Eq. 5.

For the equality relations, $p(y_i = y_j)$ can be derived similarly as follows:

$$\begin{aligned} p(y_i = y_j) &= \frac{1}{\sum p(c_k)} \sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k)[\, p(y_k < y_i = y_j) + \\ &\qquad p(y_i = y_k = y_j) + p(y_i = y_j < y_k)\,] \\ &= \frac{p(y_i = y_j)}{\sum p(c_k)} \sum p(c_k)[p(y_k < y_i) + p(y_j < y_k)] \\ &\quad + \frac{1}{\sum p(c_k)} \sum_{c_k \in \mathbf{K}_{c_i,c_j}} p(c_k) p(y_i = y_k) p(y_k = y_j) \end{aligned} \tag{25}$$

Substituting $p(y_i = y_j)$ with $\boldsymbol{R}^{=}_{i,j}$, we can obtain Eq. 6:

$$\boldsymbol{R}^{=}_{i,j} = \frac{\sum p(c_k) \boldsymbol{R}^{=}_{i,k} \boldsymbol{R}^{=}_{k,j}}{\sum p(c_k)(1 - \boldsymbol{R}^{>}_{i,k} - \boldsymbol{R}^{>}_{k,j})}$$

It should be noted that the derivation is not unique, as there are different ways to write the probability of relations containing equality relations. For example, $p(y_i < y_k = y_j)$ can also be written in $p(y_i < y_j = y_k)$, which will lead to a different formulation. However, these formulations should be equivalent to each other and can all be used to estimate $p(y_i < y_j)$.

## REFERENCES


[1] L. Yu, S. K. Yeung, and D. Terzopoulos, "The clutterpalette: an interactive tool for detailing indoor scenes," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 22, no. 2, pp. 1138–1148, 2016.

[2] M. Shinya and M.-C. Forgue, "Laying out objects with geometric and physical constraints," *The Visual Computer*, vol. 11, no. 4, pp. 188–201, 1995.

[3] L. Majerowicz, A. Shamir, A. Sheffer, and H. H. Hoos, "Filling your shelves: synthesizing diverse style-preserving artifact arrangements," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 20, no. 11, pp. 1507–1518, 2014.

[4] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, and P. Hanrahan, "Example-based synthesis of 3D object arrangements," *ACM Trans. on Graphics*, vol. 31, no. 6, pp. 135:1–135:11, 2012.

[5] S. Zhang, Z. Han, and H. Zhang, "User guided 3d scene enrichment," in *Proceedings of the ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry-Volume 1*, 2016, pp. 353–362.

[6] B. Settles, "Active learning literature survey," University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2009.

[7] X. Zhu, J. Lafferty, and Z. Ghahramani, "Combining active learning and semi-supervised learning using gaussian fields and harmonic functions," in *ICML workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, vol. 3, 2003.

[8] Z. S. Kermani, Z. Liao, P. Tan, and H. Zhang, "Learning 3D scene synthesis from annotated rgb-d images," in *Computer Graphics Forum*, vol. 35, no. 5, 2016, pp. 197–206.

[9] Y. Jiang, M. Lim, and A. Saxena, "Learning object arrangements in 3D scenes using human context," *arXiv preprint arXiv:1206.6462*, 2012.

[10] K. Wang, M. Savva, A. X. Chang, and D. Ritchie, "Deep convolutional priors for indoor scene synthesis," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, p. 70, 2018.

[11] M. Savva, A. X. Chang, and M. Agrawala, "Scenesuggest: Context-driven 3d scene design," *arXiv preprint arXiv:1703.00061*, 2017.

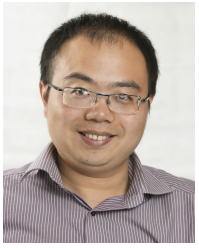
[12] R. Ma, H. Li, C. Zou, Z. Liao, X. Tong, and H. Zhang, "Action-driven 3D indoor scene evolution," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 173:1–173:13, 2016.

[13] L. Gao, Y.-P. Cao, Y.-K. Lai, H.-Z. Huang, L. Kobbelt, and S.-M. Hu, "Active exploration of large 3D model repositories," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 21, no. 12, pp. 1390–1402, 2015.

[14] A. Top, G. Hamarneh, and R. Abugharbieh, "Active learning for interactive 3D image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2011, pp. 603–610.
[15] L. Yi, V. G. Kim, D. Ceylan, I. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, L. Guibas *et al.*, "A scalable active framework for region annotation in 3D shape collections," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 6, p. 210, 2016.
[16] M. Song, Z. Sun, K. Liu, and X. Lang, "Iterative 3D shape classification by online metric learning," *Computer Aided Geometric Design*, vol. 35, pp. 192–205, 2015.
[17] M. Song, Z. Sun, and H. Li, "Accumulative categorization: Online 3D shape classification for progressive collections," *Graphical Models*, vol. 89, pp. 14–27, 2017.
[18] Y. Wang, S. Asafi, O. Van Kaick, H. Zhang, D. Cohen-Or, and B. Chen, "Active co-analysis of a set of shapes," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 6, p. 165, 2012.
[19] J. Weston, S. Bengio, and N. Usunier, "Large scale image annotation: learning to rank with joint word-image embeddings," *Machine learning*, vol. 81, no. 1, pp. 21–35, 2010.
[20] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov *et al.*, "Devise: A deep visual-semantic embedding model," in *Advances in neural information processing systems*, 2013, pp. 2121–2129.
[21] S. Bell and K. Bala, "Learning visual similarity for product design with convolutional neural networks," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, p. 98, 2015.
[22] J. Dong, X. Li, and C. G. Snoek, "Word2VisualVec: Image and video to sentence matching by visual feature prediction," *CoRR, abs/1604.06838*, vol. 2, 2016.
[23] O. Fried, S. Avidan, and D. Cohen-Or, "Patch2Vec: Globally consistent image patch representation," in *Computer Graphics Forum*, vol. 36, no. 7, 2017, pp. 183–194.
[24] R. Herzog, D. Mewes, M. Wand, L. Guibas, and H.-P. Seidel, "LeSSS: Learned shared semantic spaces for relating multi-modal representations of 3D shapes," in *Computer Graphics Forum*, vol. 34, no. 5, 2015, pp. 141–151.
[25] F. P. Tasse and N. Dodgson, "Shape2Vec: semantic-based descriptors for 3D shapes, sketches and images," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 6, p. 208, 2016.
[26] Y. Li, H. Su, C. R. Qi, N. Fish, D. Cohen-Or, and L. J. Guibas, "Joint embeddings of shapes and images via CNN image purification." *ACM Trans. Graph.*, vol. 34, no. 6, pp. 234–1, 2015.
[27] L. M. Busse, M. H. Chehreghani, and J. M. Buhmann, "Approximate sorting of preference data," in *NIPS workshop of Choice Models and Preference Learning*, 2011.
[28] L.-F. Yu, S.-K. Yeung, C.-K. Tang, D. Terzopoulos, T. F. Chan, and S. J. Osher, "Make it home: automatic optimization of furniture arrangement," *ACM Trans. on Graphics*, vol. 30, no. 4, pp. 86:1–86:12, 2011.
[29] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, and V. Koltun, "Interactive furniture layout using interior design guidelines," *ACM Trans. on Graphics*, vol. 30, no. 4, pp. 87:1–87:10, 2011.
[30] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The journal of chemical physics*, vol. 21, no. 6, pp. 1087–1092, 1953.
[31] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.
[32] Y. Aytar, L. Castrejon, C. Vondrick, H. Pirsiavash, and A. Torralba, "Cross-modal scene networks," *arXiv preprint arXiv:1610.09003*, 2016.
[33] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.
[34] L. Van Der Maaten, "Learning a parametric embedding by preserving local structure," *RBM*, vol. 500, no. 500, p. 26, 2009.
[35] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.
[36] Q. Fu, X. Chen, X. Wang, S. Wen, B. Zhou, and H. Fu, "Adaptive synthesis of indoor scenes via activity-associated object relation graphs," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 201:1–201:13, 2017.
[37] R. Ma, A. G. Patil, M. Fisher, M. Li, S. Pirk, B.-S. Hua, S.-K. Yeung, X. Tong, L. Guibas, and H. Zhang, "Language-driven synthesis of 3d scenes from scene databases," *ACM Trans. Graph.*, vol. 37, no. 6, pp. 212:1–212:16, 2018.
[38] X. Chen, J. Li, Q. Li, B. Gao, D. Zou, and Q. Zhao, "Image2Scene: transforming style of 3D room," in *Proceedings of ACM MM*, 2015, pp. 321–330.
[39] S. Lin, D. Ritchie, M. Fisher, and P. Hanrahan, "Probabilistic color-by-numbers: suggesting pattern colorizations using factor graphs," *ACM Trans. on Graphics*, vol. 32, no. 4, pp. 37:1–37:12, 2013.
[40] S. Zhang, Z. Han, R. R. Martin, and H. Zhang, "Semantic 3D indoor scene enhancement using guide words," *The Visual Computer*, vol. 33, no. 6, pp. 925–935, Jun 2017.

**Suiyun Zhang** received her B.Sc. degree from School of Software, Sun Yat-Sen University in 2013. She is currently a Ph.D. candidate at the School of Software, Tsinghua University. Her research interests include computer graphics, 3D indoor scene enhancement and semantic 3D indoor scene modeling.

**Zhizhong Han** received his PhD degree from Northwestern Polytechnical University, China, 2017. He is currently a postdoctoral researcher in Department of Computer Science, University of Maryland, College Park, USA. He is also a research member of the BIM group, Tsinghua University, China. His research interests include machine learning, pattern recognition, feature learning and digital geometry processing.



**Yu-Kun Lai** received his bachelor's and Ph.D. degrees in computer science from Tsinghua University, China, in 2003 and 2008, respectively. He is currently a Reader at the School of Computer Science & Informatics, Cardiff University, UK. His research interests include Computer Graphics, Geometry Processing, Computer Vision and Image Processing. He is on the editorial boards of Computer Graphics Forum and The Visual Computer.

**Matthias Zwicker** is a Professor at the Department of Computer Science, University of Maryland, College Park, where he holds the Reginald Allan Hahne Endowed E-nnovate chair. He obtained his PhD from ETH in Zurich, Switzerland, in 2003. Before joining University of Maryland, he was an Assistant Professor at the University of California, San Diego, and a professor at the University of Bern, Switzerland. His research in computer graphics covers signal processing for high-quality rendering, point-based methods for rendering and modeling, 3D geometry processing, and data-driven modeling and animation.

**Hui Zhang** is an Associate Professor at School of Software, Tsinghua University, China. She received her B.Sc. and Ph.D. in Computer Science from Tsinghua University, in 1997 and 2003, respectively. Her research interests include computer aided design and computer graphics.