Sketch2Scene: Sketch-based Co-retrieval and Co-placement of 3D Models

Kun Xu[1] Kang Chen[1] Hongbo Fu[2] Wei-Lun Sun[1] Shi-Min Hu[1]
[1]Tsinghua University, Beijing [2]City University of Hong Kong

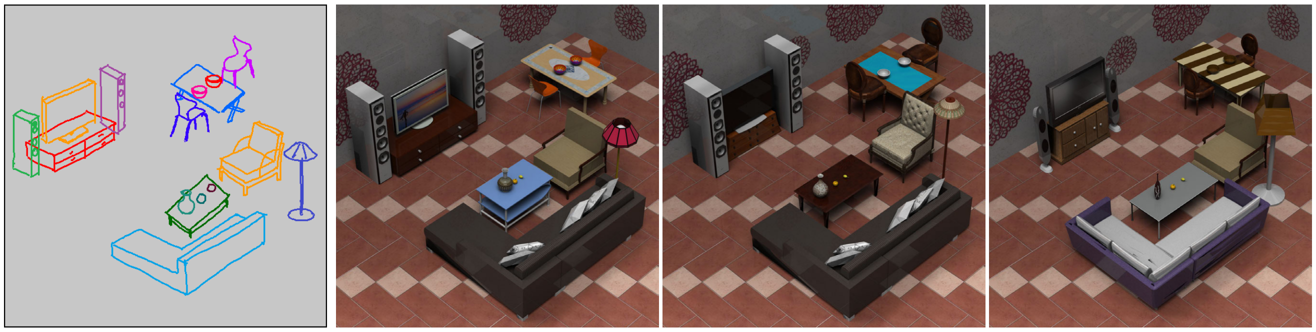


Figure 1: *Without any user intervention, our framework automatically turns a freehand sketch drawing depicting multiple scene objects (left) to semantically valid, well arranged scenes of 3D models (right). (The ground and walls were manually added.)*

Abstract

This work presents *Sketch2Scene*, a framework that automatically turns a freehand sketch drawing inferring multiple scene objects to semantically valid, well arranged scenes of 3D models. Unlike the existing works on sketch-based search and composition of 3D models, which typically process individual sketched objects one by one, our technique performs *co-retrieval* and *co-placement* of 3D relevant models by jointly processing the sketched objects. This is enabled by summarizing functional and spatial relationships among models in a large collection of 3D scenes as *structural groups*. Our technique greatly reduces the amount of user intervention needed for sketch-based modeling of 3D scenes and fits well into the traditional production pipeline involving concept design followed by 3D modeling. A pilot study indicates that the 3D scenes automatically synthesized by our technique in seconds are comparable to those manually created by an artist in hours in terms of visual aesthetics.

Links: DL PDF

1 Introduction

The availability of large collections of 3D models (e.g., Google 3D Warehouse) together with various shape retrieval techniques offers a great opportunity for easy composition of new 3D scenes or models by properly recombining the existing models or their parts. Sketch-based user interface is commonly adopted for this task mainly due to its simplicity, intuitiveness, and ease of use [Eitz et al. 2012]. It has been shown that casually drawn sketches can be used as both shape queries for model retrieval and alignment cues for model placement, greatly simplifying the modeling process.

The existing techniques for sketch-based search and composition of 3D models [Shin and Igarashi 2007; Lee and Funkhouser 2008; Xie et al. 2012] typically repeat the following process for *individual* desired models: first 2D sketch, then retrieval and finally 3D placement. This iterative *2D-3D-2D* process is not compatible with the traditional design workflow (i.e., 2D concept design followed by 3D modeling), which is largely sequential and often demands different professionals specialized for different tasks (e.g., concept artists, 3D modelers) [Chopine 2011]. In addition, their performance is highly sensitive to the quality of individual sketches. User intervention is thus often needed for every step in the process.

In this work, we focus on *joint processing* of a set of sketched objects, sketches for short, corresponding to multiple models in a 3D scene of interest. This can significantly reduce the ambiguity arising from both the steps of retrieval and placement. For example, while a single sketch itself (e.g., the computer mouse in Figure 2) is hard to recognize, the other input sketches might provide strong context cues for retrieving a desired model. Similarly, one sketch might give arrangement cues to another sketch (e.g., a keyboard *on* a desk).

Our ultimate goal is to find optimal scenes which are as locally similar to example scenes in a repository as possible and meanwhile satisfy the constraints derived from the input sketches. We reach this goal by solving two new problems: sketch-based *co-retrieval* and sketch-based *co-placement* of 3D models. We propose the concept of *structural groups*, a compact summarization of reliable relationships among objects in a large database of well-constructed 3D scenes. Our structural groups enable efficient and effective solutions to co-retrieval and co-placement, which are posed as combinatorial optimization problems. As shown in Figure 1, given a segmented sketch drawing as input corresponding to multiple scene objects of interest, our technique automatically turns the input sketches to contextually consistent, well arranged scenes of 3D models, with instant feedback (see the accompanying video).

With our tool, an artist can devote herself to concept sketching and our example-based algorithm serving as a virtual 3D modeler automatically constructs a desired scene in seconds by using the knowledge learnt from the scene repository. Our technique thus greatly reduces the amount of user intervention needed for sketch-based modeling of 3D scenes. A pilot study shows that our method is able to automatically produce 3D scenes that are comparable to those manually created by an artist in terms of visual aesthetics, but uses significantly less time.

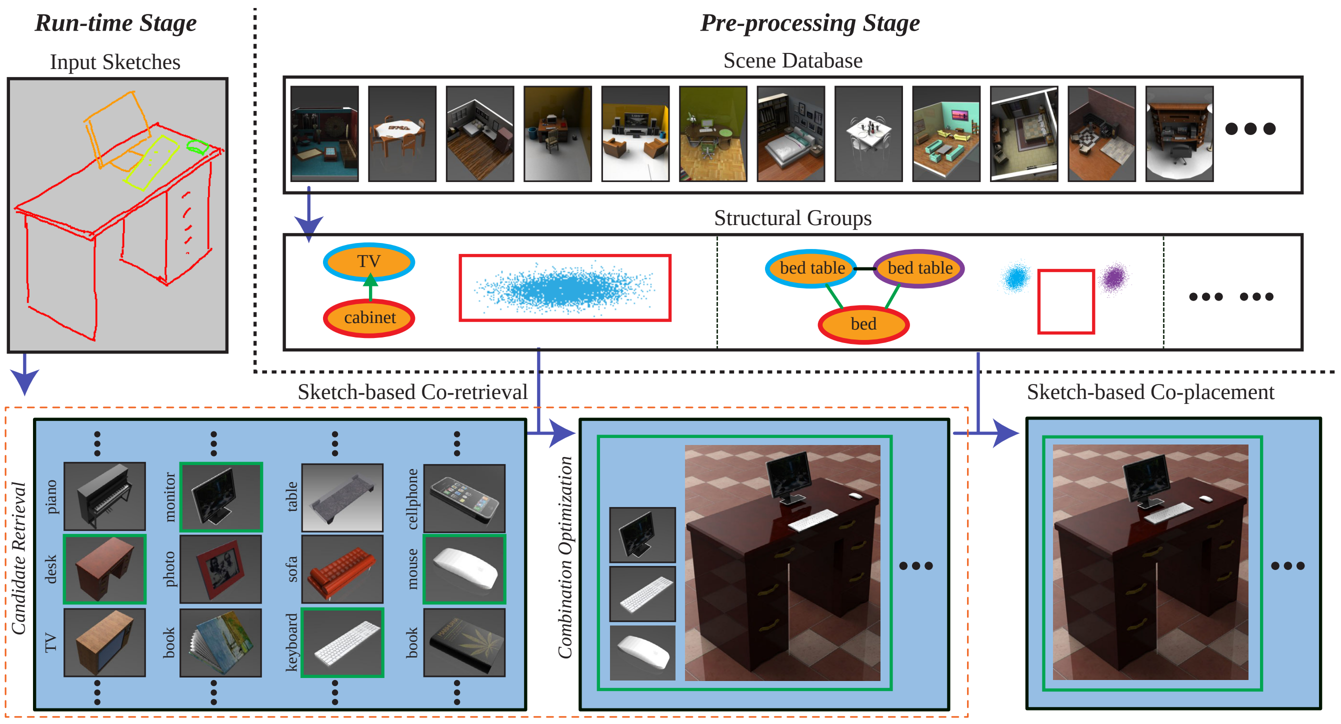


Figure 2: *Algorithm pipeline. Our technique pre-processes a large database of well-constructed 3D scenes to extract a small set of structural groups, which are used at runtime to automatically convert input segmented sketches to a desired scene. Our technique consists of two major components: sketch-based co-retrieval and sketch-based co-placement of 3D models.*

2 Related Work

There exist many techniques for sketch-based shape retrieval (see [Eitz et al. 2012] and the references therein). These techniques focus on improving retrieval performance given *individual* sketches, since they are seldom introduced in the context of scene modeling. However, even with the state-of-the-art techniques like [Eitz et al. 2012], sketch-based retrieval for realistic inputs is still challenging, thus heavily relying on users to resolve the ambiguity. Little effort has been put into combining sketch-based retrieval with sketch-based modeling for scene construction [Olsen et al. 2009; Zhu et al. 2012], which raises another problem of how to determine the position, orientation, and scale of each retrieved model given the same sketches used for retrieval. The existing methods [Shin and Igarashi 2007; Lee and Funkhouser 2008] attempt to automate this step of model placement by using heuristic cues (e.g., no penetration, in contact) but still require users to decide the sketching and modeling orders, which are vital to the use of their heuristic cues.

In recent years, several techniques have been proposed for automatic arrangement of 3D models. Their goal is more open-ended, with a focus on the generation of plausible and aesthetically pleasing arrangements, instead of a specific arrangement inferred by the user-specified sketches in our problem. Although interactive control is possible with the methods of [Merrell et al. 2011; Fisher et al. 2012], user constraints must be specified directly in the three-dimensional domain, e.g., for freezing the location of a 3D model [Merrell et al. 2011] or for encoding in desired arrangements in example scenes [Fisher et al. 2012]. In addition, these methods require either a user-specified set of 3D models to be arranged [Yu et al. 2011; Merrell et al. 2011] or 3D scene examples as input, which are often laborious to create [Fisher et al. 2012].

Our work got inspirations from assembly-based 3D modeling [Funkhouser et al. 2004], which is mainly designed for object synthesis. While recent research concentrates on open-ended 3D modeling [Chaudhuri et al. 2011; Xu et al. 2012; Kalogerakis et al. 2012], Shen et al. [2012] present a constrained part assembly framework. However, their solution cannot be adapted to our problem for the following reasons. First, their technique requires a raw 3D scan as input, though this scan might be noisy and incomplete. Second, object synthesis is very different from scene modeling [Fisher et al. 2012] (e.g., well-defined part placement versus loosely-related scene objects).

There exist numerous context-based object detection or recognition techniques in computer vision (e.g., see [Divvala et al. 2009; Galleguillos and Belongie 2010] and the reference therein). While the importance of context is clear, the term "context" lacks a clear definition, not to mention how to apply context to different problems. It is thus unclear how such existing techniques could be applied directly to our problem. Context-based search has started to show its importance in 3D computer graphics only in recent years, e.g., [Fisher and Hanrahan 2010], which however assumes the availability of explicit context, which is missing in our problem. Inspired by the recent works on data-driven suggestions [Chaudhuri and Koltun 2010; Fisher and Hanrahan 2010; Lee et al. 2011; Fisher et al. 2011], Xie et al. [2012] apply context-based search to interactive sketch-based assembly of 3D parts [Lee and Funkhouser 2008]. However, their technique also requires explicit context.

Our work is related to the recent effort towards semantic image composition [Lalonde et al. 2007; Hays and Efros 2007]. In particular, our problems and solutions bear some resemblance to those introduced by Johnson et al. [2006] and Chen et al. [2009]. For example, Chen et al. present a sketch-based UI for internet image montage. Their solution first searches for candidate images corresponding to individual sketches and then finds an optimal combination of candidate images for composition, simply based on color and texture consistency along blending boundaries. Therefore, how to adapt such solutions to our geometric problems is unclear.

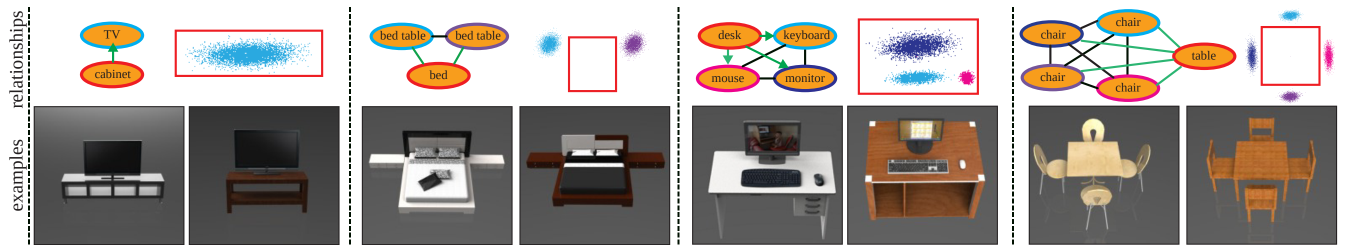


Figure 3: *Examples of extracted structural groups of size from 2 to 5. From left to right: "cabinet and TV", "bed with two bed tables", "desk, keyboard, monitor, and mouse", and "a table with four chairs". The arrows (in green) indicate supporting relationships. The spatial relationships corresponding to the edges in green (including the arrows) are illustrated at the top-right corner for each example.*

Finally, our terms *co-retrieval* and *co-placement* draw direct inspirations from *co-abstraction* [Yumer and Kara 2012] and *co-segmentation* [Huang et al. 2011; Sidi et al. 2011; Wang et al. 2012]. Instead of simply solving the respective problems for individual models, all of these works intend to achieve a certain degree of consistency across the relevant results by a joint shape analysis. Our work brings the power of such co-analysis to a completely different scenario.

3 Overview

Figure 2 gives an overview of our framework. Our problem takes as input a freehand sketch drawing, depicting a desired scene. The drawing has been pre-segmented into a set of sketches (as shown in different colors), each of which corresponds to a single scene item. Such segmentations may come with the drawing itself (e.g., constructed in layers) or can be easily obtained with interactive tools [Noris et al. 2012]. No depth order between segmented sketches is needed. The goal of our paper is to turn such an input drawing to a semantically valid, well arranged scene of 3D models, without any user intervention. We reach this goal by tackling two sub-problems: sketch-based co-retrieval and sketch-based co-placement.

Our technique requires a repository of relevant 3D scenes, rather than a database of individual 3D models as used in the previous works [Shin and Igarashi 2007; Lee and Funkhouser 2008]. Each scene has been pre-segmented into semantic objects with labels (Section 4). Given the limited number of repository scenes, it is very likely that a desired scene depicted by the input drawing would not be globally similar to any scene in the repository. However, we observe that there often exists local similarity between a desired scene and some of the repository scenes in terms of local scene configuration (e.g., object combination, layout, size, orientation). This motivates us to extract reliable building blocks which capture local semantics in the repository scenes. Such building blocks of various sizes, named as *structural groups* (Section 4), will be used in both the steps of co-retrieval (Section 5) and co-placement (Section 6).

We address the problem of sketch-based co-retrieval by first retrieving a small set of candidate objects from the repository independently for each sketch (Section 5.1) and then finding an optimal combination of the retrieved candidates (Section 5.2). The challenge here lies in the latter step, where a straightforward solution is of exponential complexity and thus computationally prohibitive. We will show that our structure groups allow us to quickly explore large parts of the combinatorial solution space corresponding to semantically meaningful scene compositions.

Given a specific set of objects obtained from the step of co-retrieval, our next step is to turn them to a well-arranged scene and meanwhile to respect the spatial constraints prescribed by the sketches (Section 6). The characteristics of good arrangement is already captured by the structural groups. We thus simultaneously optimize the positions and orientations of all the objects to match both the sketches and the structural groups in terms of local spatial configurations. Note that our unique sketch constraints prevent from using the existing object arrangement techniques [Merrell et al. 2011; Yu et al. 2011; Fisher et al. 2012].

4 Structural Groups

In this section we introduce the concept of structural groups, which give a compact and semantically meaningful summarization of exemplar scenes. This component can be potentially replaced with the probabilistic models recently proposed for instance by [Fisher et al. 2012; Kalogerakis et al. 2012]. However, we do not see that any of them is immediately applicable to our scenario, since their inputs and outputs are significantly different from ours.

Fisher et al. [2012] show how to synthesize a diverse set of new scenes by sampling a probabilistic model learnt from a database of 3D scenes including user-provided examples. Although their synthesized scenes are visually plausible, how to guarantee the best matching between the synthesized scenes and our sketch constraints is challenging. It is unclear how to mix the sketch information and scene database information for their learning pipeline. Instead, we take the problem as a *combinatorial optimization* problem, i.e., finding an optimal scene from possible object combinations induced by the input sketches such that the obtained scene is as locally similar to the repository scenes as possible, in terms of both object combination and spatial configuration.

Thus a key sub-problem here is how to efficiently evaluate the local similarity between a synthesized scene and a database of scenes. A similar problem has been recently studied in [Fisher et al. 2011], which introduces a *p-th order rooted walk graph kernel* to compare the local structure of two scene graphs rooted at particular nodes within those graphs, and computes the local similarity between the graphs by summing the kernel over all node pairs across the two graphs. While the performance of this technique is not a big issue for their applications, which typically involve only one query scene against a scene database (i.e., one to many), it would disallow instant feedback in our scenario, requiring the evaluation between every possible scene in the solution space and every scene in the repository (i.e., many to many). In addition, their similarity metric does not consider the influence of spatial relationships between objects, which we found crucial in our problem (Figure 11).

Our solution is based on the following key observation: some relationships (e.g., a lamp on a desk) often occur in multiple repository scenes and thus are more reliable, while other relationships seldom appear in the repository and are less useful for determining the degree of scene semantics. Hence we propose to extract from the scene repository a set of structural groups, which summarize the reliable local structures among the scenes in the repository. By evaluating the quality of a synthesized scene against the set of structural groups instead of individual repository scenes, our technique not only substantially reduces redundant computations for approximately repeated patterns but also avoids unnecessary evaluations for unreliable relationships.

4.1 Definition

As illustrated in Figure 3, a *Structural Group* (SG for short) consists of multiple *object categories*, among which there are reliable relationships or patterns, e.g., a dinning table surrounded by four chairs, or a monitor on a desk together with a keyboard and a mouse. We will use SGs to guide the construction of semantically valid scenes with respect to the input sketches.

We represent an SG as a *complete* graph $G = (V, E)$, with each node $v_i \in V$ $(1 \leq i \leq p)$ representing an object category, and each edge $e_{ij} \in E$ describing pairwise relationships between object categories v_i and v_j. Here $p = |V|$ is the size of this SG, i.e., the number of object categories, which might involve duplicate ones, e.g., $p = 5$ for a table with four chairs (Figure 3 (right)). p is in the range from 2 to 5 in our implementation. Each edge e_{ij} is associated with *at least one* type of pairwise relationship, listed below:

◇ **Supporting relationships** can be either *supporting* or *supported*. For example, a monitor is supported by a desk.
◇ **Spatial relationships** model the distribution of position and orientation of objects of category v_i with respect to objects of category v_j. Such pairwise relationships are defined over all the edges in an SG, i.e., between every pair of nodes.
◇ **Coplanar relationships** capture whether certain sides of the bounding boxes of objects of category v_i and category v_j are often coplanar. For example, the back of a bed table is very likely to be coplanar to one side of a bed.
◇ **Symmetric relationships** can be either rotational symmetry or axial symmetry. Such relationships are defined between v_i and v_j of the same object category with respect to another object category in the same structural group. Thus they are available only for SGs of size greater than 2 (e.g., between chairs in the SG shown in Figure 3 (right)).

4.2 Learning Structural Groups

In this subsection, we show how to learn structural groups from a large database of scenes. Note that this is a preprocessing step and needs done only once for a given set of scenes.

Dataset. We focus on indoor room scenes for their rich structural patterns and the public availability of such scene corpus in Google 3D Warehouse, which has been explored in a series of works by Fisher et al. In total, our scene database contains 743 relevant scenes, taken from 7,000 scenes downloaded from Google 3D Warehouse. Our database involves 11,000 individual objects, including different instances of the same geometry. We manually define a consistent tag (i.e., object category), reference frame, and size (with respect to the size of the object in the real world) for each object. There are totally 70 object categories.

Extraction. To capture local structures, we consider only a small group of objects which are nearby in a scene (i.e., any group of objects within a proximity threshold of one another, typically $0.5m$ in our experiments) towards SG extraction. Such an object group corresponds to a *potential* SG, denoted as a graph $G = (\{v_i\}, \{e_{ij}\})$. Object co-occurrence frequency is then adopted to capture the reliability or saliency of a structural group. Specifically, we define the reliability of a potential SG G as follows:

$$f(G) = c(G) / (\prod_{1 \leq i \leq p} c(v_i))^{1/p}, \qquad (1)$$

where $c(G)$ denotes the total counts that objects of categories $\{v_i\}$

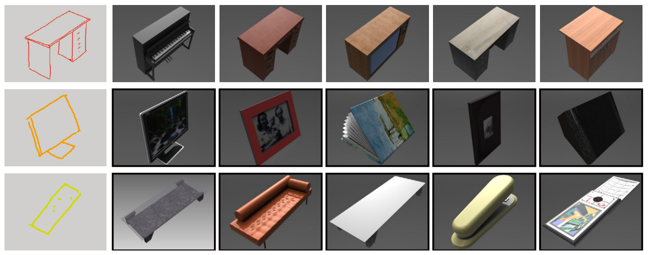


Figure 4: *Top-5 candidates retrieved independently for a subset of sketches in the* 5*-th example in Figure 14.*

co-occur in the scene database, and $c(v_i)$ denotes the total counts that objects of individual category v_i occur in the scene database. The denominator is introduced to get a normalized reliability measure. We identify the graph G as an SG only if both $f(G)$ and $c(G)$ are above certain thresholds. In total, we extract 161 SGs, including 87 size-2 SGs, 37 size-3 SGs, 19 size-4 SGs, and 18 size-5 SGs. Figure 3 shows some of the extracted SGs. Note that object co-occurrence as relationship saliency has been experimented in previous works [Fisher and Hanrahan 2010; Fisher et al. 2012], which, however, focus on pairwise relationships only (see more discussions in Section 7).

Supporting, coplanar and symmetric relationships are straightforward to establish by examining how often such relationships occur in the repository scenes. Below we describe how to extract the spatial relationships. In our implementation we consider 2D spatial relationships only, by projecting groups of objects onto a floor plane. A Gaussian mixture model [Fisher et al. 2012], denoted as $g_{i,j}$, is then learnt from the distribution of 2D center and orientation of object category v_i with respect to object category v_j (Figure 3 (top)), i.e., aligned using the reference frames associated with objects of category v_j. It is important to consider multiple same categories in an SG as individuals and separately model each of them with respect to another category in the SG. Otherwise, it may fail to distinguish multi-mode distributions for a single object (e.g., a left- or right-handed computer mouse) from co-occurring instances of the same type of object (e.g., four chairs surrounding a table). To this end, for two object groups A and B captured by an SG involving repeated categories, we establish one-to-one correspondence for objects of the same categories between A and B. In our implementation, this is achieved by minimizing the sum of angular differences: $\text{argmin}_{h(i)} \sum_{i,j} (\alpha(A, i, j) - \alpha(B, h(i), h(j)))^2$, where $\alpha(A, i, j)$ denotes the angle between the vector from the i-th object to the j-th object and the orientation of the j-th object, and $h(i)$ denotes the index of the object in B corresponding to the i-th object in A.

Manual Adjustments. While the above extraction method is able to automatically extract most of the desired SGs, it might still fail to form certain salient relationships, e.g., a TV facing towards a sofa often together with a tea table. This is because a TV is usually relatively far away from a sofa, and thus such long-range relationships cannot be automatically identified by our extraction algorithm (considering only local neighborhoods). In total we manually added 10 SGs, which could improve the final results in certain cases and were identified when we played with our preliminary sketch examples (not those shown in the paper).

5 Sketch-based Co-retrieval

Taking a set of segmented sketches as input, our co-retrieval algorithm returns a set of roughly placed 3D objects, each of which closely matches the corresponding input sketches and whose combination is semantically meaningful. Their positions and orientations will be refined in the subsequent step of co-placement (Section 6). As previously mentioned, we regard co-retrieval as a combinatorial optimization problem and solve it using a two-step approach: first finding top-N candidate objects independently for each sketch (Section 5.1) and then optimizing the combinations of the candidate objects (Section 5.2). Let M be the number of input sketches. A naïve solution would be to check semantic plausibility of all possible $O(N^M)$ combinations against the scenes in the database, and thus would be obviously computationally infeasible. We will show that our SGs constructed in the preprocessing stage (Section 4.2) together with a beam search algorithm dramatically speed up the process, making our technique support instant feedback, which is crucial for interactive modeling applications.

5.1 Candidate Retrieval

Dataset. As individual models in the collected scenes for learning SGs (Section 4.2) are often of low quality, for the application of scene composition we prepare a separate database of more detailed textured models, whose category labels and sizes are consistent with those in the scene database. In total, there are about 1,000 individual models in this new database. This also shows the potential for the application of our extracted SGs to existing shape repositories.

Retrieval. We retrieve the top-N matched objects from this model database *independently for every input sketch* using the state-of-the-art sketch-based retrieval approach [Eitz et al. 2012], but with two implementation-level changes as follows. First, for view sampling we choose camera positions with polar angles 0, 15, 30, 45, 60 degrees and with azimuthal angles of a 15-degree interval (totally 24 different azimuthal angles), and adopt an orthographic projection for simplicity. Second, we combine silhouettes, Canny lines from depth images, and Canny lines from normal images to produce the line drawing contours. Suggestive contours are not employed since it seems that the suggestive contour algorithm [DeCarlo et al. 2003] does not perform well for our models with poor mesh connectivity.

Next we reorder the top-N matched objects by comparing their matched contour images with the query sketch using *Shape Context* [Belongie et al. 2002]. Although the shape context approach is relatively slower than [Eitz et al. 2012], it is more accurate and thus gives us a better order of the matched contours as well as a more accurate matching score for each object. The sketch matching score is defined as:

$$m_S(u) = \exp(-\lambda_S \cdot x(S, u)), \tag{2}$$

where S is a query sketch, u is one of the top-N matched objects given S, $x(S, u) \in [0, 1]$ gives the shape context distance between S and the contour of u under the matched view, and λ_S ($\in [5, 15]$ in our experiments) is a user-specified parameter (Section 7). The higher value of $m_S(u)$, the closer matching. We normalize $m_S(u)$ such that the highest score among the top-N objects is scaled to one. As we will show later, shape context is also helpful for correspondence establishment between the sketch and the contours of the retrieved objects. Unsurprisingly, not all user-desired objects are ranked first, as shown in Figure 4.

5.2 Combination Optimization

In this subsection, we first introduce our combination score formulation for evaluating the quality of a specific combination of candidate objects, and then present a beam search approach to efficiently find an approximately optimal combination.

Combination Score. Given a combination C, which contains for each sketch one of top-N candidate objects, its score is given based on how likely this combination can be modeled by our extracted SGs as building blocks (Section 4). We first pretend C to be a well-constructed scene with the known supporting relationships as well as the known positions/orientations of individual objects, which, however, will be determined later. Thus our formulation below is directly applicable to the evaluation of existing 3D scenes.

Scale Consistency. We assume that our input sketches are not seriously influenced by perspective shortening. Therefore, the relative screen-space size of objects inferred by the sketches should be largely consistent with the relative size of those in the scene database. In other words, we prefer a combination of objects who undergo the same isotropic scaling when each object is uniformly scaled to match the corresponding sketch in the screen space. Let $d(u)$ be the uniform scale factor needed for object u to best match the size of the corresponding sketch. We then define the scale consistency term of a combination as follows:

$$P(C) = \exp(-\lambda_p \cdot \max_{1\le i\le M} |\log d(o_i) - \frac{1}{M}\sum_{1\le i\le M} \log d(o_i)|), \tag{3}$$

where $\{o_i\}$ denotes the set of objects in C, M is the number of sketches, and we set $\lambda_p = 3$ in our experiments. Higher values of $P(C)$ imply greater degrees of scale consistency.

Sub-combination Score. Let $U = \{u_1, u_2, \ldots, u_p\}$ $(2 \le p \le 5)$ denote a sub-combination (i.e., subset) of C. A sub-combination U is called to be *effective* if there exists one SG among the set of all the extracted SGs such that U and this SG have the same size, category labels and supporting hierarchies. The score of an effective sub-combination U is defined to measure how well it matches the corresponding SG. Specifically, it is defined as:

$$s_{\text{sub}}(U) = \prod_{1\le i\le p} m_S(u_i) \cdot \prod_{1\le i,j\le p} (g_{i,j}(u_i, u_j))^{\lambda_c}, \tag{4}$$

where $m_S(u_i)$ is the sketch matching score corresponding to u_i (Equation 2), $g_{i,j}$ is the spatial distribution modeled by a Gaussian mixture in the corresponding SG (Section 4.2). The score is the product of two terms, whose relative contributions are controlled by λ_c (Section 7). The first term constrains each object u_i to be well matched with the corresponding sketch, while the second term ensures that the spatial relationship between u_i and u_j matches well that between the corresponding nodes (with the same category labels as u_i and u_j) in the SG. For any non-effective sub-combination U, we simply set $s_{\text{sub}}(U) = 0$.

Overall Score. We finally define the combination score $s(C)$ of a combination C as follows:

$$s(C) = P(C) \cdot \sum_{\forall U\subset C, 2\le |U|\le 5} a(|U|) f(G(U)) \cdot s_{\text{sub}}(U), \tag{5}$$

where $|U|$ is the size of an effective sub-combination U, $a(|U|)$ is a monotonically increasing function of size $|U|$ to encourage SGs of larger size (Section 7), and $G(U)$ denotes the corresponding SG of U. The combination score is the product of two terms. The first term $P(C)$ is the scale consistency term (Equation 3); the second term is the sum of weighted sub-combination scores over all effective sub-combinations of C, where each sub-combination score is weighted by the reliability of the corresponding SG $f(G(U))$ (Equation 1) and a function of SG size $a(|U|)$. We prefer combinations that lead to high combination scores.

Supporting Relationships. To compute a combination score for C, we need to recover its supporting hierarchy, and the positions and orientations of individual objects. We would like to assign supporting relationships to pairs of objects in C such that $s(C)$ is maximized. By definition (Equation 5), only effective sub-combinations (i.e., who share the same category labels and supporting hierarchies as those of the extracted SGs) will contribute to the combination score. Therefore, if a supporting relationship between a pair of object categories, e.g., (v_i, v_j), is observed in the SGs, we assign the same relationship to pairs of objects of categories v_i and v_j in C.

However, we also need to respect the constraints induced by the sketches. Our current constraints are based on the following heuristics. First, if the bounding boxes of two sketches, denoted as S_1 and S_2, do not intersect, there will be no supporting relationship between them. Second, if they do intersect but the bottom edge of S_1's bounding box is below that of S_2's, the object inferred by S_2 is disallowed to support the object inferred by S_1.

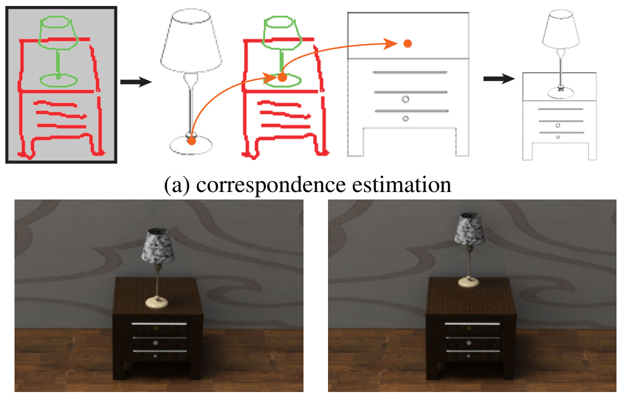
Rough Placement. Given a combination C with the derived supporting hierarchy, it is ready to estimate an initial orientation and position for each object. Recall that at the candidate retrieval step (Section 5.1), we already have each object's XY-orientation and view angle, derived from the viewpoint information for the best-matched contour image. Hence, we set each object's initial orientation as its orientation leading to the best match with the sketch [Lee and Funkhouser 2008], and set the "global view angle" as the average view angle of all objects in the combination. All the objects are uniformly scaled by the same scale factor to match the average size of the sketches and then each of them is translated to best align its corresponding sketch in the screen space.

Following the supporting hierarchy (level by level), we estimate a rough 3D position for every object in the combination. Let's start with objects without being supported by any other object. We simply assume that such objects are on a 3D floor plane, which can be assigned using the bottom face of the bounding box of any of these objects. Similar to [Shin and Igarashi 2007; Lee and Funkhouser 2008], we then determine their positions on the floor by back-projecting the screen-space bottom centers of these objects' bounding boxes to the floor plane. Possibly due to the approximation errors of the camera model and/or the rough sketches, such a simple back-projection model often fails to preserve the relative object positions inferred by the sketches, especially for objects supported by others (e.g., a lamp on a bed table as shown in Figure 5). To address the problem, we first establish sparse correspondences between the sketches and the contour images for supporting objects (e.g., the bed table in Figure 5 (a)) using shape context, and then dense correspondences using interpolation methods like thin-plate spline interpolation [Cheng et al. 2010]. Given such dense correspondences, we can easily turn a screen-space position on the supporting surface to a 3D position, where a supported object is placed. Figure 5 (b) and (c) show the difference between this method and a simple back-projection method when placing supported objects.

Since we have the position and orientation estimated for every object, it is time to evaluate a combination score for this combination. As the interactions between objects are largely ignored, the initialized orientations and positions are usually not very accurate. Therefore, we use a relatively small value of λ_c in Equation 4 (typically $\lambda_c = 0.2$). However, we will show in Section 7 that the spatial term is important to derive semantically more meaningful combinations.

Beam Search. To obtain the optimal combination (i.e., with the highest combination score defined in Equation 5), a naïve solution that enumerates all possible combinations is prohibitively expensive. We develop a beam search approach based on the following heuristic: if a combination is optimal, it is very likely that at least one or several subsets of this combination are also optimal.

The key idea is to first produce some optimal sub-combinations, and then use them to produce larger-size sub-combinations until the final full-size combination is obtained. We start with the optimal sub-combinations of size 2 (i.e., containing only 2 objects). This is done by enumerating all sketch pairs and computing the combination scores (Equation 5) of all possible object pairs for each sketch pair. Each object pair is called as a size-2 object combination. The time complexity to evaluate all size-2 object combinations is $O(N^2M^2)$. We only leave R (the choice of R will be discussed in Section 7) object pairs with the highest scores for further processing. Next, at each iteration, we produce optimal sub-combinations with size p by growing from sub-combinations with size $(p-1)$ (i.e., by adding one new object (sketch) to each size-$(p-1)$ sub-combination), or by growing from sub-combinations with size $(p-2)$ (i.e., by adding two new objects (sketches) to each size-$(p-2)$ sub-combination). Specifically, for each size-$(p-1)$ sub-combination, we enumerate over all possible new sketches (objects) to form a new size-p sub-combination. Hence it will produce at most $O(MNR)$ size-p sub-combinations. For each size-$(p-2)$ sub-combination, we enumerate over all possible size-2 sub-combinations and in practice we test against only the previously obtained R optimal size-2 sub-combinations, producing at most $O(R^2)$ size-p sub-combinations. Again we compute the combination score for each size-p sub-combination, and only leave R sub-combinations with the highest scores. The above steps are repeated until the size of sub-combination reaches M. Finally we leave the full-size object combination with the highest score to the step of co-placement.


(a) correspondence estimation

(b) by correspondence estimation (c) by simple back-projection

Figure 5: *Roughly placing a supported object (the lamp here).*

6 Sketch-based Co-placement

In this section, we present an approach to simultaneously fine-tune the positions and orientations of all objects in the optimal full-size object combination obtained from the step of co-retrieval. Let C denote that object combination, i.e., a roughly constructed scene.

Our solution is motivated by the fact that it is rather difficult to specify accurate object position and orientation using the sketches, which though should be respected; on the other hand, the extracted SGs already capture the likelihood of good arrangements. Therefore, our goal here is to refine the arrangement of the roughly constructed scene C towards the SGs while respecting the constraints from the input sketches.

We will reuse the formulation of combination score $s(C)$ (Equation 5) to optimize the positions and orientations of all objects, since it already captures the consistency of the constructed scene and the SGs in terms of arrangement. However, simply using this equation to measure the arrangement quality is insufficient, as it ignores the spatial constraints of the sketches. Note that m_S, the only term in Equation 4 directly related to the sketches, keeps a fixed sketch matching score for each object, and thus cannot be used to constrain the position and orientation of each object with respect to its corresponding sketch during optimization. Therefore, directly maximizing $s(C)$ will lead to a spatial configuration that does not respect the input sketches (see the bottom-right chair in Figure 6

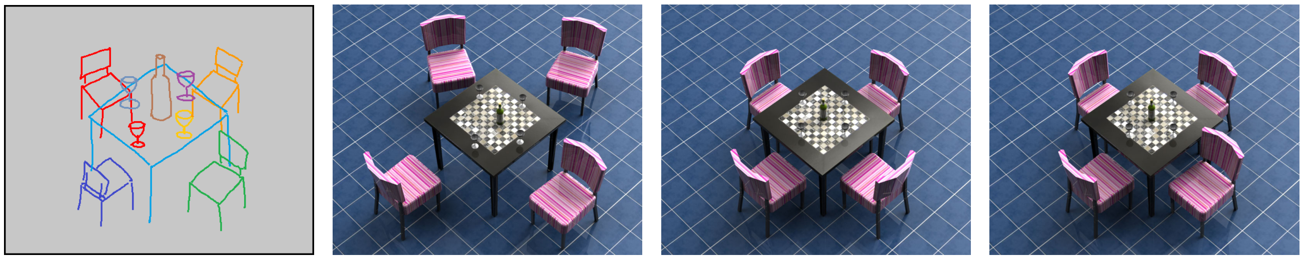

(a) input sketches (b) $w_a = 1$ (c) $w_a = 0$ (d) $w_a = 0.1$

Figure 6: *Example of sketch-based shape co-placement. Note that setting w_a to an appropriate value ($w_a = 0.1$) can lead to well-arranged scenes while still respecting the input sketches, striking a good balance.*

(c)).

Note that each object in C and its corresponding sketch are already roughly aligned in the screen space after the step of co-retrieval. To address the above issue, during the maximization of $s(C)$, we thus simply avoid large deviation from the initial object arrangement after co-retrieval. To achieve this, we change the definition of the sub-combination score $s_{\text{sub}}(U)$ defined in Equation 4 to:

$$s_{\text{sub}}(U) = \prod_{1\le i\le p} m_S(u_i)\cdot\prod_{1\le i,j\le p}((1-w_a)g_{i,j}(u_i,u_j)+w_a t(u_i,u_j))^{\lambda_c}$$

where $t(u_i, u_j)$ is newly introduced to measure how well the current arrangement matches the initial arrangement, and its formulation will be discussed shortly. w_a is a balancing weight, which we typically set as $w_a = 0.1$. The controlling weight is increased to $\lambda_c = 1$ in order to make the spatial relationships learnt from the scene database play a more important role. The term $t(u_i, u_j)$ is specifically defined as follows:

$$\begin{aligned} t(u_i,u_j) =& g(\theta(u_i,u_j);\hat{\theta}(u_i,u_j),\pi/2)\cdot \\ & g(p(u_i,u_j);\hat{p}(u_i,u_j),\max(bb(u_i),bb(u_j))), \end{aligned}$$

where $bb(u_i)$ is the diagonal length of the bounding box of u_i, $g(x;c,\sigma) = \exp(-(x-c)^2/\sigma^2)$, and $p(u_i, u_j)$ and $\theta(u_i, u_j)$ denote the *current* relative position and angle, respectively, which are obtained by projecting u_i's position and orientation to u_j's local frame in the plane parallel to the ground. Correspondingly, $\hat{p}(u_i, u_j)$ and $\hat{\theta}(u_i, u_j)$ are the *initial* relative position and angle, respectively. This term penalizes large changes in relative position and relative orientation. Our final arrangement score $s_{\text{arrangement}}(C)$ is still defined as the combination score ($s_{\text{arrangement}}(C) = s(C)$) after incorporating the above changes to s_{sub}.

Next we will maximize $s_{\text{arrangement}}(C)$ to simultaneously find the optimal positions and orientations of objects in C. The optimization is largely solved using a gradient-descent approach, for which we set the initial positions and orientations from the co-retrieval step as the initial values. It often converges quickly with around one hundred steps (see the intermediate results in the accompanying video). During the optimization, we place additional constraints. First, we impose the symmetric and coplanar relationships (see Section 4) as hard constraints. During the process of iterative optimization, if an SG with symmetry is matched, we require the involved symmetry objects to have the same geometry (e.g., the four chairs in Figure 9 (f)) and enforce the symmetry relationships in a sense of high-level shape editing [Zheng et al. 2011]. Second, we keep objects on supporting surfaces always in touch with the original supporting surfaces. Lastly, we avoid intersecting objects with each other. Figure 6 shows the effectiveness of our co-placement step as well as the impact of w_a on object arrangement.

7 Results and Discussion

We have tested our technique on dozens of input sketch drawings depicting various indoor room scene configurations. The presented sketches were created by three different users including an art student with extensive drawing experience. Before sketching, the test users were provided with the set of available tags (object categories) and for each tag several representative models, which however were not shown during sketching. All the sketches created by them are included in the paper and supplemental materials.

As shown in Figure 7, the same sketch can infer different objects in different scene contexts. In the top-row scene, the sketch in orange depicts a tea table, while in the bottom-row scene, the same sketch depicts a printer. Apparently this kind of ambiguity cannot be resolved by traditional sketch-based retrieval techniques. In contrast, our co-retrieval algorithm successfully derives the desired objects using the other sketches in the same scenes. Figure 8 shows an example with multi-layer supporting surface, which is enabled by our placement method based on correspondence estimation (Section 5.2).

In Figure 9, we illustrate the effect of adding more sketches. It is shown that additional sketches help resolve the ambiguity, noticeably improving the quality of the results. Note that the four chairs finally become the same chair (Figure 9 (f)), since they are captured by an SG with symmetric relationships (i.e., "a table and four chairs"). In contrast, the two chairs in (Figure 9 (e)) are different, mainly because they are independently interpreted by two size-2

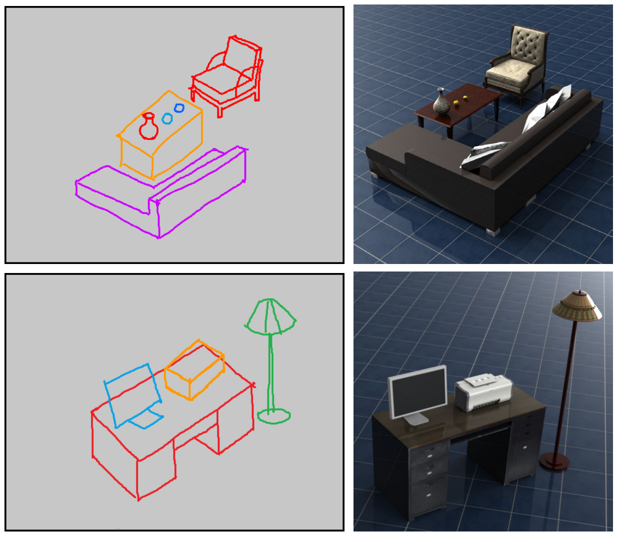


Figure 7: *For the same sketch in orange, our co-retrieval algorithm returns desired objects (tea table (top) and printer (bottom)) that match different contexts indicated by the other sketches.*

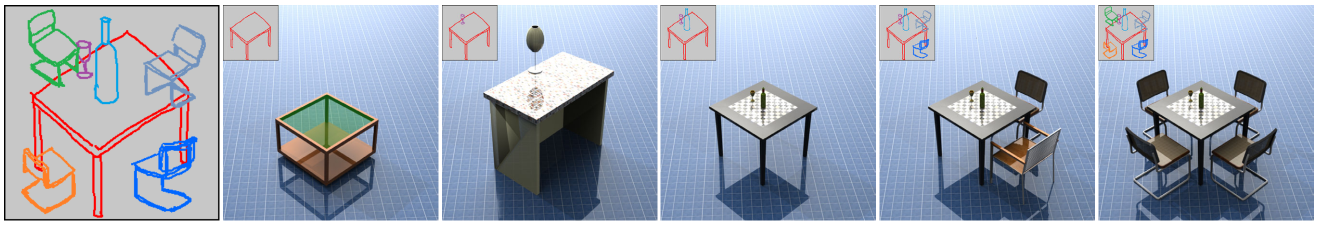


Figure 9: *The effect of adding more sketches. Note the progressively improved results when additional sketches are used.*

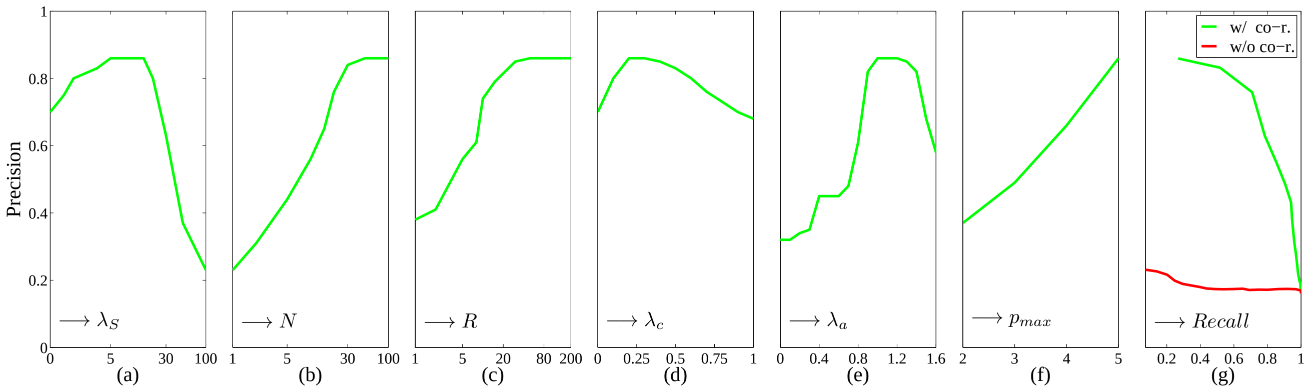


Figure 10: *Co-retrieval precision against different parameter settings.*

SGs (i.e., "a table and a chair").

Figures 1 and 14 give more results. For the examples where multiple results exist for a single set of input sketches, the first result (e.g., in the second column of Figure 1 or the third column of Figure 14) is the optimal combination returned by the step of co-retrieval. To maintain the optimal combination of object categories but increase the diversity, we replace individual objects in the first result with the other top-N retrieved candidates (Section 5.1) that have a low sketch matching score (Equation 2) and the same categories as the corresponding objects in the first result.

We experimented our technique on a PC with Intel Xeon 2.4G CPU and 8G RAM. Our technique supports instant feedback, as shown in the accompanying video. For a typical sketched scene containing around 10 sketches, it overall takes about 2 seconds, including 0.1 second for independently retrieving top-N candidates for each sketch (Section 5.1), less than 1 second for combination optimization (Section 5.2) and shape co-placement (Section 6) together.

Parameters. Our technique involves a few parameters. We use a quantitative analysis to identify the best value or range for each of the important parameters. To this end, we collect 24 sketched scenes (see the supplemental materials for all the collected sketched scenes), each of which contains from 2 to 20 sketches with ground-truth tags (category labels). We then evaluate the precision of our co-retrieval algorithm against different parameter settings, as plotted in Figure 10. The precision is the fraction of objects (in top-1 combinations for all sketched scenes) whose tags are the same as the ground-truth tags associated with the corresponding sketches. Note that each parameter is studied when the other parameters are set as default values.

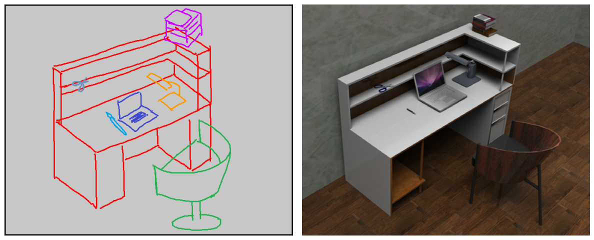
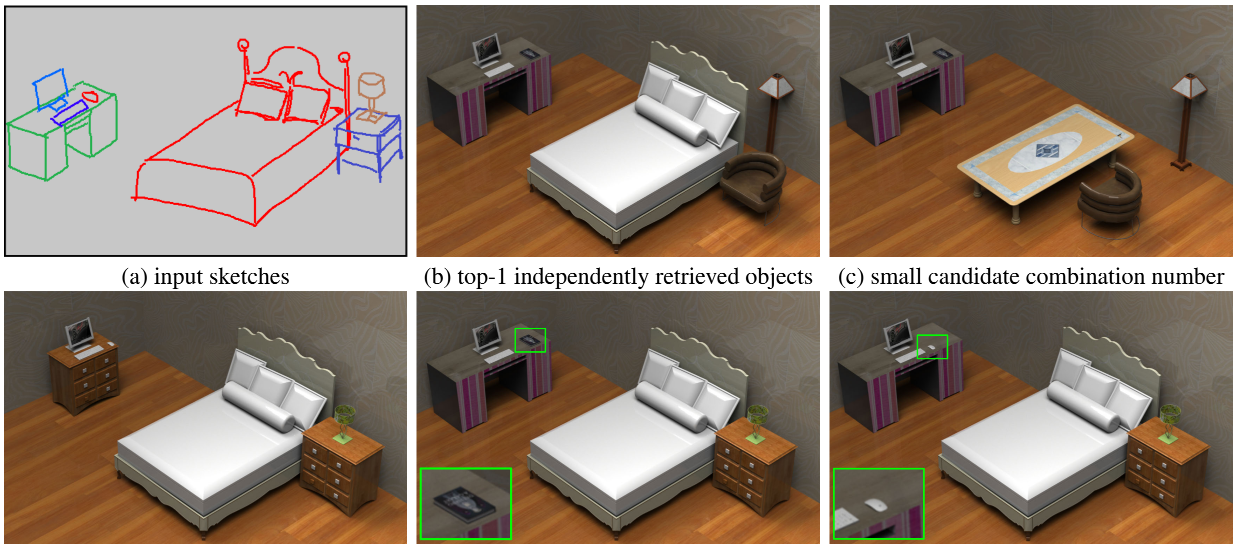


Figure 8: *A multi-layer supporting surface example.*

In Figure 10 (a), we show the influence of *sketch matching coefficient* λ_S (in Equation 2) on the precision. Generally, larger λ_S makes the "sketch matching score" play a more important role in evaluating the combination score (Equation 5). For example, $\lambda_S = 0$ means equally treating the top-N retrieved candidates for each sketch, while a very large value for λ_S (e.g., 100 or $+\infty$) would prefer to use only the top-1 retrieved candidate for each sketch. In the latter case, co-retrieval has no effects. Not surprisingly, using either a very small or very large value for λ_S is not optimal. In our experiments, we find that setting λ_S in the range $[5, 15]$ gives the best performance ($\lambda_S = 10$ by default). Figure 11 (b) gives a result composed of top-1 candidates independently retrieved for each sketch ($\lambda_S = +\infty$). Note that the bed table, desk lamp, and computer mouse are wrongly recognized as the sofa, floor lamp, and book, respectively. This result may also be obtained by simply letting $N = 1$, since N is the number of retrieved objects for each sketch. As shown in Figure 10 (b), the precision increases with N, reiterating the importance of co-retrieval. In our experiments, we set $N = 50$.

Figure 10 (c) shows the effect of *candidate combination number* R used in the beam search (in Section 5.2). A larger value of R leads to a higher precision but at the cost of time and memory consumption, both of which grow linearly with R. Note that $R = 1$ reduces our approach to a pure greedy algorithm, while $R = +\infty$ corresponds to a pure dynamic programming algorithm. From the figure, we find that the precision reaches its maximum when R is around 50. We thus set $R = 50$ in our experiments. Figure 11 (c) gives a suboptimal result with $R = 3$.

In Figure 10 (d), we plot the precision against different values of *spatial relationship coefficient* λ_c (Equation 4). This parameter controls the influence of spatial relationships on the evaluation of


(a) input sketches (b) top-1 independently retrieved objects (c) small candidate combination number

(d) no spatial relationships (e) only size-2 SGs (f) our result

Figure 11: *Examples of results under different parameter settings.*

local structure similarity. Setting $\lambda_c = 0$ would completely ignore the spatial relationships and use only the category information of each object, as similarly done in the graph-kernel algorithm [Fisher et al. 2011]. However, as shown in Figure 11 (d), without such spatial relationships, a desk might be wrongly recognized as a bed table (the left one) due to the existence of a bed in the scene, though their spatial relationship does not match the learnt spatial distribution. On the other hand, setting λ_c as a large value (e.g., 1) also decreases the precision, because the initial positions and orientations of objects are only roughly estimated. We find that setting $\lambda_c \in [0.2, 0.3]$ strikes a good balance ($\lambda_c = 0.2$ by default).

In Figure 10 (e), we show the influence of *SG size weight* $a(p)$ in Equation 5. We set $a(2) = 1^{\lambda_a}$, $a(3) = 3^{\lambda_a}$, $a(4) = 12^{\lambda_a}$, and $a(5) = 60^{\lambda_a}$, where λ_a is a coefficient varying from 0 to 1.6. We set the base weights to $(1, 3, 12, 60)$, since size-5 SG equals the sum of 5 size-4 SG (size-4 SG equals the sum of 4 size-3 SGs, etc.). Setting $\lambda_a = 1$ gives us the best precision. We also evaluate the performance when only the SGs of size $\leq p_{max}$ are used, i.e., setting $a(p) = 0$ for $p > p_{max}$. From Figure 10 (f), it can be seen that the precision increases with p_{max}. This is because SGs of small size alone cannot fully capture complex structures involving more objects. Figure 11 (e) gives an example which is generated using size-2 SGs (i.e., $p_{max} = 2$), simulating the results using only pairwise relationships [Fisher and Hanrahan 2010; Fisher et al. 2012]. Here a desired computer mouse is misinterpreted as a book, since the corresponding sketch looks similar to either a book or a mouse, and there exists a reliable SG *(book, desk)*. However, when the SGs of larger size are included (e.g., *(desk, monitor, keyboard, mouse)*), a computer mouse is correctly retrieved (Figure 11 (f)).

In Figure 10 (g), we compare the precision-recall curve of our co-retrieval results (the green curve) to that of independently retrieved results without co-retrieval (the red curve). Note that under the same recall rate the precision improves a lot by our co-retrieval optimization.

Limitations. First, our technique is less successful for generating scenes that cannot be fully modeled by our SGs. For example, the sketches in Figure 12 (top) infer five chairs symmetrically arranged around a table, which, however, is not characterized by our current set of SGs. Our implementation then leads to one chair and a symmetry group of four chairs. Second, when the input sketches deviate significantly from desired shape or size, our technique might get confused (e.g., the sketch in Figure 12 (bottom) depicting a bottle but wrongly recognized as a chair due to the unexpected size). Lastly, it is possible that desired objects are not within top-N candidates for individual sketches, making our technique fail to derive desired combinations.

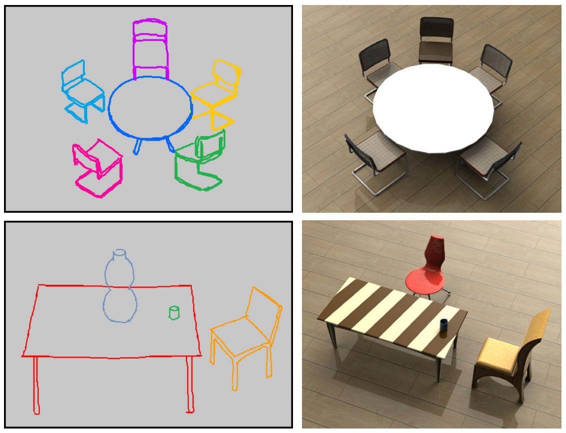


Figure 12: *Less successful examples.*

7.1 User Experience

We conducted a pilot study to evaluate whether the results automatically synthesized by our algorithm are visually comparable to those manually created by artists. Our hypothesis is that human viewers would rate our results and the artist-created results equally good.

We used 7 sketched scenes (see Figures 1 and 14), drawn by the art student, who created some other sketches in the paper and supplemental materials. These 7 sketches, out of all the sketches created by the art student, were chosen mainly for their great complexity and variety. For each sketched scene our technique used the default parameter values to automatically generate three 3D scenes. The same art student was asked to manually model a 3D scene with respect to each sketched scene, using our model database for fair comparison. On average it took the artist 30 minutes to complete a single 3D scene with a standard 3D modeling tool (Autodesk 3ds Max). We rendered one image for each constructed 3D scene, under the same camera and lighting conditions.

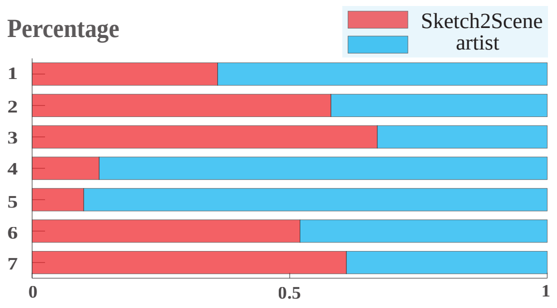


Figure 13: *Percentage in favor of our technique in the pilot study.*

We recruited 32 participants to help with the following *pairwise comparison*. For each sketched scene, each participant was given three images: the corresponding sketch drawing, the rendered image of the artist-created 3D scene, and a randomly-chosen rendered image of the three 3D scenes by our technique. The display order of the latter two images was counter-balanced across participants. The participants were asked to judge which rendered image respects the sketched scene more and is visually more pleasing.

Figure 13 plots the percentage in favor of our technique. A paired t-test finds no statistically significant difference between our technique and the completely manual method (two-tail $p = 0.023$). In other words, our results are comparable to those created by the artist, confirming our hypothesis. Overall the participants were in favor of our technique for 42.4% of all pairwise comparisons. For more than half of the sketched scenes (i.e., the 2nd, 3rd, 6th and 7th examples), the percentages in favor of our technique are above 50% on average.

The participants expressed the least preference for our results of the 4th and 5th examples. One of the reported reasons for the 4th example is that the distance between the TV and sofa is more comfortable in the artist-designed scene (see the second column in Figure 14). This might be solved by incorporating domain-specific guidelines [Yu et al. 2011; Merrell et al. 2011] into our framework. It is also interesting to note that unlike our results, where scene objects are often perfectly aligned and arranged (due to such examples in the scene database), artist-created scenes often exhibit certain degree of casualness, possibly better expressing a sense of living environment. We thus might leave the designer to enable/disable the co-placement feature, depending if he or she desires well-arranged scenes or not. It might also be interesting to offer the designer a tool for interpolating arrangement results with and without co-placement.

8 Conclusion and Future Work

This work has introduced a co-analysis approach to the task of sketch-based 3D scene modeling. We presented two new problems: sketch-based co-retrieval and co-placement of 3D models. We proposed the concept of structural groups, a compact summarization of commonly occurred relationships of objects in a large database of scenes, enabling highly efficient solutions to our problems. Without any user intervention, our technique automatically turns input rough 2D sketches to semantically valid, well arranged 3D scenes, which are visually comparable to those manually created by artists.

Our technique still has a lot of room for improvement, for instance, adding more types of supporting relationships like vertical support (e.g., for mounting a clock on a wall), allowing multiple supporting relationships between a single pair of object categories, handling input sketch drawings without pre-segmentation, using more sophisticated methods [Dutagaci et al. 2010; Secord et al. 2011; Liu et al. 2012] for viewpoint sampling, integrating with floor plan design, promoting style or color harmony etc. Applying our technique to other kinds of scenes probably would help us better understand the strength and weakness of the technique. It would also be interesting to experiment the idea of context-based search, which requires the ordering of input sketches for candidate retrieval and is potentially useful for improving co-retrieval precision or developing context-based sketching UI. We are also interested in taking annotated images for instance from LabelMe [Russell et al. 2008] or even LabelMe3D [Russell and Torralba 2009] databases as input. The latter allows us to explore qualitative relationships such as attachment and occlusion defined between labeled 2D image objects. Finally, we believe that our concept of structural groups can benefit other applications like open-ended 3D scene synthesis.

Acknowledgements

We thank the reviewers for their constructive comments and the pilot study participants for their time. This work was supported by the National Basic Research Project of China (2011CB302205), the National High Technology Research and Development Program of China (2012AA011802), the Natural Science Foundation of China (61170153 and 61120106007) and Tsinghua University Initiative Scientific Research Program. Hongbo Fu was partially supported by grants from the Research Grants Council of HKSAR (No.113610, No.113513), and the City University of Hong Kong (No.7002925 and No.7002776).

References

Belongie, S., Malik, J., and Puzicha, J. 2002. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 4 (Apr.), 509–522.

Chaudhuri, S., and Koltun, V. 2010. Data-driven suggestions for creativity support in 3d modeling. *ACM Transactions on Graphics 29*, 6, 183:1–183:9.

Chaudhuri, S., Kalogerakis, E., Guibas, L., and Koltun, V. 2011. Probabilistic reasoning for assembly-based 3d modeling. *ACM Transactions on Graphics 30*, 4 (Aug.), 35:1–35:10.

Chen, T., Cheng, M.-M., Tan, P., Shamir, A., and Hu, S.-M. 2009. Sketch2photo: internet image montage. *ACM Transactions on Graphics 28*, 5, 124:1–124:10.

Cheng, M.-M., Zhang, F.-L., Mitra, N. J., Huang, X., and Hu, S.-M. 2010. Repfinder: finding approximately repeated scene elements for image editing. *ACM Transactions on Graphics 29*, 4 (July), 83:1–83:8.

Chopine, A. 2011. *3D Art Essentials: The Fundamentals of 3D Modeling, Texturing, and Animation*. Focal Press.

DeCarlo, D., Finkelstein, A., Rusinkiewicz, S., and Santella, A. 2003. Suggestive contours for conveying shape. *ACM Transactions on Graphics 22*, 3 (July), 848–855.

Divvala, S. K., Hoiem, D., Hays, J. H., Efros, A. A., and Hebert, M. 2009. An empirical study of context in object detection. In *Proceedings of CVPR*, 1271–1278.

Dutagaci, H., Cheung, C. P., and Godil, A. 2010. A benchmark for best view selection of 3d objects. In *Proceedings of the ACM workshop on 3D object retrieval*, ACM, New York, NY, USA, 45–50.

EITZ, M., RICHTER, R., BOUBEKEUR, T., HILDEBRAND, K., AND ALEXA, M. 2012. Sketch-based shape retrieval. *ACM Transactions on Graphics 31*, 4, 31:1–31:10.

FISHER, M., AND HANRAHAN, P. 2010. Context-based search for 3d models. *ACM Transactions on Graphics 29*, 6 (Dec.), 182:1–182:10.

FISHER, M., SAVVA, M., AND HANRAHAN, P. 2011. Characterizing structural relationships in scenes using graph kernels. *ACM Transactions on Graphics 30*, 4 (July), 34:1–34:12.

FISHER, M., RITCHIE, D., SAVVA, M., FUNKHOUSER, T., AND HANRAHAN, P. 2012. Example-based synthesis of 3d object arrangements. *ACM Transactions on Graphics 31*, 6 (Nov.), 135:1–135:11.

FUNKHOUSER, T., KAZHDAN, M., SHILANE, P., MIN, P., KIEFER, W., TAL, A., RUSINKIEWICZ, S., AND DOBKIN, D. 2004. Modeling by example. *ACM Transactions on Graphics 23*, 3, 652–663.

GALLEGUILLOS, C., AND BELONGIE, S. 2010. Context based object categorization: A critical survey. *Computer Vision and Image Understanding 114*, 6, 712–722.

HAYS, J., AND EFROS, A. A. 2007. Scene completion using millions of photographs. *ACM Transactions on Graphics 26*, 3, 4:1–4:7.

HUANG, Q., KOLTUN, V., AND GUIBAS, L. 2011. Joint shape segmentation with linear programming. *ACM Transactions on Graphics 30*, 6, 125:1–125:11.

JOHNSON, M., BROSTOW, G., SHOTTON, J., ARANDJELOVIC, O., KWATRA, V., AND CIPOLLA, R. 2006. Semantic photo synthesis. *Computer Graphics Forum 25*, 3, 407–413.

KALOGERAKIS, E., CHAUDHURI, S., KOLLER, D., AND KOLTUN, V. 2012. A probabilistic model for component-based shape synthesis. *ACM Transactions on Graphics 31*, 4, 55:1–55:13.

LALONDE, J., HOIEM, D., EFROS, A., ROTHER, C., WINN, J., AND CRIMINISI, A. 2007. Photo clip art. *ACM Transactions on Graphics 26*, 3 (Aug.), 3:1–3:10.

LEE, J., AND FUNKHOUSER, T. 2008. Sketch-based search and composition of 3D models. In *Proceedings of EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, 97–104.

LEE, Y. J., ZITNICK, C. L., AND COHEN, M. F. 2011. Shadowdraw: real-time user guidance for freehand drawing. *ACM Transactions on Graphics 30*, 27:1–27:10.

LIU, H., ZHANG, L., AND HUANG, H. 2012. Web-image driven best views of 3d shapes. *the Visual Computer 28*, 3, 279–287.

MERRELL, P., SCHKUFZA, E., LI, Z., AGRAWALA, M., AND KOLTUN, V. 2011. Interactive furniture layout using interior design guidelines. *ACM Transactions on Graphics 30*, 4 (July), 87:1–87:10.

NORIS, G., SÝKORA, D., SHAMIR, A., COROS, S., WHITED, B., SIMMONS, M., HORNUNG, A., GROSS, M., AND SUMNER, R. 2012. Smart scribbles for sketch segmentation. *Computer Graphics Forum 31*, 8, 2516–2527.

OLSEN, L., SAMAVATI, F., SOUSA, M., AND JORGE, J. 2009. Sketch-based modeling: A survey. *Computers & Graphics 33*, 1, 85–103.

RUSSELL, B., AND TORRALBA, A. 2009. Building a database of 3d scenes from user annotations. In *Proceedings of CVPR*, 2711–2718.

RUSSELL, B. C., TORRALBA, A., MURPHY, K. P., AND FREEMAN, W. T. 2008. Labelme: a database and web-based tool for image annotation. *International journal of computer vision 77*, 1-3, 157–173.

SECORD, A., LU, J., FINKELSTEIN, A., SINGH, M., AND NEALEN, A. 2011. Perceptual models of viewpoint preference. *ACM Trans. Graph. 30*, 5 (Oct.), 109:1–109:12.

SHEN, C.-H., FU, H., CHEN, K., AND HU, S.-M. 2012. Structure recovery by part assembly. *ACM Transactions on Graphics 31*, 6, 180:1–180:11.

SHIN, H., AND IGARASHI, T. 2007. Magic canvas: interactive design of a 3-d scene prototype from freehand sketches. In *Proceedings of Graphics Interface*, 63–70.

SIDI, O., VAN KAICK, O., KLEIMAN, Y., ZHANG, H., AND COHEN-OR, D. 2011. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. *ACM Transactions on Graphics 30*, 126:1–126:9.

WANG, Y., ASAFI, S., VAN KAICK, O., ZHANG, H., COHEN-OR, D., AND CHEN, B. 2012. Active co-analysis of a set of shapes. *ACM Transactions on Graphics 31*, 6 (Nov.), 165:1–165:10.

XIE, X., XU, K., MITRA, N. J., COHEN-OR, D., AND CHEN, B. 2012. Sketch-to-design: Context-based part assembly. *ArXiv e-prints* (Dec.).

XU, K., ZHANG, H., COHEN-OR, D., AND CHEN, B. 2012. Fit and diverse: set evolution for inspiring 3d shape galleries. *ACM Transactions on Graphics 31*, 4, 57:1–57:10.

YU, L.-F., YEUNG, S.-K., TANG, C.-K., TERZOPOULOS, D., CHAN, T. F., AND OSHER, S. J. 2011. Make it home: automatic optimization of furniture arrangement. *ACM Transactions on Graphics 30*, 4 (July), 86:1–86:12.

YUMER, M., AND KARA, L. 2012. Co-abstraction of shape collections. *ACM Transactions on Graphics 31*, 6, 166:1–166:11.

ZHENG, Y., FU, H., COHEN-OR, D., AU, O. K.-C., AND TAI, C.-L. 2011. Component-wise controllers for structure-preserving shape manipulation. *Computer Graphics Forum 30*, 2, 563–572.

ZHU, X., JIN, X., LIU, S., AND ZHAO, H. 2012. Analytical solutions for sketch-based convolution surface modeling on the gpu. *the Visual Computer 28*, 11, 1115–1125.

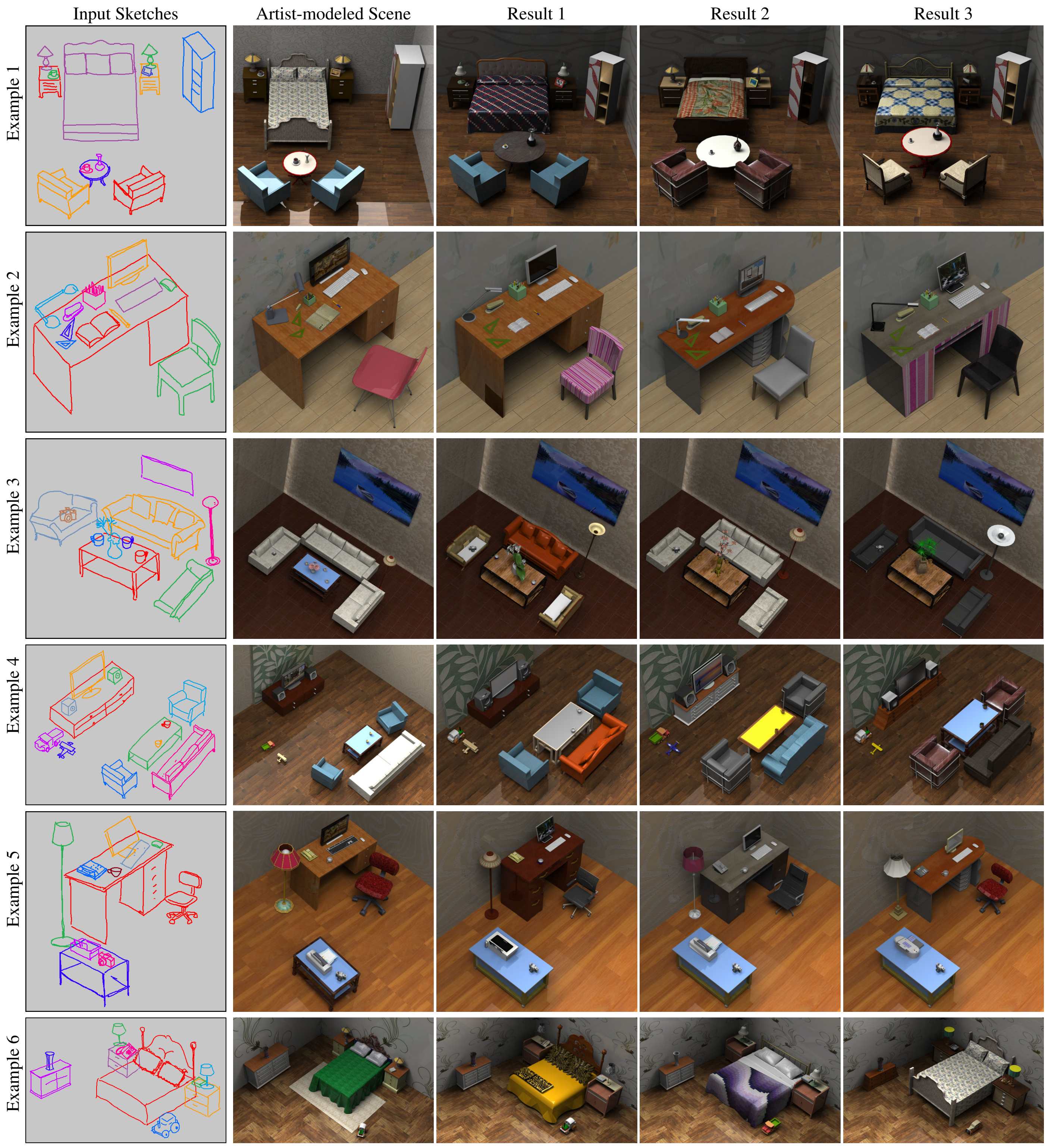


Figure 14: *Gallery of Sketch2Scene results. (The ground, walls, and the painting on the wall were manually added.) The 7th example used in the pilot study is shown in Figure 1.*