SceneCluster: Interactive Scene Synthesis by Clustering Groups of Furniture Objects

Shao-Kui Zhang, Liang Yue, Haoran Zhang, Han-Xi Zhu, Yong-Liang Yang, and Song-Hai Zhang, *Member, IEEE*

***Abstract*—Scene synthesis is crucial to computer graphics. However, the current interactive scene synthesis methods usually cost the user too much time and interactions to edit objects. This paper presents a new interactive scene synthesis method that alleviates the designer from interacting with the 3D scene and its objects. Designers only need to select an object group in an independent panel through coarse clustering and fine clustering. Then, the furniture objects will be automatically added to the scene. This paper proposes a two-level clustering that applies the Affinity Propagation Algorithm (APA) to groups of furniture objects such that the object groups can even be clustered without linear representations, latent encoding, etc. To fully apply the APA, we also propose quantitatively measuring how different the two layouts are, i.e., how quantitatively the arrangements of two object groups differ. Experiments first show that our method is more user-friendly and interactively efficient than other interactive synthesis methods. By comparing our method with recent automatic scene synthesis methods, we demonstrate that our methods still have competitive plausibility. We also verify that our method does not harm the diversity and generalization of 3D scenes.**

***Index Terms*—Multi-function design, interactive 3D modeling, scene reconfiguration.**

I. Introduction

SCENE synthesis intelligently places objects into a room. More and more researchers have applied techniques such as naive Bayes [1], [2], graph algorithms [2], [3], and neural networks [4], [5], [6] to assist designers in building indoor scenes. We recognize the above method and similar methods as *automatic scene synthesis* (Section II-A).

Other works are to recommend relevant objects to designers based on real-time operations to add user preferences, which we refer to as *interactive scene synthesis* [1], [7], [8]. For example, Yu et al. [1] and Zhang et al. [9] encode priors of individual objects. Users can suggest positions as preferences, and the methods help select and rotate appropriate objects. Dong et al. [10] and Liang et al. [7] rearrange objects given user activities. Zhang et al. [11] and Yan et al. [12] enable concurrently and interactively editing multiple objects. The idea is to facilitate users' easy manipulation of scene synthesis (Section II-B).

However, existing interactive techniques still have limitations. First, interacting with the 3D scene consumes most of the time during the synthesis. Regardless of the interactive means above, designers still need to interact with the 3D content. Second, existing literature typically recognizes each object as the unit for layout priors. Thus, the subsequent optimizations/interactions/algorithms need to guarantee the plausibility between/among all objects as much as possible. In contrast, guaranteeing the plausibility between/among groups is much easier if each unit is a group of objects.

We propose a new interactive scene synthesis method to address the limitations above, as shown in Fig. 1. With our method, users only need to operate on an independent panel out of 3D scenes. Our method enables the direct addition of furniture object groups at a time. Therefore, only a few interactions concerning the 3D scene and its objects are required.

The challenge is how we select an appropriate object group. We address this challenge by proposing a two-level clustering method for object groups. Typical clustering methods require encoding elements to latent spaces [13]. However, the objects' numbers of various groups differ. The objects' arrangements also vary, so we apply the Affinity Propagation Algorithm (APA) [14], [15] to cluster object groups (Section IV). Though the APA does not require latent encoding, it still requires quantitative measuring of how two object groups differ. This paper further proposes (1) measuring how two groups are functionally different concerning their functional distributions (Section IV-A) and (2) measuring how two groups' layouts are different concerning the involved objects' arrangements (Section IV-B).

The two-level clustering includes a coarse clustering and a fine clustering. The coarse clustering first organized the object

Received 10 July 2024; revised 6 October 2025; accepted 16 November 2025. Date of publication 19 November 2025; date of current version 6 February 2026. This work was supported by the National Key Research and Development Program of China under Grant 2023YFF0905104, in part by the National Natural Science Foundation of China under Grant 62132012 and Grant 62402262, and in part by the Tsinghua-Tencent Joint Laboratory for Internet Innovation Technology. The work of Yong-Liang Yang was supported by the UKRI grant CAMERA under Grant EP/T022523/1. The work of Shao-Kui Zhang was supported by Young Elite Scientists Sponsorship Program by CAST under Grant YESS20240286. Recommended for acceptance by Q. Huang. *(Corresponding author: Song-Hai Zhang.)*

Shao-Kui Zhang is with the School of Artificial Intelligence, Beijing Normal University, Beijing 100875, China (e-mail: shaokui@bnu.edu.cn).

Liang Yue and Han-Xi Zhu are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100190, China (e-mail: yuel22@mails.tsinghua.edu.cn; 13651325353@sina.cn).

Haoran Zhang is with the Academy of Arts & Design, Tsinghua University, Beijing 100190, China (e-mail: zhanghr23@mails.tsinghua.edu.cn).

Yong-Liang Yang is with the Department of Computer Science, University of Bath, BA2 7AY Bath, U.K. (e-mail: y.yang@cs.bath.ac.uk).

Song-Hai Zhang is with the BNRist, Department of Computer Science and Technology, Tsinghua University, Beijing 100190, China (e-mail: shz@tsinghua.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TVCG.2025.3634829, provided by the authors.

Digital Object Identifier 10.1109/TVCG.2025.3634829

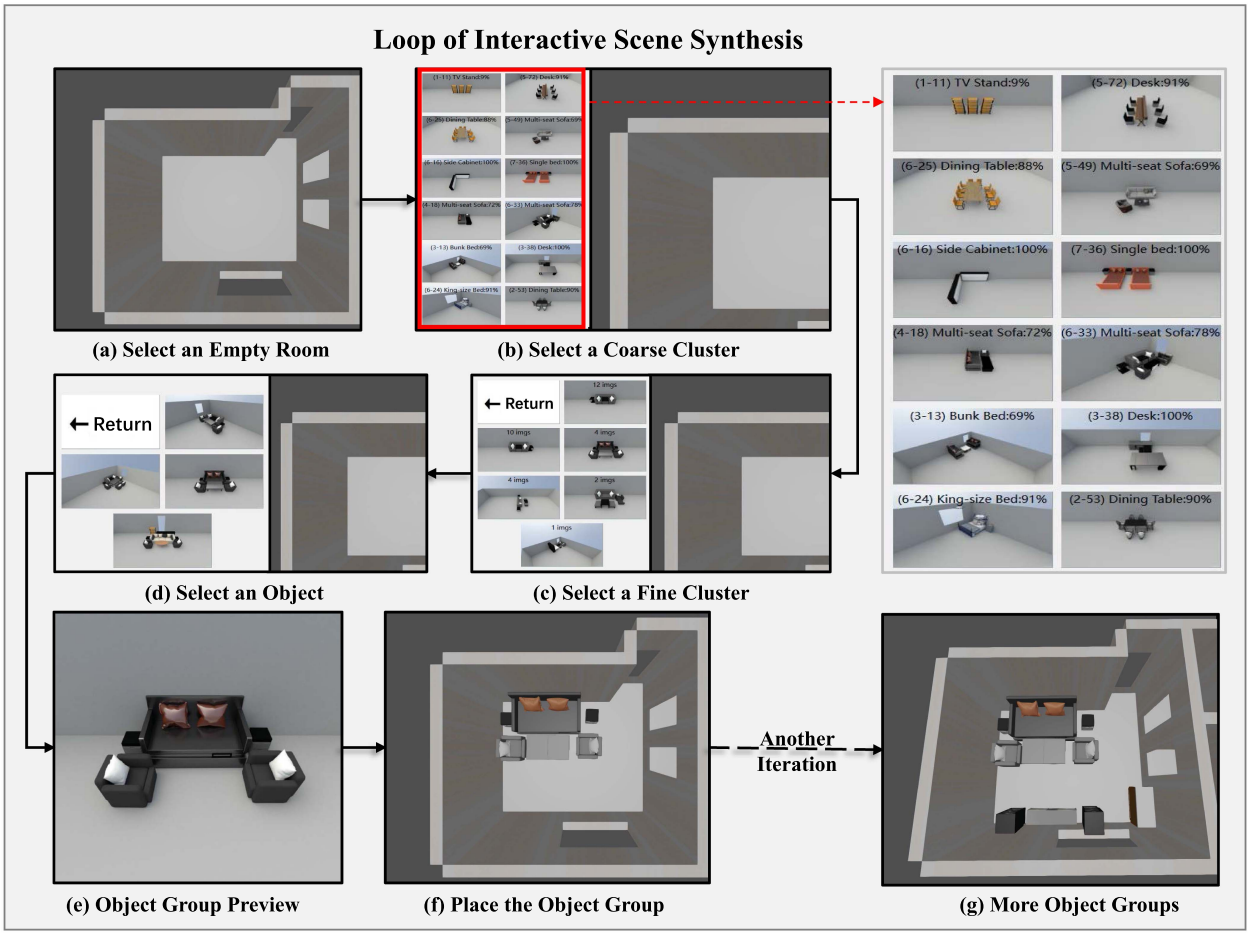


Fig. 1. We introduce an interactive scene synthesis method by clustering groups of furniture objects. Given an empty room (a), a user can successively select a coarse cluster (b), a fine cluster (c) and an object group (d), so an object group is added to the room (f). The coarse and fine clustering refers to our two-level clustering method, which clusters object groups based on their functions and layouts (objects' arrangements). Through a few interactions, our method can quickly synthesize an indoor scene (g). Please refer to a supplementary video for more interactive demos.

groups based on their functions. The fine clustering subsequently organized the object groups (within coarse clusters) based on their layouts. Fig. 1 summarizes how our interactive method is executed, i.e., repeatedly selecting appropriate object groups and adding them to the scene. A designer/user can successively choose a function-preferred cluster and a layout-preferred cluster.

To verify our method, we first compare our interactive method with existing interactive scene synthesis methods, indicating that our method significantly consumes fewer interactions and less time. Then, we compare our interactive method with existing automatic scene synthesis methods, indicating that our interactive method is still competitive with the automatic ones. Finally, we compare the generated scenes with the ones generated by other methods, indicating that our method does not affect the diversity and generalization of the generated scenes.

We made the following contributions:[1]

- We study how to apply clustering to 3D object groups and devise a two-level clustering method based on the Affinity Propagation Algorithm.
- We study how to quantitatively measure the differences between object groups concerning functions and layouts.
- We present a system that significantly reduces the interactions required to synthesize indoor scenes interactively.

II. Related Works

A. Automatic Scene Synthesis

Automatic scene synthesis yields object selections and their layouts given room contours, room functions, list of furniture, etc. Earlier works focused on the automatic layout of furniture models based on rules [16], [17], [18]. A few works utilize graphs [2], [19] or existing examples [20], [21]. Texts [22], RGB-D scans [23] and existing examples [20] are also incorporated to guide automatic scene synthesis. Please refer to a survey for more prior works on automatic scene synthesis [24].

Deep learning techniques have been applied to scene synthesis recently [25]. Early works use the convolution neural network to process the top-down view of indoor scenes and add objects

[1]The code is available at: https://github.com/Shao-Kui/3DScenePlatform. If you need to run the platform, please directly email to Shao-Kui. We will send you a stable version including all code and data. The platform can be run without any configuration.

into scenes iteratively [4], [5]. GRAINS [26] uses a variational auto-encoder (VAE) to synthesize indoor scenes by generating their semantic tree. Zhang et al. [27] use the top-down views and object lists as the input to the deep generative model. Sync2Gen [28] provides several VAEs to predict the objects' attributes and their relative attributes. Then, Sync2Gen uses a Bayesian scene optimization to combine these attributes with the prior distributions learned from the training data. Sceneformer [29] and ATISS [30] encode scenes with the object list and use transformer-based generative methods [31], while DiffuScene [32] encodes scenes with scene graphs and uses a diffusion-based generative model.

Unlike prior works on automatic scene synthesis, this paper focuses on interactive scene synthesis, i.e., giving users control over the synthetic process. Section II-B below discusses directly related literature to our method.

B. Interactive Scene Synthesis

Automatically generated scenes cannot guarantee the users' satisfaction/preference. The construction of 3D scenes also needs to meet the personalized needs of users. Designers generally need to intervene in automatic scene generation interactively. Therefore, existing literature also explores and develops interactive methods (typically systems) to help designers/users edit 3D scenes.

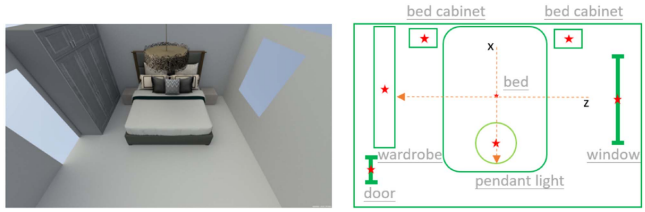
Interactive scene synthesis includes active interaction and passive interaction. Active interaction refers to methods that directly take user input and yield suggestions/recommendations/transformations of objects. For example, Yu et al. [1] add an object to a scene at a time. The subsequent objects are conditioned on the prior-added objects. Similarly, Zhang et al. [9] further enable real-time object exploration upon adding individual objects. Zhang et al. [33] enable users to easily encode relative directions between objects. Yan et al. [12] let users tune a set of objects during iterative optimizations. Zhang et al. [11] enable direct multiple object editing.

Passive interaction refers to learning the user's preferences from their behaviors/habits and generating scenarios matching their preferences. For example, Liang et al. [7] rearrange the workspace according to user activities so that users can conveniently travel in 3D scenes. Zhang et al. [8] rearrange objects concerning multiple users. Dong et al. [10] rearrange objects, preserving the user's perception of scenes.

This paper focuses on active interactions, so passive interactions are out of the scope of this paper. In experiments (Section VI), we will compare our method against the recent interactive methods.

C. Clustering Algorithm

Clustering is an unsupervised machine learning technique that classifies data points so similar data points are classified into the same cluster [34]. Various clustering algorithms are proposed, such as "K-means", "DBSCAN" [35], "Mean Shift", etc. However, these algorithms are applicable when we have data points. This paper needs to cluster object groups. Although it is possible to roughly define the distance between two groups based on the functional differences of their furniture, the data space in which the groups are located does not have a "continuous" property, so we cannot map the object group to a point in the high dimensional space. Calculating the "mean" between furniture object groups as the clustering center is impossible. We can only form a "difference matrix", where each entry suggests how two groups differ, so it is necessary to choose the Affinity Propagation Algorithm (APA) [14] and its subsequent improvements [15], [36] to gradually adjust the element of each category through the affinity between data points.



Object	Position	Orientation	Scale
Bed	[0.00, 0.00, 0.00]	0.00	[0.90, 1.00, 0.88]
Cabinet	[-0.84, 0.00, 1.45]	1.57	[1.13, 1.00, 1.14]
Cabinet	[-0.84, 0.00,-1.45]	1.57	[1.13, 1.00, 1.14]
Light	[0.32, 1.90, 0.04]	1.57	[1.05, 1.00, 0.97]
Wardrobe	[-0.17, 0.00, 2.31]	3.14	[1.44, 1.00, 0.97]
Door	[1.33, 1.05, 2.73]	0.00	[0.80, 2.10, 0.12]
Window	[0.28, 1.58,-2.33]	0.00	[1.25, 1.45, 0.12]

Fig. 2. An object group. The upper shows the corresponding scene with its orthographic view. The table below records each object's relative location (position), orientation, and scale relative to the main object (the bed in this example).

III. Overview

We define an *object group* as several specific objects (furniture) with spatial relationships. The relationships include spatial relations among objects and spatial relations between walls/windows/doors. Fig. 2 shows an example of an object group from which the spatial relationship between furniture objects is quantitatively recorded.

Fig. 3 shows an overview of our method. We first group objects based on an existing 3D scene dataset, i.e., 3D-Front [37]. Please refer to Section A of a supplementary document for how we group objects, thus forming a dataset. Then, we cluster the object groups by their functional distributions and spatial layouts, leveraging the Affinity Propagation Algorithm (Section IV). The first level clustering, i.e., coarse clustering, distinguishes object groups of different functions. Object groups with similar functions are clustered together. Within each coarse cluster, we use the second level clustering, i.e., fine clustering, to classify the object groups concerning their spatial layouts, i.e., object arrangements.

Our system interface shows the clusters for interactive scene synthesis (Section V). We aim to alleviate designers/users when interacting with the 3D scene. Designers/users only need to find an appropriate object group based on the clusters. Our method automatically adds the group into the room.

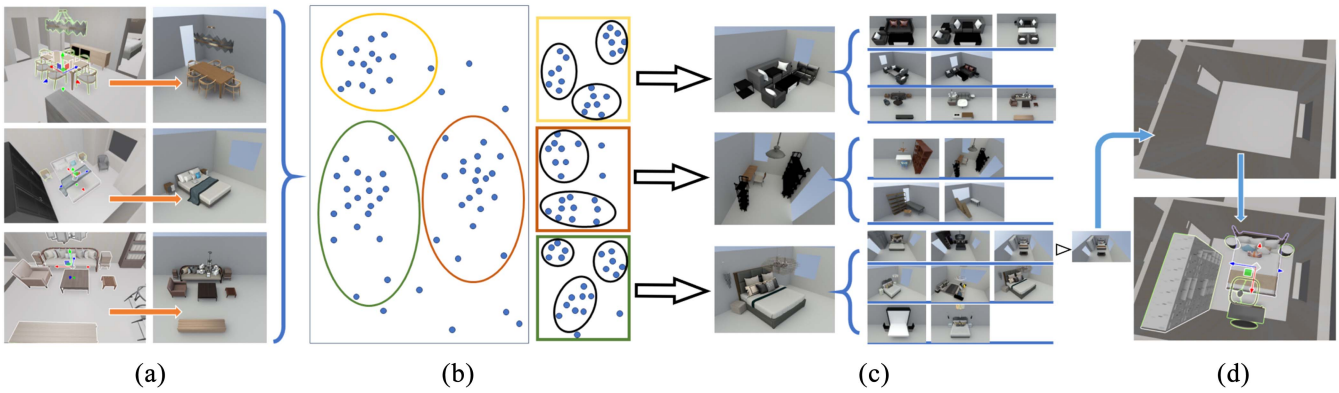


Fig. 3. The method overview. Objects are grouped (a). A two-level clustering is applied to object groups (b), so our system can use coarse/fine clusters to interactively synthesize scenes (c). When a user selects an object group based on the coarse/fine clusters, the group is automatically arranged in the room (d).

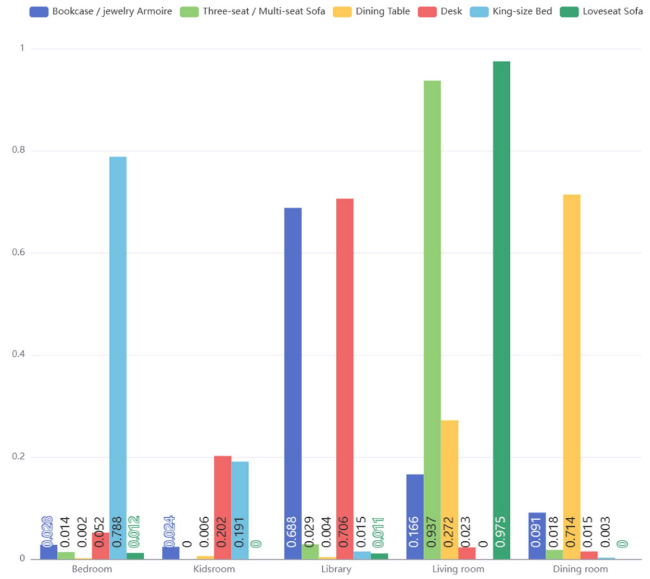


Fig. 4. The distributions of a few furniture types. For example, a loveseat sofa typically exists in a living room, but a king-sized bed in a bedroom and a kidsroom.

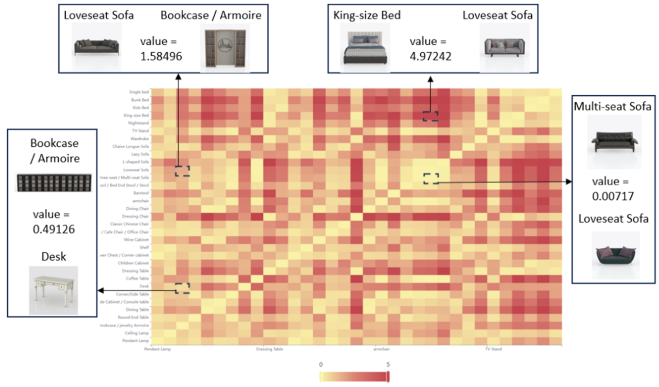


Fig. 5. The value $\hat{KL}(p,q)$ between each furniture type. The darker the color is, the larger their dissimilarity is. For example, "Desk" is similar to "Bookcase" because they relate to studies. The two "sofa" types have a similar distribution because they typically appear in living rooms. In contrast, the dissimilarity between the "Desk" and "Multi-seat Sofa" is large.

IV. Two-Level Clustering

When clustering object groups, we cannot calculate the "average" of these groups. We cannot find the mean point as the clustering center, so traditional clustering algorithms cannot be applied to our problem. We can only use the "difference" between each pair of groups and form the "difference matrix" for the clustering algorithm, leading us to utilize APA [14].

A. Coarse Clustering

Coarse clustering distinguishes the object groups in terms of their functions. Each room has a type, and each object has a semantic. An object may appear in various room types, thus formulating distributions of its semantics about the room types. Fig. 4 shows the distribution of some furniture. Note that the room distribution is not the co-existence of objects. Our method uses the room distributions for the following two-level clustering algorithm.

The difference between the two probability distributions is indicated by KL divergence. We add the forward and reverse KL divergence to $\hat{KL}(p,q)$, as shown in (1), where superscripts r refers to a room type, and p^r means the occurrence of furniture p in room r. The value of $\hat{KL}(p,q)$ is shown as a heat map in Fig. 5. Similar furniture such as "Multi-seat Sofa" and "loveseat sofa" have lower $\hat{KL}(p,q)$, while values with different functions are larger.

$$\hat{KL}(p,q) = KL(p||q) + KL(q||p)$$
$$= \Sigma_r p^r log\left(\frac{p^r}{q^r}\right) + \Sigma_r q^r log\left(\frac{q^r}{p^r}\right) = \Sigma_r (p^r - q^r) log\left(\frac{p^r}{q^r}\right) \quad (1)$$

We use a scale η to cast $\hat{KL}(p,q)$ to the dissimilarity $dsi(p,q)$ by (2). Only the dissimilarity between the same type of objects is 1.0.

$$dsi(p,q) = \begin{cases} 1, & \text{if } p == q \\ \hat{KL}(p,q) * \eta & \text{otherwise.} \end{cases} \quad (2)$$

Then, we formulate the differences between the object groups based on the object dissimilarity above. Suppose we have two

object groups. One consists of a bed and a wardrobe, and another only has an additional cabinet. They are all common bedroom settings, and they are similar. Adding a functionally similar object does not enormously increase their difference. Therefore, each object retains the dissimilarity to the most similar furniture in the opposite group. In doing so, the two groups are similar due to the similar room distribution and low dissimilarity between the cabinet and the bed.

We use the notation p_i to represent the object in an object group containing I objects and the notation q_j to represent the object in another group with J objects. Thus, we represent the smallest dissimilarity as $dsi(p_i) := min_j\{dsi(p_i, q_j)\}$ and $dsi(q_j) := min_i\{dsi(p_i, q_j)\}$. Objects such as sofas, beds, wardrobes, and TV cabinets can dominantly reflect the functions of the object group. We have assigned higher weights to them, represented in sign w_{key}. A softmax with negative scale α amplifies the similarity between more similar objects (objects pairs with lower dissimilarity), as shown in (3), where the first and the last row of (3) is the softmax, and the second row is the weighted average. The weight w_{p_i} and w_{q_j} can be w_{key} or 1.0 depending on whether the object is dominant.

$$edsi(p_i) = e^{\alpha \cdot dsi(p_i)}, edsi(q_j) = e^{\alpha \cdot dsi(q_j)}$$
$$edsi_w = \frac{\Sigma_i edsi(p_i) \cdot w_{p_i} + \Sigma_j edsi(q_j) \cdot w_{q_j}}{\Sigma_i w_{p_i} + \Sigma_j w_{q_j}}$$
$$dsi_w = \frac{1}{\alpha} log(edsi_w) \tag{3}$$

Finally, we use (4) to convert the average dissimilarity to the object group's difference. The theoretical minimum value $dsi_w = 1.0$ should be subtracted from dsi_w because the lowest difference should be zero. We then divide $dsi_w - -1.0$ by $log(I + J)$ to normalize groups with many objects.

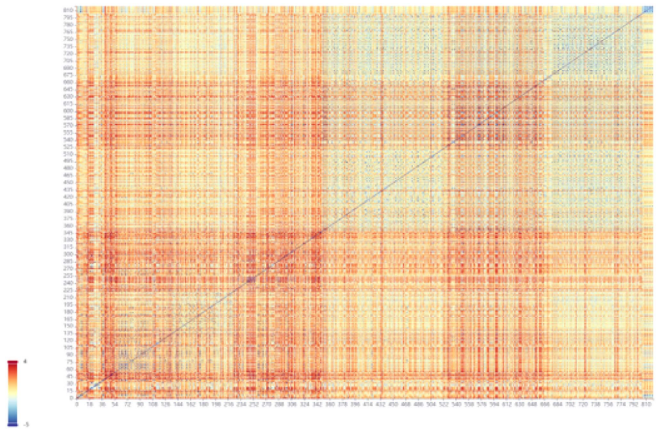
$$D = \frac{dsi_w - -1.0}{log(I + J)} \tag{4}$$

The above differences in object groups are fed into the APA [14], which forms the affiliation relationship between data points (object groups in our case) by maintaining the R matrix, A matrix and S matrix. For data points X and Y, X's affiliation relationship to Y is the tendency of X to be incorporated into the cluster with Y as its center, where Point X chooses Point Y as its exemplar. Our difference among object groups D is stored in the S matrix. The data element in the R matrix, $r_{X,Y}$, reflects how suitable point Y is to serve as the exemplar for point X. The data element in A matrix, $a_{X,Y}$, reflects how appropriate it would be for point X to choose point Y as its exemplar. $r_{X,Y}$ and $a_{X,Y}$ are iteratively updated based on the R, A and S matrix until $r_{X,Y}$ and $a_{X,Y}$ converge. Finally, we put the data points affiliating to the same point into a cluster. The APA has two hyperparameters: the iteration rounds $iter$ and damping factor λ. The hyperparameters in coarse clustering are summarized in Table I.

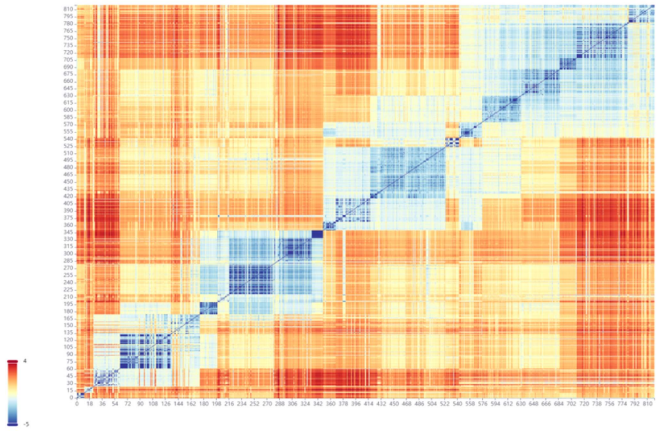
As discussed earlier, we cannot plot object groups on a coordinate system to show the clustering results, so we draw the difference matrices as a heat map in Fig. 6. Fig. 6(a) shows the difference matrix between the object groups before clustering.

TABLE I
HYPER PARAMETERS IN COARSE CLUSTERING

Notation		Value	Description
1	η	11.1	zoom factor from $KL(p, q)$ to $dsi(p, q)$
2	w_{key}	2.0	weights of key furniture
3	α	-1.0	scale factor of softmax
4	λ	0.2	damping factor in AP clustering
5	iter	300	iteration counts in AP clustering



(a) Difference Matrix Before Clustering



(b) Difference Matrix After Clustering

Fig. 6. (a) The difference matrix before clustering. (b): After clustering, we place the object groups in the same cluster together. Blue colors indicate two object groups are similar, while red colors indicate two object groups are dissimilar. Thus, four clusters are indicated by the blue colors in our dataset.

After clustering, we place the object groups of the same cluster together and rearrange the clusters to form a new difference matrix in Fig. 6(b), where the object groups' differences within each cluster are small (the blue area near the diagonal). The object group's differences across clusters (the orange area far from the diagonal) are large.

Fig. 7 marks some clusters and further illustrates Fig. 6. Because the APA is unsupervised and each cluster is not annotated, we have noted the most frequent objects in each category to roughly reflect the primary function. When users/designers interactively synthesize scenes, we also show the most frequent objects to guide them. The blue area on the diagonal of Fig. 6(b) forms four parts. We use a green dashed circle to highlight them in Fig. 7. We found that the four parts represent the common

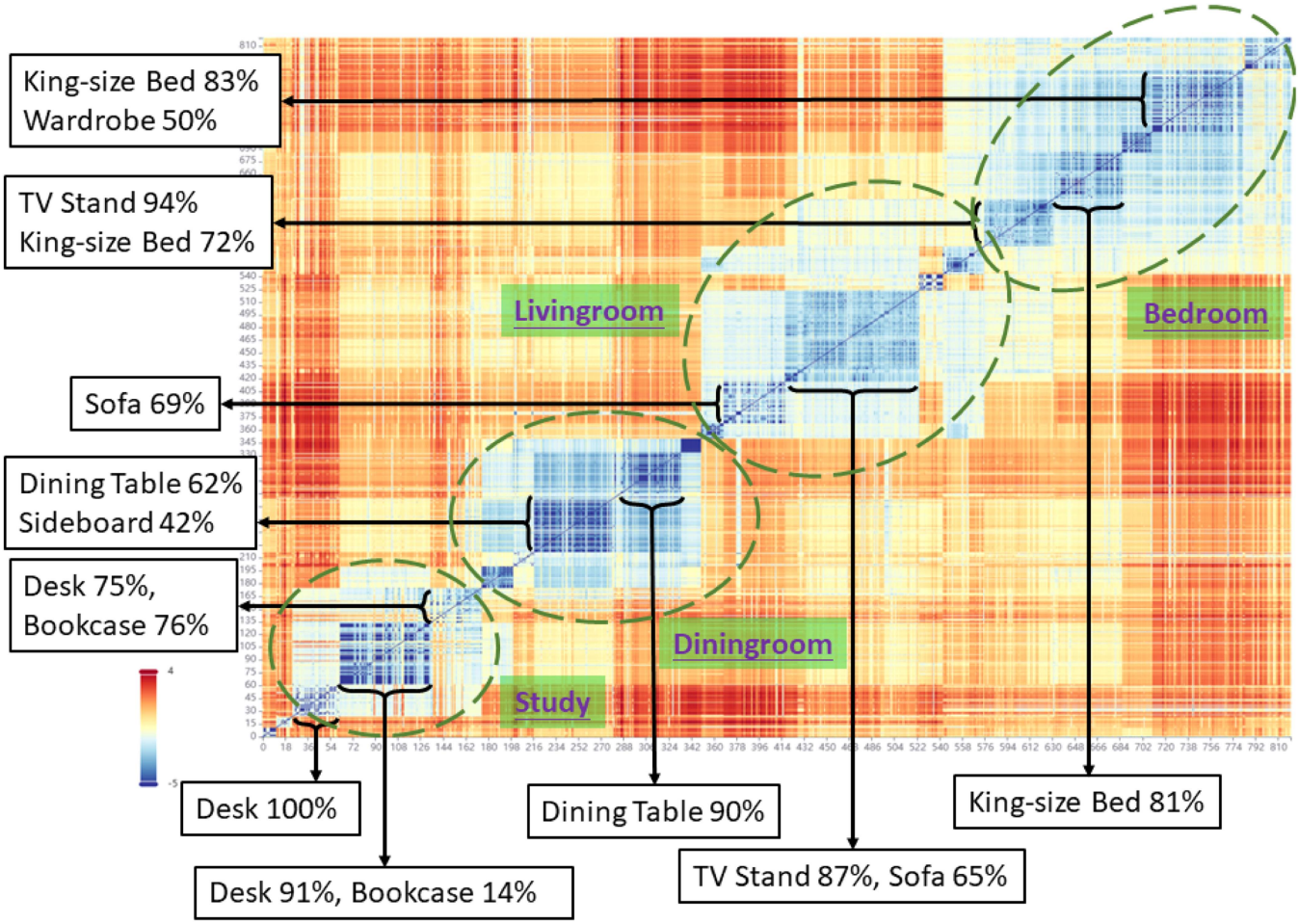


Fig. 7. An annotation of Fig. 6(b). The black arrows and boxes indicate the key furniture's proportion in some clusters. The green circles and boxes indicate the "study", "dining room", "living room", and "bedroom" clusters, respectively.

object groups in "study", "dining room", "living room", and "bedroom", respectively. This is consistent with our intuition. Furthermore, our object-group-oriented clustering is flexible on mixed functions between clusters. For example, the overlap between the living room and bedroom is due to some beds being paired with a TV cabinet, while some sofas are equipped with a nightstand.

B. Fine Clustering

There are many object groups in a cluster, which still consumes time for searching. We should divide one cluster into even smaller groups through the following fine clustering. Fine clustering is also calculated by forming a difference matrix and applying the APA.

1, Pairing. The first step is to pair the objects in two groups. We also assemble $\hat{KL}(p,q)$ in Section IV-A, but we enlarge the scale factor η, so the dissimilarity between furniture objects will be even greater, as shown in (5).

$$dsi(p,q) = \begin{cases} 1, & \text{if } p == q \\ \hat{KL}(p,q) * \eta' & \text{otherwise.} \end{cases} \tag{5}$$

For each pair, we can calculate the dissimilarity of objects and form a small matrix exclusive to the two groups' objects, as shown in Fig. 8. We select the lowest value from the small matrix and pair up the two corresponding objects, where this operation is repeatedly executed until all objects are paired. The pairs are recorded as M. Object numbers in two object groups may differ. Unpaired objects will be paired up with an empty object with a considerable dissimilarity dis_{miss}.

2, Distance Calculation. We calculate the distance between the paired furniture. We align the main objects of the two object groups and put other objects, maintaining their relative transformations concerning the main objects, as shown in Fig. 9. We calculate the distance of fine clustering by the (6). The notation M_m refers to a pair in M. M_m pairs p_i and q_j up. The first term of (6) is positional difference, the second is the square of orientational difference multiplying a weight σ_θ, and the third is the dissimilarity dsi with its weight σ_{dsi}.

$$\begin{aligned} dis(M_m) = ||\vec{p_i} - -\vec{q_j}||_2 + (p_{i\theta} - -q_{j\theta})^2 \cdot \sigma_\theta \\ + dsi(p_i, q_j) \cdot \sigma_{dsi} \end{aligned} \tag{6}$$

3, Weighted Average. Finally, we sum up the $dis(M_m)$ in M. In Fig. 8, the pairs with smaller dissimilarity, such as sofas and

Fig. 8. Pairing object groups (No.386 and No.427). We select a minimum dissimilarity (marked in red) from the matrix and make a pair of the corresponding furniture. The large sofa (Star), coffee table (Square), TV cabinet (Cross), small sofa (Triangle), and lamp (Circle) are all paired together. The side cabinet and small chair are finally paired (Diamond). The last pair is reluctant, so their relationship is insignificant.

TABLE II
HYPER PARAMETERS IN FINE CLUSTERING

Notation		Value	Description
1	η'	111.12	zoom factor from $KL(p,q)$ to $dsi(p,q)$
2	σ_θ	1.2	weight of orientation in $dis(M_m)$
3	σ_{dsi}	1.2	weight of $dsi(p,q)$ in $dis(M_m)$
4	λ'	0.5	damping factor in AP clustering
5	iter′	50	iteration count in AP clustering
6	dis_{miss}	10.0	dissimilarity of unmatched object

TV cabinets, are more critical to calculating $dis(M_m)$. Although the seats and small cabinets are paired up reluctantly, their effect on the overall difference is insignificant. Thus, pairs with smaller $dsi(p,q)$ are more significant in terms of their groups' differences. We set the weights between the more similar pairs higher, as shown in (7). The final result D' is calculated by the weighted average in (8).

$$w_m = [log(dsi(p_i, q_j))]^{-1} \tag{7}$$

$$D' = \frac{\Sigma_m dis(M_m) * w_m}{\Sigma_m w_m} \tag{8}$$

We set "$iter$" to 30 and "λ" to 0.5 since the object groups in each course cluster become fewer. The hyperparameters in fine clustering are summarized in Table II. Since the layouts are more diverse than functions, directly plotting the fine clusters is not intuitive. Please refer to Section B of the supplementary document for more details. Our metric may not perform well when most objects in the two object groups are from different categories, so in (6), the positional and orientational terms are more significant than the dsi term. However, such cases are mostly filtered by coarse clustering and rarely appear.

V. INTERACTIVE SCENE SYNTHESIS

Next, we introduce the process of interactive synthesis. An iteration of adding an object group is shown in Fig. 1. An interactive session has two level directories and four clicks: (1) Click on a room to set it as the target room. (2) Choose a coarse cluster on the panel to the left. While displaying the coarse clusters, we show the number of their fine clusters in it, the total number of object groups in it, and the most frequent key furniture (See Section IV-A). (3) Choose a fine cluster, and the number of object groups in it will be shown simultaneously. (4) After selecting an object group, its furniture will be automatically arranged in the target room. The process only requires operating in an independent panel and avoids editing 3D content. However, users can still adjust the inserted object groups to fine-tune their transformations so the users' preferences are perfectly satisfied.

When interacting with the scene, object groups are recommended based on a correlation-driven strategy, as illustrated in Fig. 1. After a group G_p is selected, we retrieve and rank candidate groups $\{G_q\}$ according to a correlation score. The pairwise score between objects $p \in G_p$ and $q \in G_q$ is computed by cooccurrence probability, where $O(p)$ denotes the frequency of p in the dataset, and $O(p,q)$ represents their co-occurrence count. The overall correlation score between groups is then aggregated. To encourage diversity and avoid recommending overly similar groups, we introduce a penalty based on the Jaccard similarity of object categories (C_p, C_q) between groups. Assuming the selected and candidate groups contain m and n objects, respectively, the final ranking score used for recommendation is defined in (9), where α is a tunable parameter to adjust the penalty. Examples of recommendations are shown in Fig. 10.

$$S(G_p, G_q) = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\frac{O(p,q)}{O(p)} - \alpha\frac{|C_p \cap C_q|}{|C_p \cup C_q|} \tag{9}$$

When arranging an object group, we consider its relative transformations concerning walls, windows, doors, and existing objects, where data priors are calculated to help yield the transformations.

In each object group, the first object is usually the main object. As long as the position of the main object is determined, other objects can be located based on the relative location recorded in the object group. There are ways to locate the main object: if there are windows or doors in the target room and the main object's relative location toward the window or door is available, we can locate the main object accordingly; otherwise, if the object group records its relative location to the nearest wall, we can choose a wall and place the object beside it.

In order to make the interactive indoor scene synthesis convenient, we provide designers with several schemes. When we place the main object according to the window or door, different doors or windows may lead to different locations; when we place it according to the wall, several walls can also lead to several schemes. We can also place the main object beside a wall, even if no window or door is recorded in this object group.

After determining the main object's transformations, other objects can be placed based on their relative transformations. However, the shape of the target room may be irregular. Therefore, after placing the main object, we use the area proportion of the object group's bounding box in the room to determine whether this scheme is feasible roughly. If the area proportion

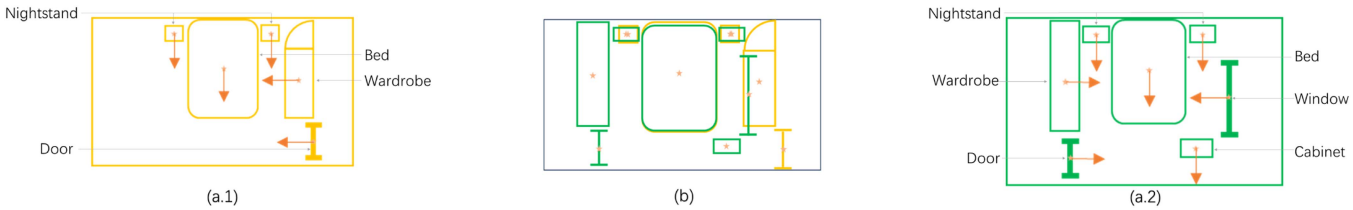


Fig. 9. Calculating distances of fine clustering. (a,1) and (a,2) are object groups where objects' categories, positions and orientations are marked. (b) shows the alignments of the two groups. The beds are the main objects in both groups, so we align the spatial positions of the beds, and other objects are transformed according to the main objects. We then calculate the distance between the other pairs after the alignment. In this example, the relative transformations between the doors/wardrobes and the beds mainly contribute to the differences between the two object groups.

Fig. 10. Examples of recommendation results. Based on the current object group in the room, the system preferentially recommends object groups that are related but not similar. This encourages diversity while maintaining contextual relevance.

is too large, the scheme will be dropped. If the proportion is sufficient for the room, we can slightly adjust the objects' transformations so the object group fits.

The above process includes how we arrange object groups and the attributes within the groups. To achieve a fully automatic version of our framework, we just let the computer select the clusters. The only difference is that our interactive framework does not "select" object. In our implementation, the automatic selections are achieved using the And-Or graph [19], i.e., the frequencies of objects conditioned on their room types.

VI. Experiment

A. Platform and Setup

We implement our interactive scene synthesis method on a platform, as shown in Fig. 11. We render the 3D scene on the front end with Three.js, a popular rendering engine on top of WebGL. The back-end server is implemented with Flask, which organizes scenes and objects.

We have implemented other interactive synthesis methods (i.e., the baselines) on this platform. We have added timers for every operation of interactive scene synthesis so we can record the time and clicks of scene synthesis with different algorithms. The interactive baselines are listed below. Please refer to Section C of a supplementary document for the floorplans used and other details in the experiments.

- *Industrial Method:* This synthesizes scenes with no intelligent suggestions, typical industrial solutions.

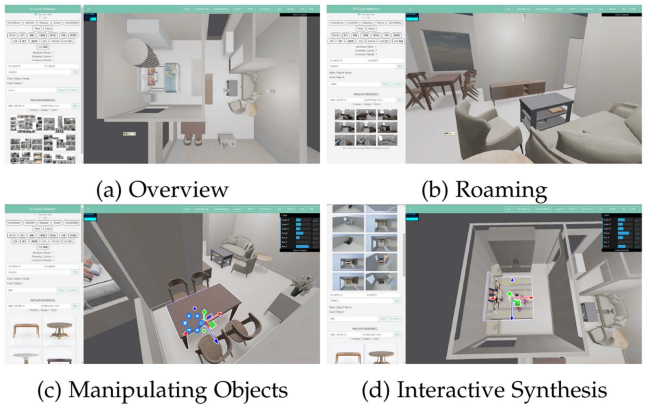
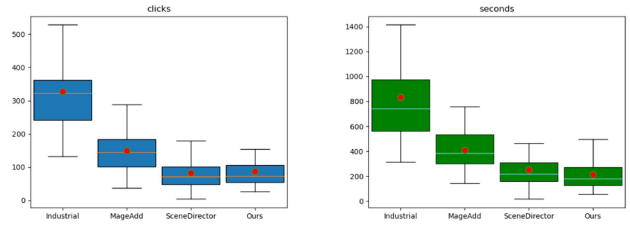


Fig. 11. The interactive scene synthesis platform supports our experiment. This platform provides basic operations for interactive scene synthesis and integrates synthetic methods we test through our user study.

- *MageAdd [9]:* This synthesizes scenes by successively placing objects when users interact with 3D scenes.
- *SceneDirector [11]:* This synthesizes scenes by successively placing multiple objects when users interact with 3D scenes.

B. Interactive Scene Synthesis

This experiment invites participants to fill empty rooms using different interactive scene synthesis methods. Over 250 scenes



(a) Total Clicks (C^m) (b) Total Time (S^m)

Fig. 12. The total clicks and time required by baselines and our method. The red dots indicate averages. The horizontal segments represent medians. The boxes represent quartiles. The two ends of each vertical thin line are two peaks.

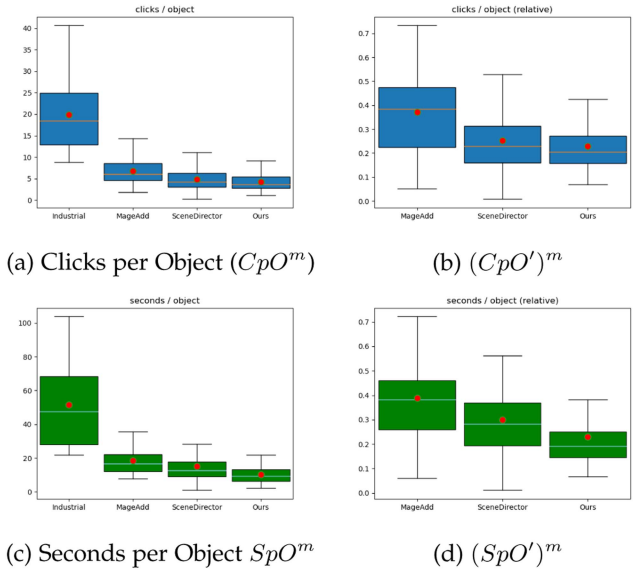
(for each baseline) are synthesized by 30 participants (composed of college students, freelancers, office workers, etc.). Each participant was taught how to use the interactive methods, and she/he could freely use our system until she/he was familiar with all the frameworks. A staff member was available to answer any technical questions during the experiment.

Note that for each interactive method, the input remains the same, but the interactive features differ. For example, MageAdd's feature is that potential objects will pop up when users use the cursor to explore the room [9]. Our feature is directly adding object groups. Despite every interactive method's way to further process data, all of the baselines are built upon the same dataset.

In order to illustrate the experimental results clearly, we define some notations. For baseline m, the total click cost is represented by C^m, and the total time cost is S^m. C^m is an integer. S^m refers to the timer. We plot the total clicks and time required by the baselines in boxplots, as shown in Fig. 12. The time and clicks cost using our method, SceneDirector and MageAdd, are much less than the Industrial Method. Our method and SceneDirector are more efficient than MageAdd. Although our method slightly requires more clicks than SceneDirector, it saves more time. This is because our method saves the interactions required concerning 3D but adds interactions to our panel (Fig. 1), where users click on the panel. Additionally, the standard deviation of SceneDirector is higher than ours, indicating our method is more stable than it.

Three factors affect the C^m and S^m besides m: the number of objects added, the designer's background and room sizes. Firstly, we calculate the "clicks per object" as $CpO^m = C^m/O^m$ and "seconds per object" as $SpO^m = S^m/O^m$ to eliminate the effect of different numbers of objects added. O^m refers to the number of objects added using method m.

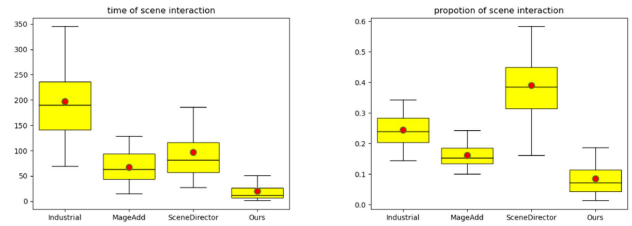
Some participants would choose and adjust the objects carefully, thus spending more time. To eliminate this, we further divide CpO^m and SpO^m by CpO^{Inds} and SpO^{Inds}, where $inds$ represents the industrial method with no intelligent interactive sessions. We have $(CpO')^m = CpO^m/CpO^{Inds}$ and $(SpO')^m = SpO^m/SpO^{Inds}$. Fig. 13 shows the statistics of CpO^m, $(CpO')^m$, SpO^m and $(SpO')^m$, where our method can save the effort of adding individual objects. Finally, we select eight typical floorplans for room sizes to conduct the experiments, where all baselines synthesize scenes given the



(a) Clicks per Object (CpO^m) (b) $(CpO')^m$

(c) Seconds per Object SpO^m (d) $(SpO')^m$

Fig. 13. Clicks and seconds per object.



(a) Total 3D Operation Time (b) Proportion

Fig. 14. The time used in interacting with the 3D objects/scenes. (a) shows the total time for 3D interactions. (b) shows the ratio of the total time for 3D interactions to the entire synthetic time.

same room sizes. The eight floorplans are shown in Section C of the supplementary document. Note that all the methods can synthesize scenes given various floorplans. The eight chosen ones are only for this experiment's fairness.

We record the time of "adding", "deleting", "moving", "rotating", and "scaling" the objects. Our system integrates timers for all such operations. Fig. 14 shows the operation time and its proportion concerning the entire synthetic time. Our method requires significantly less time to edit scenes compared to other baselines. According to the experimental results, we also realize that, though every interactive method requires manually fine-tuning objects' transformations, our method still requires the fewest operations for fine-tuning.

C. Survey by Questionnaire

We invite the participants in Section VI-B to also participate in a questionnaire on various frameworks. The participants score the convenience, flexibility, aesthetic, and overall likeness from 1 to 5 for MageAdd, SceneDirector and our method, i.e., 12 scoring questions. The results are shown in Fig. 15. Our method performs better than the baselines. The participants especially felt comfortable using our method since it alleviates interactions

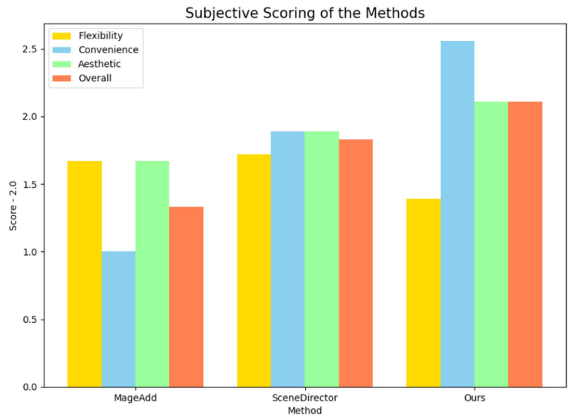


Fig. 15. The average scores of four measurements. They are scored from 1 to 5, and we minus these average scores with 2.0 to magnify the difference.

concerning 3D. However, it's harder to personally edit the object groups, which makes our method less interactively flexible than the baselines. As several participants suggested, our method may be more useful in an area that does not require personality but requires quantity. Although our interactive flexibility can be improved, the generated scenes are still diverse and general. See Section VI-E for more statistics.

D. Automatic Scene Synthesis

To further evaluate the plausibility of our method, we compare our results with the scenes generated by two novel automatic scene synthesis methods: (1) ATISS [30], a transformer-based scene synthesis method, (2) DiffuScene [32], a diffusion-based scene synthesis method. In this experiment, we only compare the final results rather than the interactive sessions. To make this evaluation as fair as possible, we adapt our method to be automatic, i.e., let the computer interact. The computer automatically and randomly selects a coarse and a fine cluster based on the room type. An object group is randomly chosen and arranged in the room according to Section V. This process is executed iteratively until the room is full.

We displayed the scenes using different methods in a questionnaire and asked 20 newly invited participants to evaluate the scenes' *balance*, *aesthetics* and *plausibility*. The *balance* refers to the spatial harmony and utilization of the room, e.g., a room with a half filled with objects and an empty half is not well balanced. The *aesthetic* refers to the degree of visual pleasure. The *plausibility* refers to whether the layout of objects in the scene can achieve the expected function. Participants compare the scenes presented and rate them individually. Each participant was asked to compare 20 pairs of scenes and choose a better one based on *balance*, *aesthetics* and *plausibility*. Each pair of scenes is from two anonymous methods.

The statistical result is shown in Fig. 16, where our method generates competitive results against generative methods based on deep learning. As the object groups originate from the scenes in 3D-Front, our synthetic results are easier to understand than the ATISS and DiffuScene. Hence, its selection rate is larger than that of these two methods.

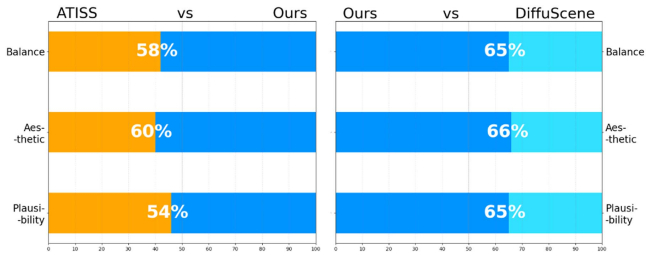
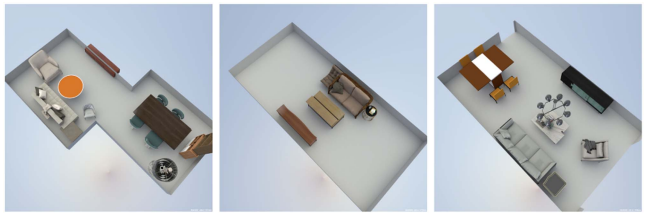


Fig. 16. The statistical result of verifying our plausibility compared with automatic scene synthesis methods. The three bars are *balance*, *aesthetic* and *plausibility*, respectively.



(a) ATISS (b) DiffuScene (c) Ours

Fig. 17. The qualitative results of comparing our method with ATISS and Diffuscene.

Note that it is challenging to fairly compare interactive and automatic methods. This experiment is to verify the plausibility of our method only. Please refer to Section VI-B for comparisons with other interactive methods. Fig. 17 shows several qualitative results. Current automatic scene synthesis methods based on deep learning cannot be directly applied to interactive scene synthesis because they cost much more time than the users' tolerance during real-time interaction.

We also provide FID, KID, SCA and CKL of our method against ATISS and Diffuscene, as shown in Table III. The results show that the rendered (visualized) results from our method are not visually different from the database concerning Diffuscene. Our category distributions of objects are also similar to the database.

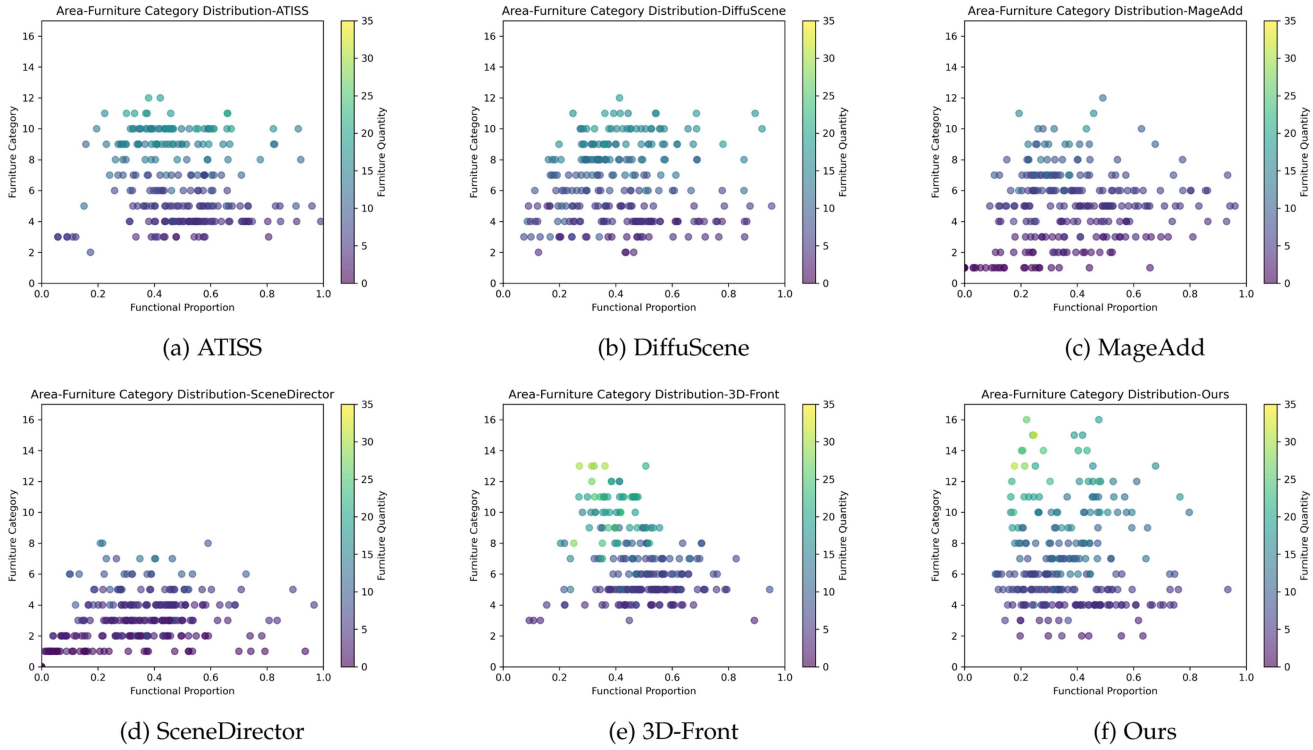
E. Diversity and Generalization

This section verifies that our method does not harm the diversity and generalization of generated scenes. We compare our method with the baselines in Sections VI-B and VI-D. Each baseline generates 250 scenes, and we plot the scenes in Fig. 18, where each point represents a scene and has three channels, including functional proportion, furniture category and furniture quantity. Please refer to Fig. 18's caption for more details.

A diverse set of scenes should be scattered dispersively in Fig. 18. For example, a scene should have various furniture categories to support diverse daily scenarios. A set of scenes should have different degrees of compactness, indicating how various furniture sizes (shapes) are used. The diversity of a scene database can be analyzed through the distribution patterns of these plots. The statistical result shows that scenes generated with our method have competitive diversity against previous methods, analogous to ATISS [30], DiffuScene [32]

TABLE III
THE QUANTITATIVE RESULTS, WHICH COMPARE OUR METHOD WITH AUTOMATIC SCENE SYNTHESIS METHODS. THE FOUR METRICS ARE *FID*, *KID*, *SCA* AND *CKL*

Method	Bedroom				Dinning Room				Living Room			
	FID	KID	SCA	CKL	FID	KID	SCA	CKL	FID	KID	SCA	CKL
ATISS	20.76	2.04	67.53	0.78	38.65	5.77	73.54	0.84	50.83	6.18	72.69	0.78
DiffuScene	18.39	1.42	53.86	0.35	32.80	0.78	55.02	0.27	36.54	1.28	57.76	0.28
Ours	18.47	1.40	55.70	0.34	32.03	1.03	54.65	0.30	35.79	1.25	58.90	0.39



(a) ATISS (b) DiffuScene (c) MageAdd

(d) SceneDirector (e) 3D-Front (f) Ours

Fig. 18. Plotting generated scenes. Each scene is represented as a point, and its color represents its furniture quantity, i.e., the number of furniture objects in each scene. The horizontal axis represents each scene's functional proportion, i.e., the area occupied by furniture divided by the total area of the room. The vertical axis represents each scene's furniture category, i.e., the number of furniture categories involved. For example, a dinning table setting with one dinning table and four chairs comprises five furniture but two categories.

and MageAdd [9]. Scenes built with SceneDirector [11] have the drawback of low furniture categories. While scenes randomly selected from the 3D-Front database [37] have the richest variation in the furniture category, our method can reach similar categorical richness.

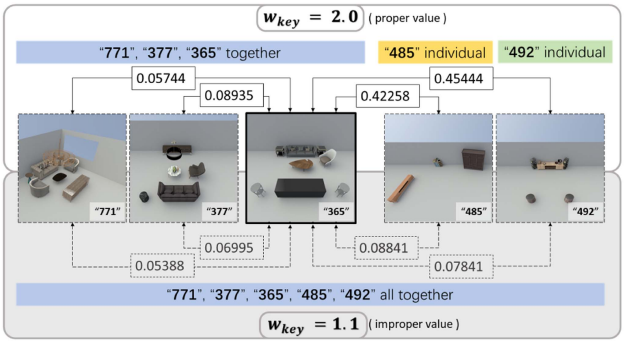
F. Ablation Study

In this section, we tune the hyper-parameters in Tables I and II to yield different clusters. Despite the hyper-parameters for the AP algorithm and the scale factors, we have w_{key}, σ_θ, σ_{dsi} and dis_{miss}.

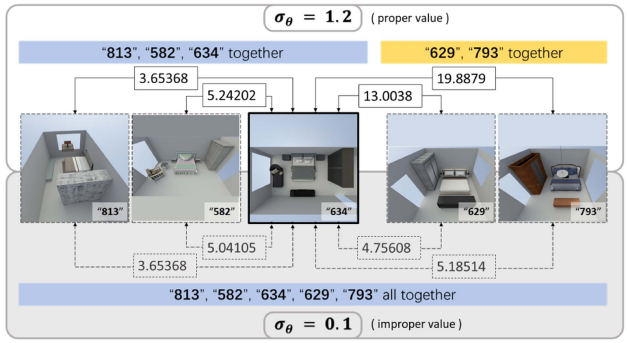
Fig. 19 illustrates the consequences of improperly setting the hyper-parameters. For example, in Fig. 19(b), we compare five bedroom object groups. The three groups on the left have left-side windows, while the two groups on the right have right-side windows. When σ_θ is set to 1.2, it can perfectly distinguish the left three and the right two, separating them into different clusters. However, if $\sigma_\theta = 0.1$, all these groups are combined into a single group. Furthermore, as the lower part of Fig. 19(d) shows, groups of tables with six chairs can be placed in the same cluster as the table with four chairs when dis_{miss} is reduced to 1.0. However, they should be separated, as in the situation in the upper part.

VII. DISCUSSIONS

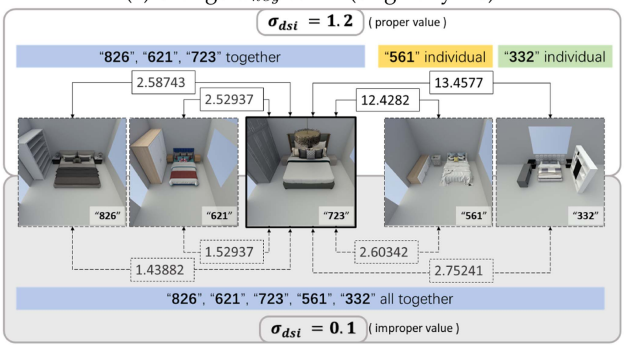
Incorporating Deep Learning based Techniques: Our method leverages data-driven priors derived from the 3D-Front dataset, which encode common spatial relationships between objects and room elements (e.g., walls, windows) based on statistical analysis of real-world scenes. While the current priors are based on dataset statistics, we recognize the potential for learned approaches to further improve generalizability, and we will address this in future work by exploring end-to-end learning models for placement. Besides, recently, more learned distances rise. Learned distance metrics, such as those derived from deep learning-based similarity learning (e.g., SceneFormer [29]), scene graphs (e.g., DiffuScene [32], InstructScene [38]) or embedding techniques (e.g., Forest2Seq [39]), could offer an



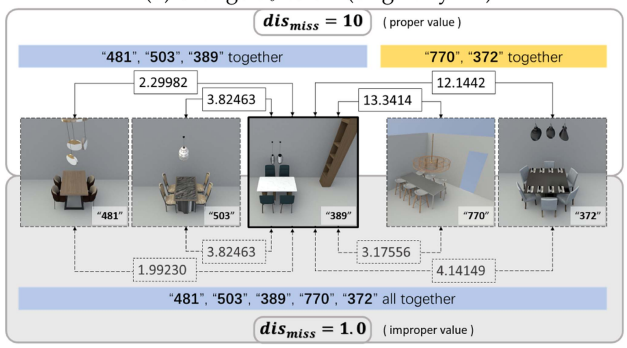
(a) change w_{key} to 1.1 (originally 2.0)



(b) change σ_θ to 0.1 (originally 1.2)

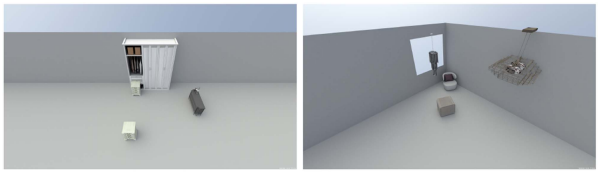


(c) change σ_{dsi} to 0.1 (originally 1.2)



(d) change dis_{miss} to 1.0 (originally 10.0)

Fig. 19. We vary the values of four hyper-parameters (w_{key}, σ_θ, σ_{dsi} and dis_{miss}) to obtain different clustering results. In each figure, we compare the distances among some object groups when a hyper-parameter changes. (a) We reduce the w_{key} from 2.0 to 1.1. The upper part displays the distances (Boxes, Numbers and Arrows) and clusters (Blue, Yellow and Green Boxes) of the five object groups when w_{key} is 2.0. The lower part is when w_{key} is 1.1. We show the distance between Group No. 365 (Middle) and each of the following groups: No. 771, No. 377, No. 485, and No. 492. The three groups on the left are related to living rooms, and the two on the right are some shelves. The three "living room" object groups should be clustered, and the shelves should be individual. Thus, $w_{key} = 2.0$ is proper. All object groups are mixed in one cluster when w_{key} is too small, e.g., $w_{key} = 1.1$. (b) A proper value of σ_θ can distinguish between object groups with left-side windows and those with right-side windows. (c) σ_{dsi} helps distinguish groups with slightly different choices of objects, e.g., a proper value of σ_{dsi} separates object groups with different objects on the left side of the bed. (d) A proper value of dis_{miss} can distinguish object groups with different subordinate objects, e.g., the chairs.



(a) A Wardrobe-Centric Group (b) An Outlier Group

Fig. 20. Our clustering method (Section IV) encounters two types of failure cases: (a) Due to the similarity in object co-occurrence, two groups may be clustered even though the objects involved are different. For instance, the object group consisting of the wardrobe and bedside table, as well as the object group consisting of the bed, have similar frequencies of occurrence in different rooms; therefore, the wardrobe-centric group is mis-clustered into the bed-centric cluster. (b) Some groups should be outliers in terms of the clusters, i.e., they should not belong to any particular cluster. However, our method still assigned them to the nearest cluster.

alternative way to measure the difference between object groups. However, object groups in 3D digital scenes are naturally heterogeneous, i.e., object numbers and arrangements are different. Using APA is our resort. In the future, given extensive data and a way to align the heterogeneity and non-linearity, we believe the learned distances will capture data-driven patterns.

Scability: Our method utilizes the Affinity Propagation Algorithm for clustering, which has a time complexity of O(n^2), where n is the number of object groups. While this quadratic complexity can be challenging for very large datasets, our experiments demonstrate that it handles the scale of typical indoor scene synthesis effectively, as the clustering is performed offline and only once. The manual annotations from the 3D-Front dataset were necessary to ensure high-quality groupings based on functional and spatial relationships, which is critical for meaningful clustering and user experience. This process may not scale effortlessly to massive datasets. Regarding algorithm complexity, APA was chosen for its ability to handle non-metric spaces and avoid the need for predefined cluster centers, which is suitable for our object group representations.

VIII. Conclusion

We proposed a new interactive scene synthesis method by clustering object groups. Our method can synthesize scenes without interacting with them, thus saving human effort. Our method spends less time and fewer clicks than existing interactive methods. Although our interactive synthesis tool provides designers with much convenience, there remain possibilities to improve our method further.

Firstly, more multimodal information should be provided, such as explanatory text when presenting the object groups'

photos to users. What kind of auxiliary information can further effectively help the designers? As many object groups already exist, how can this information be automatically generated? This should be considered in the future.

Secondly, in Section V, we only use a simple idea to arrange objects, which lacks robustness. Although we have considered the irregularity of the room shape as much as possible, it still needs to be improved to handle more complex scene shapes. We may use deep neural networks to learn how to place object groups directly in specific environments.

Thirdly, our distance computation during clustering requires further refinement to address certain corner cases. Groups containing objects from different categories may be assigned to the same cluster. As shown in Fig. 20(a), the wardrobe-centric group and bed-centric group have similar frequencies of occurrence in different rooms due to the co-occurrence of wardrobes and beds. Thus, the wardrobe-centric group is mis-clustered into the bed-centric cluster. Additionally, there are some outliers regarding the clustering Fig. 20(b) that do not belong to any specific cluster. However, they are assigned to the nearest cluster based on our method.

Finally, although our interactive framework allows inserting object groups based on existing ones, overlapping objects may still occur. Currently, when two functional groups overlap, users can either remove or retain them. Removing an object is a quick operation, but we still need a solution to prevent bordering users from being affected by that operation.

REFERENCES

[1] L.-F. Yu, S.-K. Yeung, and D. Terzopoulos, "The clutterpalette: An interactive tool for detailing indoor scenes," *IEEE Trans. Vis. Comput. Graph.*, vol. 22, no. 2, pp. 1138–1148, Feb. 2015.

[2] 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. 1–13, 2017.

[3] Y. Liang, S.-H. Zhang, and R. R. Martin, "Automatic data-driven room design generation," in *Proc. Int. Workshop Next Gener. Comput. Animation Techn.*, 2017, pp. 133–148.

[4] K. Wang, M. Savva, A. X. Chang, and D. Ritchie, "Deep convolutional priors for indoor scene synthesis," *ACM Trans. Graph.*, vol. 37, no. 4, 2018, Art. no. 70.

[5] K. Wang, Y.-A. Lin, B. Weissmann, M. Savva, A. X. Chang, and D. Ritchie, "PlanIT: Planning and instantiating indoor scenes with relation graph and spatial prior networks," *ACM Trans. Graph.*, vol. 38, no. 4, 2019, Art. no. 132.

[6] D. Ritchie, K. Wang, and Y.-A. Lin, "Fast and flexible indoor scene synthesis via deep convolutional generative models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6182–6190.

[7] W. Liang, J. Liu, Y. Lang, B. Ning, and L.-F. Yu, "Functional workspace optimization via learning personal preferences from virtual experiences," *IEEE Trans. Vis. Comput. Graph.*, vol. 25, no. 5, pp. 1836–1845, May 2019.

[8] Y. Zhang, H. Huang, E. Plaku, and L.-F. Yu, "Joint computational design of workspaces and workplans," *ACM Trans. Graph.*, vol. 40, no. 6, pp. 1–16, 2021.

[9] S.-K. Zhang, Y.-X. Li, Y. He, Y.-L. Yang, and S.-H. Zhang, "MageAdd: Real-time interaction simulation for scene synthesis," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 965–973.

[10] Z.-C. Dong et al., "Tailored reality: Perception-aware scene restructuring for adaptive VR navigation," *ACM Trans. Graph.*, vol. 40, no. 5, pp. 1–15, 2021.

[11] S.-K. Zhang, H. Tam, Y. Li, K.-X. Ren, H. Fu, and S.-H. Zhang, "Scenedirector: Interactive scene synthesis by simultaneously editing multiple objects in real-time," *IEEE Trans. Vis. Comput. Graph.*, vol. 30, no. 8, pp. 4558–4569, Aug. 2024.

[12] M. Yan, X. Chen, and J. Zhou, "An interactive system for efficient 3D furniture arrangement," in *Proc. Comput. Graph. Int. Conf.*, 2017, pp. 1–6.

[13] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[14] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007, doi: 10.1126/science.1136800.

[15] C.-D. Wang, J.-H. Lai, C. Y. Suen, and J.-Y. Zhu, "Multi-exemplar affinity propagation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 9, pp. 2223–2237, Sep. 2013.

[16] L.-F. Yu, S. K. Yeung, C.-K. Tang, D. Terzopoulos, T. F. Chan, and S. Osher, "Make it home: Automatic optimization of furniture arrangement," *ACM Trans. Graph.*, vol. 30, no. 4, 2011, Art. no. 86.

[17] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, and V. Koltun, "Interactive furniture layout using interior design guidelines," *ACM Trans. Graph.*, vol. 30, no. 4, 2011, Art. no. 87.

[18] T. Weiss et al., "Fast and scalable position-based layout synthesis," 2018, *arXiv:1809.10526*.

[19] S. Qi, Y. Zhu, S. Huang, C. Jiang, and S.-C. Zhu, "Human-centric indoor scene synthesis using stochastic grammar," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5899–5908.

[20] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, and P. Hanrahan, "Example-based synthesis of 3D object arrangements," *ACM Trans. Graph.*, vol. 31, no. 6, 2012, Art. no. 135.

[21] Q. Fu, S. He, H. Fu, X. Li, and Z. Deng, "Fuzzy-based indoor scene modeling with differentiated examples," *Comput. Vis. Media*, vol. 9, no. 4, pp. 717–732, 2023.

[22] A. Chang, W. Monroe, M. Savva, C. Potts, and C. D. Manning, "Text to 3D scene generation with rich lexical grounding," *arXiv:1505.06289*, 2015.

[23] M. Fisher, M. Savva, Y. Li, P. Hanrahan, and M. Nießner, "Activity-centric scene synthesis for functional 3D scene modeling," *ACM Trans. Graph.*, vol. 34, no. 6, 2015, Art. no. 179.

[24] S.-H. Zhang, S.-K. Zhang, Y. Liang, and P. Hall, "A survey of 3D indoor scene synthesis," *J. Comput. Sci. Technol.*, vol. 34, no. 3, 2019, Art. no. 594.

[25] Q. Fu, S. He, X. Li, and H. Fu, "PlanNet: A generative model for component-based plan synthesis," *IEEE Trans. Vis. Comput. Graph.*, vol. 30, no. 8, pp. 4739–4751, Aug. 2024.

[26] M. Li et al., "Grains: Generative recursive autoencoders for indoor scenes," *ACM Trans. Graph.*, vol. 38, no. 2, pp. 1–16, 2019.

[27] Z. Zhang et al., "Deep generative modeling for scene synthesis via hybrid representations," *ACM Trans. Graph.*, vol. 39, no. 2, pp. 1–21, 2020.

[28] H. Yang et al., "Scene synthesis via uncertainty-driven attribute synchronization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 5630–5640.

[29] X. Wang, C. Yeshwanth, and M. Nießner, "SceneFormer: Indoor scene generation with transformers," in *Proc. Int. Conf. 3D Vis.*, 2021, pp. 106–115.

[30] D. Paschalidou, A. Kar, M. Shugrina, K. Kreis, A. Geiger, and S. Fidler, "Atiss: Autoregressive transformers for indoor scene synthesis," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds. Red Hook, NY, USA: Curran Associates, Inc., 2021, pp. 12013–12026.

[31] A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon et al., Eds. Red Hook, NY, USA: Curran Associates, Inc., 2017.

[32] J. Tang, Y. Nie, L. Markhasin, A. Dai, J. Thies, and M. Nießner, "DiffuScene: Denoising diffusion models for generative indoor scene synthesis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2024, pp. 20507–20518.

[33] S. Zhang, Z. Han, Y.-K. Lai, M. Zwicker, and H. Zhang, "Active arrangement of small objects in 3D indoor scenes," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 4, pp. 2250–2264, Apr. 2021.

[34] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, Sep. 1999. doi: 10.1145/331499.331504.

[35] K. Khan, S. U. Rehman, K. Aziz, S. Fong, and S. Sarasvady, "DBSCAN: Past, present and future," in *Proc. 5th Int. Conf. Appl. Digit. Inf. Web Technol.*, 2014, pp. 232–238.

[36] K. Wang, J. Zhang, D. Li, X. Zhang, and T. Guo, "Adaptive affinity propagation clustering," *CoRR*, vol. abs/0805.1096, 2008. [Online]. Available: http://arxiv.org/abs/0805.1096

[37] H. Fu et al., "3D-front: 3D furnished rooms with layouts and semantics," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 10933–10942.

[38] C. Lin and Y. Mu, "Instructscene: Instruction-driven 3D indoor scene synthesis with semantic graph prior," *arXiv:2402.04717*, 2024.

[39] Q. Sun, H. Zhou, W. Zhou, L. Li, and H. Li, "Forest2Seq: Revitalizing order prior for sequential indoor scene synthesis," in *Proc. Eur. Conf. Comput. Vis.*, 2024, pp. 251–268.

Shao-Kui Zhang received the PhD degree of computer science and technology from Tsinghua University, Beijing, in 2023. He is currently an associate professor with the School of Artificial Intelligence, Beijing Normal University. His research interests include computer graphics, 3D scene synthesis, and intelligent 3D scene interaction.

Liang Yue received the BS degree of computer science from Peking University, Beijing, in 2022. He is currently working toward the master's degree with the Department of Computer Science and Technology, Tsinghua University, Beijing, China. His research interests include computer graphics, 3D scene synthesis, and intelligent 3D scene interaction.

Haoran Zhang received the BS degree in architecture from Tsinghua University, Beijing, in 2023. He is currently working toward the master degree with the Academy of Arts & Design, Tsinghua University, Beijing, China. His research interests include computer graphics and 3D scene synthesis.

Han-Xi Zhu is currently working toward the undergraduate degree with the Department of Computer Science and Technology, Tsinghua University, Beijing, China. His research interests include computer graphics, 3D scene synthesis, and intelligent 3D scene interaction.

Yong-Liang Yang received the BS and PhD degrees in computer science from Tsinghua University. He is a senior lecturer with the Department of Computer Science, University of Bath. His research interests include broadly in visual computing and interactive techniques.

Song-Hai Zhang (Member, IEEE) received the PhD degree of computer science and technology from Tsinghua University, Beijing, in 2007. He is currently an associate professor with the Department of Computer Science and Technology, Tsinghua University. His research interests include computer graphics and virtual reality.