Computer-Aided Layout Generation for Building Design: A Review

Jiachen Liu[1] Yuan Xue[2] Haomiao Ni[3] Rui Yu[4] Zihan Zhou[5] Sharon X. Huang[1]
[1]The Pennsylvania State University [2]The Ohio State University [3]University of Memphis
[4]University of Louisville [5]Manycore Tech Inc.

Abstract

Generating realistic building layouts for automatic building design has been studied in both the computer vision and architecture domains. Traditional approaches from the architecture domain, which are based on optimization techniques or heuristic design guidelines, can synthesize desirable layouts, but usually require post-processing and involve human interaction in the design pipeline, making them costly and time-consuming. The advent of deep generative models has significantly improved the fidelity and diversity of the generated architecture layouts, reducing the workload by designers and making the process much more efficient. In this paper, we conduct a comprehensive review of three major research topics of architecture layout design and generation: floorplan layout generation, scene layout synthesis, and generation of some other formats of building layouts. For each topic, we present an overview of the leading paradigms, categorized either by research domains (architecture or machine learning) or by user input conditions or constraints. We then introduce the commonly-adopted benchmark datasets that are used to verify the effectiveness of the methods, as well as the corresponding evaluation metrics. Finally, we identify the well-solved problems and limitations of existing approaches, then propose new perspectives as promising directions for future research in this important research area. A project associated with this survey to maintain the resources is available at awesome-building-layout-generation.

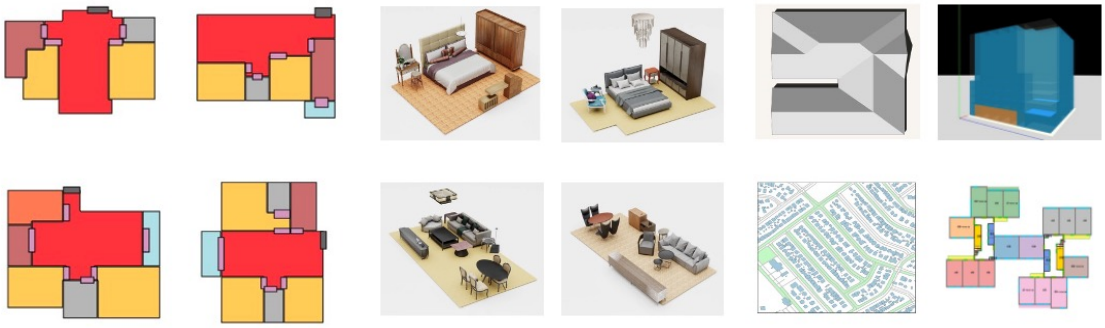
***Keywords:** Computer-aided design, building layout, machine learning, deep generative models.*

1. Introduction

Architecture design is important to ensure buildings and living spaces can fulfill their intended functions and requires careful consideration of layout, circulation, integration of amenities and furniture, as well as visual appearance. A well-designed residential house or community can provide a comfortable, salubrious and appealing environment for people to live in. However, the complete workflow of designers and architects usually involve handcrafted configurations from specific domains and multi-round refinement according to users' feedback, which makes this task time consuming and expensive.

The advancement in image generation techniques [46, 65, 6, 166, 54] has inspired designers and researchers to rethink the workflow of architectural design. How to employ advanced machine learning techniques as assistance for computer-aided architectual design has captured attention and efforts from both architects and machine learning researchers. In the area of computer-aided building layout design, there are two mainstream research directions. The first one, residential floorplan layout generation, aims to synthesize diverse and realistic residential house layouts based on various types of user input to meet specific requirements. The second one, scene layout synthesis, endeavors to generate complete scenes including furniture objects that have desired semantic and spatial attributes such as object category, location, size and orientation. In addition to these two main topics, there are also other emerging topics such as block-plan generation, volumetric building design, roof synthesis, among others. Please refer to Fig. 1 for several representative building layouts used in computer-aided design.

Different from natural image synthesis, the problem of building layout generation has its distinct characteristics. First, the layout design process typically involves user interactions and constraints. A designer can specify the spatial connectivity, dimensions of rooms or furniture, according to customer needs, preferences, and building budget. This requires the layout generation model to have the ability to condition on user inputs as constraints, other than merely perform generation in an unconditional way. Second, unlike natural images which can be captured from arbitrary locations and viewpoints with irregular distributions, building layouts usually encapsulate notable structural regularities. For instance, the surrounding walls usually hold orthogonality (also known as Manhattan-world assumption [23]) for most buildings, and the walls are typically axis-aligned. For building designs, two apartments facing each other on the same floor often have similar or symmetric residential layouts. These regularities can serve as distribution priors which can be leveraged to alleviate the challenges associ-



Floorplan Layouts Scene Furniture Layouts Other Types of Building Layouts

Figure 1. The diverse building layout types used in mainstream computer-aided architectural design methods. The floorplan and scene furniture layouts are the two most common types of layouts. Moreover, there exist other types including roofs, building volumes, community or urban-level layout, etc.

ated with data sparsity, model complexity and parameter estimation.

Last but not least, in natural image generation, a model with pixel-level encoding and decoding is usually expected to form an entire image, whereas in building layout generation, a set of vectors can usually appropriately represent the house or furniture layout. A rendering process which fills color into the generated vectorized boundary is followed for the purpose of displaying and visualization. These aforementioned key differences motivate researchers to develop diverse but distinctive design pipelines to address the layout generation problem.

Considering the importance of computer-aided layout generation for building design, it is worthwhile to have a comprehensive survey to review existing methods and provide perspectives to inspire future research directions. A couple of related survey papers have been published. Weber *et al.* [145] gives a survey on automated floorplan generation paradigms, which are categorized into bottom-up, top-down methods from the architecture domain and referential methods from other domains especially machine learning (ML). However, the scope of this survey is limited on architectural floorplan generation, rather than have a broader discussion on other types of building layout designs. Ritchie *et al.* [109] gives a comprehensive review of neurosymbolic models used in computer graphics. The paper focuses on a survey of representations for diverse geometric symbols and primitives, such as 2D or 3D shapes, materials and texture, in the architectural design area and beyond.

These existing surveys either focus on a single topic (e.g. [145]) or cover topics that are too broad to be categorized as computer-aided layout generation (such as [109]). To the best of our knowledge, there is no prior work that comprehensively provides a comprehensive introduction and overview on methods for computer-aided architecture design, specifically.

In this paper, we aim to conduct a comprehensive review of methods for computer-aided layout generation for building design. The layouts are not limited to floor plans, but also include scene layouts, building-level and site-level layouts.

On each topic of interest, we first try to generally introduce the problem to handle, and formulate the task in a mathematical way. We then present existing approaches that could be categorized by various perspectives. We also summarize associated benchmark datasets used by current works as well as widely-adopted evaluation metrics to assess solutions for completing the task. At the end of the paper, we provide a summary of current research focuses and trends in this area, and present several perspectives related to addressing open challenges and initiating new research directions.

This survey paper is organized as follows: In Sec. 2, we give a brief introduction to the preliminaries of mainstream deep generative models from the machine learning and computer vision domains, which usually serve as the fundamental frameworks used by state-of-the-art (SOTA) data-driven layout synthesis works. In Sec. 3, we formulate the task of residential floorplan generation, review methodologies as grouped by research domains or different formats of user input, then summarize widely-used datasets and evaluation metrics. In Sec. 4, we introduce the problem of scene layout synthesis and review the mainstream paradigms for tackling the problem, benchmarks datasets and evaluation protocols. In Sec. 5, we review existing works for solving other problems related to building layout design such as building-level layout generation and site-level layout generation. In each section, we discuss the advantages and limitations of existing works, as well as the remaining challenges. Finally, in Sec. 6, we present new perspectives that hopefully can inspire future research directions and outstanding works in this important area.

2. Preliminaries of Deep Generative Models

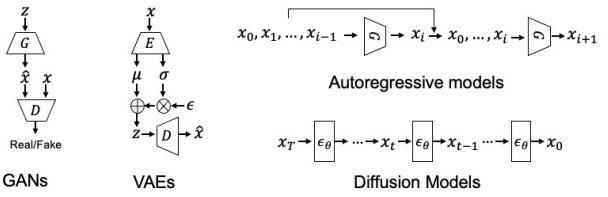


Figure 2. The typical workflow of different representative deep generative models, including GANs, VAES, Autoregressive Models and Diffusion Models (DMs).

Recent advancements in layout generation primarily leverage deep generative models, a class of techniques that employ deep neural networks to model the distribution of training samples [15]. We now present some commonly used deep generative modeling methods, such as generative adversarial networks (GANs), variational autoencoders (VAEs), autoregressive models, and the latest diffusion models (DMs). The typical workflow for each of these methods is intuitively illustrated in Fig. 2.

2.1. Generative Adversarial Networks

A typical Generative Adversarial Network (GAN) [46] comprises two sub-networks: a discriminator network distinguishing between real and generated images, and a generator network creating images to deceive the discriminator. Specifically, with the input being a noise vector $z \sim p_{\mathcal{Z}}(z)$, the generator learns a differentiable mapping function $G(z)$ from noise space $\mathcal{Z}$ to data space $\mathcal{X}$. The discriminator, $D(x)$, is trained to output a single scalar indicating the probability that x originates from real data rather than from the generator. The discriminator D is trained to maximize the probability of real data x and minimize the probability of fake data generated by the generator G. And G is trained to minimize $\log(1 - D(G(z)))$, thus maximizing $D(G(z))$ when D is fixed. The overall training objective function can be defined as:

$$\min_G \max_D \mathbb{E}_{x\sim p_{\mathcal{X}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_{\mathcal{Z}}(z)}[\log(1-D(G(z)))] \ . \tag{1}$$

The training of GANs is widely recognized as challenging due to their inherent adversarial nature [6]. As the discriminator becomes more effective, the gradients conveyed to the generator diminish; conversely, when the discriminator performs poorly, the generator does not receive informative gradients. Another issue is mode collapse, where one network becomes trapped in a bad local minimum, resulting in the learning of only a limited subset of the data distribution. To address these problems, many other loss functions have been proposed. One notable example is WGAN [7], which utilizes the Wasserstein distance to measure the difference between distributions. The training objective function of WGAN is defined as:

$$\min_G \max_D \mathbb{E}_{x\sim p_{\mathcal{X}}(x)}[D(x)] - \mathbb{E}_{z\sim p_{\mathcal{Z}}(z)}[D(G(z))] \ . \tag{2}$$

Note that unlike the original GAN model Eq. (1), where the discriminator D serves as a binary classifier, in WGAN, D is employed to approximate the Wasserstein distance, which is a regression task. Consequently, the sigmoid function, usually present in the last layer of D, is omitted in WGAN. WGAN also uses weight clipping as a strategy to enable network training with the approximated Wasserstein distance.

2.2. Variational Autoencoders

Consider a latent-based model $p_\theta(x|z)$ with a prior $p_\theta(z)$ and a posterior $p_\theta(z|x)$. As the direct computation of $p_\theta(x) = \int_z p_\theta(x|z)p_\theta(z)dz$ is not tractable, to maximize the data likelihood $p_\theta(x)$, one can leverage variational inference [65] to establish a tractable lower bound on $p_\theta(x)$. This can be achieved by introducing an approximation of the true intractable posterior, denoted as $q_\phi(z|x) = \text{argmin}_q D_{KL}(q_\phi(z|x)||p_\theta(z|x))$. In particular, variational autoencoders employ a feed-forward inference network to approximate $q_\phi(z|x)$, enabling scalability to large datasets [65, 108]. From the definition of KL divergence [25], we have:

$$D_{KL}(q_\phi(z|x)||p_\theta(z|x)) = \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{q_\phi(z|x)}{p_\theta(z|x)}\right] \ . \tag{3}$$

Thus we can get:

$$\begin{aligned}
\log p_\theta(x) = \ & D_{KL}(q_\phi(z|x)||p_\theta(z|x)) - \mathbb{E}_{q_\phi(z|x)}[\log q_\phi(z|x)] \\
& + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(z,x)] \\
\geq \ & - \mathbb{E}_{q_\phi(z|x)}[\log q_\phi(z|x)] + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(z,x)] \\
= \ & - \mathbb{E}_{q_\phi(z|x)}[\log q_\phi(z|x)] + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(z)] \\
& + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] \\
= \ & - D_{KL}(q_\phi(z|x)||p_\theta(z)) + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] \\
\equiv \ & \mathcal{L}(\theta,\phi;x) \ ,
\end{aligned} \tag{4}$$

where $\mathcal{L}$ is known as the evidence lower bound (ELBO) [137]. Hence, maximizing the data likelihood estimation can be attained by maximizing the ELBO $\mathcal{L}$ in Eq. 4. To optimize this bound with respect to parameters θ and ϕ, gradients need to be back-propagated through the stochastic sampling process $z \sim q_\phi(z|x)$. This can be achieved by reparameterizing z to move the sampling operation to an input layer. Specifically, when $z \sim q_\phi(z|x) = \mathcal{N}(z;\mu,\sigma^2\mathbf{I})$, we can sample z by first sampling $\epsilon \sim \mathcal{N}(0,\mathbf{I})$, and then computing $z = \mu + \sigma \cdot \epsilon$.

2.3. Autoregressive Models

Based on the chain rule of probability, autoregressive models [13] generate the variables in an iterative manner. Suppose the variable to be generated can be decomposed as $\mathbf{x} = x_1, x_2, ..., x_n$, the probability of generating such a sequence can be represented as:

$$p(x) = p(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} p(x_i|x_1, x_2, ..., x_{i-1}). \quad (5)$$

In practice, to make the optimization process easier for the network, the negative log-likelihood $-ln\ p(x)$ is minimized during training, which is essentially equal to maximizing the original probability $p(x)$:

$$-ln\ p(x) = -\sum_{i=1}^{n} ln\ p(x_i|x_1, x_2, ..., x_{i-1}). \quad (6)$$

The common network architectures of autoregressive models include masked MLPs [68], recurrent neural networks (RNNs) such as LSTMs [55] and GRUs [22], causal convolutions [20] and masked self-attention-based Transformers [136].

Autoregressive models are natural choices when the modalities for generation can be represented as ordered sequences, such as text or audio. For other modalities such as images, one would need to find a way to transform data in the original modalities into sequence representations. A common limitation of autoregressive models is that when the sequence turns longer, the computational cost increases drastically, leading to much slower generation for the later time steps of the sequence. On layout generation problems, since layouts can be typically represented as vectorized sequences, autoregressive models, known for their effectiveness in this representation capacity on sequential data, are widely adopted to model the generation process. We will discuss the details of such works in the next sections.

2.4. Diffusion Models

Diffusion models (DMs) [54, 123, 125] are probabilistic models designed to learn a data distribution. DMs typically consist of a forward process and a reverse process. Given a sample from the data distribution $x_0 \sim q(x_0)$, the DM *forward* process produces a Markov chain $x_1, \ldots, x_T$ by gradually adding Gaussian noise to x_0 based on a variance schedule $\beta_1, \ldots, \beta_T$, that is:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}) \ , \quad (7)$$

where variances β_t are constants. If β_t are small, the posterior $q(x_{t-1}|x_t)$ can be well approximated by diagonal Gaussian [93, 123]. Furthermore, when the T of the chain is large enough, x_T can be well approximated by standard Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$. These suggest that the true posterior $q(x_{t-1}|x_t)$ can be estimated by $p_\theta(x_{t-1}|x_t)$ defined as [94]:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t), \Sigma_\theta(x_t)) \ . \quad (8)$$

The DM *reverse* process (also known as *sampling*) then generates samples $x_0 \sim p_\theta(x_0)$ by initiating a Markov chain with Gaussian noise $x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and progressively decreasing noise in the chain of $x_{T-1}, x_{T-2}, \ldots, x_0$ using the learnt $p_\theta(x_{t-1}|x_t)$. To learn $p_\theta(x_{t-1}|x_t)$, Gaussian noise ϵ is added to x_0 to generate samples $x_t \sim q(x_t|x_0)$, then a model ϵ_θ is trained to predict ϵ using the following mean-squared error loss:

$$L_{\text{DM}} = \mathbb{E}_{t\sim\mathcal{U}(1,T), x_0\sim q(x_0), \epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}[||\epsilon - \epsilon_\theta(x_t, t)||^2] \ , \quad (9)$$

where time step t is uniformly sampled from $\{1, \ldots, T\}$. Then $\mu_\theta(x_t)$ and $\Sigma_\theta(x_t)$ in Eq. (8) can be derived from $\epsilon_\theta(x_t, t)$ to model $p_\theta(x_{t-1}|x_t)$ [54, 94]. The denoising model ϵ_θ is typically implemented using a time-conditioned U-Net [113] with residual blocks [50] and self-attention layers [136]. Sinusoidal position embedding [136] is also usually used to specify the time step t to ϵ_θ.

3. Residential Floorplan Layout Generation

In this section, we start with the task of interior floorplan layout generation, which is a representative layout format among all types of building layouts. In terms of methodology, we categorize the approaches of the generation task into two groups: traditional optimization-based approaches from the architectural domain, and data-driven learning-based approaches from the domains of machine learning and computer-aided design. We review methods that are recently proposed in each group. We also provide a summary of publicly-available benchmark datasets and widely-adopted evaluation protocols for the task. The section ends with a comparative analysis of competing methods for the task.

3.1. Task formulation

A floorplan contains both room elements as well as connection elements such as doors and windows. For simplicity, and unless specified otherwise, the remainder of this survey paper will use the term 'room' to refer to all these elements, without distinguishing between 'room' and 'connection'. Because of the geometric attributes of real-world room layouts, the vertices of a room R are usually represented as a polygonal vector which consists of a group of self-looped corner coordinates:

$$\boldsymbol{R} = \{\boldsymbol{r}_{x_i}, \boldsymbol{r}_{y_i}\}_{i=1}^{n} \quad (10)$$

where r_{x_i} and r_{y_i} are the x and y coordinates of the i-th corner of the room R, and n is the total number of room

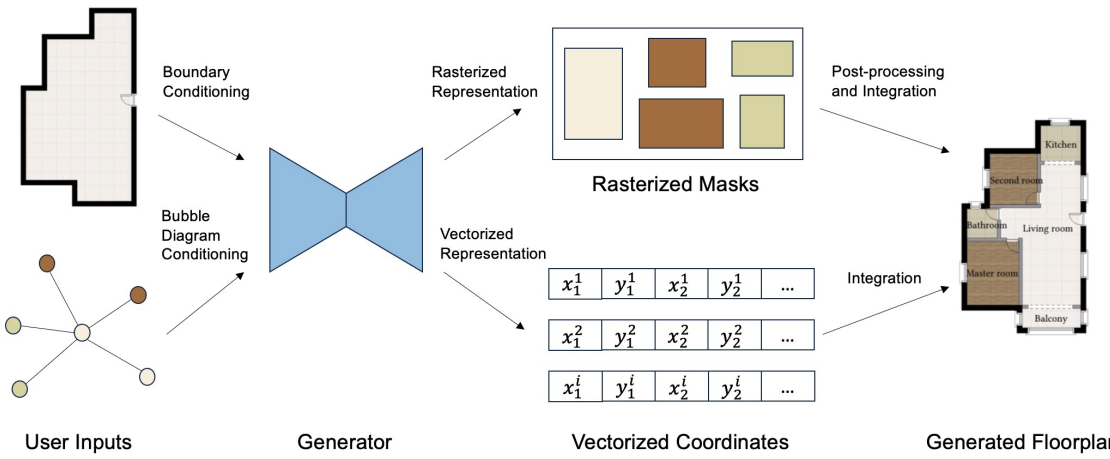


Figure 3. A generic learning-based floorplan generation pipeline with user input. We show two major user input conditions, residential boundary and bubble diagram–commonly used in current studies. Methods using rasterized representation aim to generate a set of room masks then perform post-processing and integrate the outputs into a vectorized floorplan. For vectorized methods, the outcomes can be directly integrated into a floorplan. The icons of the boundary input and the final generation are referred from the diagrams used in RPLAN.

corners. Please note that n can be different for each room. Stacking all the rooms together as a sequence, the entire floorplan L for a residential house can be represented as:

$$L = \{\boldsymbol{R}_j\}_{j=1}^{N}. \tag{11}$$

where N is the number of rooms. N can be different for each floorplan sample.

Please note that typical floorplan layouts satisfy the Manhattan-world constraint, *i.e.*, their boundaries are axis aligned. Some recent study [119] has overcome such limitation and proposed a general representation where non-Manhattan layouts can also be synthesized. The generation process is typically constrained by some specified user input or conditions such as house boundary F, and bubble diagram G (which is a graph indicating the room types and their connectivity relationships). More recently, a floorplan restoration task is proposed [56], which aims to complete and recover the entire floorplan L, from a partial reconstruction outcome L_p. Thus, without loss of generality, the generation process, which takes some of the conditions as input, can be formulated as:

$$L = M(F, G, L_p), \tag{12}$$

where M denotes the floorplan generative model.

3.2. Traditional Methods from Architectural Domain

Traditional floorplan generation methods from the architecture design domain are typically categorized as either bottom-up or top-down approaches [30], each with their own strengths and weaknesses. We introduce some representative works for each type in this subsection.

Bottom-up methods Building designs are usually constrained by specified spatial requirements, such as room dimensions and mutual adjacency. Therefore, bottom-up paradigms tend to become a natural choice for mapping spatial relations or bubble diagrams. Conceptually, a group of predefined building components are aggregated into a larger assembly following specific regulations and constraints. Rosenman *et al.* [114] design a single floorplan generation pipeline that maximizes cross ventilation and minimizes the weighted sum of room distances. [8] describes how to apply topological and geometric objectives to house design within proper boundaries. Merrell *et al.* [84] showcases how to accommodate complex input room sequences into a suitable residential layout through a Bayesian network on multi-floor buildings. Beyond spatial constraints, Yi *et al.* [159] propose designing 3D house layouts based on optimal environmental performance. Guo *et al.* [47] design a multi-agent topology-finding system and an evolutionary optimization process to first generate topology-satisfied layouts and then achieve predefined architectural criteria. GPLAN [121] employs graph-theoretical and optimization techniques to facilitate the generation of dimensioned floorplan layouts.

Top-down methods Top-down methods are inspired by predefined building massing from urban-scale considerations, with strong constraints on the envelope in real-world architectural design. As a result, subdivision, fitting, shape packing, and iterative agent-based methods have been proposed and employed to automate design problems across various architectural scales. Top-down methods take a massing or boundary as input and a set of entities as targets

Table 1. Learning-based floorplan generation methods with their floorplan representation, the types of user input, utilized benchmark datasets and publishing venue.

Methods	Representation	Framework	User Input	Benchmark Datasets	Publishing Press
Wu *et al.*	Rasterized	CNNs	House Boundary	RPLAN	SIGGRAPH Asia 2019
Graph2Plan	Rasterized	CNNs, GNNs	House Boundary, Bubble Diagram	RPLAN	SIGGRAPH 2020
HouseGAN	Rasterized	GANs	Bubble Diagram	RPLAN, LIFULL	ECCV 2020
HouseGAN++	Rasterized	GANs	Bubble Diagram	RPLAN	ICCV 2021
Para *et al.*	Vectorized	Transformers	Unconditioned	RPLAN, LIFULL	ICCV 2021
iPLAN	Rasterized	GANs, CNNs	House Boundary	RPLAN, LIFULL	CVPR 2022
Liu *et al.*	Vectorized	GNNs, Transformers	Bubble Diagram	RPLAN	ECCV 2022
Upadhyay*et al.*	Rasterized	Conv-MPNs, CNNs	House Boundary, Bubble Diagram	RPLAN	ICMEW 2022
FloorplanGAN	Vectorized	GANs	Room types and areas	RPLAN	Automation in Construction 2022
HouseDiffusion	Vectorized	Diffusion Models	Bubble Diagram	RPLAN	CVPR 2023
Tang *et al.*	Rasterized	GANs	Bubble Diagram	LIFULL	CVPR 2023
Zheng *et al.*	Vectorized	GNNs	Bubble Diagram	RPLAN	Automation in Construction 2023
Aalaei *et al.*	Rasterized	Conv-MPNs, GANs	Bubble Diagram	RPLAN	Automation in Construction 2023
Hosseini *et al.*	Rasterized	CNNs, Transformers	Partial Floorplan	RPLAN and a new restoration dataset	BMVC 2023
WallPlan	Vectorized	CNNs	House Boundary	RPLAN	ACM Transactions on Graphics (TOG) 2022
MaskPlan	Vectorized	Masked Autoencoders, Transformers	House Boundary and Partial Attributes	RPLAN	CVPR 2024
Hu *et al.*	Vectorized	Transformers, Diffusion Models	House Boundary, Bubble Diagram	RPLAN	Arxiv 2024

for insertion. The input is then subdivided based on geometric constraints to assign spaces. Compared to bottom-up methods, which require a specified optimization process to ensure boundary constraints, top-down methods naturally conform to this condition by subdividing and transforming directly from the global boundary. Medjdoub *et al.* [82] design a subdivision pipeline for single floorplan generation, considering adjacency and room scale constraints. Banerjee *et al.* [11] propose a computational model for creative design, highlighting practicality, originality, and support for interactive user input. [32] presents a mid-scale floorplan layout generation approach by optimizing with respect to human crowd properties, including mobility, accessibility, and coziness, using agent-based crowd simulation. For a more extensive review of traditional methods proposed in the architectural domain, please refer to [30].

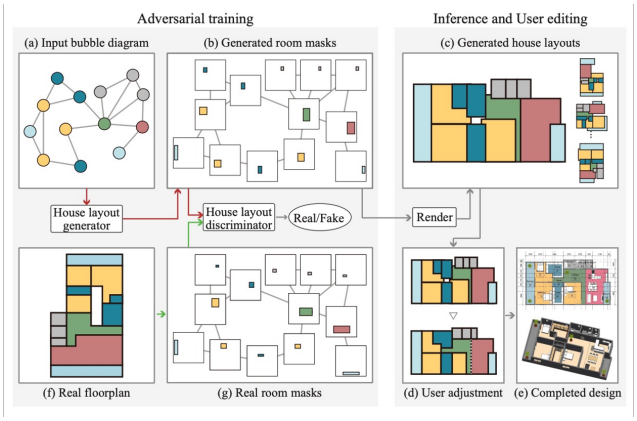


Figure 4. HouseGAN is a representative floorplan generation approach using the rasterized representation. It first parses the input bubble diagram, then generates the room masks separately with a generator and discriminator architecture. The separately generated rooms are subsequently integrated together and post-processed to finalize the floorplan design.

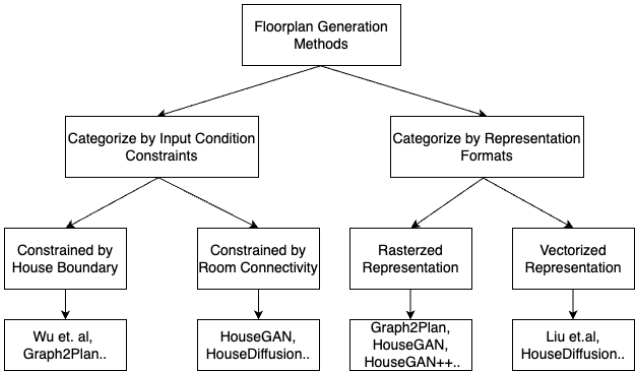


Figure 5. A tree-structured diagram to illustrate the categorization of different approaches and representative methods.

3.3. Data-Driven Learning-based Methods

In this subsection, we review learning-based approaches for computer-aided layout generation proposed in the machine learning and computer-aided design domains. We group the methods based on the utilized representation format and the user input conditions. A generic pipeline of the methods with user-input is demonstrated in Fig. 3, and a tree-structured diagram on how we categorize approaches is shown in Fig. 5.

3.3.1 Methods Categorized by Layout Representation Format

We focus on floorplan layouts and first categorize the deep generative methods by how they represent floorplans during generation. In general, there are two groups of methods, those using a rasterized representation and those using a vectorized representation. Methods using the rasterized representation [57, 91, 92, 134] treat the rooms or walls of the floorplan as rasterized masks, then composite the rooms into a floorplan image. Methods using the vectorized representation [97, 77, 80, 119] represent the room coordinates as quantized vectors, enabling a more straightforward and efficient representation of a floorplan layout.

Methods using Rasterized Representation. The most straightforward way to represent floorplans is to treat them similarly to natural images. Each room is represented as a rasterized mask, and the image pixels are the generation targets. RPLAN [149] and Graph2Plan [57] adopt convolutional neural networks (CNNs) and graph neural networks (GNNs) to generate graph-constrained rasterized floorplans, which are then converted into floorplan vectors via post-processing. Leveraging GAN-based models, HouseGAN [91] (shown in Fig. 4) learns to generate a list of rasterized masks for rooms using a bubble diagram as input. HouseGAN++[92] extends HouseGAN by introducing a GT-conditioning training scheme and a set of test-time optimization strategies. iPLAN[48] proposes a human-in-the-loop system that enables human experts and the deep learning framework to co-evolve a sketchy layout into the final floorplan design, generating segmented room masks in the process. Upadhyay *et al.* [134] take both bubble diagrams and input boundaries as conditions and process them by different feature embedding networks. The bubble diagram embeddings and boundary features are then aggregated via a cascaded alignment network to generate the final floorplan layout. Tang *et al.* [130] propose a graph transformer architecture coupled with GANs to generate floorplans in an adversarial procedure. For these methods, handcrafted post-processing strategies are usually adopted to convert the masks into vectors as the final representation. Aalaei *et al.* [1] utilize a Conv-MPN network to parse the bubble diagram and a GAN architecture to generate rasterized floorplans. The generation proceeds iteratively until the geometrical and topological constraints are approximately satisfied, after which the final output is transformed into a vectorized format.

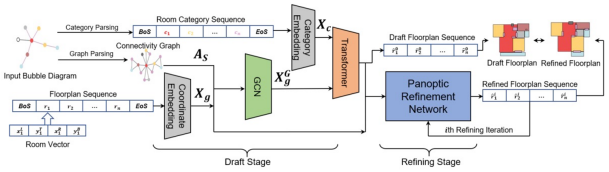


Figure 6. Liu *et al.*represents the floorplan samples as vectors and presents an end-to-end vectorized floorplan generation pipeline. In the first stage, the GCN and Transformer-based network takes the parsed bubble diagram embeddings as input and generates a draft floorplan sequence in an auto-regressive manner. In the second stage, the draft floorplan is passed to another Transformer network to achieve panoptic refinement for the synthesized floorplan.

Methods using Vectorized Representation. Instead of representing rooms as rasterized segments, methods using the vectorized representation [97, 77, 119] encode room coordinates as 1-D vectors. By utilizing 1-D vectors, this representation allows for improved generation efficiency while preserving the topological and geometric integrity of the layout. Para *et al.* [97] use an autoregressive Transformer model primarily for unconditioned floorplan layout generation. Liu *et al.* [77] propose a two-stage vectorized floorplan generation framework conditioned on bubble diagram input. The first stage generates a draft floorplan in an autoregressive manner, and the second stage refines the draft floorplan with panoptic refinement (see Fig. 6). FloorplanGAN [80] takes the room types and their relative occupancy area as input, then extracts room type and geometric embeddings from these conditions. A GAN-based architecture is then used for floorplan generation, consisting of a vector generator, a differentiable renderer, and a CNN-based discriminator. More recently, HouseDiffusion [119] employs diffusion models [54, 124] to represent and generate floorplan vectors. The framework is trained through a diffusing-then-denoising process, and during inference, the final generated floorplan is iteratively denoised from Gaussian noise, conditioned on the input bubble diagram. A significant contribution of their work is breaking the Manhattan-layout assumption commonly adopted in existing works [97, 77], allowing their proposed diffusion model to generate diverse non-Manhattan floorplans. Zheng *et al.* [173] first parse a dual graph structure from the given bubble diagram, then apply a hybrid GNN along with a topology and geometry-driven optimization approach to generate room coordinates. MaskPlan [165] employs a Masked Autoencoder architecture to generate a blend of graph-based and image-based layout attributes, conditioning on partial input formats from users. Compared to rasterized representation, vectorized representation is a more desirable choice in recent studies. Representing rooms with varying sizes using 1D vectors (as described using the format in Sec. 3.1) significantly enhances generation efficiency and improves the ability to preserve correct floorplan topology and geometry. Constraints can be naturally integrated within the vectorized format. For example, the number of rooms in a generated layout can be controlled by specifying the length of the vectors, ensuring compliance with design specifications while maintaining structural consistency. Moreover, vectorized coordinates inherently support axis-aligned properties more effectively than rasterized formats, ensuring precise spatial alignment. Vectorized representation also enables end-to-end optimization explicitly on the graphic vectors, eliminating the need for post-processing steps to transform rasterized masks into vectors.

3.3.2 Methods Categorized by User Input Conditions

Floorplan design is typically a conditional generation problem that takes various conditions from users or designers as constraints. In this subsection, we discuss the common user input conditions and categorize them into two

mainstream formats: residential boundary constraints and bubble-diagram-based constraints. We then introduce the deep generative methods associated with each respective category.

Methods Conditioned on Residential Boundary Constraints. Since residential houses are often not independent and share common building primitives, and because design budget is an important factor, the residential boundary is a common input constraint for floorplan synthesis, specified by either designers or users. In RPLAN [149], a CNN network takes the rasterized house boundary as input and predicts room types, locations, and rasterized walls. Graph2Plan [57] takes the boundary image along with the retrieved room graph from users as input, then encodes this information through Graph Neural Networks (GNN [117]) and CNN networks. iPLAN [48] takes the apartment boundary and room types as conditions, and learns to locate and segment the rooms into rasterized masks that conform to the boundary. WallPlan [126] takes the boundary as input, initializes a network to learn window locations, and then passes the window-conditioned house boundary into parallel branches to learn wall graphs and room semantics. Most recently, Hu *et al.* [58] proposes a transformer-based diffusion model for wall junction generation and wall segment prediction, with a focus on enforcing the geometric attributes from structured graphs.

Methods Conditioned on Bubble Diagram Constraints. Another line of work [91, 92, 77, 119] uses another common input format known as the bubble diagram, which includes room types and graph connectivities, as input constraints. For room types, they are either encoded as one-hot vectors in rasterized methods [91, 92] or quantized as tokens and encoded through learnable embeddings in vectorized representations [77]. Message passing networks [45] or GNNs [117, 66] are employed in [91, 92, 77, 134] to encode the element connectivity between different rooms. Tang *et al.* [130] designs a graph-constrained transformer framework to learn graph-based relations from the input bubble diagrams. HouseDiffusion [119] encodes the graph-constrained room relationship via performing graph-aware attention on the embedded room coordinates using the given bubble diagram. Zheng *et al.* [173] also leverages the bubble diagram input to parse the dual graph representation to guide the optimization stage. Aalaei *et al.* [1] uses ConvMPN [164] to exploit topology information from the input bubble diagram and yields floorplans through a GAN architecture.

3.4. Benchmark datasets

RPLAN [149] RPLAN is the first large-scale densely annotated floorplan dataset with over $80K$ real residential buildings, which is widely adopted in ML-based floorplan generation works. It provides comprehensive vectorized graphics of room types, boundaries, dimensions, and connectivity information.

LIFULL [74] LIFULL HOME is a large-scale database containing millions of real floorplan samples with corresponding room type labels. The bubble diagram can be retrieved if needed through a floorplan vectorization [76] process.

Structured3D [172] Structured3D is a photo-realistic dataset with diverse 3D structure annotations, satisfying a range of 3D modeling problems. It provides scene information from both holistic views such as floorplans and 3D meshes, and primitives such as junctions, lines and planes. Floorplans are annotated by room types and dimensions over 3,500 scenes.

Zillow [24] The Zillow Indoor Dataset (ZInD) is a large-scale real indoor dataset with over $70k$ panoramas and unfurnished houses. Compared to other benchmarks, it provides more diverse data following real-world distribution, containing both Manhattan and non-Manhattan layouts. The floorplans are annotated with room semantics and dimensions over 2,500 scenes.

CubiCasa5k [62] Cubicasa5K is a floorplan dataset containing 5,000 sampled residential floorplans. A unique feature of this dataset is that it provides furniture object-level vectorized layouts, enabling more fine-grained intelligent design for both room and furniture layouts.

ProcTHOR-10k [26] PROCTHOR is a framework for procedural generation of Embodied AI environments. PROCTHOR produces a large and diverse set of floorplans, followed by a large asset library of 108 object types and 1633 fully interactable instances that are used to automatically populate each floorplan.

DStruct2Design [81] In the DStruct2Design paper, the authors merge existing datasets, including RPLAN and ProcTHOR, and apply post-processing to format them for compatibility with language-based generative models. This enables language-guided floorplan generation that can specify both geometric structures and the absolute dimensions of rooms.

3.5. Evaluation metrics and experimental comparison

In this section, we introduce common evaluation metrics for conditioned floorplan generation. Tab. 2 illustrates the

Table 2. Quantitative comparison on SOTA bubble-diagram-constrained floorplan generative methods. The numbers (5, 6, 7, 8) or 'mixed' refer to the room number of the floorplans used in the testing split.

Methods	Diversity ↓					Compatibility ↓					Inference Memory (GB) ↓	Latency (FPS) ↑
	Mixed	5	6	7	8	Mixed	5	6	7	8	8	8
HouseGAN++	17.9/0.2	19.9/0.3	15.4/0.1	14.0/0.2	18.9/0.5	2.7/0.1	1.7/0.0	2.1/0.0	3.1/0.1	3.6/0.1	1.6	2.83
PanopticRefine	16.3/0.4	18.9/0.5	16.9/0.5	14.5/0.3	16.5/0.5	2.5/0.1	1.3/0.0	1.9/0.0	3.4/0.0	5.0/0.0	1.27	4.22
HouseDiffusion	10.5/0.2	11.2/0.2	10.3/0.2	10.4/0.2	9.5/0.1	2.0/0.0	1.5/0.0	1.2/0.0	1.7/0.0	2.5/0.0	1.9	1.34

evaluation metrics used in representative vectorized methods, including *diversity and compatibility*. Additionally, *realism* measures the generation quality by user engagement and ranking on different methods, which is another important metric.

Realism Realism is one of the most important metrics in generation tasks, as it evaluates user perception of the generated floorplans. A common practice is to invite a group of participants—either professional architects or amateurs—to provide scores or rankings for the outcomes of various generation methods, given the same set of sampled constraints. The average score or ranking is then calculated to assess the quality of the generation.

Diversity and Quality The Fréchet Inception Distance (FID)[53] is a widely used metric for assessing the quality of generated images by comparing their distribution to that of real images. It calculates the Fréchet distance between the feature representations of generated and real images, which are extracted using a pre-trained Inception network, measuring differences in both the mean and covariance. Lower FID scores indicate that the generated images are closer to the real images in terms of visual fidelity. In the context of floorplan generation, the entire test set of generated floorplans and their corresponding real counterparts are considered for FID computation. As a result, the FID score can also indirectly account for diversity, as low diversity in generated images may lead to a poor match with the real image distribution. For direct measurement of diversity in generated images, metrics such as Precision and Recall for Distributions (PRD)[116, 67] and Intra-FID (i.e., FID between generated images and real images within each class) [88] can be used.

Compatibility Compatibility with input graphs is assessed using Graph Edit Distance (GED) [2], a metric that evaluates whether the generated floorplan maintains the correct connectivity relationships specified by the input graph. GED measures the total number of connectivity errors in the placement of interior doors within rooms, or front doors with respect to rooms and outdoor areas, in a generated floorplan.

Boundary Intersection-over-Union (IOU) For methods that use residential boundaries as constraints, the Intersection over Union (IoU) score between the generated room boundary and the ground truth is used as a quantitative metric to measure the overlap. A higher IoU score indicates better alignment of the generated house boundary with the input boundary.

3.6. Discussions

We provide an overview of the existing methods, floorplan representations, frameworks used, supported user inputs, benchmark datasets, and the publication venues for learning-based floorplan generation methods in Tab. 1.

For the floorplan generation methods conditioned on bubble diagrams, we list the quantitative outcomes for current SOTA learning-based methods in Tab. 2. One can see that, GAN-based methods using the rasterized representation fall behind in terms of both diversity and compatibility compared with the vectorized approaches. The multi-stage GNN and transformer framework [77] strikes a better balance between generative fidelity and computational cost. HouseDiffusion [119] achieves SOTA generation quality by leveraging iterative optimization via a well-designed denoising process. To further demonstrate the generation fidelity of HouseDiffusion, we run their model using their released code[1], and showcase some representative generated sampled in Fig. 7. The model can represent the rooms as complex polygons (illustrated by the living rooms) and most of the rooms preserve spatial relationships consistent with the specification in the bubble diagram. For the samples of the top 2 rows, the model has demonstrated the ability to handle the complex input bubble diagrams and succeed in generating visually appealing floorplans. However, the samples in the last 2 rows display noticeable issues, including artifacts in spatial arrangement. The interior doors are sometimes incorrectly arranged inside the rooms (the left sample in the 3rd and 4th rows). The generated rooms occasionally exhibit unusual wall boundaries that do not appear in real floorplans (the right sample in the 3rd row). Moreover, the overall spatial arrangement of the rooms is not always optimal, with occasional unnecessary overlap between rooms (the right sample in the 4th row).

As demonstrated above, there exist non-negligible issues or limitations for current SOTA deep generative mod-

[1]HouseDiffusion - https://github.com/aminshabani/house_diffusion

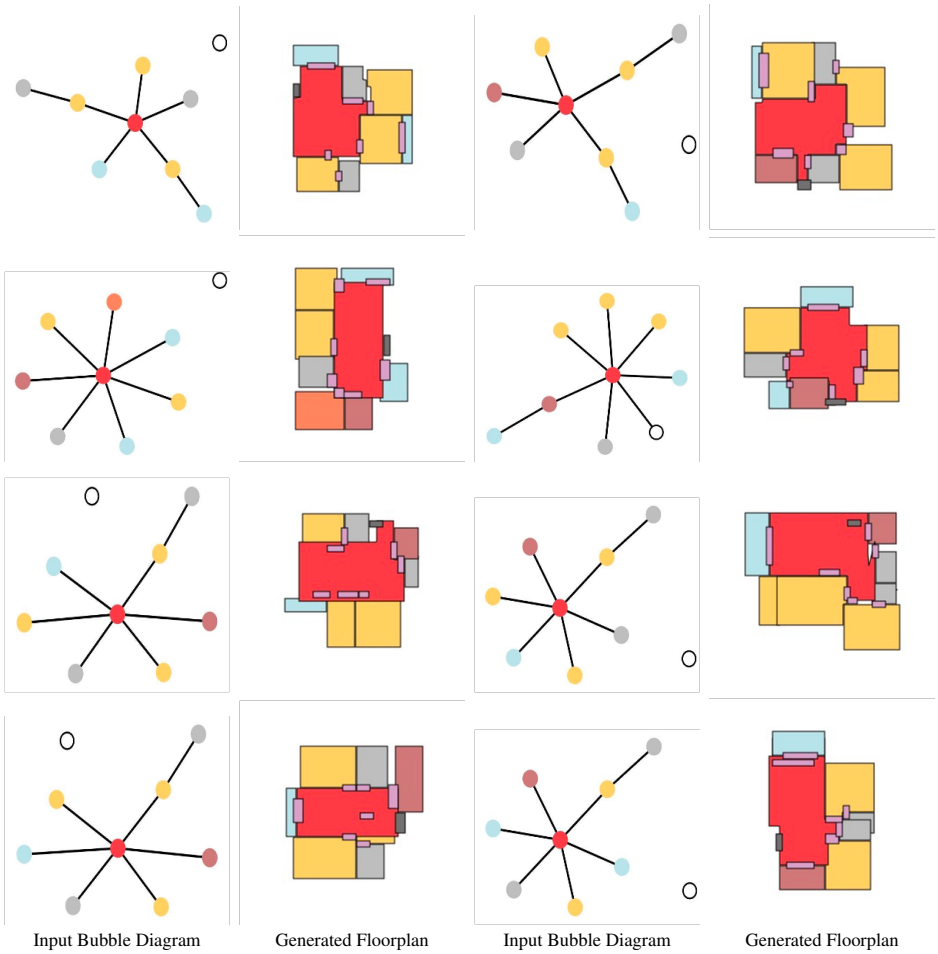


Figure 7. The generated floorplans from a state-of-the-art generative model HouseDiffusion [**?**]. We got the results from the most challenging setting on 8-room generation during inference, while the model is trained on the remaining data splits with other room numbers. The top two rows show satisfactory outcomes whereas the last two rows exist noticeable issues.

els. First, the incorrect arrangement of elements and the presence of generated artifacts should be addressed by enhancing the model's generative capacity. Additionally, the model should better capture correct spatial and geometric inter-room relationships during generation. Second, the supported user input formats are limited. In addition to boundaries and bubble diagrams, common options for user interaction and customization include text, room sketches, and audio instructions. A generation framework capable of incorporating diverse input modalities would offer greater flexibility and be more suitable for industry-level products and applications.

4. Scene Layout Generation

We take a step further into the interior layout generation problem after establishing the floorplan wall boundaries. The placement of diverse furniture categories based on the synthesized floorplans is another crucial aspect of indoor layout design. In this section, we introduce and formulate the task of indoor scene synthesis. We further categorize data-driven methods into graph-based approaches and sequential generation techniques. Finally, we present the publicly available benchmark datasets used in this field, common evaluation metrics for this task, and discussion and quantitative comparison of scene layout generation methods. A generic methodological framework is shown in Fig. 8.

4.1. Task formulation

Conditioned by some given user input constraints, *i.e.*, the room boundary F or scene graph G, the room category (e.g. bedroom, living room) C, and sometimes some input text instructions T or given incomplete object layouts L_p, scene synthesis aims to generate or complete the entire scene layout sequence $L = \{O_i\}_{i=1}^{N}$, ensuring correct furniture arrangement and functionality for daily activities. More specifically, a furniture object O_i is parameterized as a 3D bounding box with parameterized location t_i, orienta-

Table 3. Indoor scene synthesis methods categorized by the type of user input, designed framework and utilized benchmark datasets and publishing press.

Methods	Framework	User Input	Benchmark Datasets	Publishing Press
Wang *et al.*	CNNs	Input Partial Scene	SUNCG	TOG 2018
PlanIT	Graph-based CNNs	Room Boundary	SUNCG	TOG 2019
GRAINS	Graph-based VAE	Unconditioned	SUNCG	TOG 2019
SceneGraphNet	Message Passing Networks, GRU	Unconditioned	SUNCG	ICCV 2019
Ritchie *et al.*	CNNs	Room Boundary	SUNCG	CVPR 2019
3D-SLN	Graph-based VAE	Scene Graph	SUNCG	CVPR 2020
SG-VAE	Grammar VAE	Unconditioned	SUNCG	ECCV 2020
Zhang *et al.*	Feed-forward Network	Unconditioned	SUNCG	TOG 2020
SceneHGN	Hierarchical graph-based VAE	Room Boundary	3D-FRONT	TPAMI 2021
ATISS	Transformers	Room Boundary	3D-FRONT	NeurIPS 2021
Depth-GAN	GANs	Unconditioned	Structured3D, Matterport3D	ICCV 2021
Yang *et al.*	Bayesian Networks	Scene Graphs	3D-FRONT, SUNCG	ICCV 2021
Graph-to-3D	Graph-based VAE	Scene Graphs	3DSSG	ICCV 2021
Sceneformer	Transformers	Room Boundary	SUNCG	3DV 2021
LayoutEnhancer	Transformers	Layout Subsets	3D-FRONT	SIGGRAPH Asia 2022
DiffuScene	Diffusion Models	Room Boundary	3D-FRONT	Arxiv 2023
COFS	Transformers	Layout Subsets	3D-FRONT	SIGGRAPH 2023
LEGO-Net	Transformers	Messy State Layout	3D FRONT	CVPR 2023
CC3D	Neural Radiance Fields, StyleGAN	2D Floorplan	3D FRONT, KITTI 360	ICCV 2023
LayoutGPT	LLMs	Texts	3D-FRONT	NeurIPS 2023
CommonScenes	GCNs, Diffusion Models	Scene Graphs, Texts	SG-FRONT (3D-FRONT)	NeurIPS 2023
PhyScene	Diffusion Models	Room boundary	3D-FRONT, 3D-FUTURE	CVPR 2024
LTS3D	Diffusion Models	Unconditioned	3D-FRONT	Arxiv 2024
Forest2Seq	Transformers	Room boundary	3D-FRONT	ECCV 2024

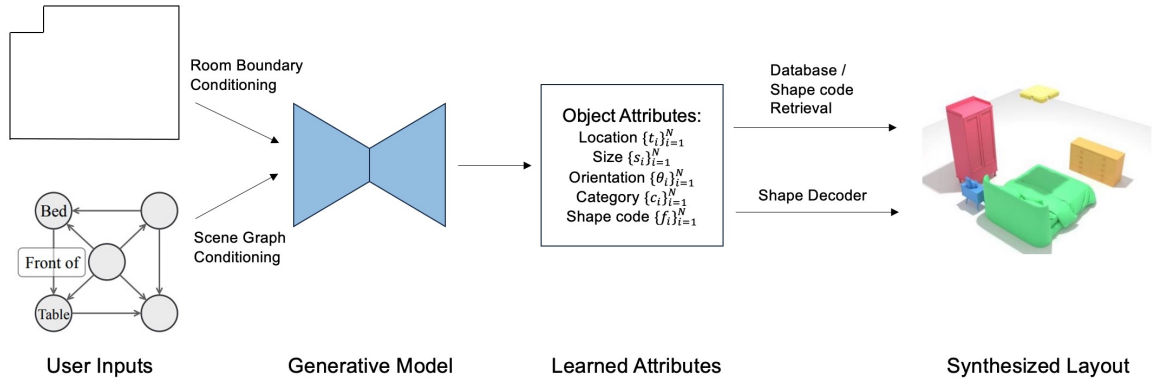


Figure 8. A generic scene synthesis framework with different types of user input. The generative model takes a room boundary or a scene graph as input condition, then generates a group of parameterized furniture object attributes. For retrieval based methods, the learned shape attributes are used for a shape retrieval stage to get object meshes. For end-to-end approaches, the shape meshes are parsed from a learned shape decoder. The icons of furniture object layouts are referred from DiffuScene.

tion r_i, dimensional size s_i, categorical type c_i and a shape latent code f_i. Denote the designed generative model as M, the generation process can be formulated as:

$$L = \{O_i\}_{i=1}^N = \{t_i, r_i, s_i, c_i, f_i\}_{i=1}^N = M(F, G, C, T, L_p). \quad (13)$$

Then, a shape retrieval process is typically employed to obtain the textured object mesh from the database that best matches the generated attributes, integrating them into the final scene layout.

4.2. Methods

4.2.1 Optimization-based Methods

Traditional scene modeling and synthesis works usually address this problem as an optimization process. Merrell *et al.* [85] identifies a set of design guidelines for house layout design and represents the scene distribution as a density function, then employs the Markov chain Monte Carlo sampler (MCMC) to get optimized outcome while respecting user input as layout constraints. Yeh *et al.* [158] formulates

the open-world layout synthesize problem as a open universe probability distribution sampling process constrained by factor graphs and proposes a reversible jump MCMC method to represent and and solve the optimization problem. Yu *et al.* [162] first extracts spatial and hierarchical relationship of the objects and obtains an plausible initialization. Then the furniture arrangements are refined through a simulated annealing optimization stage. Moreover, Zhang *et al.* [168] proposes a system that iteratively adds patterns with respect to constraints formulated by commercial design rules by employing optimization procedures. A group of works first parse structural scene graphs from natural languages [34, 17, 16] or sketches [151] as constraints, then synthesize the scene layouts by querying and matching the database with the learned spatial knowledge. Zhang *et al.* [167] optimizes the layout with learned room geometry and object distribution priors, partitioning the input objects into disjoint groups, followed by layout optimization using position-based dynamics (PBD) based on the Hausdorff metric. Some works [39, 35, 61] manage to model the distribution of human activities and incorporate such contexts into the scene layout generation process. Although having shown desirable fidelity under certain scenarios, these methods are driven by the prior knowledge of the scene distribution and regularized by hand-crafted guidelines. As a result, generation diversity is generally inferior to that of recent deep generative model-based methods, which will be discussed in the next subsection.

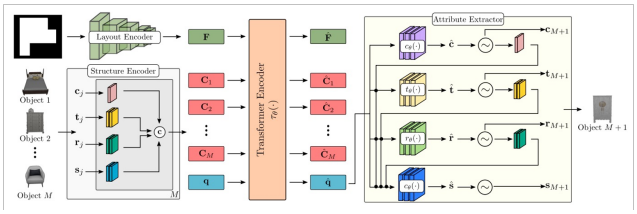


Figure 9. ATISS serves as a representative indoor scene synthesis pipeline. The top-down view residential boundary is passed into a layout encoder to extract boundary feature F. Different attributes of the furniture objects c, t, r and s are encoded by a structure encoder to learn respective embeddings regardless of the object sequence order. Then a transformer encoder processes these learned embeddings and output the refined embeddings. Then separate attribute extractors are designed for decoding the distributions of these attributes. During inference, the object attributes are sampled from the learned distributions.

4.2.2 Learning-based Methods

We categorize the methods of data-driven scene layout synthesis into feed-forward approaches and autoregressive-model-based approaches. Feed-forward methods are usually designed as graph-based networks, treating the furniture objects together with the floor boundary as a scene graph or other structural representation to exploit their contextual relationship, or seek to learn a powerful scene representation with a simple yet effective feed-forward generative network. Autoregressive-based methods regard the target objects as an ordered or unordered sequence, and generate the objects in an iterative manner, taking the previous generation as the context of the next step.

Graph-based methods The objects in a scene are not independent but interleave with each other, indicating rich spatial relationships and context information. To this end, graph-based methods [140, 79, 70, 102, 169, 175, 153] are proposed to solve the scene synthesis problems, which aim to first represent the scene layout as an abstracted graph, then exploit scene contexts and inter-object relationships via the built graph. The generated layout are naturally conformed to the relationships suggested in the scene graph. PlanIT [140] proposes to first extract a scene graph trained by a deep generative model, then employs another image-based reasoning model to iteratively insert objects into the scene. GRAINS [70] formulates the scene as a hierarchical graph, then employs a recursive VAE on learning object group encoding and generation decoding. 3D-SLN *et al.* [79] proposes a variational generative model to synthesize layout from scene graphs based on GCNs and VAE. SG-VAE [102] represents the scene as a tree structure and leverages grammar-based auto-encoder to learn the co-occurrence and appearance attributes of the scene. Depth-GAN [154] introduces a GAN-based framework to learn a 3D scene representation for scene synthesis through projection supervision from a set of 2.5D semantic-segmented depth images. Zhang *et al.* [169] proposes a hybrid representaion which capture information from both 3D object and image-based representation with a feed-forward generative model. SceneGraphNet [175] designs a deep and dense message passing network on extracting structural and spatial relationships, to obtain a probability distribution among object categories given a query location. Yang *et al.* [153] proposes a Bayesian optimization framework which first generates over-complete sets of attributes then employs a pruning stage to filter out the infeasible predictions based on consistency constraints of the attributes. SceneHGN [41] employs a recursive VAE network to learn the scene layout hierarchy from room to object and partial object level. Most recently, DiffuScene [131] first builds a complete 3D scene graph to encode the spatial context then represents and generates the object attributes with a diffusion network [54]. Different from prior work which generates object attributes then apply a database retrieval process to obtain textured object mesh which heavily restricts the generation capacity, Graph-to-3D [27] proposes a GCN-based VAE architecture to learn to directly generate object 3D meshes given the scene graph as input with

an end-to-end framework. To better handle the biased generation results due to the object category imbalance in the training set, FairScene [150] exploits unbiased object interactions with a causal reasoning framework which achieves fair scene synthesis by calibrating the long-tailed category distribution.

Sequential generation methods Graph-based methods need to build scene graphs based on prior knowledge as prerequisites. Contrastively, auto-regressive models do not require such prior knowledge, and directly treat the objects as an ordered or unordered sequence, generating the target furniture in an iterative manner. Wang *et al.* [141] proposes to first parse the top-down view input partial scene and get a scene representation. Then separate CNNs are utilized to predict object attributes and determine whether to continue to insert objects, to complete the scene layout generation. Ritchie *et al.* [110] encodes the top-down view of the scene with a CNN and predicts the attributes of the objects in a sequential manner. Sceneformer [144] introduces a set of transformers [136] to learn different targets separately and autoregressively insert objects to the scene in a pre-defined order. ATISS [100] (shown in Fig. 9) takes the house boundary layout as input constraint, and employs a transformer architecture as attribute encoder as well as separate attribute decoders to learn order-invariant sequential generation over furniture objects. In addition to auto-regressive generation, a set of work [69, 98] regard the objects as a sequence but synthesize all of the objects at a time via feed-forward networks. COFS [98] devises an order-invariant autoregressive transformer to perform cross-attention over the entire conditioning input, such that the generation can be conditioned with fine-grained input. LayoutEnhancer [69] accounts for ergonomic qualities as expert knowledge into the generation process to tackle the challenges induced by imperfect training data. RoomDesigner [171] proposes to leverage anchor latents to encode the piece-wise geometric representation of furniture, then conduct scene generation sequentially with a transformer network. CommonScenes [163] (Fig. 11) constructs a new dataset dubbed SG-FRONT on top of 3D-FRONT providing conditional scene graphs, and proposes to encapsure inter-object relationship and local shape cues with a diffusion model, and employs a diffusion model to generate the 3D scene layout with diverse shapes in an end-to-end manner without any database retrieval stage or category-level decoders. Forest2Seq [128] derives ordering information from the layout sets and designs a transformer to generate realistic 3D scenes in an autoregressive manner. GLTScene [71] incorporates the interior design principles with learning techniques and adopts a global-to-local strategy for this task, by designing two transformer networks to learn the global and local priors, respectively.

Other methods Except for the above two mainstream paradigms, some methods build their framework upon some recent innovations such as neural radiance field [86] to achieve 3D-aware generation, or leverage the generation ability of LLMs [95] as a prior. CC3D [9] takes the 2D furnitured floorplan as the input constraint and aims to generate 3D scene layout. It proposes to leverage a neural radiance field to lift the 2D features into 3D, and use the StyleGAN [63] architecture as the generator and discriminator to synthesize the layouts. LEGO-Net [146], inspired by the workflow of diffusion models, aims to refine a messy scene layout into a cleaner, more organized one. It devises a denoising transformer architecture to gradually transform the original messy state to a clean state. Most recently, leveraging the rise of large language models (LLMs), LayoutGPT [33] proposes a LLM-based training-free layout generation framework. It first prepares well-constructed text prompts such as Cascading Style Sheets (CSS) formats and specified task instructions, then generates desirable image or scene layouts through LLMs like GPTs [95], and has shown comparable or more superior generative fidelity compared with the existing SOTAs trained on a particular dataset. Other than these, Physcene [155] proposes a diffusion-based framework to generate indoor scene layouts imposing constraints such as object collision, room layout, and object reachability, integrating embodied AI environment into the scene synthesis task. LT3SD [83] introduces a latent tree representation coupling the diffusion models to generate complex 3D scene geometry. FuncScene [87] proposes a VAE network which leverages function groups as an intermediate representation to connect the local scenes and the global structure, aiming to achieve a coarse-to-fine indoor scene synthesis.

Figure 10. 3D-FRONT provides a large-scale interior house layout database with versatile house and room layouts, as well as intricate furniture CAD models.

4.3. Benchmark Datasets

3D-FRONT 3D-FRONT [37] (as shown in Fig. 10) is a large-scale indoor scene and object layout dataset contain-

Table 4. Quantitative comparison on retrieval-based deep scene layout generative methods. The numerical results are referred from the latest method DiffuScene.

Methods	Bedroom				Dining Room				Living Room			
	FID (↓)	KID (↓)	SCA (↑)	CKL (↓)	FID (↓)	KID (↓)	SCA (↑)	CKL (↓)	FID (↓)	KID (↓)	SCA (↑)	CKL (↓)
DepthGAN	40.15	18.54	96.04	5.04	81.13	50.63	98.59	9.72	88.10	63.81	97.85	7.95
Sync2Gen	31.07	11.21	82.97	2.24	46.05	8.74	88.02	4.96	48.45	12.31	84.57	7.52
Sync2Gen*	33.59	13.78	87.11	2.67	48.79	12.01	91.43	5.03	47.14	11.42	86.71	1.60
ATISS	18.60	1.72	61.71	0.78	38.66	5.62	71.34	0.64	40.83	5.18	72.66	0.69
DiffuScene	17.21	0.70	52.15	0.35	32.60	0.72	55.50	0.22	36.18	0.88	57.81	0.21

Table 5. Quantitative comparison on scene synthesis methods which take scene graphs as input. The numerical results are referred from a latest method CommonScenes.

Methods	Shape Representation	Bedroom		Living Room		Dining Room		All	
		FID (↓)	KID (↓)	FID (↓)	KID (↓)	FID (↓)	KID (↓)	FID (↓)	KID (↓)
3D-SLN	Retrieval	57.90	3.85	77.82	3.65	69.13	6.23	44.77	3.32
Progressive	Retrieval	58.01	7.36	79.84	4.24	71.35	6.21	46.36	4.57
Graph-to-Box	Retrieval	54.61	2.93	78.53	3.32	67.80	6.30	43.51	3.07
Graph-to-3D	DeepSDF	63.72	17.02	82.96	11.07	72.51	12.74	50.29	7.96
Layout+txt2shape	SDFusion	68.08	18.64	85.38	10.04	64.02	5.08	50.58	8.33
CommonScenes	rel2shape	57.68	6.59	80.99	6.39	65.71	5.47	45.70	3.84

ing diverse room categories and high-quality textured 3D models with different styles. The layout designs are sourced from professional creations while the furniture texture and styles are managed by a recommend system to ensure consistent and expert designs. This dataset is widely adopted as a benchmark for indoor scene or texture synthesis applications.

3D-FUTURE 3D-FUTURE [38] complements 3D-FRONT by providing high-quality 3D shapes, informative textures and attributes. Additionally, 3D-FRONT fill the blank of the large-scale and accurate 2D-3D alignment between realistic image and 3D objects by rendering over 20K photo-realistic images across diverse scenes. This enables its promising applications on serving as benchmarks onto tasks such as high-quality 3D shape generation, reconstruction and retrieval.

3DSSG 3DSSG [139] is a large scale 3D dataset built on top of 3RScan [138] with semantic scene graph annotations, containing spatial relations between objects, classes and other attributes. In total it contains 363k graph-image pairs, which are rendered from 3D scene graphs. This dataset enables large-scale training on scene layout synthesis conditioned on scene graphs.

SG-FRONT SG-FRONT [163] is a dataset created in CommonScenes [163] and is developed from 3D-FRONT [37] dataset. It aims to encourage future exploration on conditional scene synthesis on scene graphs. To this end, it offers a set of well-annotated scene graph labels grouped into three categories: spatial/proximity, support, and style. SG-FRONT covers 15 relationship types densely annotating scenes. More details can be found in the original paper.

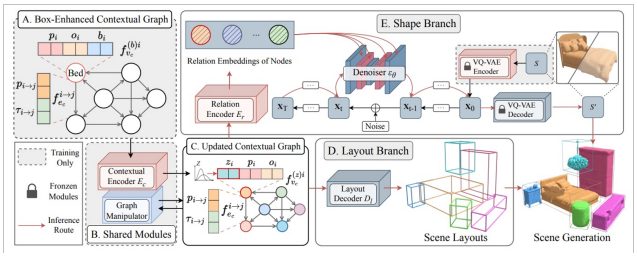


Figure 11. CommonScenes is one of the SOTA indoor scene layout generation pipelines. It leverages the CLIP-enhanced GCN feature encoder to process the scene graph constraints, a GCN decoder to synthesize layouts, and a diffusion model to decode the object mesh. In this method, the furniture meshes are generated with an end-to-end framework without any database or shape code retrieval stage.

4.4. Evaluation Metrics

As shown in Tab. 4 and Tab. 5, the commonly-used evaluation metrics in mainstream approaches on leading benchmarks include FID, KID, SCA and CKL, assessing fidelity, diversity, and adherence to graph constraints.

Fidelity and diversity. Generation fidelity and diversity are common standards for image synthesis tasks. The generated 3D scene layout is first projected as images from multiple bird-eye views, then the generation quality is measured by the Fréchet Inception Distance (FID) [53] and Kernel Inception Distance [14] between the synthesized lay-

out and the real ones. Besides, the KL divergence [3] between the object category distributions of the generated and groundtruth furniture serves as another quantitative metric. Moreover, scene classification accuracy (SCA) is used to measure whether the synthesized layouts is good enough to be indistinguishable from real scenes.

Graph constraint metrics. For the methods which take scene graphs as input constraints, object pair-wise accuracy on the assigned constraints are evaluated to achieve consistency with the input constraints. The constraints contain three categories: spatial/proximity (*e.g.*, left/right, bigger/smaller), support (*e.g.*, close by, above), and semantic or style level relationships (*e.g.*, same material as, same category as), etc. More details can be referred to from [163].

4.5. Discussions

We list the learning-based scene layout synthesis methods, their frameworks, enabled user inputs, benchmark datasets, and publishing venues in Tab. 3. It can be observed that autoregressive transformers, which represent the scene as a sequence, and graph-based generators, which represent the scene as a graph, are two mainstream frameworks for learning object attributes. More recently, researchers have also explored the potential of large language models (LLMs) and diffusion models to facilitate scene layout generation. A key limitation of current methods is scalability. Most focus solely on single-room layout generation, rather than house-level generation involving multiple rooms. The relationships between rooms may provide additional spatial and semantic cues for object arrangement. Another limitation is that most methods involve a separate retrieval stage to obtain object meshes from the database after learning object attributes. While this ensures plausible object shapes, it significantly limits generation diversity. Although some works [27, 163] propose generating object layouts and meshes in an end-to-end manner, the resulting meshes still exhibit noticeable artifacts or unavoidable collisions between objects. Optimizing layout attributes and decoding high-fidelity shapes more effectively in a fully end-to-end pipeline remains an open area for further exploration.

5. Other Building Layout Design and Synthesis

Sections 3 and 4 primarily focus on the generation and synthesis processes of apartment-level (or flat-level) layouts. In contrast, this section provides an overview of layout generation in a broader architectural context, extending from individual buildings to site-level considerations such as blocks or parcels in communities. Due to the distinct objectives associated with these tasks, there is a lack of shared or unified benchmark datasets and evaluation protocols. Consequently, this section does not delve into the specifics of datasets and evaluations. Interested readers are encouraged to consult relevant works for more detailed information.

5.1. Building-level layout generation

We begin by expanding the task scope from individual apartments to entire buildings. This expansion encompasses the comprehensive planning of multiple apartments, staircases, elevators, and other public amenities across entire floors. Simultaneously, we must address the holistic arrangement of elements within multi-story buildings. This extended scope introduces unique challenges, such as generating roofs [107, 103], shear walls [73, 78, 170], facades [42], and volumetric representation [18, 4].

To address this complex domain, we synthesize relevant literature, categorizing it into non-learning-based and learning-based methodologies. This approach allows us to gather information and methods from diverse sources, leading to a deeper understanding of the complexities of designing entire building structures.

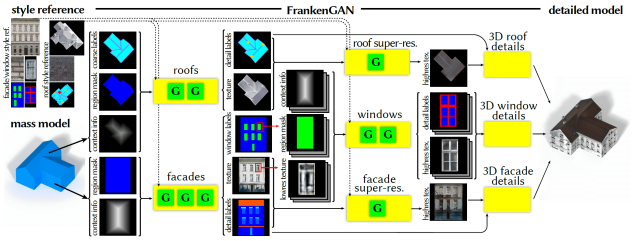


Figure 12. FrankenGAN is a framework for generating building layouts using GANs. It involves three steps: generating texture and label maps for facades and roofs, enhancing texture resolution with dedicated chains, and creating 3D details for roofs, windows, and facades based on the generated maps.

5.1.1 Non-learning-based methods

Müller *et al.* [89] introduced CGA shape grammar to create visually appealing and detailed building shells with consistent mass models in computer-generated architecture. Following that, Schwarz and Müller [118] introduced CGA++ to procedurally model architecture, providing an integrated and versatile solution that builds upon the CGA shape. Bao *et al.* [12] first located and represented good layouts as portal graph, then used this portal graph and local shape grammars to explore more building layouts. Rodrigues *et al.* [111] introduced a way to handle different levels using a mix of evolutionary techniques, where stairs and elevators adapt and interact with other spaces in the search process. Bahrehmand *et al.* [10] shared a tool that helps designers create personalized layouts based on guidelines and user preferences, with key features including addressing subjective design aspects and using a genetic algorithm

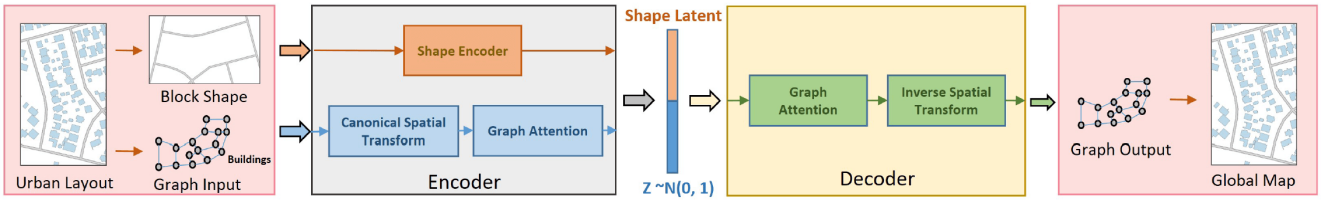


Figure 13. GlobalMapper is a representative method for site-level layout generation. It uses a VAE framework with a CNN encoder to transform arbitrary block shapes into a binary mask in latent space, allowing for the conditional generation of large urban layouts.

for better quality layouts. Wu *et al.* [148] proposed a structured approach to create indoor spaces using a mixed integer quadratic programming (MIQP) method, applicable to various building sizes like homes, offices, malls, and supermarkets. Gan [40] introduced a graph data model using Building Information Modeling (BIM) to represent key features in modular buildings. Isaac *et al.* [59] proposed a graph-based approach for offsite preassembly with clustering algorithm on BIM tool data, and providing a computer program for automated application in large and complex projects. Sharafi *et al.* [120] proposed a Unified Matrix Method for efficiently finding the optimal spatial design of multi-story modular buildings in early design stages. Fan *et al.* [30] design a two-stage genetic algorithm (GA) to automatically generate modular high-rise residential buildings (MHRBs). Gerber *et al.* [42] investigated using multi-agent systems (MAS) in architectural design to enhance and partially automate the design process.

5.1.2 Learning-based methods

Advanced techniques in deep generative models have made scalable, building-level layout generation possible. Merrell *et al.* [84] presented a Bayesian network method for automatically creating detailed multi-story building layouts based on high-level requirements. Ghannad and Lee [43] suggested a method to automatically create and set up a modular building design using GAN and building information modeling (BIM) technologies. They later [44] introduced a framework using Coupled GAN (CoGAN) for automating the generation of modular housing designs. Wang *et al.* [142] made a dataset of building layouts and graphs, propose a mix of methods for diverse graph matching, and create a neural network with U-Net and spatial attention for effective building region segmentation. FrankenGAN [64] addressed the challenge of realistically detailing mass models by employing latent style vectors through synchronized GANs guided by exemplar style images, with the pipeline shown in Fig. 12. Du *et al.* [28] proposed a 3D building fabrication method using a multi-properties GAN chain for complex architectural structures. BuildingGAN [18] utilized a volumetric representation for buildings and proposes a graph-conditioned GAN framework to generate building layouts. Alam *et al.* [4] treated the volumetric building design as a sequential problem and creates a sequential volumetric design dataset. Latent representation learning is utilized to tackle the auto-completion and reconstruction of design sequences. Vitruvio [132] introduced a conditional generation framework based on sketch and context for single-view building mesh reconstruction. Fu *et al.* [36] proposed a GAN-based approach (FrameGAN) for automating the layout design of components in steel frame-brace structures. Liao *et al.* [73] suggested using a GAN-based method to quickly and intelligently design shear walls by learning from existing design documents. Zhao *et al.* [170] introduced an approach to design layouts for reinforced concrete structures using deep neural networks, incorporating building space and element attributes as input, and generating new designs based on learned principles. Lu *et al.* [78] introduced StructGAN-PHY, a physics-boosted GAN for intelligent generative design in structural engineering, leveraging a surrogate model for physical performance assessment. Roof-GAN [103] produced a structured roof model as a graph, assessing the primitive geometry, inter-primitive, primitive geometry on building roofs, and generates structural roof primitives based on GANs. Ren *et al.* [107] proposed a roof graph representation which encodes roof topology and employs auto-regressive models to synthesis roof structures.

5.2. Site-level layout generation

We then expand the scope of layout generation tasks from single-building configurations to include multiple buildings. Following [142], we refer to this specific challenge as site-level layout generation, involving the placement of buildings within a designated block or parcel of land. While existing works predominantly target urban planning, limited research specifically addresses rural building layouts [29]. We organize the pertinent literature into two categories: non-learning-based methods and learning-based methods. In terms of benchmark, Reco [21] has created and maintained the first large-scale open-source dataset for residential community layout planning, covering 37,646 residential communities and 598,728 buildings across 60 cities.

5.2.1 Non-learning-based methods

Parish and Müller *et al.* [99] proposed a city modeling system using L-systems, which processes input maps to generate roads, divide land into lots, and create building geometries. Aliaga *et al.* [5] presented a system that uses real-world urban data and synthesis algorithms to interactively generate realistic urban layouts with both structural and visual components. Vanegas *et al.* [135] introduced a technique for user-guided creation of city parcels in urban modeling, involving the subdivision of city block interiors based on specified attributes and style parameters. Lipp *et al.* [75] proposed a method for interactive city layout modeling, combining procedural and manual modeling. Emilien *et al.* [29] proposed a method for creating village layouts on diverse terrains, utilizing a hybrid settlement/road generation process with dynamic interest maps and an anisotropic conquest process. Peng *et al.* [101] presented a solution for covering a domain with deformable templates, ensuring no overlap by considering constraints and permissible deformations. Yang *et al.* [156] offered a framework for generating quality street networks and parcel layouts using hierarchical domain splitting for urban planning and virtual environments. Wang *et al.* [143] employed a digital description framework and generative grammar to examine the morphological complexity of block forms. Nagy *et al.* [90] demonstrated how Generative Design optimizes a residential neighborhood project, showcasing its potential in solving complex urban design challenges with conflicting stakeholder demands. Sung and Jeong *et al.* [129] presented an efficient approach to organizing multiple buildings on a site using the C# script component in Rhino3d and Grasshopper, offering rapid, real-time results without complex computations.

5.2.2 Learning-based methods

Feng *et al.* [32] introduced a random forest-based method for designing mid-scale layouts in areas like shopping malls, optimizing paths and sites based on crowd flow metrics: mobility, accessibility, and coziness. Liang *et al.* [72] employed Generative Latent Optimization (GLO) and adversarial training to develop a model for effortless generation and placement of buildings on a designated map. Shen *et al.* [122] employed GANs to automatically generate urban design plans, predicting building details based on city conditions. Fedorova [31] explored using GANs to design urban blocks, employing a flexible model that learns from the existing city context rather than explicitly defining parameters. BlockPlanner [152] introduced a vectorized block-plan representation and employs a graph-constrained VAE to decode different geometric attributes. Ying *et al.* [161, 160] proposed an intelligent method using genetic algorithms and CNNs to optimize high-density residential building layouts considering local wind conditions. Quan [105] presented Urban-GAN, a user-friendly urban design system enabling individuals with minimal design expertise to select, generate, and make design decisions based on urban form cases. Jiang *et al.* [60] presented ESGAN, a GAN-based model for automated building layout generation, incorporating a conditional vector to meet project requirements in different design scenarios. Sun *et al.* [127] explored machine learning preferences in residential site plan layouts (RSPL) using the Pix2pix model, aiming to enhance applications in residential and urban planning development. GlobalMapper [51] represented city block layouts as graphs, encoding building layouts into a shape-independent canonical representation. As shown in Fig. 13, it also enables conditional generation on realistic urban layouts for arbitrary road networks.

Recently, He and Aliaga [52] have proposed a graph-based masked autoencoder (GMAE) that uses a canonical graph representation to generate large-scale, context-sensitive urban layouts with high realism and semantic consistency. Similarly, Unlu et al. [133] have introduced GroundUp, a human-centered AI tool that enables architects to easily convert 2D sketches into 3D city massing models by integrating a sketch-to-depth prediction network with a diffusion model. ControlCity [174] introduced a multimodal diffusion model that combines text, metadata, road networks, and imagery to accurately generate urban building footprint data from Volunteer Geographic Information (VGI). UrbanEvolver [104] introduced a deep generative model for function-aware urban layout regeneration. The model generates roads and building layouts for target regions, considering land use types and surrounding contexts. It employs function-layout adaptive blocks and a comprehensive loss system.

6. Summary and Discussion

6.1. Summary on existing methods

In the literature on mainstream methodologies in computer-aided architectural design, it has been recognized that learning-based generation significantly transforms and enhances the traditional paradigm, which has been characterized by manually intensive, multi-round workflows for designers. Training on large-scale, high-quality benchmark datasets enables generation networks to learn desired layout distributions, even with adaptive user input or manually imposed constraints. The evolution of backbone neural networks from CNNs to Transformers, along with deep generative models such as GANs, VAEs, and Diffusion Models, has further improved generation performance across architectural design tasks. Another important shift in method design is the adoption of vectorized representations for layouts due to their flexibility and accuracy in depicting layout

geometry. These learning-based generative paradigms offer greater flexibility, diversity, and completeness compared to traditional methods. However, significant limitations remain in these end-to-end synthesis pipelines, as discussed in the previous sections.

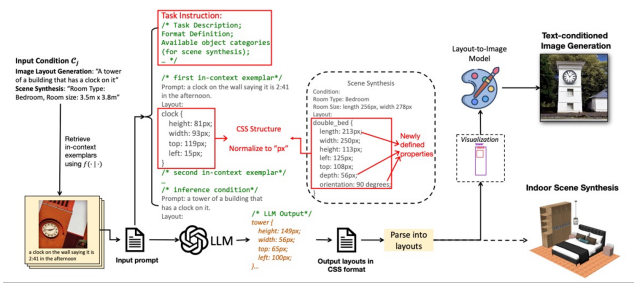


Figure 14. LayoutGPT achieves training-free layout-to-image or indoor scene layout generation using LLMs. With intricate text prompts and instructions (*i.e.*, converting the text inputs to CSS formats), it has demonstrated comparable or better generation fidelity with the SOTA methods trained on a specific dataset.

6.2. Discussion on open problems and new perspectives

Future research could focus on multi-modal input support to accommodate diverse customer requirements. In floorplan generation, existing studies [149, 91] have explored user inputs such as house boundaries and bubble diagrams. In scene synthesis, current research emphasizes graph-conditioned [27, 163, 9], text-conditioned generation [131], room boundary constraints [100], and partial layout completion [100, 98, 131]. Inspired by recent advancements in multi-modal, image-level foundational generative models [112, 115], a promising direction is to develop a unified foundational floorplan generative model that accommodates various input formats (e.g., text or audio instructions, bubble diagrams, layout sketches), enabling the model to learn more powerful representations through the integration of multi-modal inputs. Moreover, reinforcement learning (RL) techniques can be applied to incorporate embodied AI into the layout generation process, allowing for better capture of user interactions and specifications.

Another potential direction is to leverage the capabilities of large language models (LLMs) to assist the architectural design workflow. Recent advancements in combining LLMs with visual understanding [96, 157, 147] have inspired the community to explore how to fully utilize the potential of LLMs, such as ChatGPT, in supporting diverse downstream computer vision tasks. LayoutGPT [33] (as shown in Fig. 14) has demonstrated the remarkable efficacy of using LLMs (e.g., GPT-3.5 or GPT-4) for layout-to-image generation or indoor scene layouts. However, many state-of-the-art LLMs are proprietary and accessible only via paid APIs, limiting their broader adoption for handling large-scale data inputs. Furthermore, although LLMs are trained on extensive internet data to achieve superior zero-shot generative capabilities, they may not directly address the specialized needs of particular user inputs or tasks. As such, their output should be considered a preliminary result requiring further refinement. Furthermore, the generated layouts are generally limited to regular shapes, such as bounding boxes, which do not always align with the complex, often polygonal or non-Manhattan room layouts found in real-world floor plans. Further exploration is needed to adapt the broad applicability of LLMs to more specialized downstream tasks, such as floorplan generation and scene layout synthesis, using data from diverse environments with varied styles.

A shared limitation among most existing layout generative models is that they are trained and tested on a single dataset, severely limiting the generalizability of a model across different environments, object categories, scales, and styles. One potential solution is to construct large-scale benchmark datasets with more diverse data samples to enhance the model's representation capacity and achieve desirable fidelity across varied data distributions. Another strategy to address dataset limitations is to employ self-supervised pre-training approaches, such as contrastive learning [19, 49] or large foundation models [106, 95, 112], to encapsulate a general embedded representation, which can then be fine-tuned for specific downstream tasks or datasets.

7. Conclusions

In this review paper, we first provide an overview of the current progress in computer-aided architectural design, categorizing the relevant topics into three major areas. For each category, we thoroughly examine methods from both the traditional architectural domain and learning-based generative approaches. We also discuss the advantages and current bottlenecks related to data, settings, and methods. Furthermore, we identify existing open problems and propose new perspectives for the research community, with a focus on leveraging powerful techniques such as foundational generative models and LLMs. We hope this survey encourages the community to reassess the needs and limitations of real-world applications, inspiring more valuable and insightful research in the field.

References

[1] M. Aalaei, M. Saadi, M. Rahbar, and A. Ekhlassi. Architectural layout generation using a graph-constrained conditional generative adversarial network (gan). *Automation in Construction*, 155:105053, 2023. 7, 8

[2] Z. Abu-Aisheh, R. Raveaux, J.-Y. Ramel, and P. Martineau. An exact graph edit distance algorithm for solving pattern recognition problems. In *4th International Conference on*

Pattern Recognition Applications and Methods 2015, 2015. 9

[3] H. Akaike. Information theory and an extension of the maximum likelihood principle. In *Selected papers of hirotugu akaike*, pages 199–213, 1998. 15

[4] M. F. Alam, Y. Wang, L. Tran, C.-Y. Cheng, and J. Luo. Representation learning for sequential volumetric design tasks. *arXiv preprint arXiv:2309.02583*, 2023. 15, 16

[5] D. G. Aliaga, C. A. Vanegas, and B. Benes. Interactive example-based urban layout synthesis. *ACM transactions on graphics (TOG)*, 27(5):1–10, 2008. 17

[6] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017. 1, 3

[7] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223, 2017. 3

[8] S. A. Arvin and D. H. House. Modeling architectural design objectives in physically based space planning. *Automation in Construction*, 11(2):213–225, 2002. 5

[9] S. Bahmani, J. J. Park, D. Paschalidou, X. Yan, G. Wetzstein, L. Guibas, and A. Tagliasacchi. Cc3d: Layout-conditioned generation of compositional 3d scenes. *arXiv preprint arXiv:2303.12074*, 2023. 13, 18

[10] A. Bahrehmand, T. Batard, R. Marques, A. Evans, and J. Blat. Optimizing layout using spatial quality metrics and user preferences. *Graphical models*, 93:25–38, 2017. 15

[11] A. Banerjee, J. C. Quiroz, and S. J. Louis. A model of creative design using collaborative interactive genetic algorithms. In *Design Computing and Cognition'08: Proceedings of the Third International Conference on Design Computing and Cognition*, pages 397–416, 2008. 6

[12] F. Bao, D.-M. Yan, N. J. Mitra, and P. Wonka. Generating and exploring good building layouts. *ACM Transactions on Graphics (TOG)*, 32(4):1–10, 2013. 15

[13] Y. Bengio, R. Ducharme, and P. Vincent. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000. 4

[14] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 14

[15] S. Bond-Taylor, A. Leach, Y. Long, and C. G. Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE transactions on pattern analysis and machine intelligence*, 2021. 3

[16] A. Chang, M. Savva, and C. D. Manning. Learning spatial knowledge for text to 3d scene generation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 2028–2038, 2014. 12

[17] A. X. Chang, M. Eric, M. Savva, and C. D. Manning. Sceneseer: 3d scene design with natural language. *arXiv preprint arXiv:1703.00050*, 2017. 12

[18] K.-H. Chang, C.-Y. Cheng, J. Luo, S. Murata, M. Nourbakhsh, and Y. Tsuji. Building-gan: Graph-conditioned architectural volumetric design generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11956–11965, 2021. 15, 16

[19] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607, 2020. 18

[20] X. Chen, N. Mishra, M. Rohaninejad, and P. Abbeel. Pixelsnail: An improved autoregressive generative model. In *International Conference on Machine Learning*, pages 864–872, 2018. 4

[21] X. Chen, Y. Xiong, S. Wang, H. Wang, T. Sheng, Y. Zhang, and Y. Ye. Reco: A dataset for residential community layout planning. *arXiv preprint arXiv:2206.04678*, 2022. 16

[22] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 4

[23] J. Coughlan and A. L. Yuille. The manhattan world assumption: Regularities in scene statistics which enable bayesian inference. *Advances in Neural Information Processing Systems*, 13, 2000. 1

[24] S. Cruz, W. Hutchcroft, Y. Li, N. Khosravan, I. Boyadzhiev, and S. B. Kang. Zillow indoor dataset: Annotated floor plans with 360deg panoramas and 3d room layouts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2133–2143, 2021. 8

[25] I. Csiszár. I-divergence geometry of probability distributions and minimization problems. *The annals of probability*, pages 146–158, 1975. 3

[26] M. Deitke, E. VanderBilt, A. Herrasti, L. Weihs, K. Ehsani, J. Salvador, W. Han, E. Kolve, A. Kembhavi, and R. Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in Neural Information Processing Systems*, 35:5982–5994, 2022. 8

[27] H. Dhamo, F. Manhardt, N. Navab, and F. Tombari. Graph-to-3d: End-to-end generation and manipulation of 3d scenes using scene graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16352–16361, 2021. 12, 15, 18

[28] Z. Du, H. Shen, X. Li, and M. Wang. 3d building fabrication with geometry and texture coordination via hybrid gan. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–12, 2020. 16

[29] A. Emilien, A. Bernhardt, A. Peytavie, M.-P. Cani, and E. Galin. Procedural generation of villages on arbitrary terrains. *The Visual Computer*, 28:809–818, 2012. 16, 17

[30] Z. Fan, J. Liu, L. Wang, G. Cheng, M. Liao, P. Liu, and Y. F. Chen. Automated layout of modular high-rise residential buildings based on genetic algorithm. *Automation in Construction*, 152:104943, 2023. 5, 6, 16

[31] S. Fedorova. Generative adversarial networks for urban block design. In *SimAUD: A Symposium on Simulation for Architecture and Urban Design*, 2021. 17

[32] T. Feng, L.-F. Yu, S.-K. Yeung, K. Yin, and K. Zhou. Crowd-driven mid-scale layout design. *ACM Trans. Graph.*, 35(4):132–1, 2016. 6, 17

[33] W. Feng, W. Zhu, T.-j. Fu, V. Jampani, A. Akula, X. He, S. Basu, X. E. Wang, and W. Y. Wang. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in Neural Information Processing Systems*, 36, 2024. 13, 18

[34] M. Fisher and P. Hanrahan. Context-based search for 3d models. In *ACM SIGGRAPH Asia 2010 papers*, pages 1–10. 2010. 12

[35] M. Fisher, M. Savva, Y. Li, P. Hanrahan, and M. Nießner. Activity-centric scene synthesis for functional 3d scene modeling. *ACM Transactions on Graphics (TOG)*, 34(6):1–13, 2015. 12

[36] B. Fu, Y. Gao, and W. Wang. Dual generative adversarial networks for automated component layout design of steel frame-brace structures. *Automation in Construction*, 146:104661, 2023. 16

[37] H. Fu, B. Cai, L. Gao, L.-X. Zhang, J. Wang, C. Li, Q. Zeng, C. Sun, R. Jia, B. Zhao, et al. 3d-front: 3d furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10933–10942, 2021. 13, 14

[38] H. Fu, R. Jia, L. Gao, M. Gong, B. Zhao, S. Maybank, and D. Tao. 3d-future: 3d furniture shape with texture. *International Journal of Computer Vision*, 129:3313–3337, 2021. 14

[39] 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 Transactions on Graphics (TOG)*, 36(6):1–13, 2017. 12

[40] V. J. Gan. Bim-based graph data model for automatic generative design of modular buildings. *Automation in Construction*, 134:104062, 2022. 16

[41] L. Gao, J.-M. Sun, K. Mo, Y.-K. Lai, L. J. Guibas, and J. Yang. Scenehgn: Hierarchical graph networks for 3d indoor scene generation with fine-grained geometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 12

[42] D. J. Gerber, E. Pantazis, and A. Wang. A multi-agent approach for performance based architecture: design exploring geometry, user, and environmental agencies in façades. *Automation in construction*, 76:45–58, 2017. 15, 16

[43] P. Ghannad and Y.-C. Lee. Developing an advanced automated modular housing design system using deep learning and building information modeling (bim). In *Computing in Civil Engineering*, pages 587–595. 2021. 16

[44] P. Ghannad and Y.-C. Lee. Automated modular housing design using a module configuration algorithm and a coupled generative adversarial network (cogan). *Automation in construction*, 139:104234, 2022. 16

[45] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272, 2017. 8

[46] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 1, 3

[47] Z. Guo and B. Li. Evolutionary approach for spatial architecture layout design enhanced by an agent-based topology finding system. *Frontiers of Architectural Research*, 6(1):53–62, 2017. 5

[48] F. He, Y. Huang, and H. Wang. iplan: interactive and procedural layout planning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7793–7802, 2022. 7, 8

[49] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 18

[50] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[51] L. He and D. Aliaga. Globalmapper: Arbitrary-shaped urban layout generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 454–464, 2023. 17

[52] L. He and D. Aliaga. Coho: Context-sensitive city-scale hierarchical urban layout generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 17

[53] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 9, 14

[54] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 1, 4, 7, 12

[55] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 4

[56] S. Hosseini and Y. Furukawa. Floorplan restoration by structure hallucinating transformer cascades. 5

[57] R. Hu, Z. Huang, Y. Tang, O. Van Kaick, H. Zhang, and H. Huang. Graph2plan: Learning floorplan generation from layout graphs. *ACM Transactions on Graphics (TOG)*, 39(4):118–1, 2020. 6, 7, 8

[58] S. Hu, W. Wu, Y. Wang, B. Xu, and L. Zheng. Advancing architectural floorplan design with geometry-enhanced graph diffusion. *arXiv preprint arXiv:2408.16258*, 2024. 8

[59] S. Isaac, T. Bock, and Y. Stoliar. A methodology for the optimal modularization of building design. *Automation in construction*, 65:116–124, 2016. 16

[60] F. Jiang, J. Ma, C. J. Webster, X. Li, and V. J. Gan. Building layout generation using site-embedded gan model. *Automation in Construction*, 151:104888, 2023. 17

[61] Y. Jiang, M. Lim, and A. Saxena. Learning object arrangements in 3d scenes using human context. *arXiv preprint arXiv:1206.6462*, 2012. 12

[62] A. Kalervo, J. Ylioinas, M. Häikiö, A. Karhu, and J. Kannala. Cubicasa5k: A dataset and an improved multi-task model for floorplan image analysis. In *Image Analysis: 21st Scandinavian Conference, SCIA 2019, Norrköping, Sweden, June 11–13, 2019, Proceedings 21*, pages 28–40, 2019. 8

[63] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 13

[64] T. Kelly, P. Guerrero, A. Steed, P. Wonka, and N. J. Mitra. Frankengan: guided detail synthesis for building mass

models using style-synchonized gans. *ACM Transactions on Graphics (TOG)*, 37(6):1–14, 2018. 16

[65] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1, 3

[66] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 8

[67] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila. Improved precision and recall metric for assessing generative models. *Advances in neural information processing systems*, 32, 2019. 9

[68] H. Larochelle and I. Murray. The neural autoregressive distribution estimator. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 29–37, 2011. 4

[69] K. Leimer, P. Guerrero, T. Weiss, and P. Musialski. Layoutenhancer: Generating good indoor layouts from imperfect data. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–8, 2022. 13

[70] M. Li, A. G. Patil, K. Xu, S. Chaudhuri, O. Khan, A. Shamir, C. Tu, B. Chen, D. Cohen-Or, and H. Zhang. Grains: Generative recursive autoencoders for indoor scenes. *ACM Transactions on Graphics (TOG)*, 38(2):1–16, 2019. 12

[71] Y. Li, P. Xu, J. Ren, Z. Shao, and H. Huang. Gltscene: Global-to-local transformers for indoor scene synthesis with general room boundaries. In *Computer Graphics Forum*, volume 43, page e15236, 2024. 13

[72] J. Liang, H. Liu, Y. Zhao, M. Sanjabi, M. Sardari, H. Chaput, N. Aghdaie, and K. Zaman. Building placements in urban modeling using conditional generative latent optimization. In *IEEE International Conference on Image Processing (ICIP)*, pages 3249–3253, 2020. 17

[73] W. Liao, X. Lu, Y. Huang, Z. Zheng, and Y. Lin. Automated structural design of shear wall residential buildings using generative adversarial networks. *Automation in Construction*, 132:103931, 2021. 15, 16

[74] L. LIFULL Co. Lifull home's dataset (collection), nov 2015. 8

[75] M. Lipp, D. Scherzer, P. Wonka, and M. Wimmer. Interactive modeling of city layouts using layers of procedural content. In *Computer Graphics Forum*, volume 30, pages 345–354, 2011. 17

[76] C. Liu, J. Wu, P. Kohli, and Y. Furukawa. Raster-to-vector: Revisiting floorplan transformation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2195–2203, 2017. 8

[77] J. Liu, Y. Xue, J. Duarte, K. Shekhawat, Z. Zhou, and X. Huang. End-to-end graph-constrained vectorized floorplan generation with panoptic refinement. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XV*, pages 547–562, 2022. 6, 7, 8, 9

[78] X. Lu, W. Liao, Y. Zhang, and Y. Huang. Intelligent structural design of shear wall residence using physics-enhanced generative adversarial networks. *Earthquake Engineering & Structural Dynamics*, 51(7):1657–1676, 2022. 15, 16

[79] A. Luo, Z. Zhang, J. Wu, and J. B. Tenenbaum. End-to-end optimization of scene layout. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3754–3763, 2020. 12

[80] Z. Luo and W. Huang. Floorplangan: Vector residential floorplan adversarial generation. *Automation in Construction*, 142:104470, 2022. 6, 7

[81] Z. H. Luo, L. Lara, G. Y. Luo, F. Golemo, C. Beckham, and C. Pal. Dstruct2design: Data and benchmarks for data structure driven generative floor plan design. *arXiv preprint arXiv:2407.15723*, 2024. 8

[82] B. Medjdoub and B. Yannou. Separating topology and geometry in space planning. *Computer-aided design*, 32(1):39–61, 2000. 6

[83] Q. Meng, L. Li, M. Nießner, and A. Dai. Lt3sd: Latent trees for 3d scene diffusion. *arXiv preprint arXiv:2409.08215*, 2024. 13

[84] P. Merrell, E. Schkufza, and V. Koltun. Computer-generated residential building layouts. In *ACM SIGGRAPH Asia 2010 papers*, pages 1–12. 2010. 5, 16

[85] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, and V. Koltun. Interactive furniture layout using interior design guidelines. *ACM transactions on graphics (TOG)*, 30(4):1–10, 2011. 11

[86] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 13

[87] W. Min, W. Wu, G. Zhang, and L. Zheng. Funcscene: Function-centric indoor scene synthesis via a variational autoencoder framework. *Computer Aided Geometric Design*, 111:102319, 2024. 13

[88] T. Miyato and M. Koyama. cGANs with projection discriminator. In *International Conference on Learning Representations*, 2018. 9

[89] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool. Procedural modeling of buildings. *ACM Transactions on Graphics (TOG)*, 25(3):614–623, 2006. 15

[90] D. Nagy, L. Villaggi, and D. Benjamin. Generative urban design: integrating financial and energy goals for automated neighborhood layout. In *Proceedings of the Symposium for Architecture and Urban Design Design*, pages 265–274, 2018. 17

[91] N. Nauata, K.-H. Chang, C.-Y. Cheng, G. Mori, and Y. Furukawa. House-gan: Relational generative adversarial networks for graph-constrained house layout generation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 162–177, 2020. 6, 7, 8, 18

[92] N. Nauata, S. Hosseini, K.-H. Chang, H. Chu, C.-Y. Cheng, and Y. Furukawa. House-gan++: Generative adversarial layout refinement network towards intelligent computational agent for professional architects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13632–13641, 2021. 6, 7, 8

[93] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen. Glide: Towards

photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 4

[94] A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171, 2021. 4

[95] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022. 13, 18

[96] Z. Pang, Z. Xie, Y. Man, and Y.-X. Wang. Frozen transformers in language models are effective visual encoder layers. *arXiv preprint arXiv:2310.12973*, 2023. 18

[97] W. Para, P. Guerrero, T. Kelly, L. J. Guibas, and P. Wonka. Generative layout modeling using constraint graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6690–6700, 2021. 6, 7

[98] W. R. Para, P. Guerrero, N. Mitra, and P. Wonka. Cofs: Controllable furniture layout synthesis. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023. 13, 18

[99] Y. I. Parish and P. Müller. Procedural modeling of cities. In *Proceedings of the 28th annual conference on computer graphics and interactive techniques*, pages 301–308, 2001. 17

[100] D. Paschalidou, A. Kar, M. Shugrina, K. Kreis, A. Geiger, and S. Fidler. Atiss: Autoregressive transformers for indoor scene synthesis. *Advances in Neural Information Processing Systems*, 34:12013–12026, 2021. 13, 18

[101] C.-H. Peng, Y.-L. Yang, and P. Wonka. Computing layouts with deformable templates. *ACM Transactions on Graphics (TOG)*, 33(4):1–11, 2014. 17

[102] P. Purkait, C. Zach, and I. Reid. Sg-vae: Scene grammar variational autoencoder to generate new indoor scenes. In *European Conference on Computer Vision*, pages 155–171, 2020. 12

[103] Y. Qian, H. Zhang, and Y. Furukawa. Roof-gan: Learning to generate roof geometry and relations for residential houses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2796–2805, 2021. 15, 16

[104] Y. Qin, N. Zhao, J. Yang, S. Pan, B. Sheng, and R. W. Lau. Urbanevolver: function-aware urban layout regeneration. *International Journal of Computer Vision*, pages 1–20, 2024. 17

[105] S. J. Quan. Urban-gan: An artificial intelligence-aided computation system for plural urban design. *Environment and Planning B: Urban Analytics and City Science*, 49(9):2500–2515, 2022. 17

[106] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763, 2021. 18

[107] J. Ren, B. Zhang, B. Wu, J. Huang, L. Fan, M. Ovsjanikov, and P. Wonka. Intuitive and efficient roof modeling for reconstruction and synthesis. *arXiv preprint arXiv:2109.07683*, 2021. 15, 16

[108] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286, 2014. 3

[109] D. Ritchie, P. Guerrero, R. K. Jones, N. J. Mitra, A. Schulz, K. D. Willis, and J. Wu. Neurosymbolic models for computer graphics. In *Computer Graphics Forum*, volume 42, pages 545–568, 2023. 2

[110] D. Ritchie, K. Wang, and Y.-a. Lin. Fast and flexible indoor scene synthesis via deep convolutional generative models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6182–6190, 2019. 13

[111] E. Rodrigues, A. R. Gaspar, and Á. Gomes. An approach to the multi-level space allocation problem in architecture using a hybrid evolutionary technique. *Automation in Construction*, 35:482–498, 2013. 15

[112] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 18

[113] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241, 2015. 4

[114] M. Rosenman. Case-based evolutionary design. *AI EDAM*, 14(1):17–29, 2000. 5

[115] L. Ruan, Y. Ma, H. Yang, H. He, B. Liu, J. Fu, N. J. Yuan, Q. Jin, and B. Guo. Mm-diffusion: Learning multi-modal diffusion models for joint audio and video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10219–10228, 2023. 18

[116] M. S. Sajjadi, O. Bachem, M. Lucic, O. Bousquet, and S. Gelly. Assessing generative models via precision and recall. *Advances in neural information processing systems*, 31, 2018. 9

[117] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008. 8

[118] M. Schwarz and P. Müller. Advanced procedural modeling of architecture. *ACM Transactions on Graphics (TOG)*, 34(4):1–12, 2015. 15

[119] M. A. Shabani, S. Hosseini, and Y. Furukawa. Housediffusion: Vector floorplan generation via a diffusion model with discrete and continuous denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5466–5475, 2023. 5, 6, 7, 8, 9

[120] P. Sharafi, B. Samali, H. Ronagh, and M. Ghodrat. Automated spatial design of multi-story modular buildings using a unified matrix method. *Automation in Construction*, 82:31–42, 2017. 16

[121] K. Shekhawat, N. Upasani, S. Bisht, and R. Jain. Gplan: Computer-generated dimensioned floorplans for given adjacencies. *arXiv preprint arXiv:2008.01803*, 2020. 5

[122] J. Shen, C. Liu, Y. Ren, and H. Zheng. Machine learning assisted urban filling. In *International Conference on Computer-Aided Architectural Design Research in Asia, CAADRIA*, pages 679–688, 2020. 17

[123] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265, 2015. 4

[124] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 7

[125] Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019. 4

[126] J. Sun, W. Wu, L. Liu, W. Min, G. Zhang, and L. Zheng. Wallplan: synthesizing floorplans by learning to generate wall graphs. *ACM Transactions on Graphics (TOG)*, 41(4):1–14, 2022. 8

[127] P. Sun, F. Yan, Q. He, and H. Liu. The development of an experimental framework to explore the generative design preference of a machine learning-assisted residential site plan layout. *Land*, 12(9):1776, 2023. 17

[128] Q. Sun, H. Zhou, W. Zhou, L. Li, and H. Li. Forest2seq: Revitalizing order prior for sequential indoor scene synthesis. In *European Conference on Computer Vision*, pages 251–268, 2025. 13

[129] W. Sung and Y. Jeong. Site planning automation of apartment complex through grid-based calculation in grasshopper. *Automation in construction*, 138:104216, 2022. 17

[130] H. Tang, Z. Zhang, H. Shi, B. Li, L. Shao, N. Sebe, R. Timofte, and L. Van Gool. Graph transformer gans for graph-constrained house generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2173–2182, 2023. 7, 8

[131] J. Tang, Y. Nie, L. Markhasin, A. Dai, J. Thies, and M. Nießner. Diffuscene: Denoising diffusion models for generative indoor scene synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20507–20518, 2024. 12, 18

[132] A. Tono, H. Huang, A. Agrawal, and M. Fischer. Vitruvio: Conditional variational autoencoder to generate building meshes via single perspective sketches. *Automation in Construction*, 166:105498, 2024. 16

[133] G. E. Unlu, M. Sayed, Y. Gryaditskaya, and G. Brostow. Groundup: Rapid sketch-based 3d city massing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 17

[134] A. Upadhyay, A. Dubey, V. Arora, S. M. Kuriakose, and S. Agarawal. Flnet: graph constrained floor layout generation. In *2022 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pages 1–6, 2022. 6, 7, 8

[135] C. A. Vanegas, T. Kelly, B. Weber, J. Halatsch, D. G. Aliaga, and P. Müller. Procedural generation of parcels in urban modeling. In *Computer graphics forum*, volume 31, pages 681–690, 2012. 17

[136] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4, 13

[137] M. J. Wainwright and M. I. Jordan. Introduction to variational methods for graphical models. *Foundations and Trends in Machine Learning*, 1:1–103, 2008. 3

[138] J. Wald, A. Avetisyan, N. Navab, F. Tombari, and M. Nießner. Rio: 3d object instance re-localization in changing indoor environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7658–7667, 2019. 14

[139] J. Wald, H. Dhamo, N. Navab, and F. Tombari. Learning 3d semantic scene graphs from 3d indoor reconstructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3961–3970, 2020. 14

[140] 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 Transactions on Graphics (TOG)*, 38(4):1–15, 2019. 12

[141] K. Wang, M. Savva, A. X. Chang, and D. Ritchie. Deep convolutional priors for indoor scene synthesis. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 13

[142] L. Wang, J. Liu, Y. Zeng, G. Cheng, H. Hu, J. Hu, and X. Huang. Automated building layout generation using deep learning and graph algorithms. *Automation in Construction*, 154:105036, 2023. 16

[143] X. Wang, Y. Song, and P. Tang. Generative urban design using shape grammar and block morphological analysis. *Frontiers of Architectural Research*, 9(4):914–924, 2020. 17

[144] X. Wang, C. Yeshwanth, and M. Nießner. Sceneformer: Indoor scene generation with transformers. In *2021 International Conference on 3D Vision (3DV)*, pages 106–115, 2021. 13

[145] R. E. Weber, C. Mueller, and C. Reinhart. Automated floorplan generation in architectural design: A review of methods and applications. *Automation in Construction*, 140:104385, 2022. 2

[146] Q. A. Wei, S. Ding, J. J. Park, R. Sajnani, A. Poulenard, S. Sridhar, and L. Guibas. Lego-net: Learning regular rearrangements of objects in rooms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19037–19047, 2023. 13

[147] C. Wu, S. Yin, W. Qi, X. Wang, Z. Tang, and N. Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*, 2023. 18

[148] W. Wu, L. Fan, L. Liu, and P. Wonka. Miqp-based layout design for building interiors. In *Computer Graphics Forum*, volume 37, pages 511–521, 2018. 16

[149] W. Wu, X.-M. Fu, R. Tang, Y. Wang, Y.-H. Qi, and L. Liu. Data-driven interior plan generation for residential buildings. *ACM Transactions on Graphics (TOG)*, 38(6):1–12, 2019. 7, 8, 18

[150] Z. Wu, Z. Wang, S. Liu, H. Luo, J. Lu, and H. Yan. Fairscene: Learning unbiased object interactions for indoor scene synthesis. *Pattern Recognition*, 156:110737, 2024. 13

[151] K. Xu, K. Chen, H. Fu, W.-L. Sun, and S.-M. Hu. Sketch2scene: Sketch-based co-retrieval and co-placement of 3d models. *ACM Transactions on Graphics (TOG)*, 32(4):1–15, 2013. 12

[152] L. Xu, Y. Xiangli, A. Rao, N. Zhao, B. Dai, Z. Liu, and D. Lin. Blockplanner: city block generation with vectorized graph representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5077–5086, 2021. 17

[153] H. Yang, Z. Zhang, S. Yan, H. Huang, C. Ma, Y. Zheng, C. Bajaj, and Q. Huang. Scene synthesis via uncertainty-driven attribute synchronization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5630–5640, 2021. 12

[154] M.-J. Yang, Y.-X. Guo, B. Zhou, and X. Tong. Indoor scene generation from a collection of semantic-segmented depth images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15203–15212, 2021. 12

[155] Y. Yang, B. Jia, P. Zhi, and S. Huang. Physcene: Physically interactable 3d scene synthesis for embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16262–16272, 2024. 13

[156] Y.-L. Yang, J. Wang, E. Vouga, and P. Wonka. Urban pattern: Layout design by hierarchical domain splitting. *ACM Transactions on Graphics (TOG)*, 32(6):1–12, 2013. 17

[157] Z. Yang, L. Li, K. Lin, J. Wang, C.-C. Lin, Z. Liu, and L. Wang. The dawn of lmms: Preliminary explorations with gpt-4v (ision). *arXiv preprint arXiv:2309.17421*, 9, 2023. 18

[158] Y.-T. Yeh, L. Yang, M. Watson, N. D. Goodman, and P. Hanrahan. Synthesizing open worlds with constraints using locally annealed reversible jump mcmc. *ACM Transactions on Graphics (TOG)*, 31(4):1–11, 2012. 11

[159] H. Yi, Y. K. Yi, and T. Chan. Performance based architectural design optimization: Automated 3d space layout using simulated annealing. In *Proceedings of the 2014 ASHRAE/IBPSA-USA Building Simulation Conference, Atlanta, GA, USA*, pages 10–12, 2014. 5

[160] X. Ying, X. Qin, J. Chen, and J. Gao. Generating residential layout based on ai in the view of wind environment. In *Journal of Physics: Conference Series*, volume 2069, page 012061, 2021. 17

[161] X. Ying, X. Qin, L. Shen, C. Yu, and J. Zhang. An intelligent planning method to optimize high-density residential layouts considering the influence of wind environments. *Heliyon*, 9(1), 2023. 17

[162] L. F. Yu, S. K. Yeung, C. K. Tang, D. Terzopoulos, T. F. Chan, and S. J. Osher. Make it home: automatic optimization of furniture arrangement. *ACM Transactions on Graphics (TOG)-Proceedings of ACM SIGGRAPH 2011, v. 30,(4), July 2011, article no. 86*, 30(4), 2011. 12

[163] G. Zhai, E. P. Örnek, S.-C. Wu, Y. Di, F. Tombari, N. Navab, and B. Busam. Commonscenes: Generating commonsense 3d indoor scenes with scene graphs. *Advances in Neural Information Processing Systems*, 36, 2024. 13, 14, 15, 18

[164] F. Zhang, N. Nauata, and Y. Furukawa. Conv-mpn: Convolutional message passing neural network for structured outdoor architecture reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2798–2807, 2020. 8

[165] H. Zhang, A. Savov, and B. Dillenburger. Maskplan: Masked generative layout planning from partial input. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8964–8973, 2024. 7

[166] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 5907–5915, 2017. 1

[167] S.-H. Zhang, S.-K. Zhang, W.-Y. Xie, C.-Y. Luo, Y.-L. Yang, and H. Fu. Fast 3d indoor scene synthesis by learning spatial relation priors of objects. *IEEE Transactions on Visualization and Computer Graphics*, 28(9):3082–3092, 2021. 12

[168] S.-K. Zhang, J.-H. Liu, Y. Li, T. Xiong, K.-X. Ren, H. Fu, and S.-H. Zhang. Automatic generation of commercial scenes. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 1137–1147, 2023. 12

[169] Z. Zhang, Z. Yang, C. Ma, L. Luo, A. Huth, E. Vouga, and Q. Huang. Deep generative modeling for scene synthesis via hybrid representations. *ACM Transactions on Graphics (TOG)*, 39(2):1–21, 2020. 12

[170] P. Zhao, W. Liao, H. Xue, and X. Lu. Intelligent design method for beam and slab of shear wall structure based on deep learning. *Journal of Building Engineering*, 57:104838, 2022. 15, 16

[171] Y. Zhao, Z. Zhao, J. Li, S. Dong, and S. Gao. Roomdesigner: Encoding anchor-latents for style-consistent and shape-compatible indoor scene generation. In *2024 International Conference on 3D Vision (3DV)*, pages 1413–1423, 2024. 13

[172] J. Zheng, J. Zhang, J. Li, R. Tang, S. Gao, and Z. Zhou. Structured3d: A large photo-realistic dataset for structured 3d modeling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 519–535, 2020. 8

[173] Z. Zheng and F. Petzold. Neural-guided room layout generation with bubble diagram constraints. *Automation in Construction*, 154:104962, 2023. 7, 8

[174] F. Zhou, H. Li, R. Hu, S. Wu, H. Feng, Z. Du, and L. Xu. Controlcity: A multimodal diffusion model based approach for accurate geospatial data generation and urban morphology analysis. *arXiv preprint arXiv:2409.17049*, 2024. 17

[175] Y. Zhou, Z. While, and E. Kalogerakis. Scenegraphnet: Neural message passing for 3d indoor scene augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7384–7392, 2019. 12