HouseDiffusion: Vector Floorplan Generation via a Diffusion Model with Discrete and Continuous Denoising

Mohammad Amin Shabani, Sepidehsadat Hosseini, Yasutaka Furukawa
Simon Fraser University
{mshabani, sepidh, furukawa}@sfu.ca

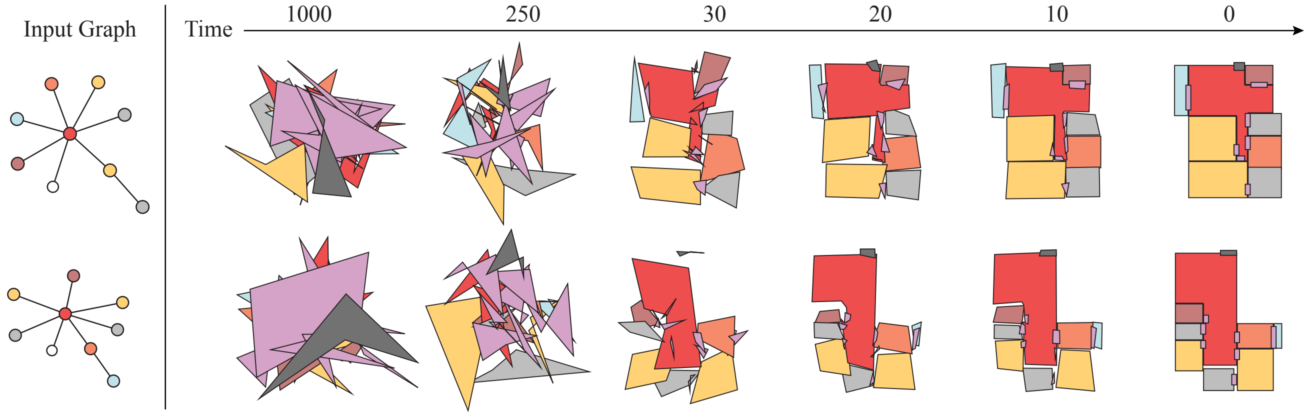


Figure 1. Given a bubble diagram as the input constraint, HouseDiffusion directly generates a vector floorplan by initializing the room/door coordinates with Gaussian noise and iteratively denoising them. Qualitative and quantitative evaluations demonstrate that HouseDiffusion significantly outperforms the current state-of-the-art with large margins.

Abstract

The paper presents a novel approach for vector-floorplan generation via a diffusion model, which denoises 2D coordinates of room/door corners with two inference objectives: 1) a single-step noise as the continuous quantity to precisely invert the continuous forward process; and 2) the final 2D coordinate as the discrete quantity to establish geometric incident relationships such as parallelism, orthogonality, and corner-sharing. Our task is graph-conditioned floorplan generation, a common workflow in floorplan design. We represent a floorplan as 1D polygonal loops, each of which corresponds to a room or a door. Our diffusion model employs a Transformer architecture at the core, which controls the attention masks based on the input graph-constraint and directly generates vector-graphics floorplans via a discrete and continuous denoising process. We have evaluated our approach on RPLAN dataset. The proposed approach makes significant improvements in all the metrics against the state-of-the-art with significant margins, while being capable of generating non-Manhattan structures and controlling the exact number of corners per room. A project website with supplementary video and document is here https://aminshabani.github.io/housediffusion.

1. Introduction

Automated floorplan generation made tremendous progress in the last few years. While not being fully autonomous yet, the state-of-the-art techniques help architects to explore the space of possible designs quickly [33, 34]. 90% of buildings do not have dedicated architects for floorplan design due to their cost in North America. This technology will make the work of professional architects affordable to more house buyers.

Despite recent progress, state-of-the-art floorplan generative models produce samples that are incompatible with the input constraint, lack in variations, or do not look like floorplans [34]. The issue is the raster geometry analysis via convolutions, where a room is represented as a binary image. The raster analysis is good at local shape refinement but lacks in global reasoning, and requires non-trivial post-processing for vectorization [7, 34]. On the other hand, direct generation of vector floorplans is not trivial either. Different from the generation of images or natural languages, structured geometry exhibit precise incident relationships among architectural components (e.g., doors and rooms).

For example, a wall is usually axis-aligned, where the coordinate values of adjacent corners are exactly equal. A wall might be shared with adjacent rooms further. Direct regression of 2D coordinates currently faces limitations in achieving these relationships. One could use a discrete representation such as one hot encoding over possible coordinate values with classification, but this causes a label imbalance (i.e., most values are 0 in the encoding) and fails the network training.

This paper presents a novel approach for graph-constrained floorplan generation that directly generates a vector-graphics floorplan (i.e., without any post-processing), handles non-Manhattan architectures, and makes significant improvements on all the metrics. Concretely, a bubble-diagram is given as a graph, whose nodes are rooms and edges are the door-connections. We represent a floorplan as a set of 1D polygonal loops, each of which corresponds to a room or a door, then generate 2D coordinates of room/door corners (See Fig. 1). The key idea is the use of a Diffusion Model (DM) with a careful design in the denoising targets. Our approach infers 1) a single-step noise amount as a continuous quantity to precisely invert the continuous forward process; and 2) the final 2D coordinate as the discrete quantity to establish incident relationships. The discrete representation after the denoising iterations is the final floorplan model.

Qualitative and quantitative evaluations show that the proposed system outperforms the existing state-of-the-art, House-GAN++ [34], with significant margins, while being end-to-end and capable of generating non-Manhattan floorplans with exact control on the number of corners per room. We will share all our code and models.

2. Related Work

Floorplan generation: Generation of 3D buildings and floorplans has been an active area of research from a pre-deep learning era [11,30–32,37]. The research area has further flourished with the emergence of deep learning. Nauata *et al.* [33] proposed House-GAN as a graph constrained floorplan generative model via Generative Adversarial Network [10]. House-GAN generates segmentation masks of different rooms and combines them to a single floorplan. The authors further improved the quality of the generation by House-GAN++ [34], which iteratively refines a layout. Given the boundary of a floorplan, Upadhyay *et al.* [44] used the embedded input boundary as an additional input feature to predict a floorplan. Hu *et al.* [16] proposed Graph2Plan that retrieves a graph layout from a dataset and generates room bounding boxes as well as a floorplan in an ad-hoc way. Sun *et al.* [42] proposed to iteratively generate connectivity graphs of rooms and a floorplan semantic segmentation mask. Given a set of room types and their area sizes as the constraint, Luo and Huang [28] proposed a vector generator and a raster discriminator to train a GAN model using differential rendering. Although their method generates vector floorplans directly, it is limited to rectangular shapes. Along with the adjacency graph as the input, Yin *et al.* [4] use graph-theoretic and linear optimization techniques to generate floorplans. Our paper also tackles a graph-constrained floorplan generation with a bubble diagram as the constraint [34]. The key difference is that HouseDiffusion processes a vector geometry representation from start to finish, and hence, directly generating vector floorplan samples.

Diffusion models: Deep generative models have seen great success in broader domains [10,23,36,38,46], where a Diffusion Model (DM) [6, 41, 49] is an emerging technique. Ho *et al.* [13] used a DM to boost image generation quality. Dhariwal and Nichol [35] made improvements by proposing a new noise schedule and learning the variances of the reverse process. The same authors made further improvements by novel architecture and classifier guidance [9]. DMs have been adapted to many other tasks such as Natural Language Processing [22], Image Captioning [8], Time-Series Forecasting [43], Text-to-Speech [19, 21], and finally Text-to-Image as seen in the great success of DALL-E 2 [39] and Imagen [40].

Molecular Conformation Generation [15,17,27,48] and 3D shape generation [25, 26, 29, 50] are probably the closest to our task. What makes our task unique and challenging is the precise geometric incident relationships, such as parallelism, orthogonality, and corner-sharing among different components, which continuous coordinate regression would never achieve. In this regard, several works use discrete state space [3,5,14] or learn an embedding of discrete data [8,22] in the DM formulation. However, we found that these pure discrete representations do not train well, probably because the diffusion process is continuous in nature. In contrast, our formulation simultaneously infers a single-step noise as the continuous quantity and the final 2D coordinate as the discrete quantity, achieving superior generation capabilities (See Sect. 5.3 for more analysis). To our knowledge, our work is the first in using DMs to generate floorplans as vector graphics images.

3. Preliminary

Diffusion models (DMs) denoise a Gaussian noise x_T towards a data sample x_0 in T steps, whose training consists of the forward and the reverse processes. The forward process takes a data sample x_0 and generates a noisy sample x_t at time step t by sampling a Gaussian noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$:

$$x_t = \sqrt{\gamma_t} x_0 + \sqrt{(1-\gamma_t)}\epsilon. \quad (1)$$

γ_t is a noise schedule that gradually changes from 1 to 0. The reverse process starts from a pure Gaussian noise $x_T \sim \mathcal{N}(0, 1)$ and learns to denoise a sample step by step until reaching x_0, where the denoising process takes x_t and estimates x_{t-1} by inferring x_{t-1}, ϵ, or x_0 [13].

4. HouseDiffusion

HouseDiffusion solves a graph-constrained floorplan generation problem (See Fig. 1). The constraint is a bubble diagram, whose nodes are rooms and edges are door connections. A room node is associated with a room type. [1]

The choice of a data representation is crucial for the success of any generative model. A floorplan is a graph, whose general representation is a set of room-corners and their connections as an adjacency matrix. However, this is not an easy representation to generate with the bubble diagram constraint. Instead, we represent a floorplan as a set of 1D polygonal loops one for each room/door. The challenge is to ensure geometric consistencies among the loops. For example, room corners and walls must be shared exactly without gaps or overlaps. The section explains our solution, namely, the floorplan representation and the network architecture.

4.1. Floorplan Representation

Let $P = \{P_1, P_2, ..., P_N\}$ denote the set of polygonal loops for each room/door to be generated. Each loop P_i is defined by a sequence of corners with 2D coordinates:

$$P_i = \{C_{i,1}, C_{i,2}, ..., C_{i,N_i} | C_{i,j} \in \mathcal{R}^2\}. \tag{2}$$

N_i denotes the number of corners in P_i, which needs to be specified or generated. A common approach is to set the greatest possible number and let the network decide how many corners to use via output flags. However, this significantly increases the representation size and makes training harder and inefficient. Instead, we construct a histogram of the number of corners for each room/door type from the training samples, then probabilistically pick N_i where the probabilities are proportional to their histogram counts. This heuristic works well in practice. A user could also directly specify the number of corners to control the room shape complexity.

Coordinate values in our data are integers in the range of $[0, 255]$. In the forward process, we affinely map the range to $[-1, 1]$ and treat as continuous values $\{C_{i,j}\}$ to be mixed with the Gaussian noise $\mathcal{N}(0, 1)$. In the reverse process, we also represent a coordinate value as a discrete integer in the binary representation, that is, as 8 binary numbers.

4.2. HouseDiffusion Architecture

HouseDiffusion is a diffusion model based architecture. The forward process follows (1), where γ_t is a standard cosine noise schedule [35]. The reverse process is a Transformer [45] based neural network, which takes the floorplan representation at time t and infers the representation at time $t-1$ (See Fig. 2). We append a superscript t to denote the floorplan sample at time t (*e.g.*, P_i^t or $C_{i,j}^t$).

Feature embedding: Given a floorplan sample $\{P^t\}$ at time t, every room/door corner $C_{i,j}^t$ becomes a node in the Transformer architecture with a $d(=512)$ dimensional embedding vector $\hat{C}_{i,j}^t$. We initialize the embedding as

$$\hat{C}_{i,j}^t \leftarrow \text{Linear}([\text{AU}(C_{i,j}^t), R_i, \mathbf{1}(i), \mathbf{1}(j), t]) \tag{3}$$

AU augments the corner coordinate by 1) uniformly sampling $L(=8)$ points along the wall to the next corner; and 2) concatenating the sampled point coordinates. The augmentation helps to reason incident relationships along the walls. R_i is a 25D room-type one hot vector. $\mathbf{1}(\cdot)$ denotes a 32D one-hot vector for a room index i and a corner index j. t is a scalar. A linear layer converts the embedding to a 512D vector.

Continuous denoising: Embedding vectors $\{\hat{C}_{i,j}^t\}$ will go through attention layers with structured masking (See Fig. 3). There are three types of attentions in our attention layer: 1) Component-wise Self Attention (CSA), limiting attentions among nodes in the same room or door 2) Global Self Attention (GSA), a standard self-attention between every pair of corners across all rooms; and 3) Relational Cross Attention (RCA), limiting attentions to from-room-to-door or from-door-to-room that are connected in the constraint graph. We learn three sets of key/query/value matrices in each of the attention layer per head, while we use four multi-heads. The results of the three attentions are summed, followed by a standard Add & Norm layer. The continuous denoising repeats the block of this attention and Add & Norm layers four times. At the end, a single linear layer infers noise $\epsilon_\theta(C_{i,j}, t)$ at each node.

Discrete denoising: Geometric incident relationships (e.g., colinearity, orthogonality, or corner sharing) rarely emerge through coordinate regression. For example, two coordinates almost never become the same by regression. Our approach infers coordinates in a discrete form. Concretely, after obtaining $C_{i,j}^{t-1}$ and $C_{i,j}^0$ from the continuous denoising process, we affinely map $C_{i,j}^0$ back to the range $[0, 255]$, apply rounding, and use an "int2bit" function [8] to convert to a binary representation, that is, 8-dimensional binary vector. We use a similar formula as formula (3) to obtain a 512D embedding vector for each corner:

$$\text{Linear}([\text{AU}(C_{i,j}^0), \text{int2bit}(C_{i,j}^0), R_i, \mathbf{1}(i), \mathbf{1}(j), t]) \tag{4}$$

[1] Room types are "Kitchen", "Living-room", "Bedroom", "Dining-room", "Bathroom", "Study-room", "Balcony", "Entrance", "Storage", and "Unknown". Door types are "Interior door" and "Front door".

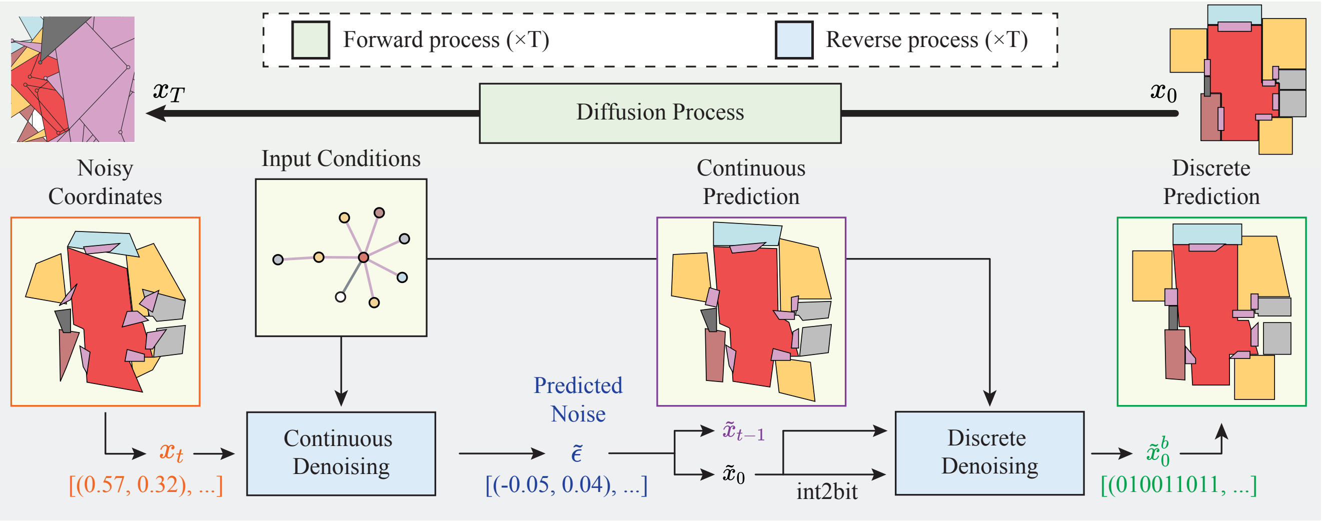
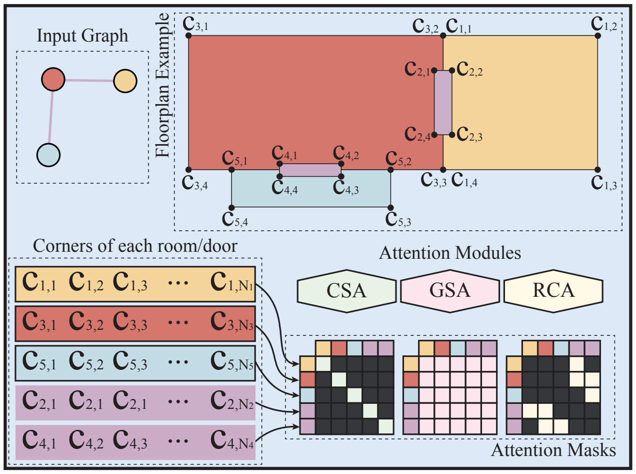


Figure 2. The forward process takes the ground-truth floorplan x_0 and adds a Gaussian noise to create a noisy floorplan sample x_t. The reverse process takes a noisy floorplan at time t with a bubble diagram as the condition. The process infers the corresponding noise $\tilde{\epsilon}$ and $\tilde{x}_0$ in the continuous and the discrete (binary) representations, respectively.



Input Graph
Floorplan Example
$C_{3,1}$
$C_{3,2}$ $C_{1,1}$
$C_{1,2}$
$C_{2,1}$ $C_{2,2}$
$C_{2,4}$ $C_{2,3}$
$C_{5,1}$ $C_{4,1}$ $C_{4,2}$ $C_{5,2}$
$C_{3,4}$ $C_{4,4}$ $C_{4,3}$ $C_{3,3}$ $C_{1,4}$ $C_{1,3}$
$C_{5,4}$ $C_{5,3}$
Corners of each room/door
$C_{1,1}$ $C_{1,2}$ $C_{1,3}$ ··· C_{1,N_1}
$C_{3,1}$ $C_{3,2}$ $C_{3,3}$ ··· C_{3,N_3}
$C_{5,1}$ $C_{5,2}$ $C_{5,3}$ ··· C_{5,N_5}
$C_{2,1}$ $C_{2,1}$ $C_{2,1}$ ··· C_{2,N_2}
$C_{4,1}$ $C_{4,2}$ $C_{4,3}$ ··· C_{4,N_4}
Attention Modules
CSA
GSA
RCA
Attention Masks

Figure 3. Given the input bubble diagram, our model benefits from three attention modules explicitly processing different levels of relations between coordinates. The figure shows how a group of coordinates share information with other groups in each attention module. The black cells represent the masked-out attentions.

We pass $C^0_{i,j}$ as the input to the augmentation and we pass the binary representation obtained by int2bit function along with the conditions to help the network by providing the initial binary representation. We repeat two blocks of attention with structured masking, followed by a linear layer to produce an 8-dimensional vector as $C^0_{i,j}$. During testing, we binary threshold $C^0_{i,j}$ and obtain the integer coordinate. During training, we directly use the values without thresholding for a loss function.

Loss functions: We train our model end-to-end with the simple L2-norm regression loss by Ho *et al.* [13] on both continuous and discrete regressions with the same weight. Concretely, $\epsilon_\theta(C_{i,j}, t)$ is compared with the ground-truth noise ϵ from the forward process. $C^0_{i,j}$ is compared with the ground-truth corner coordinate in binary representation. The inference of $C^0_{i,j}$ becomes accurate only near the end of the denoising process. Therefore, the discrete branch is used during training only when $t < 20$. At testing, we use the denoised integer coordinates from the discrete branch to pass to the next iteration only when $t < 32$.

5. Experiments

We use PyTorch to implement the proposed approach based on a public implementation of Guided-Diffusion [9]. [2] Adam [20] is the optimizer with decoupled weight decay [24] for 250k steps with batch-size of 512 on a single NVIDIA RTX 6000. An initial learning rate is 1e-3. We divide the learning rate by 10 after every 100k steps. We set the number of diffusion steps to 1000 (unless otherwise noted) and uniformly sample t during training.

We compare against other graph-constrained floorplan generative models (House-GAN++ [34] and House-GAN [33]) and scene-graph constrained image generative models (Ashual *et al.* [2], and Johnson *et al.* [18]). House-GAN++ is the current state-of-the-art for the task.

We use 60,000 vector floorplans from RPLAN [47] dataset, and the same pre-processing steps as in House-GAN++, where floorplan images have a resolution of 256 × 256. We divide the floorplan samples into four groups based on the number of rooms (i.e., 5, 6, 7, or 8 rooms). For

[2] https://github.com/openai/guided-diffusion

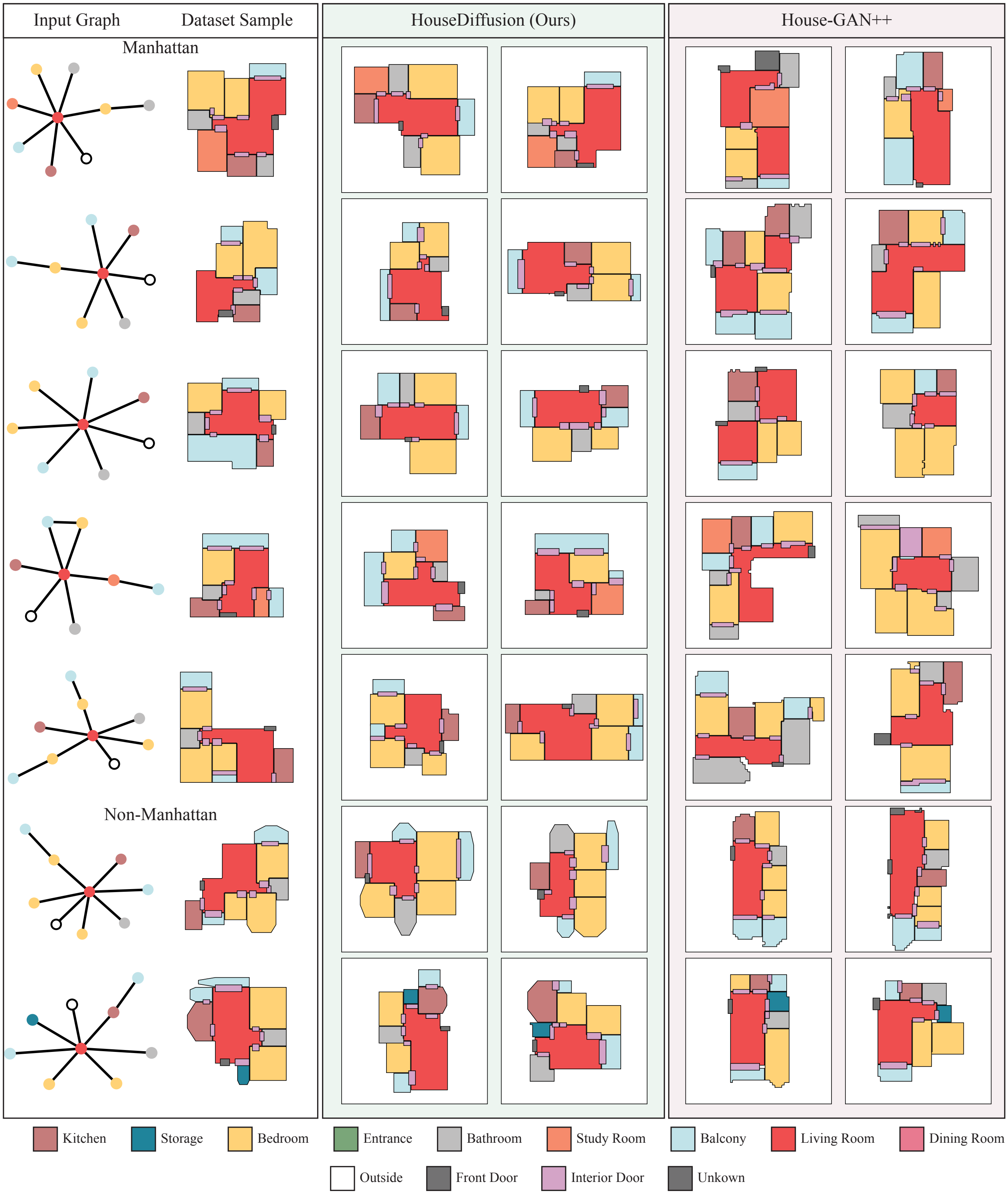


Figure 4. Generated floorplan samples against House-GAN++ [34]. See supplementary for more examples. Our results look more diverse and higher quality, where the major issue of House-GAN++ is duplicate or missing rooms, ignoring the input constraint.

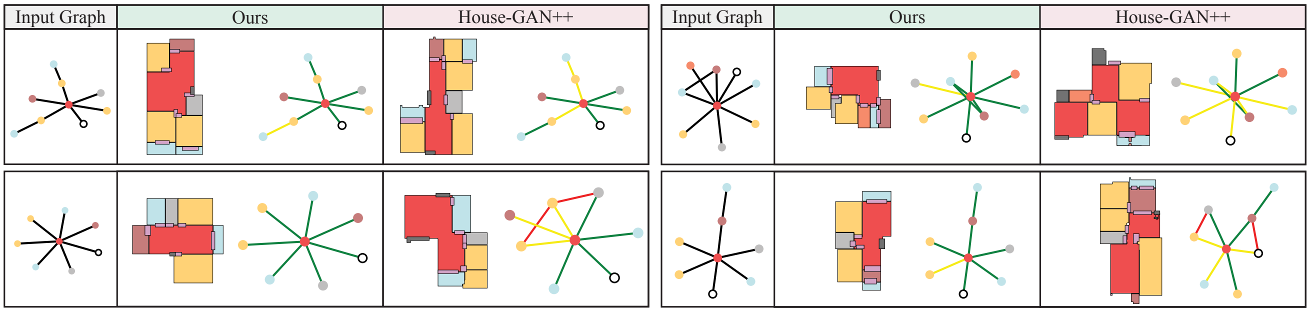


Figure 5. HouseDiffusion results are more compatible with the input bubble diagram by ensuring the generation of a single polygonal loop for each room/door. Green , Yellow , and Red indicate correct, missing, and extraneous connections, respectively.

generating floorplans in each group, we exclude samples in the group from the training so that methods cannot simply memorize samples. Following House-GAN++, the same three metrics are used for evaluations: Diversity, Compatibility, and Realism. Diversity is the Frechet Inception Distance (FID) [12]. Compatibility is the modified Graph Edit Distance [1] between the input bubble diagram and the one reconstructed from the generated floorplan. Realism is based on user studies (See Section 5.1 for more details).

To verify the capability of non-Manhattan floorplan generation, we create a new benchmark "Non-Manhattan-RPLAN" based on RPLAN by randomly adding two corners to an outer wall. See supplementary for more details.

5.1. Quantitative Evaluations

Table 1 shows the main quantitative evaluations. For existing methods, we copy the numbers reported in the House-GAN++ paper [34]. For the Realism, their paper shows the average score against all the other methods, while we use the score against just the ground-truth, because comparisons between weaker baselines do not provide useful information. Our system, HouseDiffusion, consistently outperforms all the previous methods in all the metrics. Compared to the current state-of-the-art House-GAN++ [34], HouseDiffusion makes an average improvement of 67% in diversity and 32% in compatibility. We also make significant improvements in realism, which we discuss next in detail. For non-manhattan RPLAN dataset, HouseDiffusion also makes significant improvements in diversity and compatibility, where House-GAN++ cannot handle non-manhattan structures. The realism score is not used, because the non-manhattan structures were added by heuristics and even the ground-truth may not look realistic.

We follow the same process as in House-GAN++ to obtain the realism scores. We generate 1000 floorplan samples by each system, present two samples (from different systems) to a participant, and ask to choose "A is better", "B is better", or "both are equal". A method "A" earns +1, -1, or 0 point with the above answers, respectively. Each pair of systems are evaluated 150 times by 10 participants (i.e., 15 times by each participant). Table 1 shows the average realism score of each system against the ground-truth. Table 2 shows the direct comparisons between House-GAN++, HouseDiffusion, and the ground-truth. Both tables show that HouseDiffusion achieves significant improvements with a score of -0.19 even against the ground-truth, which is very good. The number implies that participants choose our result to be as realistic as the Ground-Truth ("both are equal") for 81% of the time (assuming they either preferred Ground-Truth or chose equal). Note that House-GAN++ paper reports -0.18 for their best configuration against the ground-truth, but this variant (denoted as "Ours static*") requires expensive post-processing [7] and filters out incompatible samples based on the ground-truth bubble-diagram that account for 90% of the generated samples. Our results are direct outputs from our network architecture.

5.2. Qualitative Evaluations

Figure 4 qualitatively compares HouseDiffusion with House-GAN++ by two generated floorplans per bubble-diagram in both Manhattan and Non-Manhattan cases. Please refer to the supplementary for more examples and the supplementary video for the animations of the denoising process. HouseDiffusion consistently generates higher-quality samples. The major issue of House-GAN++ is that the system tends to miss or generate too many rooms, ignoring the input bubble-diagram constraint. For example, the top example in Fig. 4 should have one living-room (red), two bedrooms (yellow), and one study (orange), but House-GAN++ makes errors in every single sample. The issue is also highlighted in Fig. 5, which visualizes the bubble-diagrams of generated floorplan samples. HouseDiffusion guarantees exactly the correct set of rooms in the output and produces clean wall structures without artifacts (e.g., jagged boundaries) thanks to the vector representation.

5.3. Ablation Studies

We conduct a series of ablation studies to verify the effectiveness of our technical contributions and provide an in-depth analysis of our system.

Table 1. Main quantitative results, comparing HouseDiffusion (ours) with the previous methods on the three metrics. In all experiments, our method significantly outperform all the other methods.

		Realism (↑)	Diversity (↓)				Compatibility (↓)			
Dataset	Model	8	5	6	7	8	5	6	7	8
RPLAN	Ashual *et al.* [2]	-1.00	120.6±0.5	172.5±0.2	162.1±0.4	183.0±0.4	7.5±0.0	9.2±0.0	10.0±0.0	11.8±0.0
	Johnson *et al.* [18]	-1.00	167.2±0.3	168.4±0.4	186.0±0.4	186.0±0.4	7.7±0.0	6.5±0.0	10.2±0.0	11.3±0.1
	House-GAN [33]	-0.95	37.5±1.1	41.0±0.6	32.9±1.2	66.4±1.7	2.5±0.1	2.4±0.1	3.2±0.0	5.3±0.0
	House-GAN++ [34]	-0.52	30.4±4.4	37.6±3.0	27.3±4.9	32.9±4.9	1.9±0.3	2.2±0.3	2.4±0.3	3.9±0.5
	Ours	**-0.19**	**11.2**±0.2	**10.3**±0.2	**10.4**±0.4	**9.5**±0.1	**1.5**±0.0	**1.2**±0.0	**1.7**±0.0	**2.5**±0.0
NM-RPLAN	House-GAN++ [34]	—	77.3±0.8	60.0±0.7	73.8±0.8	58.2±1.0	1.5±0.0	2.9±0.0	2.1±0.0	3.2±0.0
	Ours	—	**12.0**±0.2	**11.0**±0.1	**10.3**±0.2	**10.5**±0.3	**1.2**±0.0	**1.3**±0.0	**1.6**±0.0	**2.5**±0.0

Table 2. Realism scores among House-GAN++, ours, and the ground-truth. Our method achieves 0.71 against House-GAN++, indicating that the participants choose ours to be more realistic 85% of the times (i.e., $0.71 \approx 0.85 - 0.15$).

	House-GAN++	Ours	Ground Truth
House-GAN++ [34]	—	-0.71	-0.87
Ours	0.71	—	-0.19
Ground Truth	0.87	0.19	—

Table 3. The effectiveness of our discrete and continuous denoising scheme, in comparison to an existing discrete only representation AnalogBits and our system without the discrete branch/loss that reduces to a standard denoising scheme.

	Cont.	Disc.	Divers. (↓)	Compat. (↓)
AnalogBits [8]		✓	14.5±0.3	6.0±0.03
Ours w/o disc.	✓		38.8±0.9	2.2±0.0
Ours	✓	✓	9.5±0.1	2.5±0.0

Discrete and continuous denoising: To verify the effectiveness of our discrete and continuous denoising scheme, we compare with a recent work AnalogBits [8], which employs a binary representation to generate discrete numeric values in a diffusion model framework. We also create HouseDiffusion only with continuous denoising by simply dropping the discrete branch/loss. Table 3 demonstrates that the proposed discrete and continuous denoising significantly improves the diversity with a small sacrifice on the compatibility. We hypothesize that AnalogBits merely learns the rounding with the binary representation alone when t is small, which reduces the generation quality.

Table 4. Quantitative evaluation of the three attention mechanisms. The second last row triples the number of GSA layers instead of the combination of CSA, GSA, and RCA.

CSA	GSA	RCA	Diversity (↓)	Compatibility (↓)
	✓	✓	9.9±0.1	2.9±0.0
✓		✓	12.8±0.2	4.0±0.0
✓	✓		11.4±0.2	6.8±0.0
	✓✓✓		10.8±0.2	6.5±0.0
✓	✓	✓	**9.5**±0.1	**2.5**±0.0

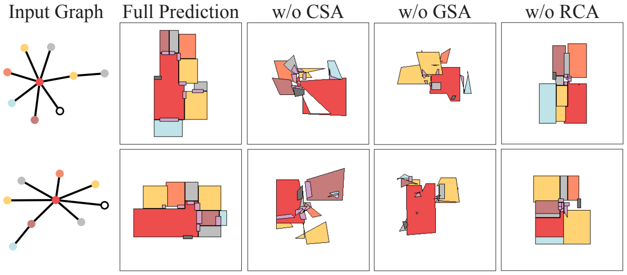


Figure 6. Qualitative evaluation of the three attention mechanisms. The shape quality degrades significantly in the absence of CSA or GSA. Door placements become corrupted in the absence of RCA.

Attention modules: HouseDiffusion employs three different types of attentions. Table 4 and Fig. 6 qualitatively and quantitatively measure their effectiveness by dropping the three attentions one by one. Global Self Attention (GSA) has the highest impact on realism (qualitatively in Fig. 6) and diversity as expected, because this accounts for the attentions between every pair of nodes. Relational Cross Attention (RCA) enforces connections between rooms and doors and has the most impact on the compatibility metric. Component-wise Self Attention (CSA) focuses on individual rooms and reveals a significant impact on the room shape quality in Fig. 6, causing self-intersections and "impossible" shapes without it. The second last row of Table 4 is a version with three times more GSA attention layers instead of the combination of CSA, GSA, and RCA, which demonstrates the importance of using all the attention types.

Geometry modification: Figure 7 demonstrates a new capability of HouseDiffusion, allowing us to specify the ex-

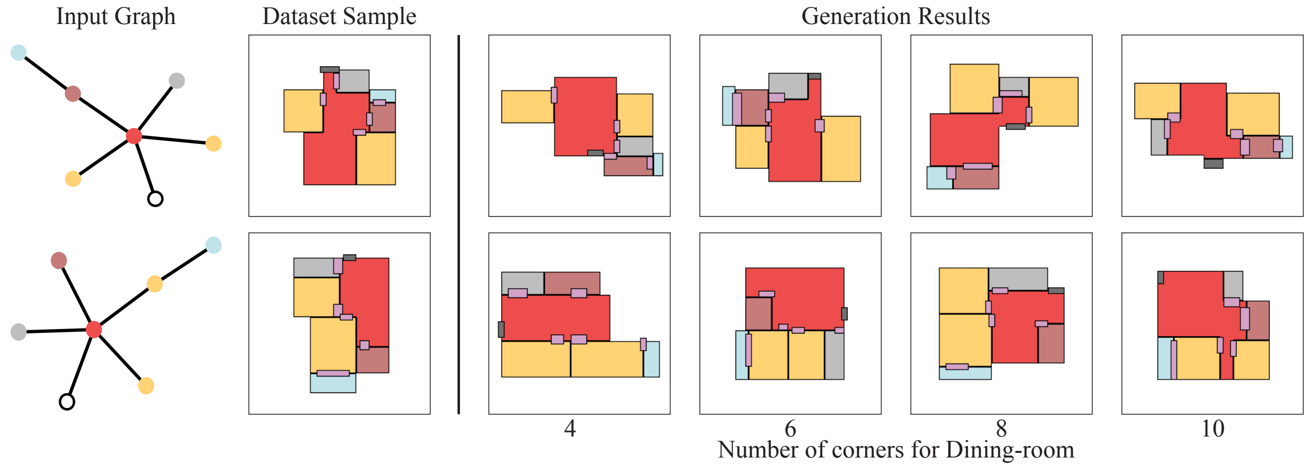


Figure 7. HouseDiffusion allows us to specify the number of corners per room. In this above examples, we increase the number of corners for the dining-room (red room), while keeping all the other rooms to have 4 corners.

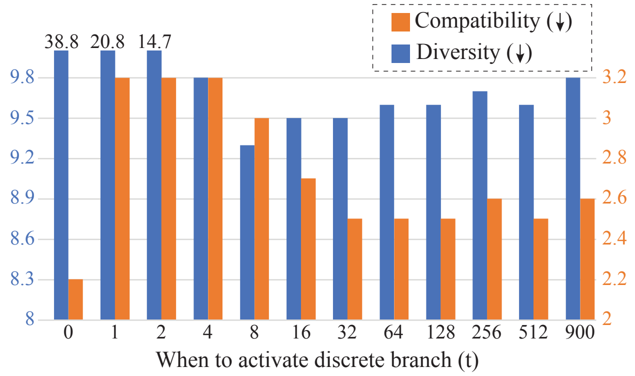


Figure 8. Using discrete head in a few steps gives less opportunity to the network to refine the discrete coordinates, while using it early can also lead to removing necessarily continuous informations. We fix discrete head to activate when $t < 32$.

act number of corners per room, which is not possible by a raster-based system such as House-GAN++. The figure shows that HouseDiffusion generates floorplans with increasingly more complex dining-room, where the surrounding rooms change their layouts to be consistent.

Discrete steps: Having both discrete and continuous denoising branches is essential for generating high-quality floorplans, while the discrete branch is active in the last 32 iterations at test time. We vary this hyperparameter and measure the performance change in Fig. 8. Limiting to fewer iterations reduces the opportunities of discrete analysis and harms performance. On the other hand, activating the discrete branch too early also degrades the performance, because the discrete analysis is on the final floorplan shape at time 0, whose inference is not accurate at early iterations. We found that 32 is a good number overall, which is used throughout our experiments.

Table 5. The effects of the corner augmentation in the feature embedding (3).

Domain	Diversity (↓)	Compatibility (↓)
Single Corner	10.0±0.3	3±0.0
Augmented Corner	9.5±0.1	2.5±0.0

Corner augmentation: Table 5 shows the corner augmentation effects in the feature embedding (3). While both the diversity and compatibility improve, the effect is higher for compatibility where the augmented points on the walls enable better analysis of geometric incident relationships.

6. Conclusion

This paper presents a novel floorplan generative model that directly generates vector-graphics floorplans. The approach uses a Diffusion Model with a Transformer network module at the core, which denoises 2D pixel coordinates both in discrete and continuous numeric representations. The discrete representation ensures precise geometric incident relationships among rooms and doors. The transformer module has three types of attentions that exploit the structural relationships of architectural components. Qualitative and quantitative evaluations demonstrate that the proposed system makes significant improvements over the current state-of-the-art with large margins in all the metrics, while boasting new capabilities such as the generation of non-Manhattan structures or the exact specification of the number of corners. Our future work is the handling of large-scale buildings. We will share all our code and models.

Acknowledgement: This research is partially supported by NSERC Discovery Grants with Accelerator Supplements and DND/NSERC Discovery Grant Supplement.

References

[1] Zeina Abu-Aisheh, Romain Raveaux, Jean-Yves Ramel, and Patrick Martineau. An exact graph edit distance algorithm for solving pattern recognition problems. In *4th International Conference on Pattern Recognition Applications and Methods 2015*, 2015. 6

[2] Oron Ashual and Lior Wolf. Specifying object attributes and relations in interactive scene generation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4561–4569, 2019. 4, 7

[3] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021. 2

[4] Sumit Bisht, Krishnendra Shekhawat, Nitant Upasani, Rahil N Jain, Riddhesh Jayesh Tiwaskar, and Chinmay Hebbar. Transforming an adjacency graph into dimensioned floorplan layouts. In *Computer Graphics Forum*. Wiley Online Library, 2022. 2

[5] Andrew Campbell, Joe Benton, Valentin De Bortoli, Tom Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. *arXiv preprint arXiv:2205.14987*, 2022. 2

[6] Hanqun Cao, Cheng Tan, Zhangyang Gao, Guangyong Chen, Pheng-Ann Heng, and Stan Z Li. A survey on generative diffusion model. *arXiv preprint arXiv:2209.02646*, 2022. 2

[7] Jiacheng Chen, Chen Liu, Jiaye Wu, and Yasutaka Furukawa. Floor-sp: Inverse cad for floorplans by sequential room-wise shortest path. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2661–2670, 2019. 1, 6

[8] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022. 2, 3, 7

[9] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021. 2, 4

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 2

[11] Mark Hendrikx, Sebastiaan Meijer, Joeri Van Der Velden, and Alexandru Iosup. Procedural content generation for games: A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 9(1):1–22, 2013. 2

[12] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 6

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 2, 3, 4

[14] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, 34:12454–12465, 2021. 2

[15] Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International Conference on Machine Learning*, pages 8867–8887. PMLR, 2022. 2

[16] Ruizhen Hu, Zeyu Huang, Yuhan Tang, Oliver Van Kaick, Hao Zhang, and Hui Huang. Graph2plan: Learning floorplan generation from layout graphs. *ACM Transactions on Graphics (TOG)*, 39(4):118–1, 2020. 2

[17] Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi Jaakkola. Torsional diffusion for molecular conformer generation. *arXiv preprint arXiv:2206.01729*, 2022. 2

[18] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228, 2018. 4, 7

[19] Heeseung Kim, Sungwon Kim, and Sungroh Yoon. Guided-tts: A diffusion model for text-to-speech via classifier guidance. In *International Conference on Machine Learning*, pages 11119–11133. PMLR, 2022. 2

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[21] Yuma Koizumi, Heiga Zen, Kohei Yatabe, Nanxin Chen, and Michiel Bacchiani. Specgrad: Diffusion probabilistic model based neural vocoder with adaptive noise spectral shaping. *arXiv preprint arXiv:2203.16749*, 2022. 2

[22] Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. Diffusion-LM improves controllable text generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2

[23] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017. 2

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018. 4

[25] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2837–2845, 2021. 2

[26] Shitong Luo and Wei Hu. Score-based point cloud denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4583–4592, 2021. 2

[27] Shitong Luo, Chence Shi, Minkai Xu, and Jian Tang. Predicting molecular conformation via dynamic graph score matching. *Advances in Neural Information Processing Systems*, 34:19784–19795, 2021. 2

[28] Ziniu Luo and Weixin Huang. Floorplangan: Vector residential floorplan adversarial generation. *Automation in Construction*, 142:104470, 2022. 2

[29] Zhaoyang Lyu, Zhifeng Kong, Xudong Xu, Liang Pan, and Dahua Lin. A conditional point diffusion-refinement paradigm for 3d point cloud completion. *arXiv preprint arXiv:2112.03530*, 2021. 2

[30] Chongyang Ma, Nicholas Vining, Sylvain Lefebvre, and Alla Sheffer. Game level layout from design specification. In *Computer Graphics Forum*, volume 33, pages 95–104. Wiley Online Library, 2014. 2

[31] Paul Merrell, Eric Schkufza, and Vladlen Koltun. Computer-generated residential building layouts. In *ACM SIGGRAPH Asia 2010 papers*, pages 1–12. 2010. 2

[32] Pascal Müller, Peter Wonka, Simon Haegler, Andreas Ulmer, and Luc Van Gool. Procedural modeling of buildings. In *ACM SIGGRAPH 2006 Papers*, pages 614–623. 2006. 2

[33] Nelson Nauata, Kai-Hung Chang, Chin-Yi Cheng, Greg Mori, and Yasutaka Furukawa. House-gan: Relational generative adversarial networks for graph-constrained house layout generation. In *European Conference on Computer Vision*, pages 162–177. Springer, 2020. 1, 2, 4, 7

[34] Nelson Nauata, Sepidehsadat Hosseini, Kai-Hung Chang, Hang Chu, Chin-Yi Cheng, and Yasutaka 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. 1, 2, 4, 5, 6, 7

[35] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 2, 3

[36] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016. 2

[37] Chi-Han Peng, Yong-Liang Yang, and Peter Wonka. Computing layouts with deformable templates. *ACM Transactions on Graphics (TOG)*, 33(4):1–11, 2014. 2

[38] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2

[39] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 2

[40] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022. 2

[41] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015. 2

[42] Jiahui Sun, Wenming Wu, Ligang Liu, Wenjie Min, Gaofeng Zhang, and Liping Zheng. Wallplan: synthesizing floorplans by learning to generate wall graphs. *ACM Transactions on Graphics (TOG)*, 41(4):1–14, 2022. 2

[43] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816, 2021. 2

[44] Abhinav Upadhyay, Alpana Dubey, Veenu Arora, Suma Mani Kuriakose, and Shaurya Agarawal. Flnet: graph constrained floor layout generation. In *2022 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pages 1–6. IEEE, 2022. 2

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3

[46] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European conference on computer vision (ECCV)*, pages 52–67, 2018. 2

[47] Wenming Wu, Xiao-Ming Fu, Rui Tang, Yuhan Wang, Yu-Hao Qi, and Ligang Liu. Data-driven interior plan generation for residential buildings. *ACM Transactions on Graphics (TOG)*, 38(6):1–12, 2019. 4

[48] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022. 2

[49] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *arXiv preprint arXiv:2209.00796*, 2022. 2

[50] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5826–5835, 2021. 2