LEGO-Net: Learning Regular Rearrangements of Objects in Rooms

Qiuhong Anna Wei[1] Sijie Ding*[1] Jeong Joon Park*[2] Rahul Sajnani[1]
Adrien Poulenard[2] Srinath Sridhar[1] Leonidas Guibas[2]

Brown University[1] Stanford University[2]

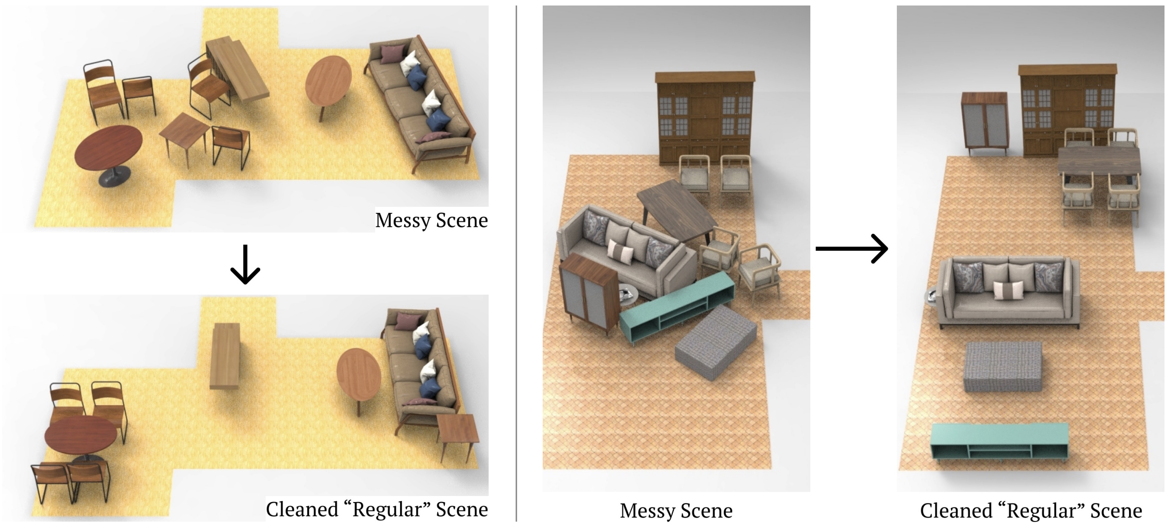


Figure 1. **LEGO-Net** **LE**arns to re**G**ularly rearrange **O**bjects in a messy indoor scene via an iterative denoising process. Different from scene synthesis or methods that require goal state specification, our method learns clean re-arrangements directly from data, retains the flavor of the original scene, and minimizes object travel distance.

Abstract

Humans universally dislike the task of cleaning up a messy room. If machines were to help us with this task, they must understand human criteria for regular arrangements, such as several types of symmetry, co-linearity or co-circularity, spacing uniformity in linear or circular patterns, and further inter-object relationships that relate to style and functionality. Previous approaches for this task relied on human input to explicitly specify goal state, or synthesized scenes from scratch – but such methods do not address the rearrangement of existing messy scenes without providing a goal state. In this paper, we present LEGO-Net, a data-driven transformer-based iterative method for ***LE****arning re****G****ular rearrangement of* ***O****bjects in messy rooms. LEGO-Net is partly inspired by diffusion models – it starts with an initial messy state and iteratively "de-noises" the position and orientation of objects to a regular state while reducing distance traveled. Given randomly perturbed object positions and orientations in an existing dataset of professionally-arranged scenes, our method is trained to recover a regular re-arrangement. Results demonstrate that our method is able to reliably rearrange room scenes and outperform other methods. We additionally propose a metric for evaluating regularity in room arrangements using number-theoretic machinery.*

1. Introduction

What makes the arrangement of furniture and objects in a room appear regular? While exact preferences may vary, humans have by-and-large universally shared criteria of regular room arrangements: for instance, heavy cabinets are arranged to align with walls, chairs are positioned evenly around a table in linear or circular configurations, or night stands are placed symmetrically on the two sides of a bed. Humans also share a common dislike of physically performing the task of rearranging a messy room. To build automated robotic systems that can guide or actually rearrange objects in a room, we first need methods that understand the shared human criteria for regular room rearrangements and respect the physical constraints of rearrangements.

*Core contribution.

Human criteria for regular rearrangements can be subtle and complex, including geometric rules of reflexional, translational, or rotational symmetry, linear or circular alignments, and spacing uniformity. Functional and stylistic inter-object relationships are also important: for example, a TV tends to be in front of and facing a sofa, chairs are next to a table, etc. Many of these criteria interact and, at times, conflict with one another. As a result, in general, there is more than one desirable clean arrangement for any given messy arrangement. In our setting, we further desire that the clean rearrangement we create to be informed by the initial messy arrangement – and not be entirely different – for multiple reasons. First, there may have been a particular clean arrangement that gave rise to the messy one – and it may be desirable to recover a similar arrangement. Second, we want to minimize the motion of objects as much as possible to respect the physical constraints and effort involved – especially the motion of big and heavy furniture. Unfortunately, extant methods fail to capture these criteria: methods for scene synthesis from scratch [25, 29, 37, 69, 70, 72] ignore the initial state of objects in a room, and rearrangement methods often require scene-specific human input in the form of a goal state [1, 46] or language description [27, 47].

In this paper, we present LEGO-Net, a method for **LE**arning re**G**ular rearrangement of **O**bjects in rooms directly from data. Different from work that focuses on arranging new objects from scratch or requires goal state specification, we focus on **rearranging** existing objects without any additional input at inference time. We take as input the position, orientation, class label, and extents of room objects in a specific arrangement, and output a room with the same objects but regularly re-arranged. LEGO-Net uses a transformer-based architecture [53] that is, in part, motivated by recent denoising diffusion probabilistic models that learn a reverse diffusion process for generative modeling [16, 49, 50]. We learn human criteria for regular rearrangements from a dataset of professionally designed *clean* (regular) scenes [15], and represent each scene as a collection of objects and a floor plan. Prior to training, we perturb the regular scenes to generate noisy configurations. During training, our transformer learns to predict the original, denoised arrangement from the perturbed scene and its floor plan. During inference, instead of directly re-arranging scenes with our model, which would amount to naïve regression, we run a Langevin dynamics-like reverse process to iteratively denoise object positions and orientations. This iterative process retains the flavor of original room state, while limiting object movement during re-arrangement.

We conduct extensive experiments on public datasets to show that our approach realistically rearranges noisy scene arrangements, while respecting initial object positions. We also demonstrate that our method is able to generalize to previously unseen collection of objects in a wide variety of floor plans. Furthermore, we include extensive experimental results (e.g., Fig. 1 and Fig. 4), including a new metric to evaluate regularity of re-arrangements, aimed at measuring the presence of sparse linear integer relationships among object positions in the final state (using the PSLQ algorithm [13]). To sum up, we contribute:

- A generalizable, data-driven method that learns to **regularly re-arrange** the position and orientation of objects in various kinds of messy rooms.
- An iterative approach to re-arrangement at inference time that retains flavor of the original arrangement and minimizes object travel distance.
- An in-depth analysis of the performance and characteristics of the denoising-based scene rearrangement.
- A new metric to measure the regularity of object arrangements based on integer relation algorithms.

2. Related Work

In this section, we discuss literature in two related areas: (1) scene synthesis from scratch, (2) scene rearrangement where an end goal is specified, and (3) diffusion models.

Indoor 3D Scene Synthesis: Indoor room synthesis is the problem of synthesizing the layout of objects in a scene from scratch. Many classical methods in the computer graphics literature use heuristics and guidelines to constrain the location of pre-specified objects [4, 62, 64, 67]. [31] identified a collection of functional, visual, and design constraints and formulated an optimization problem. Work has also focused exclusively on inter-object relationships [26]. Other methods [65] address the open world layout problem when objects are not pre-specified.

An alternative approach is to adopt procedural modeling using generative grammars [3, 5, 8, 35, 37, 52]. Some methods adopt the scenegraph representation and formulate it as a graph problem [25, 29, 33, 57, 69, 70, 72]. Both procedural and graph-based methods often rely on curated data [14]. Some methods learn directly from data using neural networks, for instance from images [39]. Both SceneFormer [60] and ATISS [34] introduce autoregressive methods for scene generation. Different from all these methods, we focus on *rearranging* rooms given an initial messy state.

Scene Rearrangement: Scene rearrangement takes an initial state of the scene and aims to bring it to a goal state specified by the user. This task is deeply connected to planning in robotics [1, 42, 48]. Some works consider robot pushing and manipulation for rearrangement [2, 6, 7, 11, 19, 20, 22, 23, 45, 46]. Many of these methods require datasets for training and often use datasets like AI2-THOR [21], Habitat [51] or Gibson [24]. Some of these methods operate on visual observations [19], while others assume fully-observed synthetic environments [6]. To specify the goal state, some recent methods use language input [27, 47] driven by large

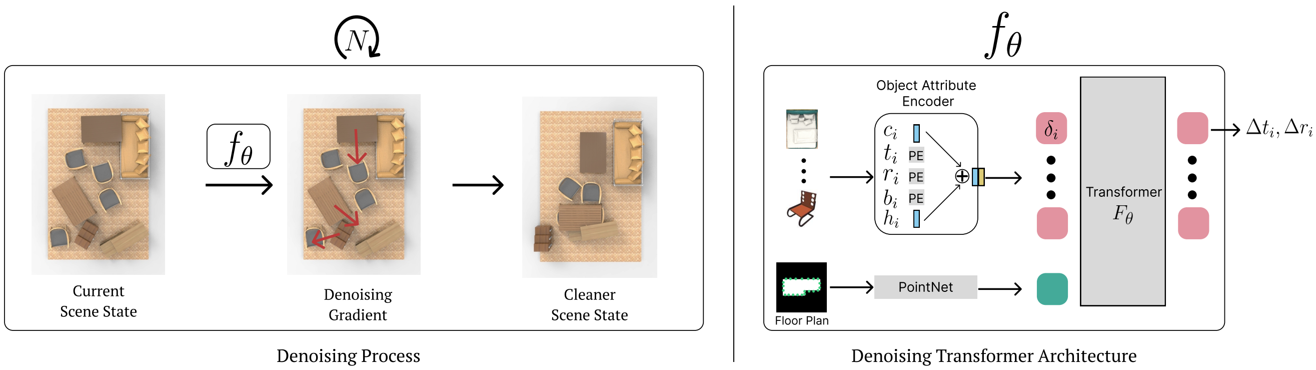


Figure 2. Pipeline overview. LEGO-Net takes an input messy scene and attempts to clean it via iterative denoising. Given the current scene state, it computes the denoising gradient towards the clean manifold, and changes the scene accordingly. This denoising step is repeated until the scene is "regular." On the right, we show our backbone transformer block f_θ that computes the denoising gradient at each step. It takes the scene attributes of the current state and outputs 2D transformations of each object that would make the scene "cleaner".

language models [9, 38]. Related to these advances in robotics, there have also been attempts to apply these specifically for room rearrangements [56, 61].

In this paper, we focus on the task of room rearrangements without the need to specify the goal state. We directly learn arrangements that satisfy human criteria from professionally arranged dataset provided by 3D-FRONT [15]. Note that a concurrent work [63] addresses the same problem but with a focus on physical simulation, incorporating reinforcement learning and path planning.

Denoising Diffusion Models: 2D Diffusion models [16, 49, 50] have emerged as a powerful technique for unconditional image synthesis, outperforming existing 2D generative models [12, 18]. Diffusion models have also seen great success in conditional image generations, receiving conditions in the form of class labels [10], text [32, 40], or input images [43]. Various methods [28, 30, 43] apply diffusion models for *restoring* corrupted or user-provided images to realistic images. Our method shares the same philosophy and adopts related techniques from the diffusion models, e.g., Langevin Dynamics, to project messy object configurations onto the manifold of "clean" scenes.

3. Method

3.1. Preliminaries

Our method takes the position, orientation, class label, and extents of objects in a 'messy' room as input and outputs a rearranged version in a 'regular' state. Since objects in rooms primarily move on the floor, we only consider 2D object pose, but our method can be combined with existing instance segmentation [59, 66] and canonicalization methods [44] to directly operate from a 3D mesh or point cloud. We represent each scene X as an unordered set of n objects and their attributes:

$$X = \{o_1, ..., o_n\}, \quad o_i = (c_i, t_i, r_i, b_i, h_i), \tag{1}$$

where $c_i \in \mathbb{R}^k$, $t_i \in \mathbb{R}^2$, $r_i \in SO(2)$, and $b_i \in \mathbb{R}^2$ respectively denote the semantic class, translation, rotation, and bounding box dimensions of the object o_i. $h_i \in \mathbb{R}^{128}$ is the pose-canonicalized shape features obtained by running ConDor [44] on each object's point cloud (see supplementary for details). The furniture semantic class labels c_i's are represented as one-hot vectors of the k classes. We represent the rotation $r_i \in SO(2)$ by the first column vector $[\cos(\theta), \sin(\theta)]^\intercal$ of its rotation matrix, following [71] to represent $SO(2)$ without any discontinuity. Note that t_i is normalized to be in $[-1, 1]$ to have the same range as r_i's sinusoidal representation to balance their importance during training. We define that a scene X^a is a rearrangement of X^b (denoted $X^a \sim X^b$) iff there exists a bijection ρ between object indices such that $h^a_i \approx h^b_{\rho(i)}$.

3.2. LEGO-Net: Learning Regular Room Rearrangements

Fig. 2 shows our approach to solving the regular room rearrangement task. Our method takes an input 'messy' scene $\tilde{X}$ and outputs a rearranged, 'regular' scene X. Towards this goal, we design a denoising Transformer [53] f_θ that is trained to predict a clean scene given its perturbed version. During inference, we take an iterative approach as it gives us rich control of the rearrangement process, *e.g.*, moving lighter objects farther. At each time step τ of the denoising process, we pass the current scene state $\tilde{X}_\tau$ to the denoising Transformer f_θ that provides gradients towards the manifold of 'clean' scenes. We repeat the denoising process until the magnitude of the predicted gradient is small enough to finally obtain a clean manifold projection of the input scene.

Manifold Projection via Denoising Autoencoder: We now describe our approach from a manifold learning perspective. We can consider the input to our method as an off-manifold point $\tilde{X}$ (*i.e.*, a messy scene) and aim to project it to the closest point X on the manifold of 'regular' scenes. Our objective is to learn a function $f_\theta(\tilde{X})$ (with network parameters θ) that finds such manifold projected point X, *i.e.*, $f_\theta(\tilde{X}) \approx X$. Motivated by the denoising autoencoders [55] and their recent extensions to score-matching models [16, 49], we train such $f(\tilde{X})$ by perturbing the regular data X employing a noise kernel $q_\sigma(\tilde{X}|X)$ with noise parameter σ. The training is done by minimizing a denoising objective function:

$$\mathcal{E}_{dn}(\theta) = \mathbb{E}_{q_\sigma(\tilde{X},X)}\left[\mathcal{L}_{\text{dn}}\left(f_\theta(\tilde{X}), X\right)\right]. \tag{2}$$

The joint distribution $q_\sigma(\tilde{X}, X) = n(\sigma)q_\sigma(\tilde{X}|X)q_0(X)$, where $n(\sigma)$ is the distribution of the noise parameter and $q_0(X)$ is the discrete uniform distribution of the training examples. Here, the loss $\mathcal{L}_{dn}$ is defined as the average distance between the pairs of objects in $f_\theta(\tilde{X})$ and X:

$$\begin{aligned}\mathcal{L}_{\text{dn}} = \frac{1}{n}\sum_{i=1}^{n} ||\tilde{t_i} - t_i||_2^2 + ||\tilde{r_i} - r_i||_2^2 \\ +\lambda_1(||\tilde{t_i} - t_i||_1 + ||\tilde{r_i} - r_i||_1),\end{aligned} \tag{3}$$

where $\tilde{t}_i$ and $\tilde{r}_i$'s are the object rotation and translation parameters of $f_\theta(\tilde{X})$, and λ_1 is a balancing parameter for the L1 loss. While we can use the object correspondences from the perturbation process, we choose to re-establish object pairing by computing Earth Mover's Distance [41] between the same class of objects in original and perturbed scenes.

Intuitively, the network $f_\theta(\tilde{X})$ learns to project $\tilde{X}$ to the clean manifold. However, it is not trained to find a random point in the clean manifold, but rather tries to find X that shares similarities with $\tilde{X}$, depending on the noise level.

Connection to Score-based Models: While the trained denoising network f_θ can theoretically be applied to clean a messy scene directly, in practice, the quality of the output is suboptimal, as shown in Sec. 4. This is because the network f_θ is trained with a *regression* loss, which is known to fit to the average state of the conditional distribution $q_\theta(X|\tilde{X})$, leading to blurry predictions [17, 68].

Recently, score-based generative models (and the closely related diffusion models) [16, 49] have shown impressive image generation results using a trained denoiser. The score-based approaches [49, 54] approximate the gradient of likelihood of the perturbed data distribution $q_\sigma(\tilde{X})$ with a neural network s_ϕ, and showed that the optimal network s^*_ϕ for the denoising objective $\mathbb{E}_{q_\sigma(\tilde{X},X)}\left[\left\|s_\phi(\tilde{X}) - \nabla_{\tilde{X}} \log q_\sigma(\tilde{X}|X)\right\|^2\right]$ satisfies $s^*_\phi(\tilde{X}) \approx \nabla_{\tilde{X}} \log q_\sigma(\tilde{X})$. Assuming a zero-mean Gaussian noise kernel $q_\sigma(\tilde{X}|X)$, the score-based network training objective becomes:

$$\mathcal{L}_{score}(\phi) = \mathbb{E}_{q_\sigma(\tilde{X},X)}\left[\left\|s_\phi(\tilde{X}) - \frac{X - \tilde{X}}{\sigma^2}\right\|^2\right]. \tag{4}$$

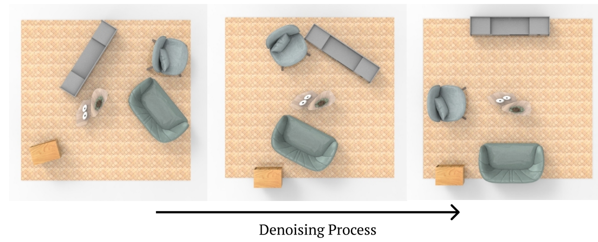


Figure 3. Instead of directly regressing the final rearranged state which can lead to non-diverse, suboptimal results, we adopt an iterative strategy based on Langevin Dynamics. At each step in our process (left to right), we gradually "de-noise" the scene until it reaches a regular state. During training, we follow the reverse process, *i.e.*, perturb clean scenes to messy state (right to left).

Since the trained score network s^*_ϕ approximates the gradient of the data distribution, it can be used for autoregressively optimizing noisy data onto the manifold of clean data. In our context, $X - \tilde{X}$ amounts to the difference in object transformations between the clean and perturbed scenes, and the direction $\frac{X-\tilde{X}}{\sigma^2}$ predicted by $s^*_\phi(\tilde{X})$ is clearly towards the clean scene manifold.

Rearrangement with Langevin Dynamics: After training with Eq. 2, we have a function that is optimized to approximate the denoised projection of an input, *i.e.*, $f^*_\theta(\tilde{X}) \approx X$. Given that $s^*_\phi(\tilde{X}) \approx \frac{X-\tilde{X}}{\sigma^2}$, we have that $s^*_\phi(\tilde{X}) \propto f^*_\theta(\tilde{X}) - \tilde{X}$. We then follow the score-based methods to adopt Langevin dynamics [16, 49] to recursively denoise the scene (see Fig. 3) using the estimated gradients:

$$\tilde{X}_{\tau+1} = \tilde{X}_\tau + \alpha(\tau)\left(f^*_\theta(\tilde{X}_\tau) - \tilde{X}_\tau\right) + \beta(\tau)z_\tau, \tag{5}$$

where $\alpha(\tau)$ and $\beta(\tau)$ are monotonically-decreasing functions of time τ that is heuristically designed to balance the Langevin dynamics and $z_\tau \sim \mathcal{N}(0, 1)$. We run the recursive computation until the magnitude of the gradient is small enough (*i.e.*, $\|f^*_\theta(\tilde{X}_{\tau_i}) - \tilde{X}_{\tau_i}\| < \kappa$, for constant κ) for k consecutive iterations for some constant k.

3.3. Architecture

LEGO-Net follows the recent success in the scene synthesis community to adopt the Transformer architecture [53] to represent our denoising function f_θ, as illustrated in Fig. 2. Given an input scene $\tilde{X}$, a Transformer encoder network F_θ takes in $|\tilde{X}| + 1$ number of 512-dimensional tokens δ_i's corresponding to the objects in the scene, as well as the room floor plan. Then, the network outputs absolute translation and rotation predictions for all object tokens (excluding the layout token), *i.e.*, $F_\theta : \mathbb{R}^{(|\tilde{X}|+1)\times 512} \mapsto \mathbb{R}^{|\tilde{X}|\times 4}$. We then apply the outputs to guide the translation and rotation of each object in $\tilde{X}$. The denoiser f_θ is defined to include both operations.

Therefore, the final processed scene is a rearrangement of the input scene: $f_\theta(\tilde{X}) \sim \tilde{X}$.

Input Object Attribute Encoding: We use the following process to abstract o_i into a token vector δ_i. We employ positional encodings of 32 frequencies, and an additional linear layer for r_i, to independently process t_i, r_i, b_i into vectors in $\mathbb{R}^{128}$. For object class c_i, we employ a 2-layer MLP with leaky ReLU activation to process the one-hot encoding into an attribute in $\mathbb{R}^{128}$. Finally, we optionally process a pose-invariant shape feature h_i from ConDor [44] with a 2-layer MLP to obtain a feature in $\mathbb{R}^{128}$. The above attribute features are then concatenated and processed with a 2-layer MLP to form an object token $\delta_i \in \mathbb{R}^{512}$. We refer readers to supplementary for the full details of the processing.

Floor Plan Encoder: Floor plans designating room boundaries both impose important realistic constraints and provide regularity information for the scene rearrangement task. Therefore, we pass the room layout in the form of an object token to the transformer so that other objects can attend to it. We employ a floor plan encoder to tokenize the floor plans as follows. We uniformly sample 250 points from the contour of the floor plan. These points along with their 2D surface normals are then processed with a simplified version of PointNet [36]. Finally, we specifically assign one bit of the 512 transformer input dimensions (for both objects and floor plans) to distinguish floor plan 'objects' from normal 'objects'. The final output of the floor plan encoder for each scene is a feature in $\mathbb{R}^{512}$.

Transformer Architecture: We use our custom positional encodings and procedures to prepare the tokens but use the original Transformer encoder architecture without notable modifications. We use 8 multi-headed attentions with 512-dimensional hidden layers and 512-dimensional key, query, and value vectors. The output of the transformer network is the estimated object transformations, a $|X| \times 4$ matrix.

3.4. Training and Inference

Data: We employ the 3D-FRONT dataset [15] for the task of indoor scene rearrangement. For each valid clean scene in the dataset, we preprocess it into $X = \{o_1, ..., o_n\}$ and extract the contour of its floor plan.

Training: We use the denoising auto-encoder formulation of Eq. 2 to train our denoiser function f_θ. We uniformly randomly sample training examples and sample a noise level σ from a normal distribution. The sampled examples are perturbed using an independent Gaussian kernel with standard deviation σ. For the perturbation, we do not consider objects going outside of the floor plans or colliding with one another. Each perturbed scene uses its original clean scene as the source of ground truth but re-establishes object correspondence through Earth Mover's Distance assignment to enable invariance among identical objects and further promote distance minimization in movement prediction. We use Adam optimizer with a learning rate of 10^{-4} to train.

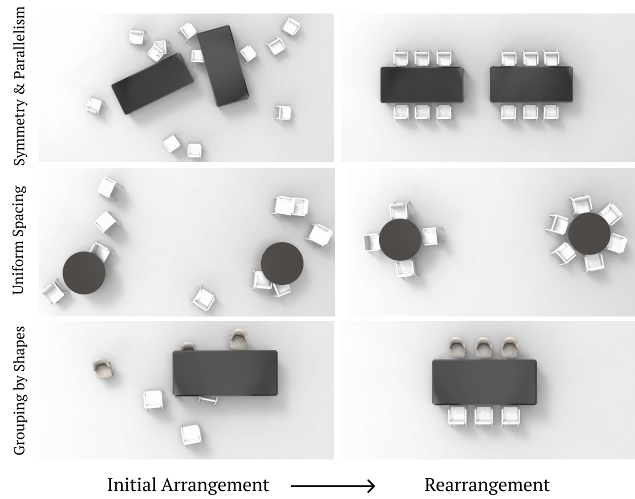


Figure 4. Regularities learning results. We train our denoising network to learn three different regularities. LEGO-Net successfully learns the complex regularity rules as demonstrated by the iterative denoising results shown on the right.

Inference: During inference, we use the Langevin Dynamics scheme of Eq. 5 to iteratively project a messy scene input $\tilde{X}$ onto the manifold of clean scenes. We select a hyperbolic function $\alpha(\tau) = \alpha_0/(1 + a_1 * \tau)$ to regulate the step size. We additionally select an exponential $\beta(\tau) = \beta_0 * b_1^{\lfloor \tau/b_2 \rfloor}$, where effectually β_0 is multiplied by b_1 every b_2 iterations, to adjust the level of noise as the denoising process proceeds. Refer to supplementary for details.

4. Experiments

We conduct a number of experiments to test LEGO-Net's ability to automatically capture scene regularities from data. To this end, we prepare two testbeds for experiments: our custom-designed **Table-Chair** environment where we mathematically constructed the regularities among objects and **3D-Front** [15], which contains tens of thousands of synthetic rooms designed by professionals. The Table-Chair dataset is useful because we can model one regularity at a time and quantify network performance. 3D-Front dataset exhibits complex and subtle rules that designers commonly perceive as ideal configurations, *e.g.*, geometry, semantic relations, styles, and functionalities.

4.1. Capturing Regularities from Data

In the Table-Chair environment, we study four main regularities: symmetry, parallelism, uniform spacing, and grouping by shapes. For each of the proposed experiments, we generate clean scenes based on the designed rules. Then, for training, we perturb the scenes on the fly to generate clean-messy pairs and re-associate objects within each class

Figure 5. Comparison against ATISS [34] on 3D-FRONT dataset. As the state-of-the-art scene synthesis method, ATISS is able to produce a realistic scene (3rd column) given the floor plan, but the generated objects and arrangements are entirely different. On the other hand, we solve the problem of re-arranging the given messy scene, directly using the existing objects. While ATISS has shown failure correction technique that solves similar problem to ours, we observe that when the scene is highly noisy, their algorithm tends to deteriorate significantly. Moreover, being a one-shot prediction method, ATISS-failure does not consider the moving distance of the new arrangement.

through Earth Mover's Distance assignment to train a network with the loss of Eq. 2. We measure each task with the success rate of the rearrangement, whose specific criteria we discuss in the supplementary.

Symmetry and Parallelism: One of the most important notions of regular arrangement is symmetry, which involves both object-object and room-level symmetries. We use a setup of 2 groups of rectangular tables and chairs. We vertically align the 2 tables and horizontally distribute them at a distance uniformly drawn from a fixed range. We arrange 3 chairs in a linear row on one side of the table and 3 chairs in another linear row on the opposite side.

Uniform Spacing: We prepare a highly-challenging setup to stress test LEGO-Net's ability to capture the concept of uniform spacing. In this setup, we have 2 circular tables, each with 2-6 chairs randomly and uniformly rotated around them. The network has to deal with the unknown number of chairs and the pair-wise spacing.

Grouping by Shape: We test LEGO-Net's ability to group objects based on their pose-invariant shapes. We augment our setup in 'Symmetry and Parallelism' to include 2 types of chairs with different shapes. We arrange the scenes such that chairs with the same shapes are on the same side of the table. The pose-invariant shape features $h_i \in o_i$ from Eq. (1) provide the necessary shape information.

Results: We visualize the rearrangement results of LEGO-Net for the above three cases in Fig. 4. Across the board, the denoising network successfully learns to capture these important regularities from data, without explicit supervision about the underlying rules. The success rate of each task is shown in Tab. 1. As expected, directly applying the regression-trained network f_θ results in the worst results.

4.2. 3D-Front Experiments

We benchmark LEGO-Net's ability to conduct regular scene rearrangements on the bedrooms and livingrooms of the 3D-FRONT dataset, which respectively contains 2338/587 and 5668/224 scenes for train/test splits.

	Symmetry & Parallelism ↑	Uniform Spacing↑	Grouping by Shape↑
Direct	16%	18.6%	23.6%
Grad. w/o noise	91.2%	96%	87.8%
Grad. w/ noise	**91.4%**	**97.2%**	**89.2%**

Table 1. Denoising success rate by regularities and inference strategies. For the three regularities shown in Fig. 4, we measure the success rate of LEGO-Net for each of the inference variants.

We train our LEGO-Net as described in Sec. 3 with $\sigma \sim \mathcal{N}(0, 0.1^2)$. While we maintain a single denoising network f_θ, we explore three variants of inference algorithms to provide greater insight of our approach: (1) LEGO-Net *direct*, (2) *grad. with noise*, and (3) *grad. w/o noise* respectively denote the inference strategy of predicting the clean outcome with one network pass, running Langevin Dynamics of Eq. (5) with noise term $\beta \neq 0$, and $\beta = 0$.

Baselines. We compare our rearrangement results against the current SOTA scene synthesis method, ATISS [34]. While ATISS is designed to synthesize a scene from scratch rather than to rearrange one, it provides an auto-regressive generative model that can be flexibly applied to our task. Specifically, we use three variants of ATISS that share the same network weights. First, ATISS *vanilla* performs its original scene synthesis task given a floor plan. Second, ATISS with *labels* performs object placement using a predefined set of objects per scene. Third, ATISS *failure-correction* takes a noisy scene and cleans it up by iteratively finding an object with low probability and re-placing it within the current scene. This variant of ATISS is given the same perturbed scene as LEGO-Net and aims to clean the scene. Note that we omit to compare against prior works that have already been compared against ATISS, e.g., [39, 57, 60]. We could not find a prior data-driven method that is designed to solve the same rearrangement problem as ours.

		Living Room				Bedroom			
		KID↓	FID↓	Distance↓ Moved	EMD ↓ to GT	KID↓	FID↓	Distance↓ Moved	EMD ↓ to GT
ATISS [34]	*vanilla*	96	44.55	—	—	33	50.49	—	—
	labels	119	45.45	—	0.3758	49	52.62	—	0.5482
	failure-correction	280	61.55	0.1473	0.3378	240	73.95	0.2025	0.4673
LEGO-NET (ours)	grad. w/ noise	67	39.19	0.091	0.125	37	49.76	0.052	0.086
	grad. w/o noise	**51**	**37.47**	**0.086**	**0.117**	**27**	**48.43**	**0.0492**	**0.0815**

Table 2. Quantitative experiment results using KID $\times 10,000$, FID, distance moved, and Earth Mover's Distance (EMD) against the ground truth arrangements. All scenes are situated within $[-1, 1]^2$ canvas. Note that ATISS *vanilla* and ATISS *labels* start from empty floor plans and thus the distance moved metric is not applicable. ATISS *failure-correction* takes a noisy scene and iteratively resamples low-probable objects, and is thus directly comparable to our method.

Metrics. To gauge how well LEGO-Net captures datasets' regularities, we adopt the popular FID and KID scores. These metrics compare the closeness of statistics of two data distributions. We follow prior works [34, 58] to render ground truth and generated scene arrangements from top-down views and compute the metrics in the image space. Note that KID is more applicable to our setting because FID is known to present huge bias when the number of data is low. Another important criterion for our rearrangement task is how much distance the objects travel between the initial and final scene states. Similarly, when applicable, we measure the Earth Mover's Distance (EMD) between the ground truth scene and our cleaned-up scene.

Finally, we introduce a new metric that measures scene regularities by finding integer relations among object positional coordinates t_i's. To do this, we select two or three random objects within a scene and check if we can find integral a_i's that satisfy:

$$a_1 t_1 + ... + a_n t_n = 0, \quad 0 < |a_i| < \eta \,, \forall a_i \qquad (6)$$

where η sets the maximum magnitude of the coefficients. Intuitively, these integer relations can capture regularities such as colinearities ($-t_1 + t_2 = 0$) and symmetries ($t_1 - 2t_2 + t_3 = 0$). See supplementary for detailed descriptions.

Results. We conduct the 3D-FRONT arrangement experiments with five algorithms (three ATISS variants and two of ours) and compute their metrics. The main numerical results, which can be found in Tab. 2, show that LEGO-Net outperforms all variants of ATISS, including the failure-correction variant that tackles the same object cleaning problem as demonstrated in the original paper.

In Fig. 6, we plot the chance of finding integer relations in scenes perturbed with different noise levels, which peaks for the original clean 3D-FRONT scenes and sharply decreases as noise is added. Also, note that the rearranged scenes of LEGO-Net demonstrate high regularities according to this measure, outperforming the results of ATISS variants. See supplementary for more experiment details.

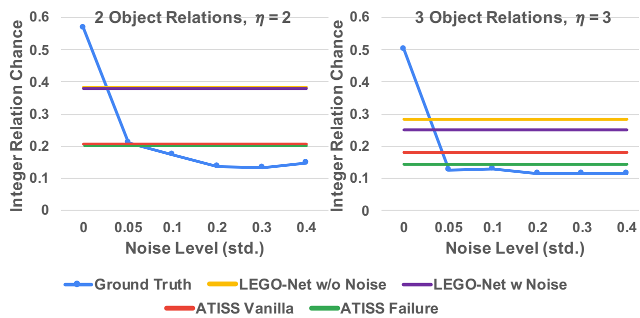


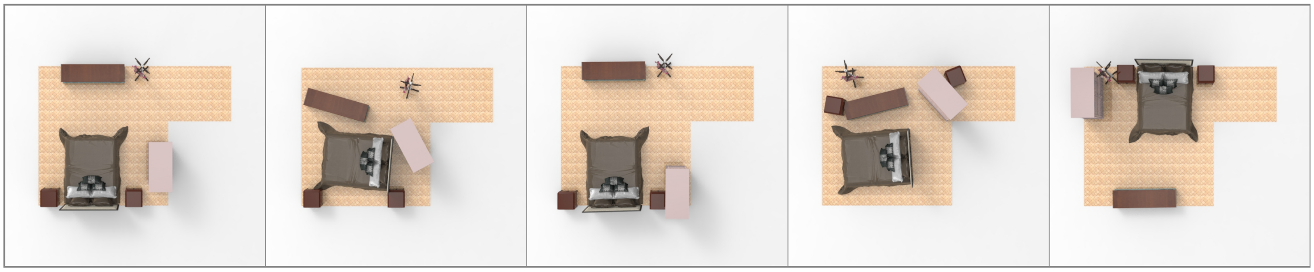
Figure 6. Integer relation occurences. We measure the chance of finding integer relations between the coordinates of two (left) and three (right) objects within a Living Room scene. Perturbing ground truth scenes sharply decreases the integer relation occurences, showing they are useful metric of regularities. Note that LEGO-Net outperforms ATISS variants in this metric.

Qualitatively, as shown in Fig. 5, LEGO-Net is able to robustly project messy scenes onto clean manifolds. While ATISS and ATISS with *labels* were able to synthesize realistic rooms, their object arrangements are entirely different from the original input scene. Importantly, we notice that ATISS *failure correction* leads to unexpectedly low-quality results. We hypothesize that this is due to their discrete, one-object-at-a-time strategy, which can easily fall into the local minimum of the likelihood space. In contrast, our score-based iterative denoising leads to robust success rates.

4.3. Analysis

To more deeply understand the behavior of our system, we analyze and discuss important aspects of LEGO-Net. We refer to supplementary for more analysis of our method.

Denoising Strategy. As we discuss throughout Sec. 4, we explore three inference strategies, namely direct, gradient with noise, and gradient without noise. For the 3D-FRONT experiment, we report that the *grad. without noise* variant consistently outperforms the other variants. However, we believe that this is likely because we used relatively



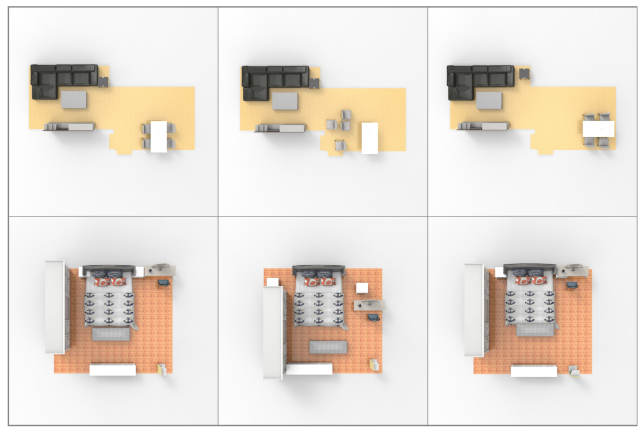
Ground Truth | Low Noise Input | Low Noise Cleaned | High Noise Input | High Noise Cleaned

Figure 7. LEGO-Net denoising results on different noise levels. When the perturbation added to the scene is low, LEGO-Net is able to closely reconstruct the clean version of the scene. In contrast, when the noise level is high, our denoising process finds a different realization of a regular scene, behaving more like an unconditional model. Similar phonemena have been observed by 2D diffusion projects, e.g., SDEdit [30].

low noise to the scenes (std 0.1) to more naturally simulate messy indoor rooms. Indeed, our experiment on the synthetic environments (Tab. 1) with larger noise (std 0.25) shows that the *grad. with noise* variant outperforms. The results suggest that adding noise during Langevin dynamics allows a better success rate for highly noisy data, but at the cost of losing accuracy in recovering the originals (as shown in Tab. 2).

Cleaning Uncertainty. LEGO-Net is trained to handle input perturbations at various noise levels. In the high-noise regime, there is high uncertainty on the structure of the original information. As input noise increases, our denoising process converges into an unconditional generative model. On the other hand, LEGO-Net has the capacity to capture original regularities when the noise is low, leading to almost precise reconstruction of the original scenes. We visually show these insights in Fig. 7.

Out-of-Distribution Inputs. We showcase LEGO-Net's ability to handle scenes perturbed with noise patterns significantly different from the one used in training, *i.e.*, zero-mean Gaussian. In the first example, we only perturb chairs. Secondly, we perturb the scene only in the translation dimensions without rotations. Shown in Fig. 8, LEGO-Net can successfully handle out-of-distribution inputs, demonstrating the robustness and versatility of our algorithm.



Ground Truth | O.O.D. Perturbed | Our Denoised

Figure 8. Out-of-distribution test. While our model is trained to denoise Gaussian noise, it demonstrates strong robustness to out-of-distribution inputs. In the first row, only the chairs are perturbed. In the second row, we perturb the scene with translation noise only. Zoom-in for details.

5. Conclusion

In this paper, we presented LEGO-Net, a method for regular rearrangement of objects in a room. Different from previous methods, LEGO-Net learns human notions of regularity (including symmetry, alignments, uniform spacing, and stylistic and functional factors) directly from data without the need to explicitly specify a goal state. During training, we learn from a large dataset of professionally-designed room layouts that are randomly perturbed. During inference, we follow a Langevin Dynamics-like strategy to iteratively "denoise" the scene. Quantitative results including comparisons and ablations show that our method performs well, which qualitative results confirm.

Limitations & Future Work: Our method has important limitations that provide extensive opportunities for future work. First, our method is currently limited to 2D room rearrangement and cannot perform 3D rearrangement, for instance in kitchen shelves. However, we do incorporate 3D shape features which can be used to extend our method to 3D. We also currently do not handle interpenetration of objects during denoising, which future work should explore.

Acknowledgements

This work was supported by AFOSR grant FA9550-21-1-0214, NSF CloudBank, an AWS Cloud Credits award, ARL grant W911NF-21-2-0104, a Vannevar Bush Faculty Fellowship, and a gift from the Adobe Corporation. We thank Kai Wang, Daniel Ritchie, Rao Fu, and Selene Lee.

References

[1] Dhruv Batra, Angel X Chang, Sonia Chernova, Andrew J Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, et al. Rearrangement: A challenge for embodied ai. *arXiv preprint arXiv:2011.01975*, 2020. 2

[2] Ohad Ben-Shahar and Ehud Rivlin. Practical pushing planning for rearrangement tasks. *IEEE Transactions on Robotics and Automation*, 14(4):549–565, 1998. 2

[3] Martin Bokeloh, Michael Wand, Hans-Peter Seidel, and Vladlen Koltun. An algebraic model for parameterized shape editing. *ACM Transactions on Graphics (TOG)*, 31(4):1–10, 2012. 2

[4] Richard W Bukowski and Carlo H Séquin. Object associations: a simple and practical approach to virtual 3d manipulation. In *Proceedings of the 1995 symposium on Interactive 3D graphics*, pages 131–ff, 1995. 2

[5] Siddhartha Chaudhuri, Daniel Ritchie, Jiajun Wu, Kai Xu, and Hao Zhang. Learning generative models of 3d structures. In *Computer Graphics Forum*, volume 39, pages 643–666. Wiley Online Library, 2020. 2

[6] Akansel Cosgun, Tucker Hermans, Victor Emeli, and Mike Stilman. Push planning for object placement on cluttered table surfaces. In *2011 IEEE/RSJ international conference on intelligent robots and systems*, pages 4627–4632. IEEE, 2011. 2

[7] Michael Danielczuk, Andrey Kurenkov, Ashwin Balakrishna, Matthew Matl, David Wang, Roberto Martín-Martín, Animesh Garg, Silvio Savarese, and Ken Goldberg. Mechanical search: Multi-step retrieval of a target object occluded by clutter. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1614–1621. IEEE, 2019. 2

[8] Jeevan Devaranjan, Amlan Kar, and Sanja Fidler. Metasim2: Unsupervised learning of scene structure for synthetic data generation. In *European Conference on Computer Vision*, pages 715–733. Springer, 2020. 2

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 3

[10] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021. 3

[11] Mehmet R Dogar, Michael C Koval, Abhijeet Tallavajhula, and Siddhartha S Srinivasa. Object search by manipulation. *Autonomous Robots*, 36(1):153–167, 2014. 2

[12] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021. 3

[13] Helaman R. P. Ferguson and David H. Bailey. A polynomial time, numerically stable integer relation algorithm. Number RNR-91-032, 1991. 2

[14] Matthew Fisher, Daniel Ritchie, Manolis Savva, Thomas Funkhouser, and Pat Hanrahan. Example-based synthesis of 3d object arrangements. *ACM Transactions on Graphics (TOG)*, 31(6):1–11, 2012. 2

[15] Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang 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. 2, 3, 5

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

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 4

[18] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33:12104–12114, 2020. 3

[19] Jennifer E King, Marco Cognetti, and Siddhartha S Srinivasa. Rearrangement planning using object-centric and robot-centric action spaces. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3940–3947. IEEE, 2016. 2

[20] Jennifer E King, Vinitha Ranganeni, and Siddhartha S Srinivasa. Unobservable monte carlo planning for nonprehensile rearrangement tasks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4681–4688. IEEE, 2017. 2

[21] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017. 2

[22] Athanasios Krontiris, Rahul Shome, Andrew Dobson, Andrew Kimmel, and Kostas Bekris. Rearranging similar objects with a manipulator using pebble graphs. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 1081–1087. IEEE, 2014. 2

[23] Yann Labbé, Sergey Zagoruyko, Igor Kalevatykh, Ivan Laptev, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Monte-carlo tree search for efficient visually guided rearrangement planning. *IEEE Robotics and Automation Letters*, 5(2):3715–3722, 2020. 2

[24] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, et al. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. *arXiv preprint arXiv:2108.03272*, 2021. 2

[25] Manyi Li, Akshay Gadi Patil, Kai Xu, Siddhartha Chaudhuri, Owais Khan, Ariel Shamir, Changhe Tu, Baoquan Chen, Daniel Cohen-Or, and Hao Zhang. Grains: Generative recursive autoencoders for indoor scenes. *ACM Transactions on Graphics (TOG)*, 38(2):1–16, 2019. 2

[26] Qi Li, Kaichun Mo, Yanchao Yang, Hang Zhao, and Leonidas Guibas. Ifr-explore: Learning inter-object functional relationships in 3d indoor scenes. *arXiv preprint arXiv:2112.05298*, 2021. 2

[27] Weiyu Liu, Chris Paxton, Tucker Hermans, and Dieter Fox. Structformer: Learning spatial structure for language-guided semantic rearrangement of novel objects. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6322–6329. IEEE, 2022. 2

[28] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022. 3

[29] Andrew Luo, Zhoutong Zhang, Jiajun Wu, and Joshua 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. 2

[30] Chenlin Meng, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 3, 8

[31] Paul Merrell, Eric Schkufza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. Interactive furniture layout using interior design guidelines. *ACM transactions on graphics (TOG)*, 30(4):1–10, 2011. 2

[32] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 3

[33] Wamiq Para, Paul Guerrero, Tom Kelly, Leonidas J Guibas, and Peter Wonka. Generative layout modeling using constraint graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6690–6700, 2021. 2

[34] Despoina Paschalidou, Amlan Kar, Maria Shugrina, Karsten Kreis, Andreas Geiger, and Sanja Fidler. Atiss: Autoregressive transformers for indoor scene synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 6, 7

[35] Pulak Purkait, Christopher Zach, and Ian Reid. Sg-vae: Scene grammar variational autoencoder to generate new indoor scenes. In *European Conference on Computer Vision*, pages 155–171. Springer, 2020. 2

[36] 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. 5

[37] Siyuan Qi, Yixin Zhu, Siyuan Huang, Chenfanfu Jiang, and Song-Chun Zhu. Human-centric indoor scene synthesis using stochastic grammar. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5899–5908, 2018. 2

[38] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 3

[39] Daniel Ritchie, Kai Wang, and Yu-an 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. 2, 6

[40] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn 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. 3

[41] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000. 4

[42] Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010. 2

[43] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–10, 2022. 3

[44] Rahul Sajnani, Adrien Poulenard, Jivitesh Jain, Radhika Dua, Leonidas J Guibas, and Srinath Sridhar. Condor: Self-supervised canonicalization of 3d pose for partial shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16969–16979, 2022. 3, 5

[45] Jonathan Scholz and Mike Stilman. Combining motion planning and optimization for flexible robot manipulation. In *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pages 80–85. IEEE, 2010. 2

[46] Rahul Shome and Kostas E Bekris. Synchronized multi-arm rearrangement guided by mode graphs with capacity constraints. In *International Workshop on the Algorithmic Foundations of Robotics*, pages 243–260. Springer, 2020. 2

[47] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022. 2

[48] Herbert A Simon and Allen Newell. Computer simulation of human thinking and problem solving. *Monographs of the Society for Research in Child Development*, pages 137–150, 1962. 2

[49] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019. 2, 3, 4

[50] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 2, 3

[51] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in Neural Information Processing Systems*, 34:251–266, 2021. 2

[52] Jerry O Talton, Yu Lou, Steve Lesser, Jared Duke, Radomír Měch, and Vladlen Koltun. Metropolis procedural modeling. *ACM Transactions on Graphics (TOG)*, 30(2):1–14, 2011. 2

[53] 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. 2, 3, 4

[54] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011. 4

[55] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008. 4

[56] Hanqing Wang, Wei Liang, and Lap-Fai Yu. Scene mover: Automatic move planning for scene arrangement by deep reinforcement learning. *ACM Transactions on Graphics*, 39(6), 2020. 3

[57] Kai Wang, Yu-An Lin, Ben Weissmann, Manolis Savva, Angel X Chang, and Daniel Ritchie. Planit: Planning and instantiating indoor scenes with relation graph and spatial prior networks. *ACM Transactions on Graphics (TOG)*, 38(4):1–15, 2019. 2, 6

[58] Kai Wang, Manolis Savva, Angel X Chang, and Daniel Ritchie. Deep convolutional priors for indoor scene synthesis. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 7

[59] Weiyue Wang, Ronald Yu, Qiangui Huang, and Ulrich Neumann. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2569–2578, 2018. 3

[60] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. Sceneformer: Indoor scene generation with transformers. In *2021 International Conference on 3D Vision (3DV)*, pages 106–115. IEEE, 2021. 2, 6

[61] Luca Weihs, Matt Deitke, Aniruddha Kembhavi, and Roozbeh Mottaghi. Visual room rearrangement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5922–5931, 2021. 3

[62] Tomer Weiss, Alan Litteneker, Noah Duncan, Masaki Nakada, Chenfanfu Jiang, Lap-Fai Yu, and Demetri Terzopoulos. Fast and scalable position-based layout synthesis. *IEEE Transactions on Visualization and Computer Graphics*, 25(12):3231–3243, 2018. 2

[63] Mingdong Wu, Fangwei Zhong, Yulong Xia, and Hao Dong. Targf: Learning target gradient field for object rearrangement. *arXiv preprint arXiv:2209.00853*, 2022. 3

[64] Ken Xu, James Stewart, and Eugene Fiume. Constraint-based automatic placement for scene composition. In *Graphics Interface*, volume 2, pages 25–34, 2002. 2

[65] Yi-Ting Yeh, Lingfeng Yang, Matthew Watson, Noah D Goodman, and Pat Hanrahan. Synthesizing open worlds with constraints using locally annealed reversible jump mcmc. *ACM Transactions on Graphics (TOG)*, 31(4):1–11, 2012. 2

[66] Li Yi, Wang Zhao, He Wang, Minhyuk Sung, and Leonidas J Guibas. Gspn: Generative shape proposal network for 3d instance segmentation in point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3947–3956, 2019. 3

[67] Lap Fai Yu, Sai Kit Yeung, Chi Keung Tang, Demetri Terzopoulos, Tony F Chan, and Stanley 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. 2

[68] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016. 4

[69] Song-Hai Zhang, Shao-Kui Zhang, Wei-Yu Xie, Cheng-Yang Luo, and Hong-Bo Fu. Fast 3d indoor scene synthesis with discrete and exact layout pattern extraction. *arXiv preprint arXiv:2002.00328*, 2020. 2

[70] Zaiwei Zhang, Zhenpei Yang, Chongyang Ma, Linjie Luo, Alexander Huth, Etienne Vouga, and Qixing Huang. Deep generative modeling for scene synthesis via hybrid representations. *ACM Transactions on Graphics (TOG)*, 39(2):1–21, 2020. 2

[71] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019. 3

[72] Yang Zhou, Zachary While, and Evangelos 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. 2