LEGO-Net: Learning Regular Rearrangements of Objects in Rooms
Supplementary Document

A. Overview

This supplementary document contains extended technical details, along with qualitative and quantitative results that supplement the main document. After introducing our video results, we cover details of our network architectures and their applications (Sec C). We then provide detailed explanations of our two main experiment setups: Table-Chair (Sec. D) and 3D-FRONT (Sec. E). Next, we conduct additional analysis experiments in Sec. F. Finally, we discuss failure modes (Sec. G) and future directions (Sec. H).

B. Video Results

In order to better visualize the 3D structures of the outputs and the denoising process, we provide videos of these processes in the format of an HTML website. We highly encourage the viewers to open the file “LEGO.html” and watch the videos for a direct view of the denoising process.

C. Architecture Details

C.1. Input Object Attribute Encoding

For each object attribute $o_i = (c_i, t_i, r_i, b_i, h_i)$, we process it into an object token $\delta_i \in \mathbb{R}^{512}$ to input into the transformer. The details are as follows.

We embed the translation t_i, rotation $r_i = [\cos(\theta_i), \sin(\theta_i)]^T$, and bounding box dimension b_i with a sinusoidal positional encoding of 32 frequencies. The frequencies are a geometric sequence with initial term 1 and common ratio $128^{\frac{1}{31}}$, which gives an ending term of 128. The positional encoding is therefore

$$PE(x) = \{\sin(128^{\frac{j}{31}}x), \cos(128^{\frac{j}{31}}x) \mid 0 \leq j \leq 31\} \in \mathbb{R}^{64}.$$

Applying PE to $t_i = [t_{i,x}, t_{i,y}]$ and $b_i = [b_{i,x}, b_{i,y}]$ gives 128-dimensional embeddings, whereas applying it to θ_i gives a 64-dimensional embedding. We then additionally process $PE(\theta_i)$ with a linear layer mapping to $\mathbb{R}^{128}$.

As mentioned, for object class c_i, we utilize a 2-layer MLP with leaky ReLU activation to process the one-hot encoding into a 128-dimensional attribute. The above four features are concatenated to form a 512-dimensional vector.

We also optionally process a pose-invariant shape feature h_i from ConDor [11]. More specifically, we pretrain ConDor on the 3D point clouds provided by 3D-FRONT [4], and extract the product of the Tensor Field Network layer output and spherical harmonics coefficients from ConDor to provide pose-invariant features in $\mathbb{R}^{1024\times128}$ capturing the shape of each object. Taking the mean across the 1024 points gives a 128-dimensional feature, which we then pass through a 2-layer MLP with leaky ReLU activation to obtain the final shape feature in $\mathbb{R}^{128}$. We apply the shape feature h_i in the *Grouping by Shapes* experiment to demonstrate its effectiveness.

Finally, the concatenated attributes for each object lie in $\mathbb{R}^{640}$ and are processed through 2 linear layers with leaky ReLU activation to produce a 512-dimensional object token for the transformer. Note that if floor plan is utilized, we modify the final layer to produce a 511-dimensional feature, and utilize the last bit to distinguish object tokens from floor plan tokens.

C.2. Floor Plan Encoder Architecture

We represent each floor plan as 250 randomly sampled contour points. We represent each point through its 2D position coordinate and the 2D normal of the line it is on. In aggregate, we represent each floor plan with a feature in $\mathbb{R}^{250\times4}$.

We employ a simplified PointNet [10] as the floor plan encoder to extract one unified floor plan feature from this representation. The encoder first processes the feature with 3 linear layers and Leaky ReLU activation, mapping the feature dimension through $[4, 64, 64, 512]$. Then, we max pool the resulting embedding in $\mathbb{R}^{250\times512}$ to obtain a global floor-plan encoding in $\mathbb{R}^{512}$. We then pass this global representation through one final linear layer and append a binary bit distinguishing it from object tokens. Finally, we combine this floor plan token with the $|\tilde{X}|$ number of 512-dimensional object tokens and pass the resulting $(|\tilde{X}| + 1) \times 512$ input matrix to the transformer.

C.3. Output Layers

The backbone transformer outputs $(|\tilde{X}| + 1) \times 512$ feature tokens. We ignore the floor plan token and use the $|\tilde{X}|$

object tokens to predict denoising transformations. We apply 2 linear layers with leaky ReLU activation to the output tokens to map to 256 dimensions then 4 dimensions, and finally obtain the absolute transformation predictions.

C.4. Inference Langevin Dynamics Parameters

During inference, we use the Langevin Dynamics scheme to iteratively denoise a messy scene input. As mentioned, for time step τ, we select $\alpha(\tau) = \alpha_0/(1 + a_1 * \tau)$ to regulate the step size and $\beta(\tau) = \beta_0 * b_1^{\lfloor \tau/b_2 \rfloor}$ to regulate the noise added at each iteration. We empirically select $a_1 = 0.005$ and $b_1 = 0.9$. For the living room, we adopt $\alpha_0 = 0.1$, $\beta_0 = 0.01$, and $b_2 = 10$. For bedroom, we adopt $\alpha_0 = 0.08$, $\beta_0 = 0.008$, and $b_2 = 8$.

We break from the iterative denoising process upon any one of two conditions: (1) if for 3 consecutive iterations, both the predicted translation displacement vector has Frobenius norm less than 0.01 and the predicted rotation angle displacement is less than 0.005 radians, or (2) we have reached 1500 iterations.

D. Table-Chair Experiment

D.1. Data Generation

To analyze the regularities our model can capture, we propose three synthetic Table-Chair experiment settings, with a focus on Symmetry and Parallelism, Uniform Spacing, and Grouping by Shapes respectively.

For each of the proposed experiments, we generate clean scenes based on designed rules and take a bimodal approach at perturbations when generating clean-messy training data pair. More specifically, for half of the synthesized clean scenes, we employ a Gaussian noise kernel whose standard deviation is drawn from a zero-mean Gaussian distribution with a small standard deviation (0.01 for translation and $\pi/90$ for rotation angle). For the other half of the clean scenes, we employ a Gaussian noise kernel whose standard deviation is drawn from a zero-mean Gaussian distribution with a relatively larger standard deviation (0.25 for translation and $\pi/4$ for rotation angle). For the other training details, we follow the same paradigm as in the 3D-FRONT experiments.

D.2. Inference Parameters

As for the 3D-FRONT experiments, we employ the Langevin Dynamics scheme to rearrange a given perturbed Table-Chair arrangement. We empirically adjust the parameters to slightly increase the step size, accelerate the noise decay schedule, and loosen the termination condition. In particular, we select $\alpha_0 = 0.12$, $a_1 = 0.005$, $\beta_0 = 0.01$, $b_1 = 0.9$, $b_2 = 2$, and terminate once the predicted displacements are small enough in magnitude for 1 iteration.

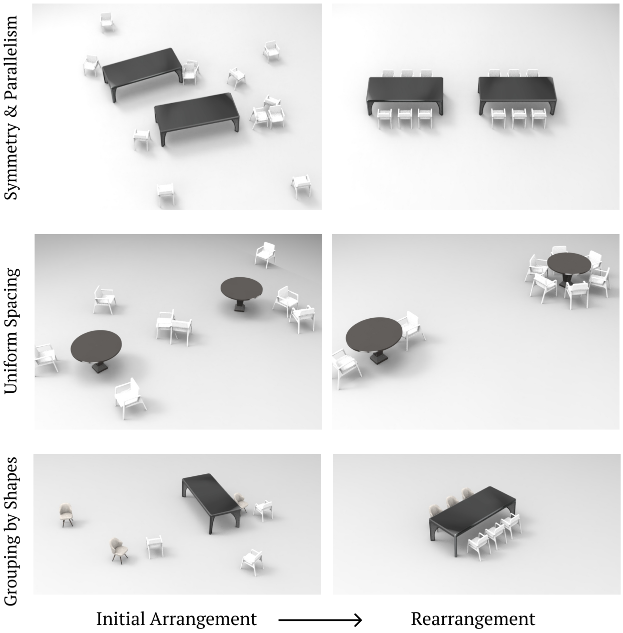


Figure 1. Regularities learning results (additional visualizations). We train our denoising network to learn three different regularities. LEGO-Net learns the complex regularity rules as demonstrated by the iterative denoising results shown on the right. Zoom in for details, especially for the shape-based grouping.

D.3. Success Rate

As mentioned, we measure LEGO-Net's performance in each Table-Chair environment through the success rate of its rearrangement. We will now elaborate on its criteria.

Symmetry and Parallelism: For a rearrangement to be classified as a success, it must satisfy the following:

- The mean euclidean distance of per-object movement averaged across scenes is less than 0.5.
- For each chair, the angular offset between its orientation and the table-facing orientation is less than $\pi/60$ radians.
- Given the two rearranged table positions, we compute their respective chair positions and perform Earth Mover's Distance assignment using these as target and the final predicted chair positions as source. The total distances summed across all 12 chairs for the 2 tables need to be less than 0.08. Note that this metric integrates colinearity, parallelism, and symmetry, and penalizes collision.

Uniform Spacing: For a rearrangement in the Uniform Spacing experiment to be classified as a success, it must satisfy the first two criteria for the Symmetry and Parallelism experiment. For the third criteria, because the number of chairs arranged around each of the 2 circular tables is variable, we cannot formulate the Earth Mover's Distance assignment as in the Symmetry and Parallelism experiment. Instead, to measure how well an arrangement captures the object-object relationships and regular relative positioning, we compute two other metrics.

For each table, we compute the angular distances between each adjacent pair of its chairs and measure the variance of these distances. We designate that a successful arrangement must have an angular distance variance of less than 0.009 radians. Additionally, given we utilize a fixed radius to generate clean chair arrangements around tables, we can measure the mean difference between the chair-to-closest-table distance and this radius. We designate that the magnitude of this difference needs to be less than 0.01 for an arrangement to be considered successful.

Grouping by Shapes: Similarly, for a rearrangement in the Grouping by Shapes experiment to be classified as a success, it must satisfy the first two criteria for the Symmetry and Parallelism experiment. Additionally, we once again can compute the exact regular arrangement of chairs with respect to the table, enabling us to calculate the Earth Mover's Distance from the final predicted chair positions to the clean configuration with respect to the predicted table position. We designate that the distance summed across the 6 chairs needs to be less than 0.05.

Furthermore, to measure success at grouping, we require that each row must be assigned exactly 3 chairs and that all chairs assigned to the same row must have the same shape feature.

D.4. Additional Qualitative Results

In Fig. 1, we provide additional qualitative renderings for the three Table-Chair experiments.

E. 3D-FRONT Experiment

To process the 3D-FRONT dataset [4], we closely follow the preprocessing protocol of ATISS [9]. For each scene, we extract from the given meshes and parameters the translation, rotation, class, and bounding box size for every object, and we normalize all lengths to be in $[-1, 1]$. We additionally extract accurate contours of the floor plans by running an iterative closest point algorithm [1], using the contour corner points of ATISS's binary floor plan masks as source and the relevant vertices from 3D-FRONT floor meshes as target.

E.1. Baseline Description

We compare against three variants of ATISS: *vanilla, labels*, and *failure-correction*. As described in the main text, *vanilla* is the original ATISS approach that generates a scene from scratch given the floor plan. ATISS *labels* is given the floor plan, as well as a set of furniture labels and the transformations and sizes of the labeled objects. ATISS *failure-correction* is proposed as an application to the probabilistic generative modeling of ATISS. It identifies which object is likely to be a failure and resamples that object given all the other objects. While the original paper only showed the technique to work when a single object is perturbed, we find it reasonable to extend the algorithm to multi-object perturbation cases. Specifically, we provide a scene with all objects perturbed (same input as LEGO-Net) and iteratively resample the lowest-probable object. We stop the iteration when it reaches 1,000 times or when the minimum probability is higher than a manually set threshold. We note that while *failure-correction* did not perform as expected when all of the objects are perturbed, as shown in Fig. 2, it is the closest baseline we could find in the literature that performs data-driven denoising of a scene.

For the comparisons, we use the official training and testing code provided by the authors of ATISS without modifications. For ATISS *failure-correction*, we add a for-loop and stopping criteria on top of their implementation, and maintain the scales of objects as fixed to be coherent with the rearrangement task.

E.2. Metric Description

For FID and KID computation, we first generate the same number of scenes as in the test dataset and randomly select 500 from the generated scenes. We then randomly select 500 real scenes from the 3D-FRONT dataset to compare against. For both FID and KID, we repeat the metric computation 5 times and report the average. Note that for computing the FID scores for ATISS, we used the officially provided code and followed their exact evaluation procedure, but we failed to reproduce their numbers. Hence, we use our own way of computing the metrics and report ours.

We note that the Frechet Inception Distance (FID) [5] is known to present significant positive bias when the number of images is small (e.g., ≤ 2000). In our case, we're dealing with an even smaller number of test images. Therefore, to compute a metric that is less biased in the small-data regime, we adopt the Kernel Inception Distance [2], which is known to address the bias problem and present small variance even at a few hundred samples.

Distance Traveled. As discussed in the main text, we aim at rearranging the messy scenes while retaining the flavor of the original scenes. Practically, cleaning a room should move objects as minimally as possible while realizing regularities. Therefore, we measure and report the mean of the average distance traveled for scenes. More specifically, for each scene, we compute the average Euclidean distance between the corresponding objects in the initial and final states, and take the mean across scenes.

Note that for ATISS *vanilla*, this metric is not applica-

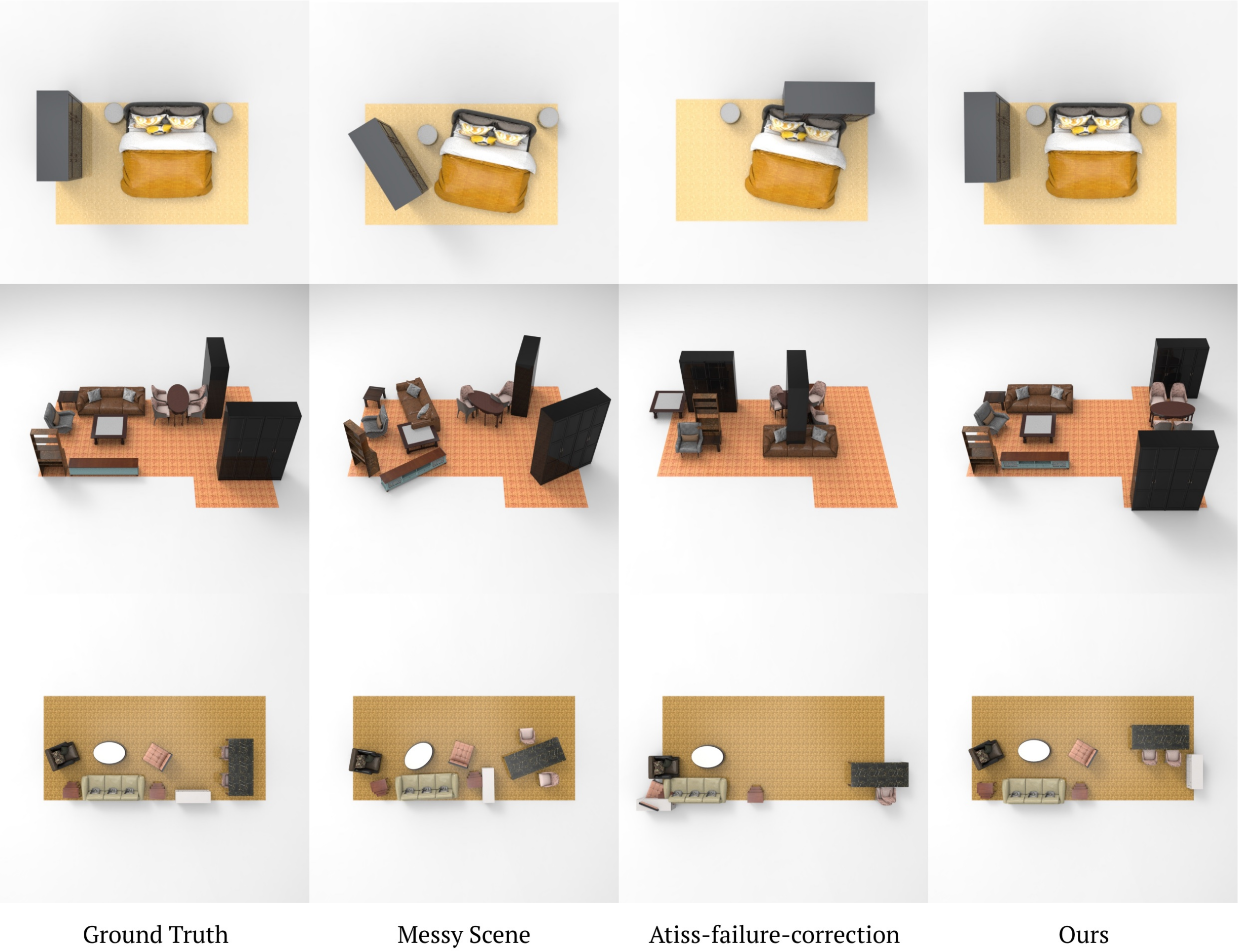


Figure 2. Qualitative comparison against ATISS-failure-correction. We show qualitative examples illustrating the difference in behaviors between our method and the closest method on the same task, ATISS *failure-correction*. Being a generative model, ATISS is able to compute the probability of the current object transformations and resample the ones with low probability. While ATISS has demonstrated great results in their paper when only a single object is perturbed, we observe that when *all* objects are perturbed, ATISS *failure-correction* has a hard time fixing the scene to a reasonable regular configuration. We hypothesize that this is due to the one-prediction-at-a-time nature of ATISS – it is difficult to find a good location to put the current object when all the other objects are perturbed. On the other hand, our method simultaneously optimizes for all the objects, avoiding such difficulty. Even for the highly challenging scene structure of the second row, our method is able to provide a high-quality layout, while ATISS *failure-correction* fails to find regular rearrangement.

ble as the method randomly places objects into the scene. For ATISS *failure-correction*, we calculate this metric by computing the distance between the initial position of each object and its final position after applying the algorithm.

EMD to GT. In order to measure how accurately our method recovers the original scene configuration, we measure the Earth Mover's Distance between the final and the ground truth scene states. Note that computing the difference between the denoising prediction and the ground truth is widely used in the image-denoising literature, using such metrics as PSNR or SSIM [7]. ATISS *vanilla* and *labels* do not receive the messy scene as input, so this metric is not applicable to them. On the other hand, ATISS *failure-correction* directly fixes the input scene, thus we may measure how accurately it recovers the original clean scene. Finally, we note that the EMD to GT metric becomes highly noisy and irrelevant when the noise added to perturb the clean scenes becomes too high, as then, there is scarsely any locational information left in the messy inputs.

E.3. Additional Qualitative Results

We provide additional qualitative results on the 3D-FRONT dataset. We show more comparisons against the closest method on the scene rearrangement task, ATISS *failure-correction*, in Fig. 2, and more results of our method in Fig. 9.

	Bedroom ↑	Living Room ↑
PointNet W/O Noise	**84.4%**	**54.4%**
PointNet W/ Noise	84.2%	54.2%
ResNet W/O Noise	83.8%	54%
ResNet W/ Noise	83.6%	54.2%

Table 1. Percentage of denoised scenes with 90% of its furniture within the floor plan boundaries. We train and test our method using different floor plan encoding architectures (PointNet vs. ResNet), and measure the percentage of the denoised scenes where most furniture respect the room boundaries.

F. Analysis (Continued)

F.1. Enforcing Floor Plan Constraints

In this section, we analyze our choice of floor plan encoder. As described in the main text, we extract points on the boundary of the binary floor plan mask, and process them with a PointNet [10] to obtain a unified feature vector describing the floor plan. We note that, in ATISS [9], a 2D convolutional network with residual connections (ResNet) was used to process the floor plans. Here, we conduct an experiment to justify our use of PointNet architecture. As a baseline, we use the ResNet architecture from ATISS, but augment the input floor plan with two additional channels corresponding to the *xy* coordinate for each pixel center, which is known to provide "spatial awareness" to the 2D CNN (in CoordConv [8]). We expect this variant of ResNet to work at least as well as the vanilla ResNet with binary mask input used in ATISS.

To compare the two methods of floor plan encoding, we train two variants of the model, using ResNet or PointNet floor plan encoding architecture. To test the effectiveness of the two methods, we measure how often furniture is moved outside the floor boundaries. Specifically, a scene rearrangement is considered successful when 90% of objects are placed inside the floor boundary within 4% of its length margin. While respecting the floor boundaries does not necessarily lead to high-quality, regular scenes, we empirically find this metric as a reasonable proxy. As can be seen from the numerical results of Tab. 1, the use of PointNet outperforms that of ResNet by a slight margin. However, we note that using PointNet is significantly faster, having almost no computational overhead for operating on the sampled 250 boundary points. We, therefore, choose to use the simpler but similar-performing PointNet to encode the scene floor plans.

F.2. Relative vs Absolute

2D diffusion models perform better at predicting the noise than the un-noised images [6]. However, for our setting, we observed that the absolute prediction models generally outperform their relative counterparts in the Table-Chair environment. We trained 3 variants of the same architecture network for the Uniform & Parallelism Table-Chair setting: (i) absolute translation and rotation prediction, (ii) relative translation and rotation prediction, and (iii) relative translation and absolute rotation prediction.

	Distance Moved ↓	Direction Offset ↓	EMD to GT ↓
Absolute	2.98e-1	1.10e-3	**5.16e-2**
Relative	**2.47e-1**	**4.21e-4**	1.58e-1
Relative Translation & Absolute Rotation	2.73e-1	5.18e-4	2.83e-1

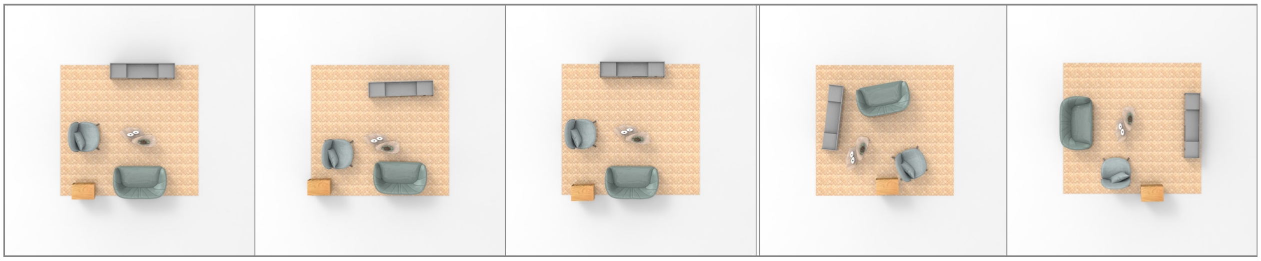
Table 2. Mean statistics from denoising results using the gradient with noise strategy for the Uniform & Parallelism Table-Chair experiment. The relative prediction model slightly outperforms the absolute prediction model in terms of distance moved and angular orientation offset, but it performs much worse in terms of EMD to GT as although it maps objects to the right general region, it does not place them as precisely as the absolute model.

Following the criteria in D.3, both variants (ii) and (iii) surprisingly report success rates of 0%. Upon further investigation (shown in Tab. 2), variants (ii) and (iii) perform comparably, if not better, at limiting the distance of movement and orienting chairs to face the tables, but they significantly underperform (i) in terms of Earth Mover's Distance to Ground Truth chair positions with respect to the predicted table position. With the same denoising parameters, the relative variants consistently fail to place the chairs as precisely as the absolute variant. The superior performance of the absolute variant may partly be explained by the fact that this setting has a relatively limited space of possible regular arrangements. The observed deficiency of relative predictions may diminish as the complexity of the scene increases.

We believe that relative transformation prediction is an important future direction to explore, as they offer translation invariance, which is particularly valuable for large-scale scenes. Currently, the position and orientation information in our input object attributes are global, and thus our system is not translationally invariant. An interesting direction for future investigation is to explore a sliding-window-style input processing to ensure translation invariance and to apply positional encoding to the output transformations.

F.3. Distance to Ground Truth and Distance Moved vs. Noise

Since the task of rearrangement values affinity to the starting configuration of objects, one question of interest is how closely we recover the original clean arrangement when given a perturbed version of it, versus another possibly equally valid, clean arrangement. Its correlation with



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

Figure 3. LEGO-Net denoising results on different noise levels (additional visualization to Fig.7 of the main text). 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. We provide numerical evidence for this phenomenon in Tab. 4.

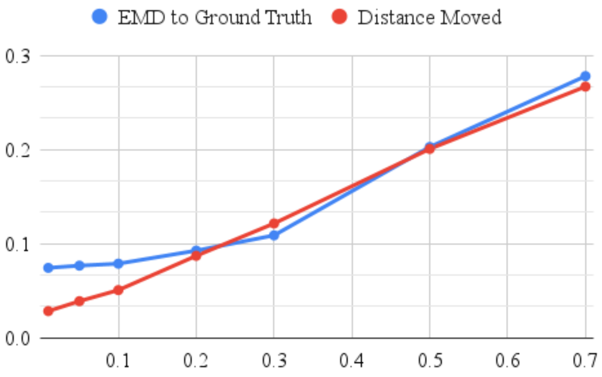


Figure 4. LEGO-Net denoising results on different noise levels (supplementing Fig.8 of the main text). The horizontal axis specifies the standard deviation of the zero-mean Gaussian distribution from which the (translation) noise level is drawn, and the vertical axis specifies distance in scenes normalized to $[-1, 1]$. We compute mean statistics across 100 scenes at $[0.01, 0.05, 0.1, 0.2, 0.3, 0.5, 0.7]$ for the standard deviation of the translation noise level distribution and at $[\pi/90, \pi/24, \pi/12, \pi/4, \pi/3, \pi/2, \pi]$ for that of rotation angle.

the level of noise added is intuitive–we expect that when the perturbation is low, we more closely reconstruct the original scen with a low distance moved whereas when the perturbation is high, our model may choose a regular arrangement different from that of the original scene in an effort to minimize the distance moved (see Fig. 3). This is indeed what we have observed numerically, as shown in Fig 4.

Note that with a low degree of noise, our model performs the task of rearrangement, but as the degree of noise increases, our model gradually transitions to the task of arrangement. In the extreme case where we give LEGO-Net a scene with objects outside of the floor plan, LEGO-Net is able to perform scene arrangement from scratch (see Fig. 5).

The open-endedness in the definition of regularity is one of the reasons why this task is both challenging and interesting. The interpolation-like behavior of LEGO-Net conditioned on the degree of noise signifies the learnable relationship between rearrangement and synthesis, and it demonstrates that a diffusion-like approach holds promising potential at such open tasks.

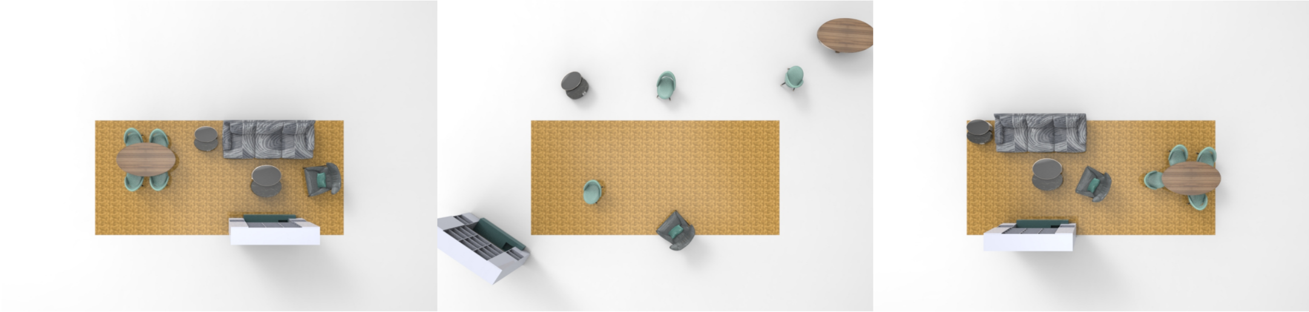
F.4. Integer Relations

The task of evaluating regularity in an object arrangement is itself an interesting research problem because, in general, multiple regular solutions are possible. To evaluate and quantify the notion of "regularity" in object arrangements, we propose using number-theoretic machinery for detecting and evaluating sparse linear integer relations among object coordinates. That is, given coordinates t_i's for n objects, we seek to find integral coefficients a_i's such that:

$$a_1 t_1 + ... + a_n t_n = 0, \quad 0 < |a_i| < \eta\,, \forall a_i, \tag{1}$$

For simplicity, we consider each dimension of the cooridnates separately, i.e., $t_i \in \mathbb{R}$. Additionally, in practice, we introduce an additional parameter ϵ to control the precision of the solutions found. In particular, we seek to find relationships such that $|a_1 t_1 + ... + a_n t_n| < \epsilon$. For our evaluation, we fix $\epsilon = 0.01$ to focus on the near-perfect relations while allowing some leeway for insignificant offsets. To efficiently find these integer solutions, we employ the PSLQ algorithm [3].

When the subset size n and maximum coefficient magnitude constraint η are small, the integer relations can be intuitively understood (e.g. representing co-linearity, symmetry, uniform spacing among few objects). To capture more complex and more general notions of regularity, we increase n and η. Doing so preicipitates two challenges. First, the number of possible subsets of size n for each scene increases rapidly as n increases, compromising efficiency. Secondly, experimentally running the algorithm on pure noise shows that with looser constraints, we may find many trivial relations that do not appear to correspond to



Ground Truth

Messy Scene

Ours

Figure 5. Extreme case rearrangement. When the amount of noise added is exceedingly high, the original scene structure is gone. Running our denoising algorithm then amounts to unconditional sampling and leads to an arrangement significantly different from the ground truth.

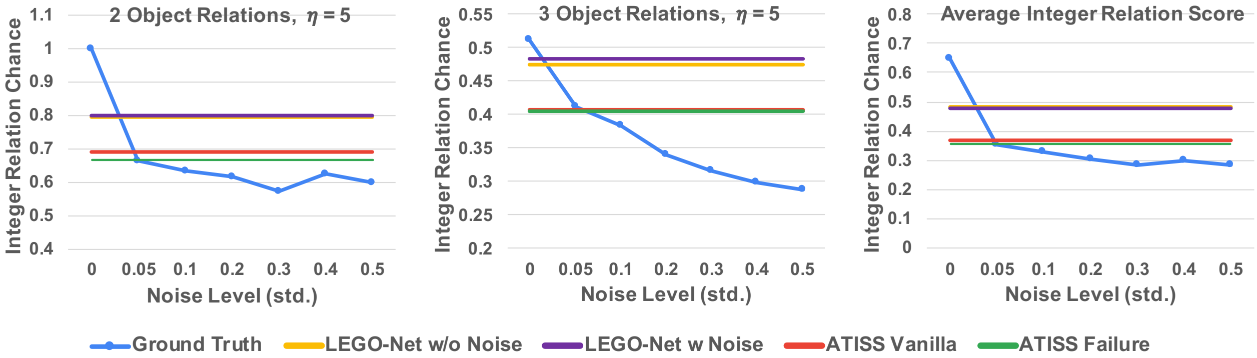


Figure 6. We measure the chance of finding integer relations among the coordinates of 2 (left) and 3 (middle) objects within a living room scene for $\eta = 5$. We aggregate these two settings with the two presented in the main text to produce an overall average integer relation score (right) for general regularity. We normalize all raw chance measures by the maximal chance across the four settings, making all numerical values directly comparable. Note that LEGO-Net outperforms the ATISS variants.

high-level 'cleanness'. To counter these, we introduce two filtering mechanisms to increase the subset sampling efficiency and to filter out the insignificant relations.

We observe that regualrities of interest to us mostly occur among objects in close proximity to one another, such as tables and chairs. Therefore, for $n > 2$, instead of sampling from all possible subsets of the objects in the scene, we iterate through each object and sample from the object's positional neighborhood. For $n = 3$, for each object, we sample 2 from the closest 4 neighboring objects to form $\{t_1, t_2, t_3\}$. This greatly improves sampling efficiency, and it also helps eliminate irrelevant candidates as integer relations satisfied by objects in vicinity to one another are more likely to be meaningful for the purpose of our evalutions.

Additionally, to filter out relations that may have been satisfied by numerical coincidence, we require all relations to be translation-invariant. Specifically, for each subset, we sample a noise μ from uniform distribution $U(-1, 1)$ and apply the PSLQ algorithm to $\{t_1 + \mu, ..., t_n + \mu\}$. We repeat the process 10 times and only deem a subset to have a valid relation if the algorithm succeeds for all 10 times. This helps the metric to focus on regularities with respect to the relative positions instead of the absolute positions of objects.

In Fig 6, we demonstrate that with these two filtering mechanisms, our metric is still meaningful for $\eta = 5$ and potentially larger parameters, which would be useful for measuring wider ranges of regularities. We also show that averaging all the integer relation metrics across the various settings suggest more general notions of regularity, for which LEGO-Net outperforms the ATISS variants.

G. Failure Modes

While LEGO-Net generates unprecedented-quality indoor scenes through the iterative denoising process, we notice that it often suffers from objects going out of the floor boundaries and objects penetrating each other. These failure modes are illustrated in Fig. 7 and 8. In fact, in Tab 1, we measure that around half of living room realizations have at least one object outside of the boundaries.

We propose two possible remedies for these related issues. First, we could apply post-processing steps to phys-

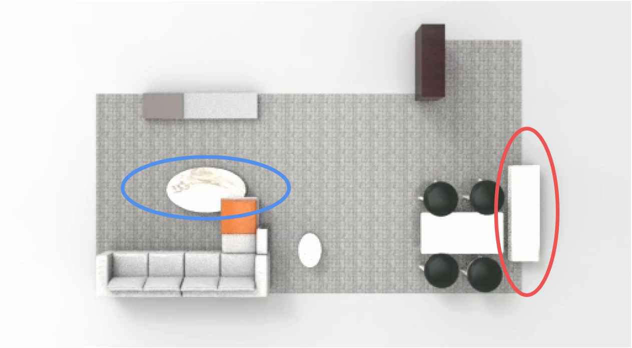


Figure 7. Failure modes. While LEGO-Net generates high-quality scenes, it often exhibits the two failure modes of placing objects outside of the floor plan boundaries (red ellipse) and inducing penetration between objects (blue ellipse). We leave post-processing steps to resolve these issues as future work.

ically resolve the two problems. That is, we optimize the locations of the objects within the scene such that penetrations and out-of-boundary issues are resolved with minimal required movement. We believe a possible formulation would involve a signed distance function (SDF), with which it is easy to compute the gradient to minimize the penetrations. When a point lies on the negative territory of another shape's SDF, we can optimize the location of that point out towards the SDF's gradient directions.

Secondly, one can consider richer encoding of the floor plan. One possibility is to encode each line segment of the floor plan separately as a token. This will essentially treat each line segment as an object in the scene and could enforce stricter constraints on the boundaries. At a glance, this strategy might increase the computational cost significantly, due to the quadratic nature of Transformer time complexity. However, one could consider limiting the communication between the line segment tokens to prevent quadratic scaling of the complexity.

We leave these two potential remedies for our failure modes as future work.

H. Future Work

In this work, we introduced LEGO-Net, an iterative-denoising-based method for tackling scene rearrangement task, which is relatively understudied compared to the scene synthesis task. We show through extensive experiments that our method is able to capture regularities of complex scenes, generating high-quality object rearrangements that could not be achieved by existing approaches to date.

However, LEGO-Net in its current form only operates on a 2D plane for the rearrangement. Extending our work to operate on the $SE(3)$ transformation space would make it more applicable to real-world scenes. A significant barrier to achieving 3D rearrangement is the lack of data. We notice that most of the indoor scene arrangement datasets deal with objects laid on the floor plan, which could limit the progress of studying $SE(3)$ scene arrangements. Designing and collecting such a dataset, e.g., small objects on top of one another is an interesting future direction.

Moreover, the trajectories generated during the denoising process of LEGO-Net are not meant to respect physical constraints, e.g., penetrations and collisions. We find that enforcing the physical constraints during the denoising steps could significantly limit the space of possible scene rearrangement. Currently, if one wants to move objects in a scene according to the initial and final states of our algorithm, one needs to run a motion planning algorithm. Extending our work to output motion plans, along with the final states, is worth pursuing.

Finally, LEGO-Net has only been shown to work well on relatively small room-scale scenes. Extending our work to operate on larger-scale scenes such as warehouses might require changes to some of our architecture choices, including strengthening translational invariance. Exploring such strategies remains an understudied challenge, which we continue to explore.

References

[1] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. Spie, 1992. 3

[2] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 3

[3] Helaman R. P. Ferguson and David H. Bailey. A polynomial time, numerically stable integer relation algorithm. Number RNR-91-032, 1991. 6

[4] 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. 1, 3

[5] 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. 3

[6] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 5

[7] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010. 4

[8] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *Advances in neural information processing systems*, 31, 2018. 5

[9] 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. 3, 5

[10] 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. 1, 5

[11] 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. 1

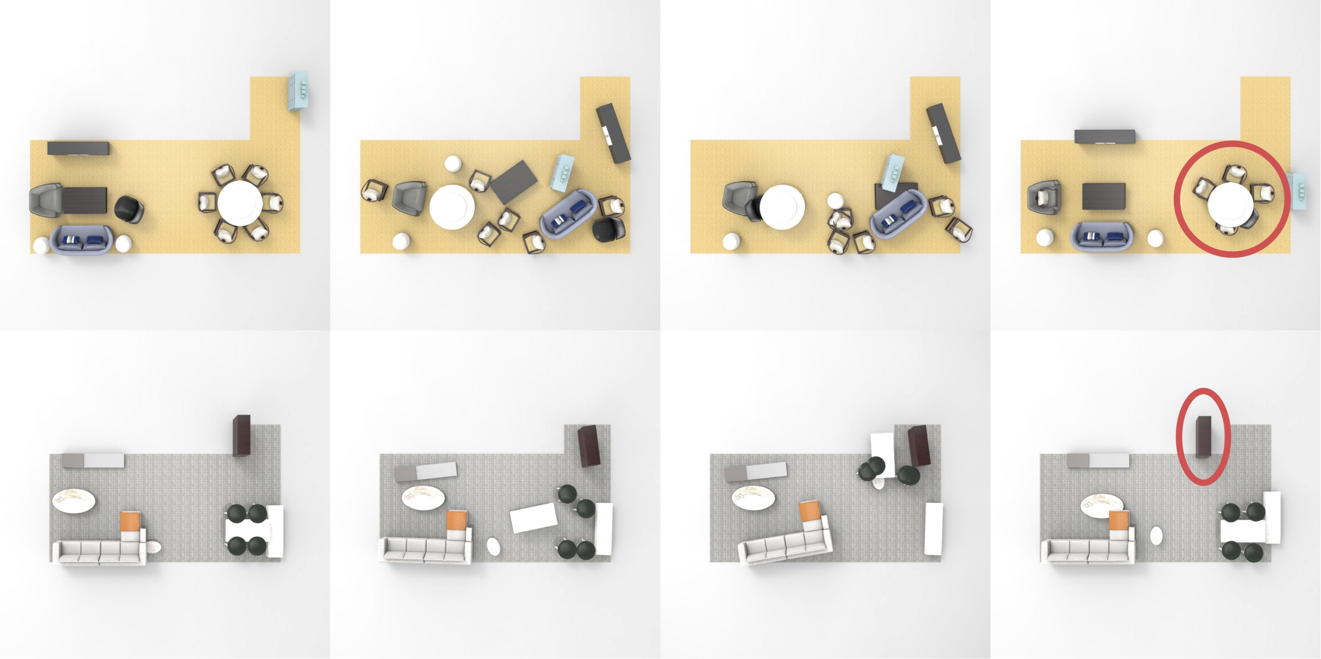
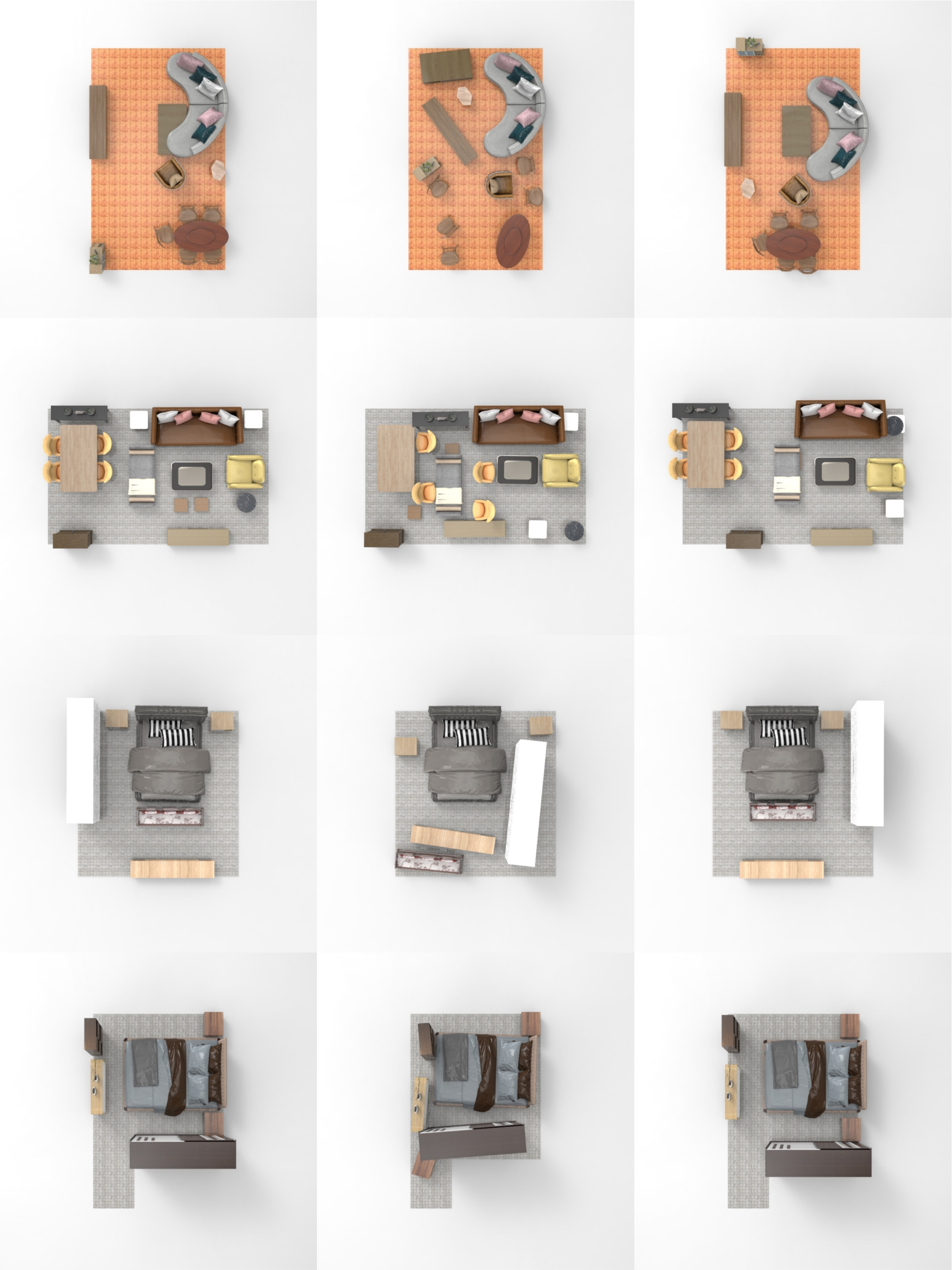


Figure 8. Additional demonstrations of failure modes of our method. In the top row, note that two chairs are missing due to perfect collision of their position predictions with those of the other chairs. In the bottom row, the cabinet is placed outside of the floor boundaries.



Ground Truth Messy Scene Our Rearrangement

Figure 9. Additional rearrangement results of LEGO-Net on the 3D-FRONT dataset.