

StructDiffusion: Language-Guided Creation of Physically-Valid Structures using Unseen Objects

WeiYu Liu¹, Yilun Du², Tucker Hermans³, Sonia Chernova¹, Chris Paxton⁴

“Set the table in the center left, relative to you.”



“Make a tower in the middle and center of the table”



“Make a short line out of mugs in the middle and center of the table”



Start Done

Fig. 1: Real-world rearrangement with unseen objects, given a language instruction. We use StructDiffusion to predict possible goals that satisfy physical constraints such as avoiding collisions between objects. At the core of StructDiffusion is an object-centric multimodal transformer backbone combined with a diffusion model, capable of sampling diverse high-level motion goals for language-guided rearrangement.

Abstract—Robots operating in human environments must be able to rearrange objects into semantically-meaningful configurations, even if these objects are previously unseen. In this work, we focus on the problem of building physically-valid structures without step-by-step instructions. We propose StructDiffusion, which combines a diffusion model and an object-centric transformer to construct structures given partial-view point clouds and high-level language goals, such as “set the table”. Our method can perform multiple challenging language-conditioned multi-step 3D planning tasks using one model. StructDiffusion even improves the success rate of assembling physically-valid structures out of unseen objects by on average 16% over an existing multi-modal transformer model trained on specific structures. We show experiments on held-out objects in both simulation and on real-world rearrangement tasks. Importantly, we show how integrating both a diffusion model and a collision-discriminator model allows for improved generalization over other methods when rearranging previously-unseen objects. For videos and additional results, see our website: <https://structdiffusion.github.io/>.

I. INTRODUCTION

Structures are everywhere in the real world: shelves are stocked, tables are set, furniture is assembled. For robots to be

successful assistants and collaborators, they must understand object structures and build these structures based on human commands. In this work, we predict how previously-unseen objects should be rearranged in order to realize language instructions, given only partial-view point cloud of a scene. Solving this task requires models that can reason about different constraints over where objects should be at once (e.g., object geometry, language-driven task semantics, physics) and generate solutions that respect all these constraints.

Assume a robot is given a language instruction such as “set the table.” First, the objects must be in the correct relative positions in order to satisfy desired spatial and semantic relations: utensils on the sides, plate in the center, for example. Second, arrangements must be physically valid: even if the plate is bigger than the robot saw in training data, it should not collide with the utensils. These constraints correspond to two overlapping, but not identical, sets of possible solutions where a robot can place objects.

One potential approach for this task is to train a language-conditioned multi-modal transformer to directly predict the best sequence of actions the robot should take [1, 2, 3, 4, 5].

¹Georgia Tech, ²MIT, ³University of Utah and NVIDIA, ⁴Meta AI

Such models have achieved impressive results on pick-and-place tasks for known objects [1, 3]; however, for structure generation of unseen objects, regressing to a single solution can often create problems when there are multiple, potentially conflicting, constraints (e.g., place objects “tightly” but avoid collisions) [4]. By contrast, planning-inspired approaches work by generating and refining a distribution over estimated poses [6], most notably diffusion models [7, 8]. Such diffusion models have shown applications not just to image generation [9, 10, 11, 12], but to motion planning and pose estimation [8, 7], and to rearrangement [12], achieving better results than previous policies due to their ability to more accurately capture the space of potential solutions.

We hypothesize that by iteratively refining predicted goals from a diffusion model, subject to learned constraints, we can similarly find better solutions for language-conditioned structure creation. In our approach, called *StructDiffusion*, we first use unknown object instance segmentation to break up our scene into objects, as per prior work [6, 13, 14, 15]. Then, we use a multi-modal transformer to combine both word tokens and object encodings from Point Cloud Transformer [16] in order to make 6-DoF goal pose predictions. These predictions are both refined iteratively via diffusion and the best goal is selected with a “discriminator” model that is trained to recognize unrealistic samples. Because goals are predicted relative to the observed partial-view point clouds of the objects, it’s possible to get strong results without any object models or object foreknowledge. We train two main components:

- 1) **An object-centric, language-conditioned diffusion model**, which learns how to construct different types of multi-object structures from observations of novel objects and language instructions.
- 2) **A learned discriminator**, which drastically improves performance by rejecting samples violating physical constraints.

This approach allows *StructDiffusion* to resolve problems not possible in previous work. Compared to an existing single-task model and a strong multi-task baseline we introduce, *StructDiffusion* achieves 16.3% and 13.5% higher success rates respectively. To our knowledge, *StructDiffusion* is the first work that can build multiple structures requiring stacking and 3D reasoning, while generalizing over different unseen objects given high-level language instructions. We believe that this approach will allow for language-guided control of robots, to be applied to more complex tasks and environments.

II. RELATED WORK

Language-conditioned manipulation is a fast-growing area of research [2, 17, 4, 3, 18, 19, 20]. Recently, Say-Can [21] showed how a large language model (LLM) can be used to sequence robot skills to respond to a wide range of natural-language queries. This work has been extended to use a map and object-centric representation of the world [22]. As this line of work leverages LLMs for reasoning, they use a purely language-based version of the world and require efforts in prompt engineering [13, 23, 21]. Another thread of work looks

more closely at learning language-conditioned skills, and has achieved impressive results on a variety of pick-and-place and articulated object manipulation tasks [2, 18, 3]; however, these approaches yet to demonstrate whether the skills can be sequenced to create more complex structures we study here. It’s worth noting that several of these works have used an object-centric representation of the world [1, 22, 4], where objects are first segmented or detected and then encoded separately. For example, VIMA [1] used encoded object patches as input to a multimodal transformer. These works, however, do not look specifically at generating *physically realistic* structures: in our experiments, we show how these direct-regression-first approaches do not generate the same quality of structures, and that, in particular, simply predicting the best placement poses or actions will lead to more failures.

Our work is related to planning with unknown objects. Simeonov et al. [24] propose a planning framework for rigid objects, but they do not study structures with complex dependencies. Curtis et al. [25] investigate task and motion planning with unknown objects, which relies on a similar segmentation and grasping pipeline we use, but does not look at learning goals, instead assume access to a set of predicates (e.g., Red, on) which can be evaluated at planning time.

Furthermore, there is a set of works which learn object-object relations for planning [6, 26, 27, 28], which is relevant to our method’s refinement process. Many of these do so explicitly. In particular, learning object skill preconditions is very useful for sequential manipulation, so some works look at predicting relationships in this context [29, 27, 30]. For example, SORNet [27] learns to predict relations between objects given a canonical image view of the objects; similarly a predictive model from image inputs is learned for capturing relationships [30]. These object-object relations are an important part of planning sequential manipulation as per *StructDiffusion*, but we pay attention to implicitly classifying object relationships as a whole.

Finally, a set of recent works have explored how diffusion models may be applied to robotics [7, 31, 8, 12, 32]. In [7, 31, 32], the diffusion model is used to parameterize the motion planning procedure, while our approach focuses on generating rearrangement goals for manipulation. Furthermore, these works require known object models and are not conditioned on flexible language goals. Similar to our work, in DALL-E-Bot [12], DALL-E is used to generate a goal image for an arrangement of multiple objects, from which object matching is used to obtain an rearrangement of objects. However, this approach assumes that underlying state of world can be captured using an overhead image, as a generated image is used to parameterize the final goal and a motion plan. In our case, we directly generate object placement poses given point cloud observations, and are thus unrestricted by which position or angle the image is generated from.

III. PRELIMINARIES

We provide background information on diffusion models [10, 11] and transformers [33]. These two neural network

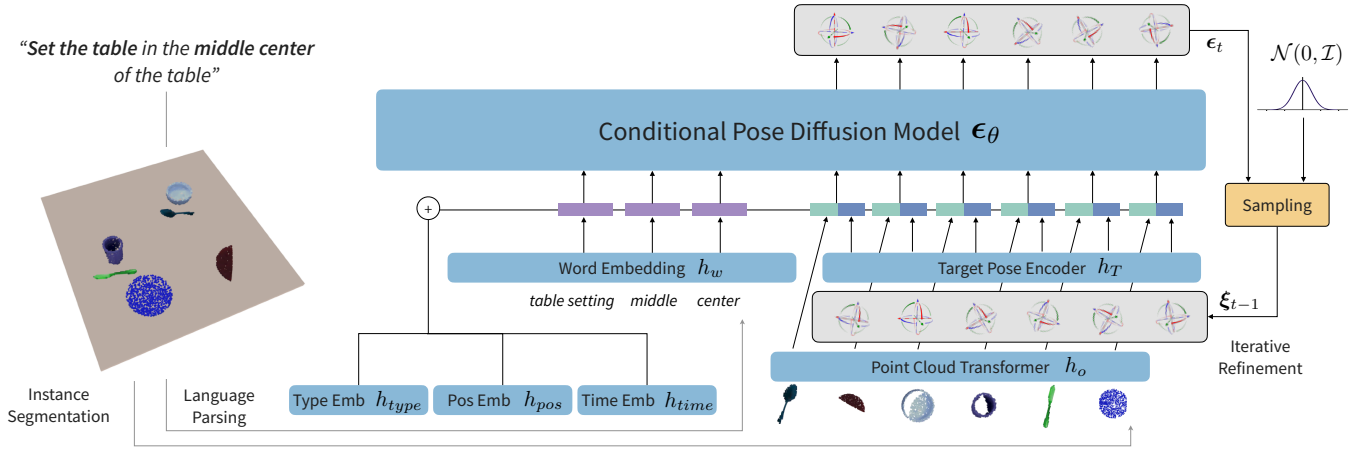


Fig. 2: Overview of the object-centric diffusion model. We combine a diffusion model with an object-centric multimodal transformer to iteratively reason about both 3D object embeddings and task specification in language, and to predict goal poses of objects.

structures provide core components for our approach.

A. Diffusion Models

Denoising Diffusion Models are a class of generative models [10, 11]. Given a sample $x \sim q(x_0)$ from the data distribution. The *forward* diffusion process is a Markov chain that creates latent variables x_1, \dots, x_T by gradually adding Gaussian noise to the sample:

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

Here β_t follows a fixed variance schedule such that the variance at each step is small and the total noise added to the original sample in the chain is large. These two conditions allows sampling $x_0 \sim p_\theta(x_0)$ from a *reverse* process that starts with a Gaussian noise x_T and follows a learned Gaussian posterior

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

In this work, we adopt the simplified model introduced in [11] that fixes the covariance $\Sigma_\theta(x_t, t)$ to an untrained time-dependent constant and reparameterize the mean $\mu_\theta(x_t, t)$ with a noise term ϵ_t . Diffusion models can be trained to minimize the variational lower bound on the negative log-likelihood $\mathbb{E}[-\log p_\theta(x_0)]$. A simplified training objective with the reparameterized mean can be derived as:

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

B. Transformers

Transformers were proposed in [33] for modeling sequential data. At the heart of the Transformer architecture is the scaled dot-product attention function, which allows elements in a sequence to attend to other elements. Specifically, an attention function takes in an input sequence $\{x_1, \dots, x_n\}$ and outputs a sequence of the same length $\{y_1, \dots, y_n\}$. Each input x_i is linearly projected to a query q_i , key k_i , and value v_i . The output y_i is computed as a weighted sum of the values, where the weight assigned to each value is based on the compatibility of the query with the corresponding key. In this work, we use the encoder layers in the original transformer architecture.

Each encoder layer includes an attention layer and a position-wise fully connected feed forward network. With the use of attention mask, the encoder layer can process sequences with different lengths.

IV. StructDiffusion FOR OBJECT REARRANGEMENT

Given a single view of an initial scene containing objects $\{o_1, \dots, o_N\}$ and a language specification containing word tokens $\{w_1, \dots, w_M\}$, our goal is to rearrange the objects into a goal configuration that satisfies the language goal. We assume the objects are rigid and we are given a partial-view point cloud of the scene with segment labels for points to identify the objects. We can extract the initial poses of the objects $\{\xi_1^{pc}, \dots, \xi_N^{pc}\}$ from the segmented object point clouds $\{x_1, \dots, x_N\}$ by setting the rotation to zero and the position to the centroid of each object point cloud in the world frame. To rearrange the objects, the robot needs to move the objects to their respective goal poses ξ_i^{goal} .

In this work, our robot can execute pick and place actions. For each object, we can sample a set of stable grasps $\mathcal{G} = \{g_1, \dots, g_M\}$. Given a target pose for object ξ_i^{goal} and a stable grasp g_j , the robot can move its end effector to $\xi_i^{ee} = \xi_i^{goal}(\xi_j^{pc})^{-1}g_j$ to place the object at the goal pose. We only use pick and place actions in our setup to simplify the problem. However, our object-centric actions can be integrated with sampling-based TAMP solutions [25] to also leverage other motion primitives, such as pushing and regrasping, to reach the goal poses predicted by our system.

Below, we describe our approach for sampling goal poses for objects based on partial point clouds, given a language goal. Our framework combines a generator based on a diffusion model and a learned discriminator that filters out invalid samples. As shown in Fig. 2, our diffusion model is integrated with a transformer model that maintains an individual attention stream for each object. This object-centric approach allows us to focus on learning the interactions between objects based on their geometric features, as well as the grounding of abstract concepts on spatio-semantic relations between objects (e.g., large, circle, top). The discriminator model operates on the

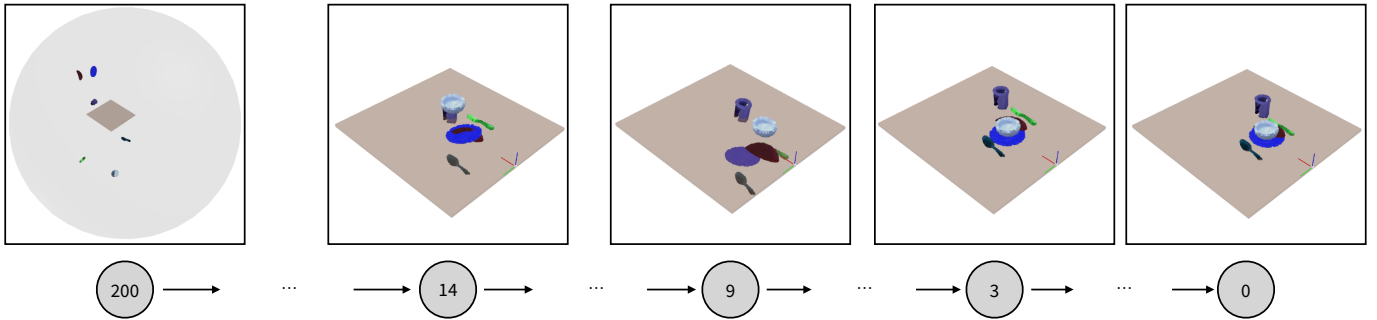


Fig. 3: We model goal generation for semantic rearrangement as a diffusion process. For each sample, we start from the last step of the reverse diffusion process, which places objects randomly in space, and jointly predict goal poses for all objects in the scene. This formulation allows our model to reason about object-object interactions in a generalizable way, which outperforms simply predicting goal poses from multi-modal inputs.

imagined scene after transforming the objects to their predicted goal poses to further reject invalid samples.

A. Encoders

We leverage modality-specific encoders to convert the multimodal inputs to latent tokens that are later processed by the transformer network.

Object encoder. Given the segmented point cloud x_i of an object o_i , we learn an encoder $h_o(x_i)$, in order to obtain the latent representation of the object. This is based on Point Cloud Transformer (PCT) [16], which has been shown to be effective at shape classification and part segmentation. We process the centered point cloud with PCT and learn a separate multilayer perceptron (MLP) to encode the mean position of the original point cloud. Encodings from the two networks are concatenated to give $h_o(x_i)$. We rely on this latent representation of objects for semantic, geometric, and spatial reasoning.

Language. To map the language goal to its latent representation, we map each unique word token from the language instructions separately to an embedding with a learned mapping $h_w(w_i)$. This method helps establish a fine-grained correspondence between each part of the language specification and the respective constraint on the generated structure. We further investigate encoding whole sentences using a sentence transformer in Appendix A.

Diffusion encodings. Since the goal poses of objects are iteratively optimized by the diffusion model and need to feed back to the model, we use a MLP to encode the goal poses of the objects $h_T(\xi_i^{goal})$. To compute the time-dependent Gaussian posterior for reverse diffusion, we combine a latent code for t in the feature channel by learning a time embedding $h_{time}(t)$.

Positional encoding. To differentiate the multimodal data, we use a learned position embedding $h_{pos}(i)$ to indicate the position of the words and objects in input sequences and a learned type embedding $h_{type}(v_i)$ to differentiate object point clouds ($v_i = 1$) and word tokens ($v_i = 0$).

B. Conditional Pose Diffusion Model

Combining a diffusion model and an object-centric transformer, *StructDiffusion* can sample diverse yet realistic ob-

ject structures while accounting for the complex constraints imposed by the object geometry and language goal. The conditional diffusion model predicts the goal poses for the objects $\xi_0 = \{\xi_i\}_i^N$ starting from the last time step of the reverse diffusion process $\xi_T \sim \mathcal{N}(0, T)$, as illustrated in Fig. 3. We use the bold symbol here because we jointly optimize the poses of all objects.

Object-centric transformer. Different from most existing diffusion models that directly generate goal images and do not explicitly model individual objects [9, 10, 11, 12], we use the transformer model to build an object-centric representation of the scene and reason about the higher-order interactions between multiple objects. This approach allows us to account for both global constraints and local interactions between objects. Leveraging attention masks, a single transformer model can also learn to rearrange different numbers of objects.

Diffusion. The use of the diffusion model helps us capture diverse structures since we are sampling from a series of Gaussian noises at different scales when going from ξ_T to our goal ξ_0 . The resulting samples, therefore, is diverse at different levels of granularity (e.g., different placements of the structures and different orientations of the individual objects). Diversity is also crucial when dealing with the inherent ambiguity in language instructions. For example, a *large* circle of plates and a *large* circle of candles impose different constraints on the proportions of the structures because the objects being arranged have different sizes.

Combining the advantages of the object-centric transformer and the diffusion model, we propose to model the conditional reverse process as

$$p_\theta(\xi_0 | \{x_i\}, \{w_i\}) = p(\xi_t) \prod p_\theta(\xi_{t-1} | \xi_t, \{x_i\}, \{w_i\})$$

The generation process depends on the point clouds of the objects and language instruction. As discussed in III-A, we learn the time-dependent noise ϵ_t , which can be used to compute ξ_t . We use the transformer as the backbone to predict the conditional noise $\epsilon_\theta(\xi_t, t, \{x_i\}, \{w_i\})$ for each object. We obtain the transformer input for the language part and the object part as

$$c_{i,t} = [h_w(x_i); h_{pos}(i); h_{type}(v_i); h_{time}(t)]$$

$$e_{i,t} = [h_o(x_i); h_T(\xi_i^{goal}); h_{pos}(i); h_{type}(v_i); h_{time}(t)]$$

Algorithm 1 Goal generation with *StructDiffusion*

```
1: for  $t \in \text{range}(T, 1)$  do
2:    $\epsilon_t \sim \epsilon_\theta(\xi_t, t, \{x_i\}, \{w_i\})$ 
3:    $z \sim \mathcal{N}(0, \mathcal{I})$  if  $t > 1$  else  $z = 0$ 
4:    $\xi_{t-1} = \frac{1}{\sqrt{\beta_t}}(\xi_t - \frac{\beta_t}{\sum_{s=1}^t 1-\beta_s} \epsilon_t) + \sqrt{\beta_t}z$ 
5: Transform object points:  $x_i^{goal} = \xi_i^{goal}(\xi_i^{pc})^{-1}x_i$ 
6: Compute discriminator scores
7: return ranked  $\xi_0$ 
```

where $[\cdot]$ is the concatenation at the feature dimension. The model takes in the sequence $\{c_{1,t}, \dots, c_{M,t}, e_{1,t}, \dots, e_{N,t}\}$ and predicts $\{\epsilon_{1,t}, \dots, \epsilon_{N,t}\}$ for the object poses. We parameterize 6-DoF pose target ξ as $(t, R) \in SE(3)$. We directly predict $t \in \mathbb{R}^3$ and predict two vectors $a, b \in \mathbb{R}^3$, which are used to construct the rotation matrix $R \in SO(3)$ using a Gram-Schmidt-like process proposed in [34].

C. Discriminators

In addition to the generator, we also use a learned discriminator model to further filter the predictions for realism. The discriminator works on imagined scenes, where the point clouds of objects are rigidly transformed to the respective goal poses following $x_i^{goal} = \xi_i^{goal}(\xi_i^{pc})^{-1}x_i$. As the discriminator is trained to predict a score $s \in [0, 1]$, it can be used to rank the generated samples from the diffusion model during inference.

Here we also have the opportunity to leverage a spatial abstraction different from the one used by the generator. The generator operates on the latent object-centric representation that are suitable to *imagine* possible structures. The discriminator can directly reason about the interactions between the transformed point cloud objects at the point level. To maintain the ability to distinguish each individual object, we add a one-hot encoding to each point feature.

In our preliminary experiments, we found that this point-level collision model has more discrimination power than an object-centric model that operates on latent representations of the objects. We also investigated whether a discriminator can be integrated into the diffusion process using a technique called *classifier guidance* [35], but observed no significant benefits. Details are presented in Appendix B.

We explore two potential discriminators. The *collision discriminator* learns to predict pairwise collisions between two objects from their partial point clouds. Second, the *structure discriminator* learns to classify the validity of the whole multi-object structure. The structure discriminator is also language-conditioned, so it can learn structure-specific constraints. We found that the structure discriminator works better when it is only required to predict if local constraints are satisfied. Therefore, we normalize the scene point cloud and drop parts of the language instruction that specify global constraints such as where to place the structure on the table.

D. Planning and Inference

In Alg. 1, we show how to combine the different components of our framework to sample object structures. We first

initialize a batch of goal poses $\mathbb{R}^{B \times N \times (3+3+3)}$ with random noise. We use batch operation on a GPU to efficiently perform diffusion and transform point clouds of multiple objects for different samples. For the discriminators, we also generate point clouds combining all objects after the diffusion process and score them in batches. The ranked samples are returned. Each sample corresponds to a physically and semantically valid multi-object structure that can be used by other components of the manipulation pipeline for planning.

E. Training Details

We train the diffusion model and the discriminators separately. For the diffusion model, we use the dataset from [4] containing high-level language instructions, segmented object point clouds, and sequences of rearrangement actions. We extract the goal poses from the rearrangement actions. We train a single model for all structures where the number of examples for different classes of structures are balanced. We use a batch size of 128 and train the diffusion model on an RTX3090 GPU for about 12 hours. To train the collision discriminator, we randomly sample 100,000 pairwise configurations for objects from the dataset. For the structure discriminator, we generate negative examples by randomly perturbing the ground truth target poses ξ_i^{goal} . For each negative example, we randomly select a subset of objects to perturb so that the negative examples have different numbers of objects out of place. We augment the data by using point clouds from different time steps of the rearrangement sequence to create the imagined scenes as they usually occlude different objects.

V. SIMULATION EXPERIMENTS

We first evaluate our method in simulation by comparing it to existing methods and also a collection of strong baselines we introduce for language-guided multi-object rearrangement.

A. Baselines

We compare our approach against the following baselines:

StructFormer: This prior method uses a multimodal transformer network to generate multi-object structures based on segmented object point clouds and language instructions [4]. The transformer network autoregressively predicts the goal poses of each object. We follow the original work to train a separate model for each class of structure.

Conditional Variational Autoencoder (CVAE): CVAEs have been used to capture different modes for multi-task learning and language-conditioned manipulation [17, 18]. Our CVAE baseline uses the object-centric transformer backbone as a strong baseline for semantic rearrangement. To prevent the latent variable from being ignored when combining the transformer with CVAE, the transformer network predicts the object goal poses in a single forward pass (i.e., not autoregressively). A single model is trained for four classes of structures.

Optimization with Learned Discriminator: This baseline iteratively optimizes the goal poses of objects with the *structure discriminator* that is trained to classify valid rearranged scenes and invalid ones. This general approach has been used



Fig. 4: Testing objects from Google Scanned Objects [40], ReplicaCAD dataset [41], and the YCB Object Set [42]. The test object dataset contains a wide range of textured objects belonging to various classes. None of these objects appear in the training data.

extensively for learning language-conditioned manipulation from offline data [36], grasping [37, 38], and predicting stable placements of objects [6], but not for language-conditioned multi-object rearrangement. We use the cross-entropy method for optimization [39], and only optimize the object poses and not the structure pose to simplify the optimization problem. We initialize samples from the baseline generative models because initializing with random values does not lead to meaningful performance.

B. Experimental Setup

We evaluate all models in the PyBullet physics simulator [43]. Point cloud observations are rendered with NViSII [44]. We test on novel object models from both known and unknown categories as our goal is to transfer the model learned in simulation directly to real-world objects. Fig. 4 shows the testing objects, which are collected from Google Scanned Objects [40], ReplicaCAD dataset [41], and the YCB object Set [42]. To generate the test scenes, we use the same data collection pipeline that is used to collect ground truth data from prior work [4]. This ensures that a valid rearrangement can be found for each scene. The set of objects and the language goal for each scene are randomly sampled. Distractor objects are randomly placed in the scene to simulate occlusions.

We report success rate for the rearrangements. To isolate the pose prediction problem from other components of the system (e.g., grasp sampling and motion planning), we directly place objects 3cm above the the predicted target poses. We check whether the rearrangement is physically valid by running the simulation loop after placing each object. We check possible collisions and intersections between objects using approximate convex decompositions of the 3D object models. We also implement model-based classifiers to evaluate whether the rearrangement satisfy the language goal. For example, we check whether the objects are in a line using the centroids of the models. A rearrangement is considered as successful if the placements of objects are not preempted due to physics-related failures and the goal scene satisfies all semantic constraints determined by the given language goal. On average, there are 5 constraints for different types of structures.

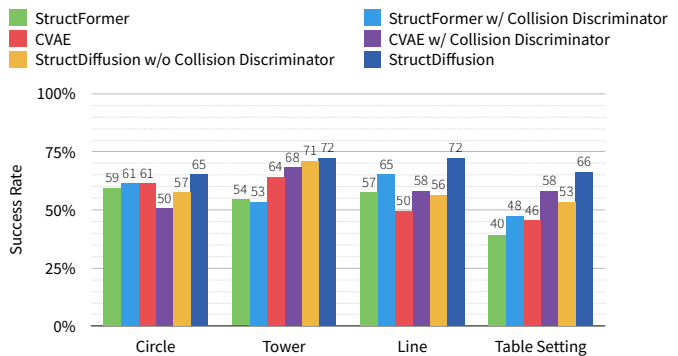


Fig. 5: Success rates for four different classes of structures on held-out objects. Models are evaluated in a physics simulator using unseen objects. A rearrangement is successful only if all objects are placed in physically valid poses and the rearranged scene satisfies the language goal. Compared to StructFormer [4], the model previously proposed for semantic rearrangement, *StructDiffusion* obtains a 16% average improvement in success rate.

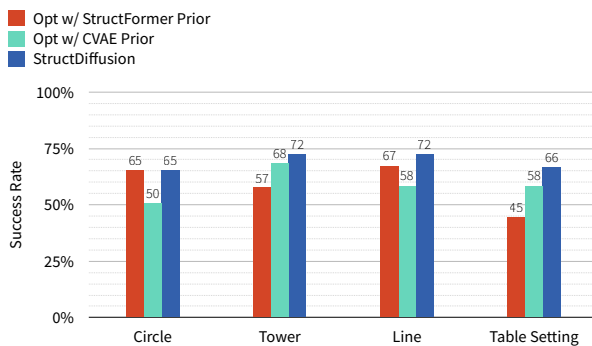


Fig. 6: Comparing *StructDiffusion* with other iterative methods. The two baselines initialize samples of target object poses using either the StructFormer or the CVAE model. The predicted scores of a learned discriminator is then used to guide iterative optimization of the samples. In comparison, *StructDiffusion* directly predicts the noises ϵ_t that need to be removed from the samples at each step.

C. Comparison with Other Generative Models

In Fig. 5, we compare with other generative models and gain insights into the generator-discriminator design of our model. We see that our complete model, *StructDiffusion*, significantly outperformed all baselines on all structure types. The improvement was most significant for structures that required precise placements of objects and modeling contacts between objects. The generator-discriminator design was necessary because the diffusion model alone still generated invalid samples, especially for the line structures. The performance difference between *StructDiffusion* and the ablated model that does not use discriminator supports that our model can leverage the complementary strengths of object-centric representation and scene-level representation that preserves the point-to-point interactions. In our preliminary experiments, we observed that using the collision discriminator is enough to create high-quality samples; however, this is not the case for the iterative methods discussed in the following section.

Although applying the collision discriminator also improved

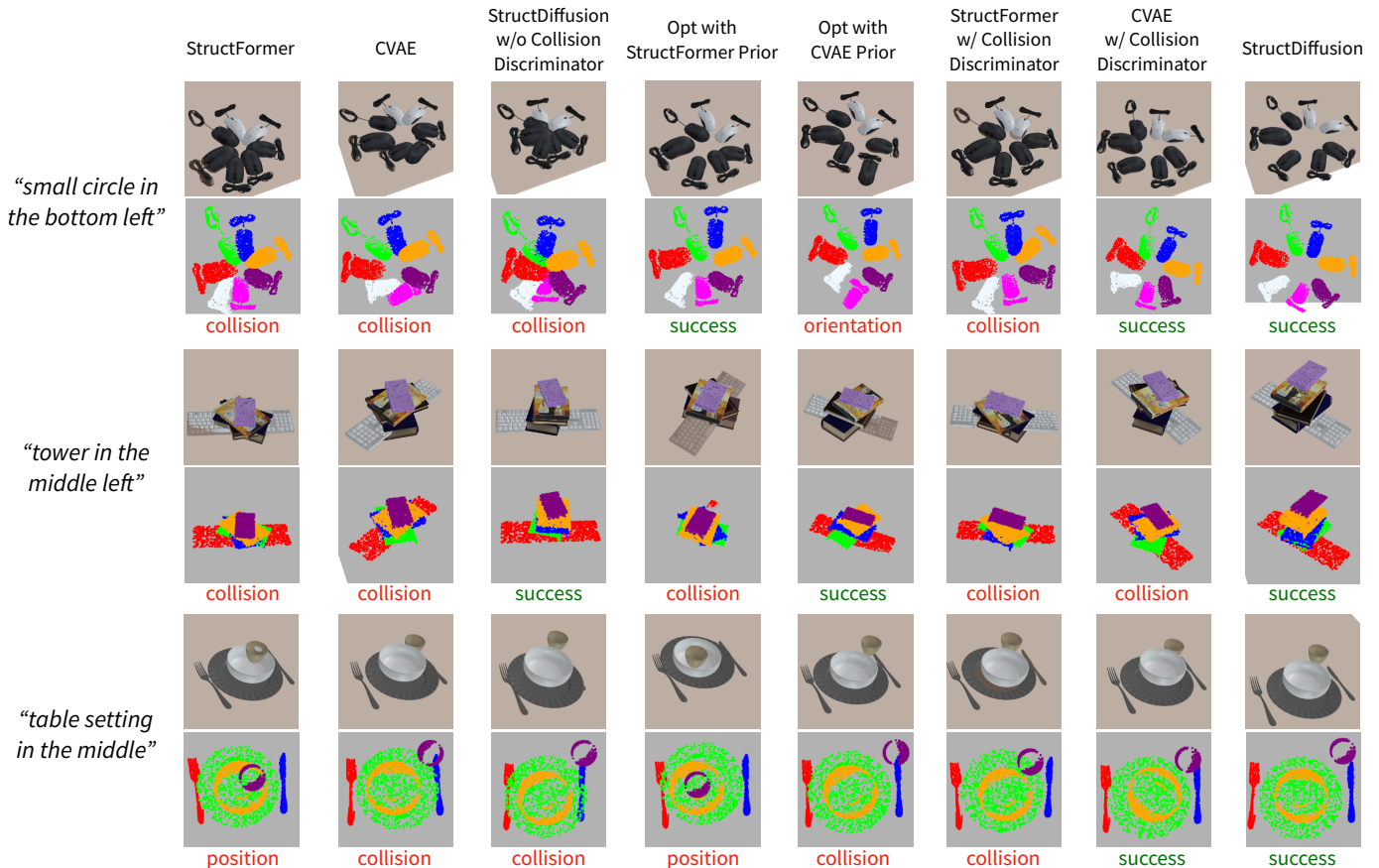


Fig. 7: Comparison between *StructDiffusion* and the baselines on partial views of held-out objects, given language commands from four different categories. *StructDiffusion* is better at resolving constraints involving contact and precise arrangement of objects, avoiding collisions and creating physically realistic placements. The labels indicate whether the structures can be successfully built in the simulation environment and also satisfy the language goal.

the performance of StructFormer and the CVAE, our diffusion model benefited the most from the addition. We attribute this difference to the different diversities of samples from these three classes of generative models. The autoregressive transformer underlying StructFormer does not explicitly model uncertainty, therefore leading to similar samples for each scene. The single source of stochasticity from the latent variable of the CVAE model is also not enough. As the diffusion model incorporates uncertainties at different scales, it has the ability to generate different classes of structures but also generate hypotheses of object placements given only partial, and even heavily occluded, point clouds of objects. We provide a qualitative comparison in Fig. 7. We further break down the failure cases based on structure types and methods to support the insights discussed above in Appendix C.

D. Comparison with Other Iterative Methods

In Fig. 6, we compare *StructDiffusion* with other optimization-based baselines that can take advantage of the additional computational time to iteratively refine the prediction. The result shows that *StructDiffusion* outperformed the other two baselines. Even though we observed strong performance when applying an optimization-based method [6] to other manipulation tasks, we do not see significant benefit in our

task. Looking more closely, we observe that the challenging cases that are not yet solved by the non-iterative variants are cases where the placements of objects are highly related. In these cases, the guidance from the discriminator can be ambiguous and leads to local minimal without reaching valid solutions. We hypothesize that leveraging guidance at different scales is necessary, as studied in a recent work that directly learns to predict scores (i.e., gradients) at different scales for 2D object rearrangement [45].

VI. REAL WORLD EXPERIMENTS

In this section, we report on real-world experiments, testing structure assembly on a real robot.

A. Perception and Hardware

We deployed our system on a 7-DoF JACO arm with an Asus Xtion RGB-D Camera. We obtained segmented object point clouds by identifying clusters-of-interest through table surface detection and Euclidean distance clustering, using the Point Cloud Library [46]. For most objects, we calculated antipodal grasps over each object point cloud [47], which are then ordered and executed using pairwise ranking [48]. For objects involved in the table setting arrangements (e.g., forks and plates), we sampled grasps using the Contact-GraspNet [49] because antipodal grasp sampling failed to

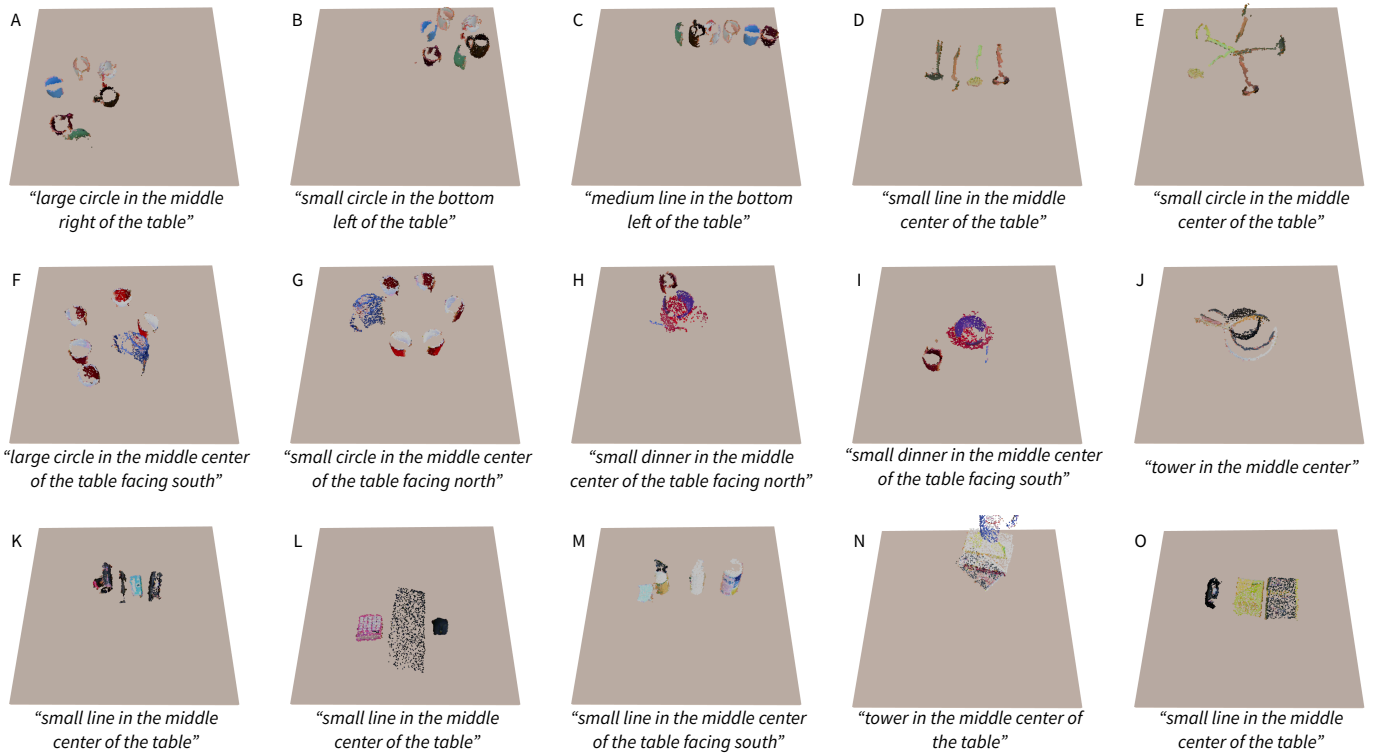


Fig. 8: Examples of predicted structures for real-world objects. We can predict structures from raw point clouds for a wide range of language instructions fitting into four different broad classes. Note the point clouds are incomplete because they were captured from a single camera.

generate successful grasps for these objects. We used RRT-Connect [50] for motion planning. We released each object 3cm above the predicated target pose for placement.

B. Predictions for Real-World Objects

We first show qualitative examples of the predicted structures for real-world objects in Fig. 8. These examples are created by rigidly transforming the segmented object point clouds from an initial scene with the target poses of the highest ranked structure. Even though our model is trained only on simulation data, it can be directly used to generate semantically diverse and physically valid structures for real-world objects. Our model can generate different variations of the same structure type, as shown in (A, B). The same set of objects can be arranged into completely different classes of structures conditioning on the language, as shown in (A, C) and (D, E). Besides changing the positions and sizes of the structures, the orientations of the structures can also be specified in language (F, G) and (H, I). Note that even though table settings in the training data are only aligned horizontally as shown in I, the use of language and training on other orientation-specific structures enable compositional generalization to a new orientation shown in H. Finally, we see non-symmetrical objects (e.g., mugs, knives, and spatulas) are correctly aligned in B, D, E, J, H.

C. Rearrangement with Pick and Place

To reliably rearrange multiple objects, we combined *StructDiffusion* with grasp and motion planning. We performed

TABLE I: Robot experiments with real-world objects. We perform each of the task multiple times with different objects and initial placements. We show the number of times that valid grasp and motion plans are found and that the plans are executed successfully by the robot.

Object Categories (#Trials)	Structure	Grasp and Motion Planning	Placement
Bowl, Pan (3)	Small Circle	3	3
Bowl, Pan (3)	Tower	3	2
Bowl, Pan (3)	Small line	2	2
Plate, Spoon, Fork, Knife, Cup (12)	Table Setting	5	4
Overall Success Rate		61.9%	52.4%

nested search to find the target structure to execute. Specifically, we iterate through the generated and ranked structures. For each structure, we sample a set of grasp poses for each object and compute corresponding pre-grasp, standoff, and placement poses based on the prediction. We searched for valid motion plans between these waypoints. If all motion plans have been found, we execute them on the robot.

In Table I, we show success counts and average success rate for trials with different objects and different language goals. For circle, tower, and line structure, valid motion and grasp plans can be found most of the time due to the diverse structures generated by *StructDiffusion*. For table settings, grasp planning is still challenging due to the small tolerance of the grasp regions. The limited diversity of sampled grasps also leads to no valid inverse kinematic solutions in some cases. We observed that partial point clouds due to noisy real sensor and self-occlusions for large objects led to a small

number of invalid structure predictions. While planning, we make the assumption that the objects are rigidly attached to the gripper after grasping without slippage. This assumption generally did not hold in the real world and led to occasional failures. This assumption can be relaxed by predicting a post-grasp displacement, using learned models such as [51]. We show examples of successful executions and failure cases in Appendix D.

VII. CONCLUSIONS

We introduced *StructDiffusion*, an approach for creating physically-valid structures using multimodal transformers and diffusion models. *StructDiffusion* operates on point cloud images of previously-unseen objects, and can create structures for a range of language instructions.

Specifically, we compared to a wide variety of strong baselines, including the previous state of the art [4] and to a conditional variational autoencoder. End-to-end policies do not perform as well, because they cannot refine placement poses that are *nearly* correct. Using diffusion models for generating diverse samples and a trained discriminator to filter object goal poses significantly improves performance.

In this work, we did not look at optimally planning. In the future, we could look at combining this approach with task and motion planning for unknown objects, as in [25]. Additional experiments presented in Appendix A showed that our model could easily be combined with a pretrained language model to deal with more natural sentences; therefore, we also hope to apply our work to a far wider range of structures and more complex language commands.

REFERENCES

- [1] Y. Jiang, A. Gupta, Z. Zhang, G. Wang, Y. Dou, Y. Chen, L. Fei-Fei, A. Anandkumar, Y. Zhu, and L. Fan, “Vima: General robot manipulation with multimodal prompts,” *arXiv preprint arXiv:2210.03094*, 2022. 1, 2
- [2] M. Shridhar, L. Manuelli, and D. Fox, “Cliport: What and where pathways for robotic manipulation,” in *Conference on Robot Learning*. PMLR, 2022, pp. 894–906. 1, 2
- [3] —, “Perceiver-actor: A multi-task transformer for robotic manipulation,” *arXiv preprint arXiv:2209.05451*, 2022. 1, 2
- [4] W. Liu, C. Paxton, T. Hermans, and D. Fox, “Struct-former: Learning spatial structure for language-guided semantic rearrangement of novel objects,” in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6322–6329. 1, 2, 5, 6, 9
- [5] P.-L. Guhur, S. Chen, R. Garcia, M. Tapaswi, I. Laptev, and C. Schmid, “Instruction-driven history-aware policies for robotic manipulations,” *arXiv preprint arXiv:2209.04899*, 2022. 1
- [6] C. Paxton, C. Xie, T. Hermans, and D. Fox, “Predicting stable configurations for semantic placement of novel objects,” in *Conference on Robot Learning (CoRL)*, 2021. 2, 6, 7
- [7] M. Janner, Y. Du, J. Tenenbaum, and S. Levine, “Planning with diffusion for flexible behavior synthesis,” in *International Conference on Machine Learning*, 2022. 2
- [8] S. Huang, Z. Wang, P. Li, B. Jia, T. Liu, Y. Zhu, W. Liang, and S.-C. Zhu, “Diffusion-based generation, optimization, and planning in 3d scenes,” *arXiv preprint arXiv:2301.06015*, 2023. 2
- [9] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695. 2, 4
- [10] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, “Deep unsupervised learning using nonequilibrium thermodynamics,” in *International Conference on Machine Learning*. PMLR, 2015, pp. 2256–2265. 2, 3, 4
- [11] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851, 2020. 2, 3, 4
- [12] I. Kapelyukh, V. Vosylius, and E. Johns, “Dall-e-bot: Introducing web-scale diffusion models to robotics,” *arXiv preprint arXiv:2210.02438*, 2022. 2, 4
- [13] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, “Prog-prompt: Generating situated robot task plans using large language models,” *arXiv preprint arXiv:2209.11302*, 2022. 2
- [14] A. Goyal, A. Mousavian, C. Paxton, Y.-W. Chao, B. Okorn, J. Deng, and D. Fox, “Ifor: Iterative flow minimization for robotic object rearrangement,” *arXiv preprint arXiv:2202.00732*, 2022. 2
- [15] A. Qureshi, A. Mousavian, C. Paxton, M. Yip, and D. Fox, “Nerp: Neural rearrangement planning for unknown objects,” in *Proceedings of Robotics: Science and Systems*, 2021. 2
- [16] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, “PCT: Point cloud transformer,” *Computational Visual Media*, vol. 7, no. 2, pp. 187–199, 2021. 2, 4
- [17] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, “Learning latent plans from play,” in *Conference on robot learning*. PMLR, 2020, pp. 1113–1132. 2, 5
- [18] O. Mees, L. Hermann, and W. Burgard, “What matters in language conditioned robotic imitation learning over unstructured data,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 4, pp. 11 205–11 212, 2022. 2, 5
- [19] C. Lynch, A. Wahid, J. Tompson, T. Ding, J. Betker, R. Baruch, T. Armstrong, and P. Florence, “Interactive language: Talking to robots in real time,” *arXiv preprint arXiv:2210.06407*, 2022. 2
- [20] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog,

- J. Hsu *et al.*, “Rt-1: Robotics transformer for real-world control at scale,” *arXiv preprint arXiv:2212.06817*, 2022. [2](#)
- [21] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog *et al.*, “Do as i can, not as i say: Grounding language in robotic affordances,” *arXiv preprint arXiv:2204.01691*, 2022. [2](#), [12](#)
- [22] B. Chen, F. Xia, B. Ichter, K. Rao, K. Gopalakrishnan, M. S. Ryoo, A. Stone, and D. Kappler, “Open-vocabulary queryable scene representations for real world planning,” *arXiv preprint arXiv:2209.09874*, 2022. [2](#)
- [23] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, “Code as policies: Language model programs for embodied control,” *arXiv preprint arXiv:2209.07753*, 2022. [2](#), [12](#)
- [24] A. Simeonov, Y. Du, B. Kim, F. R. Hogan, J. Tenenbaum, P. Agrawal, and A. Rodriguez, “A long horizon planning framework for manipulating rigid pointcloud objects,” *arXiv preprint arXiv:2011.08177*, 2020. [2](#)
- [25] A. Curtis, X. Fang, L. P. Kaelbling, T. Lozano-Pérez, and C. R. Garrett, “Long-horizon manipulation of unknown objects via task and motion planning with estimated affordances,” in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 1940–1946. [2](#), [3](#), [9](#)
- [26] A. Bobu, C. Paxton, W. Yang, B. Sundaralingam, Y.-W. Chao, M. Cakmak, and D. Fox, “Learning perceptual concepts by bootstrapping from human queries,” *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 260–11 267, 2022. [2](#)
- [27] W. Yuan, C. Paxton, K. Desingh, and D. Fox, “Sornet: Spatial object-centric representations for sequential manipulation,” in *5th Annual Conference on Robot Learning*. PMLR, 2021, pp. 148–157. [2](#)
- [28] A. Simeonov, Y. Du, L. Yen-Chen, , A. Rodriguez, , L. P. Kaelbling, T. L. Perez, and P. Agrawal, “Se(3)-equivariant relational rearrangement with neural descriptor fields,” in *Conference on Robot Learning (CoRL)*. PMLR, 2022. [2](#)
- [29] M. Sharma and O. Kroemer, “Relational learning for skill preconditions,” *arXiv preprint arXiv:2012.01693*, 2020. [2](#)
- [30] T. Migimatsu and J. Bohg, “Grounding predicates through actions,” in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 3498–3504. [2](#)
- [31] J. Urain, N. Funk, G. Chalvatzaki, and J. Peters, “Se(3)-diffusionfields: Learning smooth cost functions for joint grasp and motion optimization through diffusion,” <https://arxiv.org/pdf/2209.03855.pdf>, 2022. [2](#)
- [32] A. Ajay, Y. Du, A. Gupta, J. Tenenbaum, T. Jaakkola, and P. Agrawal, “Is conditional generative modeling all you need for decision-making?” *arXiv preprint arXiv:2211.15657*, 2022. [2](#)
- [33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, 2017, pp. 5998–6008. [2](#), [3](#)
- [34] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, “On the continuity of rotation representations in neural networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5745–5753. [5](#)
- [35] P. Dhariwal and A. Nichol, “Diffusion models beat gans on image synthesis,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 8780–8794, 2021. [5](#)
- [36] S. Nair, E. Mitchell, K. Chen, S. Savarese, C. Finn *et al.*, “Learning language-conditioned robot behavior from offline data and crowd-sourced annotation,” in *Conference on Robot Learning*. PMLR, 2022, pp. 1303–1315. [6](#)
- [37] A. Murali, A. Mousavian, C. Eppner, C. Paxton, and D. Fox, “6-dof grasping for target-driven object manipulation in clutter,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6232–6238. [6](#)
- [38] Q. Lu, M. Van der Merwe, B. Sundaralingam, and T. Hermans, “Multifingered grasp planning via inference in deep neural networks: Outperforming sampling by learning differentiable models,” *IEEE Robotics & Automation Magazine*, vol. 27, no. 2, pp. 55–65, 2020. [6](#)
- [39] M. Chen, S. L. Herbert, H. Hu, Y. Pu, J. F. Fisac, S. Bansal, S. Han, and C. J. Tomlin, “Fastrack: a modular framework for real-time motion planning and guaranteed safe tracking,” *IEEE Transactions on Automatic Control*, vol. 66, no. 12, pp. 5861–5876, 2021. [6](#)
- [40] L. Downs, A. Francis, N. Koenig, B. Kinman, R. Hickman, K. Reymann, T. B. McHugh, and V. Vanhoucke, “Google scanned objects: A high-quality dataset of 3d scanned household items,” *arXiv preprint arXiv:2204.11918*, 2022. [6](#)
- [41] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra, “Habitat 2.0: Training home assistants to rearrange their habitat,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. [6](#)
- [42] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, “The ycb object and model set: Towards common benchmarks for manipulation research,” in *2015 international conference on advanced robotics (ICAR)*. IEEE, 2015, pp. 510–517. [6](#)
- [43] E. Coumans and Y. Bai, “Pybullet, a python module for physics simulation in robotics, games and machine learning,” 2017. [6](#)
- [44] N. Morricall, J. Tremblay, Y. Lin, S. Tyree, S. Birchfield, V. Pascucci, and I. Wald, “Nvisii: A scriptable tool for photorealistic image generation,” *arXiv preprint arXiv:2105.13962*, 2021. [6](#)

- [45] M. Wu, F. Zhong, Y. Xia, and H. Dong, “Targf: Learning target gradient field for object rearrangement,” *arXiv preprint arXiv:2209.00853*, 2022. [7](#)
- [46] R. B. Rusu and S. Cousins, “3d is here: Point cloud library (pcl),” in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 1–4. [7](#)
- [47] A. Ten Pas and R. Platt, “Using geometry to detect grasp poses in 3d point clouds,” in *Robotics research*. Springer, 2018, pp. 307–324. [7](#)
- [48] D. Kent and R. Toris, “Adaptive autonomous grasp selection via pairwise ranking,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2971–2976. [7](#)
- [49] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox, “Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 13 438–13 444. [7](#)
- [50] J. J. Kuffner and S. M. LaValle, “RRT-connect: An efficient approach to single-query path planning,” in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001. [8](#)
- [51] J. Zhao, D. Troniak, and O. Kroemer, “Towards robotic assembly by predicting robust, precise and task-oriented grasps,” in *CoRL*, 2022. [9](#)
- [52] N. Reimers and I. Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. [Online]. Available: <https://arxiv.org/abs/1908.10084> [12](#)

A. Towards Natural Language

In the main experiments, we evaluate the model that encodes word tokens individually. Here, we want to see if our framework can deal with more natural language without predefined vocabularies and at different levels of abstraction (e.g., “make a line” vs “make a short line at the bottom of the table”) by combining with a pretrained language model.

1) *Language Data*: We procedurally generate sentences to train the model. At test time, we leverage sentence embeddings to generalize to new sentences. Given the word tokens in sentences, we first enumerate all possible combinations of the tokens. Note in our definition each token corresponds to a type and a value. For example, [(shape, circle), (size, small), (x_position, center), (y_position, middle), (rotation, south)] can create combinations such as [(shape, circle)], [(size, small), (x_position, center)], [(shape, circle), (x_position, center), (y_position, middle)]. In total, there are 669 unique combinations of word tokens. We then manually created templates for each combination of token types. For example, “build a [shape]”, “put the objects into [shape]”, and “build a [size] [shape] on the [x_position] of the table facing [rotation]”. Combining the enumerated combinations and sentence templates, we generated 3345 unique sentences. We then map all the sentences to pretrained sentence embeddings using a sentence transformer [52], specifically the all-MiniLM-L6-v2 model. Note that although the generated sentences could have grammar errors, the pretrained sentence transformer should be robust to those errors.

2) *Model and Training*: We modify the diffusion model to take in a single sentence embedding instead of the embeddings of individual word tokens. The sentence embedding from the pretrained model has a dimension of 384. We use a linear layer to map it to the same dimension as the point cloud embeddings. During training, we randomly sample a combination of the word tokens and map them to predefined sentence embeddings. Therefore, the model should be able to map ambiguous sentences (e.g., build a structure in the middle) to different structures (e.g., a line, a circle, and etc).

3) *Inference*: We evaluate the performance of the model using fully-specified sentences. As shown in Table A1, we see within 5% decrease in performance across structure types compared to the original model. Leveraging the generalization capability of the pretrained language model, the model should also be able to generalize to sentences not in the rearrangement training data at test time. We show some qualitative examples in Figure A1.

4) *Using the Large Language Model to Parse Natural Language*: We also explore the prompting techniques [23, 21] to map natural language to word tokens that our original model knows. Examples in Listing 1 shows that this is also a promising direction to bridge the gap between natural language and domain-specific commands.

Listing 1: Example GPT-3 Prompt to map natural language descriptions of target structures to word tokens that our model consumes. Prompt context is in gray, input language descriptions are green, and generated word tokens are purple.

```
Map each natural language instruction about an
object arrangement to available shape, size, and
location. If any element is not specified,
return NONE.

Available shapes: circle, line, table setting, tower
Available size: small, medium, large
Available location: top, middle, bottom

Example: rearrange the objects into a small circle.
Output: circle, small, NONE

Example: put the objects far apart and as a line
on top of the table.
Output: line, large, top

Example: there is a chair near the bottom edge
of the table. Arrange the dishes near the chair.
Output: table setting, NONE, bottom

Example: There is a teapot on the top right corner
of the table. Place the teacups surrounding it.
Output: circle, small, top
```

TABLE A1: Comparing the success rate % in simulation for the two types of language inputs. We observe a small drop in performance when we use a single sentence embedding comparing to individually embed each word tokens for each input sentence.

Language	Circle	Tower	Line	Table Setting
Word Tokens	65	72	72	66
Sentence	63	70	67	67

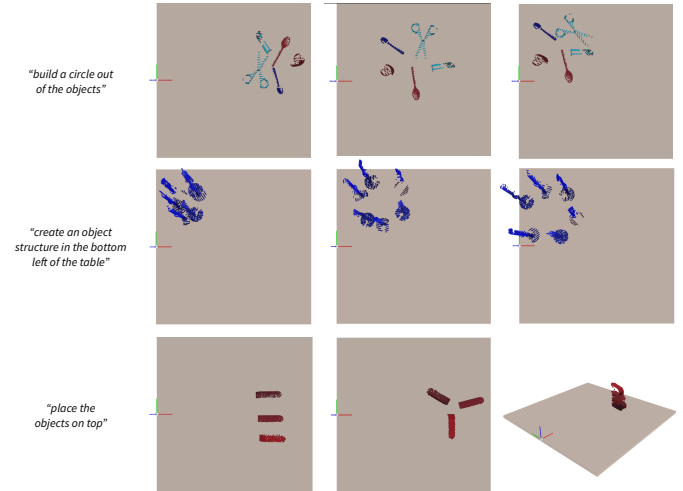


Fig. A1: Each row shows three sampled goals from the Diffusion model that is trained with procedurally generated sentences. The model generalizes to input sentences never seen during training by using a pretrained sentence encoder. Because the model is trained on language inputs at different levels of abstractions, the generated samples correspond strongly to structure properties specified in the sentences and at the same time show variations for the properties that are left unspecified.

TABLE A2: The percentage of failure cases due to different types of errors for different models averaged over all structure types.

Model	Intersection	Abnormal Velocity	Not Upright	Semantic Failure
StructDiffusion w/o Collision Discriminator	17.33	14.36	5.20	5.94
StructDiffusion	6.44	17.57	4.46	3.71
CVAE	28.71	14.11	5.20	5.69
CVAE w/ Collision Discriminator	20.05	12.62	3.22	3.47
Opt w/ CVAE Prior	17.33	13.12	5.69	6.93

TABLE A3: The percentage of failure cases due to different types of errors for StructDiffusion.

Structure Type	Intersection	Abnormal Velocity	Not Upright	Semantic Failure
Circle	4.95	24.75	4.95	0.00
Tower	12.87	8.91	4.95	8.91
Line	5.94	17.82	4.95	0.0
Table Setting	1.98	18.81	2.97	5.94

B. Classifier-Based Collision Guidance

In addition to utilizing a separate collision discriminator, we may directly integrate a classifier into the underlying sampling procedure of the diffusion model by using classifier based guidance. We present results in Table A4. We find that integrating a noisy collision discriminator with sampling can improve performance for towers and table settings but leads to limited gains for other structures.

C. Failure Mode Analysis

We analyze the failure modes for our model based on the simulation experiment. In Table A2, we first compare the number of failure cases in all simulation experiments for different methods. We observe that *StructDiffusion*, which combines a collision discriminator and a diffusion model, drastically reduces errors due to intersections between objects. Comparing CVAE and Optimization with CVAE prior, we also observe that optimizing the goal poses with a learned discriminator can reduce the intersection errors but at the expense of generating more semantically incorrect structures. In Table A3, we further break down the failure modes for our method on different structure types. Building circles tend to have a higher failure rate due to abnormal velocity because large circles may have objects occasionally placed outside of the table. In comparison, towers suffer less from this issue

because objects are more closely packed. Reasoning about the environment (e.g., the size of the table) is a necessary next step to address this issue.

D. Real World Results

Here we show results from each of the real-world experiments in Table I. Figure A3 shows some additional success examples. Success cases show how StructDiffusion allows us to rearrange previously-unseen objects that did not appear in the training data. Figure A2 includes examples of cases where our method fails. In summary, we see that most of our failures were due to simple motion planning or grasping issues; these could be solved in the future via better integration with task and motion planning algorithms, and with reactive replanning.

TABLE A4: Effect of Collision Classifier Guidance on success rate. This table shows the effect noisy classifier guidance on performance on each of the provided tasks. We do not see clear advantages to using the collision classifier.

Task	Guidance Weight				
	0	1	2	4	8
Circle	65	60	58	58	61
Tower	72	76	74	78	73
Line	72	73	69	73	69
Table Setting	66	70	73	66	70

“short line in the middle and center of the table”



Grasp slippage caused a motion planning failure.

“tower in the middle and center of the table”



Grasp slippage made the predicted goal pose invalid.

“set the table in the center left of the table”



The planned motion failed to avoid another object.

“set the table in the center left of the table”



The grasp failed.

“set the table in the center left of the table”



The grasp failed.

“set the table in the center left of the table”



The predict goal does not create enough space between the fork and the cup.

Fig. A2: Real-world failure cases from our robot experiments. A minority of failures come from StructDiffusion; the majority from grasping, planning, and execution failures. This points to future work integrating StructDiffusion with task and motion planning or with

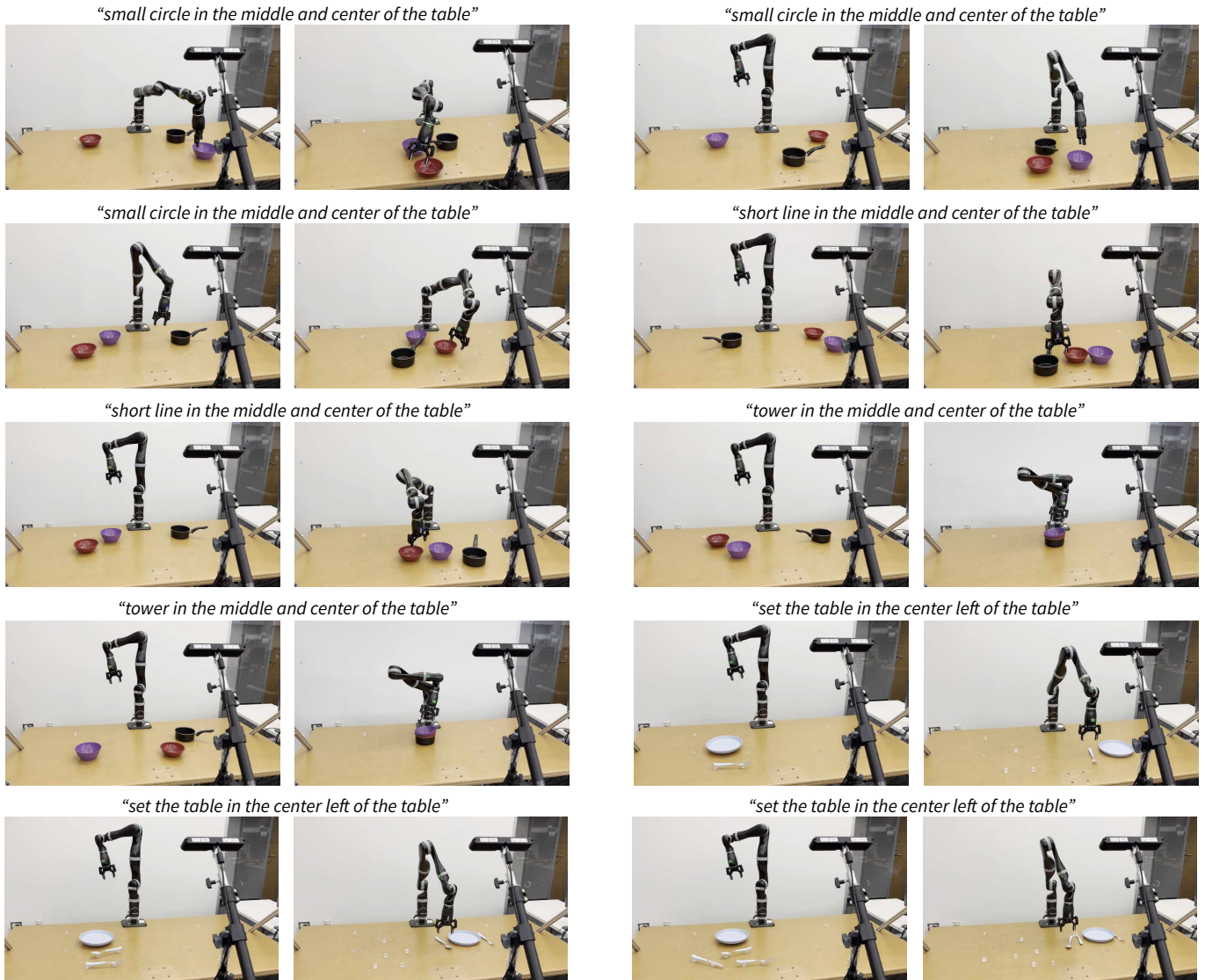


Fig. A3: Real-world success cases from our robot experiments. Out of 20 experiments, we saw 11 successes. Most failures were due to grasping and motion planning, not due to issues with our method. All experiments were performed on unseen objects, and StructDiffusion was not trained on any real-world training data.