CabiNet: Scaling Neural Collision Detection for Object Rearrangement with Procedural Scene Generation

Adithyavairavan Murali
NVIDIA
admurali@nvidia.com

Arsalan Mousavian
NVIDIA
amousavian@nvidia.com

Clemens Eppner
NVIDIA
ceppner@nvidia.com

Adam Fishman*
NVIDIA
afishman@nvidia.com

Dieter Fox*
NVIDIA
dieterf@nvidia.com

Abstract

We address the important problem of generalizing robotic rearrangement to clutter without any explicit object models. We first generate over 650K cluttered scenes—orders of magnitude more than prior work—in diverse everyday environments, such as cabinets and shelves. We render synthetic partial point clouds from this data and use it to train our CabiNet model architecture. CabiNet is a collision model that accepts object and scene point clouds, captured from a single-view depth observation, and predicts collisions for $\mathrm{SE}(3)$ object poses in the scene. Our representation has a fast inference speed of 7μs/query with nearly 20% higher performance than baseline approaches in challenging environments. We use this collision model in conjunction with a Model Predictive Path Integral (MPPI) planner to generate collision-free trajectories for picking and placing in clutter. CabiNet also predicts waypoints, computed from the scene's signed distance field (SDF), that allows the robot to navigate tight spaces during rearrangement. This improves rearrangement performance by nearly 35% compared to baselines. We systematically evaluate our approach, procedurally generate simulated experiments, and demonstrate that our approach directly transfers to the real world, despite training exclusively in simulation. Robot experiments in completely unknown scenes and objects are shown in the supplementary video.

1 Introduction

Object rearrangement is an important challenge in robotic manipulation and decision making [2, 21]. It requires the skills of picking, placing and generating complex collision-free motions in a cluttered environment. The Task-And-Motion-Planning (TAMP) [22, 49] and Embodied AI literature [13, 36, 52, 57] have reported impressive results of rearrangement in complex human environments like kitchens. But, they are largely limited to simulation [13, 21, 52] or make the strong assumption of state estimation of the environment [55] and objects [22] in the real world. Recent neural rearrangement methods [25, 39, 44, 47] generalize from sensed observations, without requiring state information, and have been demonstrated in the real world. Yet, they are limited to a specific type of scene that are not densely cluttered. Overall, there is no rearrangement system that generalizes to different challenging environments and works out of the box, or with minimal engineering effort, for each new scene type [27]. The cost of system integration, a major task of which is environment modelling, comes up to 3X of the price of the robot itself [3]. In this work, we aim to learn a single representation, trained over 650K scenes in simulation, that enables rearrangement in diverse unknown environments.

*Author is also affiliated with the University of Washington

Foundation Models for Decision Making Workshop at Neural Information Processing Systems, 2022.

One of the primary challenges in achieving generalization in object rearrangement is the limited availability of rearrangement datasets. Large data has been the driving force behind the success of visual learning [12] and large language models [48]. There have been some recent attempts at large-scale learning in simulation, such as the Habitat [52] and ManipulaTHOR [13] efforts. However, these actions are abstract and are not realistic. For example, objects simply stick to the gripper based on proximity, thereby sidestepping the complex dynamics of pick-and-place that are explicitly addressed in prior work in the robotic grasping literature [38, 43, 44, 51]. Additionally, the policies learned in these simulated environments are unproven in the real world. To facilitate performance in a physical system, our approach is conditioned on 3D point clouds instead of the RGBD representation commonly used in [13, 52]. Prior work [9, 44] has demonstrated that point clouds are an effective representation to transfer from simulation-based training to real world observations.

In robotic rearrangement, a fundamental component for planning is collision detection with an unknown environment. Classical TAMP methods typically rely on the complete geometric model of the scene for planning [21, 46] in the form of a triangular mesh or Signed Distance Field (SDF). Visual reconstruction systems such as SLAM [33], KinectFusion [45] and more recently NERF [41] are needed to generate a geometric model of the scene. Each system has its drawbacks, which may include a long start-up time [41, 45], multi-view requirements [33, 41], costly updates in dynamic scenes [33, 41, 45], or poor generalization [41]. Instead of explicitly reconstructing the scene for a traditional collision checker, recent neural methods speed up collision detection with learning [10, 32] and generalize to partial real-world observations [9, 44]. [44] proposed the CollisionNet model to predict collisions from scene point clouds for 6-DOF grasping. [9] extended this to an architecture that allowed fast collision checking from point clouds, enabling fast sampling-based placing in cluttered tabletop scenes. In this work, we extend this 3D implicit representation to scale to multiple cluttered environments, which we call CabiNet and learn a SDF-based waypoint sampler from this 3D representation. We then apply a Model Predictive Path Integral (MPPI) [19] algorithm on the GPU to use our CabiNet model to generate pick-and-place motion trajectories in clutter. In summary, our contributions are as follows:

Figure 1: CabiNet is able to perform complex rearrangement tasks in novel, cluttered scenes from just partial point cloud observations without object or environment models.

- Scaling up neural collision checking by 30X compared to prior work [9], training over nearly 60 billion collision queries. We also learn from over 650K cluttered scenes generated procedurally, which is six orders of magnitude more scene data than prior work on learning rearrangement in simulation [13]. We train a implicit 3D scene encoder CabiNet from this dataset.
- We demonstrate that CabiNet achieves fast collision detection inference of around 7μs/query and is 19.7% mAP higher than baselines when tested on 2.5 million queries in five diverse sets of environments. Using the same CabiNet encoding, we learn a scene SDF-based waypoint sampler and show that it is crucial for transitioning between pick and place actions.
- We demonstrate sim2real transfer for our model on completely unknown scenes and objects in the real world.

2 Related Work

Collision Detection from Point Clouds: There are a variety of options to check for collision with known object meshes or fully visible point clouds. One can use computational geometry libraries [46] if the object mesh is known. Alternatively, one can voxelize or spherize [29] the point

clouds and formulate the collision checking problem as evaluating whether any of the elements is in collision with robot links. However, voxel-based approaches suffer from occlusion which is mitigated through use of multiple views. This unfortunately constraints the robot workspace or requires mapping of the the environments which needs to be dynamically updated as objects are moving around. SceneCollisionNet [9] frames the collision checking problem as a hybrid of classical voxel based method and data driven methods. It encodes the scene to coarse voxels where each voxel is represented by a deep embeddding vector. Each collision query is defined as a pair of query object and scene point clouds. Collision checking is done with a binary classifier which takes as input the scene voxel embedding, object embedding, and the relative $\mathrm{SE}(3)$ transformation to that voxel. SceneCollisionNet was trained on only the table top scenes. In this paper, we build on top of SceneCollisionNet. By scaling up the training data to go beyond table top settings, we observe a boost in performance and generalization across variety of different scenes.

Neural Rearrangement Planning: Traditionally, solutions to object rearrangement have been dominated by model-based methods, such as TAMP [21] which use an explicit 3D world representation estimated from sensor observations. More recently, there has been an increase in learning-based vision-centric rearrangement approaches [2], although the majority assumes simplified action spaces that abstract away the actual grasping and placing motion and often focus on navigation. In [35] a learned visual state estimator is combined with a Monte-Carlo tree search planner, to efficiently solve planar tabletop rearrangements. When goals are provided as target images, transporter networks [60], their equivariant version [28] or goal-conditioned transporter networks [56] can be used for pick-and-place. The problem of matching object instances between goal and initial image can also be simplified with vision-language models [24]. Closest to our approach for rearrangement are NeRP [47] and IFOR [25].

Large-scale Procedural Scene Generation: Probabilistic models for indoor scene generation have been originally developed in computer graphics [15, 40, 58]. Apart from appealing visually, simulating scenes physically requires more care when arranging objects. As a result most data available for learning robot manipulation in simulation is limited to a fixed number of artist designed scenes: iGibson [36], Meta-World [59], RLBench [30], Sapien [57], Habitat [52], or AI2 ManipulaTHOR [13]. Those scenes mostly consist of assets and arrangements from datasets such as PartNet [42], ReplicaCAD [50], and 3D-Front [17]. More recently, ProcTHOR [11] has shown to procedurally generate large amounts of entire apartment layouts with room-specific object arrangements. Nevertheless, our use case focuses on smaller-scale clutter for which ProcTHOR's scenes are not dense enough. We will present our procedural scene generation pipeline next.

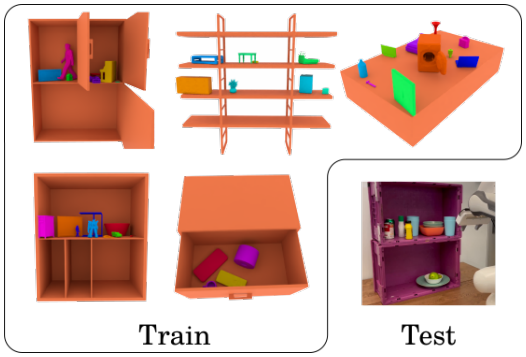

Figure 2: CabiNet is trained with over 650K synthetic scenes. We show that CabiNet transfers well to real-world settings, where we evaluated 24 different rearrangement problems on the real robot, each from a unique, cluttered scene.

3 Procedural Data Generation

Synthetic Scene Generation: We procedurally generate synthetic data in simulation. To generate our cluttered scenes, we first assemble a set of environment assets $\mathcal{E}$, object assets $\mathcal{O}$ and fixed robot manipulator $\mathcal{R}$ (Franka Panda in our case). We have a probabilistic grammar [31] P which dictates how the assets can be organized into random scene graphs $\mathcal{S} \sim P(\mathcal{E}, \mathcal{O}, \mathcal{R})$. This grammar is composed of the following key components: 1) sampling potential supports surfaces γ in an environment asset to place objects 2) rejection sampling to sequentially place objects on these surfaces without colliding with the scene and 3) fixing the robot base in a region where there is sufficient intersection over union (IoU > 0.8) between the workspace of the robot (approximated by a cuboid volume) and γ. Once the scene S is generated, collision queries are sampled with free-floating object meshes (computed in a straight-line trajectory) and the scene. The synthetic point clouds X

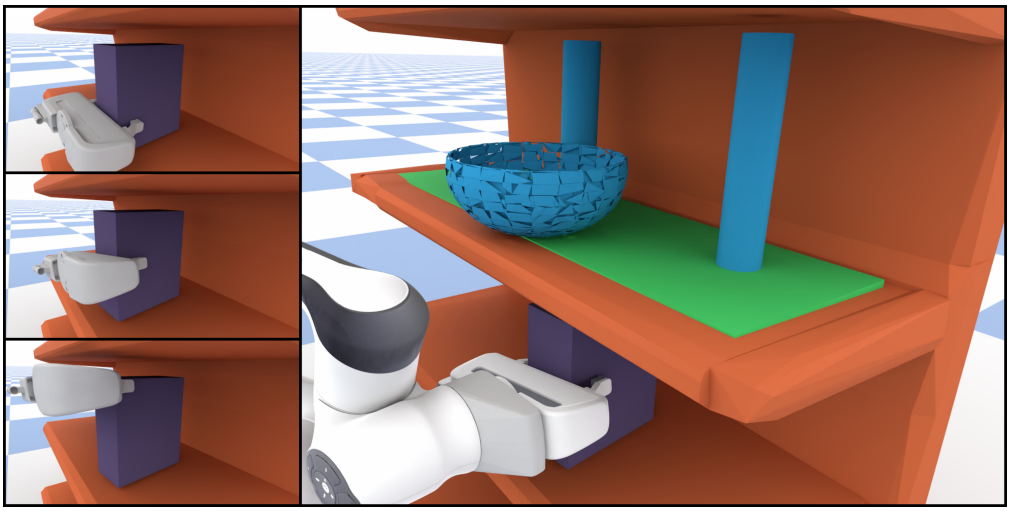

Figure 3: An example of a procedurally generated CabiNet rearrangement scene. The target object (here in purple) is chosen if it has a selection of valid collision-free grasp poses. The green region represents the placement shelf, which is chosen if a) it is different from the shelf the object originates from and b) has a valid placement pose for the target object.

are rendered online during training. The code for the scene generation will be publicly released upon publication.

Dataset: Our dataset of object assets for training comes from ACRONYM [14], that contains wide range of object geometries from 262 categories as well as high-quality $SE(3)$ grasps which we use for picking objects. We split the dataset for training and testing. Fig 2 depicts examples of our environment assets, which we chose from common categories such as shelves, cubby, cabinet, drawers and table. All the assets are procedurally generated with the exception of shelves. For shelves, we aggregate the shelf categories from ShapeNetCore[5] and filter assets which cannot be made watertight or if proper support surfaces cannot be extracted. Nonetheless, the object placements on all the environments, including the shelves, are procedurally generated. We include more examples of scenes in the appendix. For testing, we only consider assets from the shelf environment dataset and the objects are from a novel dataset [43] unseen during training.

Rearrangement Problem Generation: A valid rearrangment problem needs a scene, a target object that the robot needs to grasp and the placement shelf that the robot needs to place the object. Given a scene, a target object is sampled if there exists a set of ground truth grasps associated with that object where they do not collide with the environment and have valid collision free inverse kinematic configuration. Once the target object is sampled, we check if a collision free placement pose for the target object exists within the placement shelf. To make sure that our rearrangement problems are challenging, the placement shelf is going to be different from the shelf that has the pick object. A problem is chosen if it passes both stages of sampling target object and also having a valid placement location. Overall, this process has a success rate of 20.8% in finding successful problems. An example of rearrangement problem is shown in Fig 3.

4 Neural Rearrangement Planning

Our approach and model architecture is summarized in Fig 4. We first learn a implicit 3D encoding of the scene point cloud. We use the encoded scene feature along with learned object features for fast point-cloud based collision detection with CabiNet. The same scene feature is then used for predicting waypoints for the rearrangement task with a simple feedforward network. Both these models are then used to generate robot trajectories for object rearrangement with a Model Predictive Path Integral (MPPI) policy [19].

Collision Prediction: This is a learned collision model that accepts as input the scene point cloud X_S and the object point cloud X_O. The point cloud is encoded with voxelization and 3D convolution layers $\Psi_S = Enc(X_S)$. This is followed by a MLP binary classifier $c = g_\theta(\Psi_S, \Psi_O, T_{O\to S})$ that

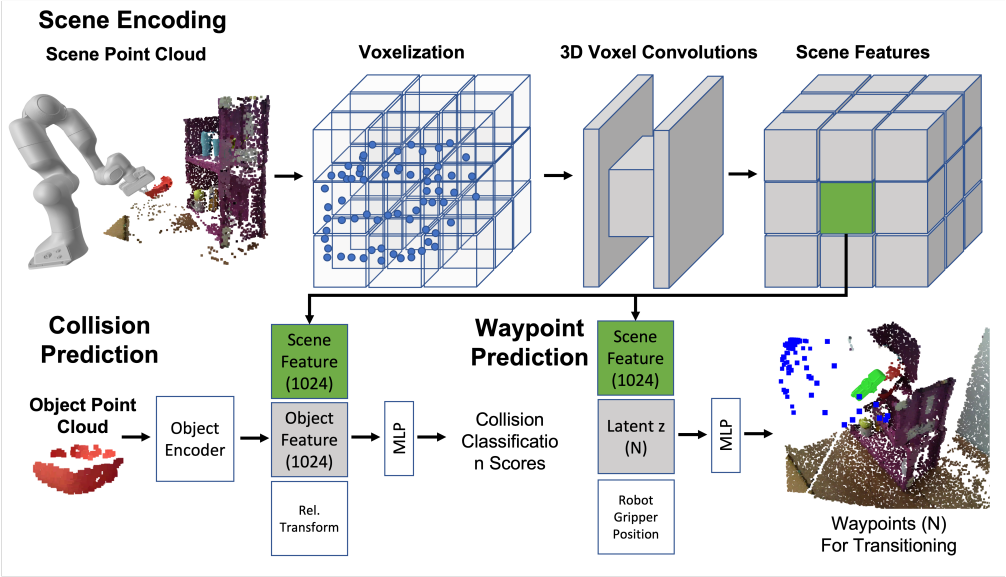

Figure 4: Our CabiNet architecture first encodes the scene point cloud with voxelization and 3D convolutions, shown in the top. The robot is only used for visualization and the robot point cloud is removed from the scene in practice. The scene features are then used with the object features to predict scene-object collision queries. We also predict waypoints (points colored in blue) for rearrangement, conditioned on latent vector z and the current gripper position (shown in green).

predicts if the object collide with the scene, where $T_{O \to S}$ is the relative transformation between the object and the scene and Ψ_O are the object features encoded with PointNet++ [6] layers akin to [9]. We adopt the model architecture of prior work [9] but it was only trained on table top scenes which results in poor generalization to other type of scenes such as shelves. We showed that by scaling up the training data to more diverse set of environments the model generalizes to different type of scenes. CabiNet is trained with binary cross entropy loss and with SGD with constant learning rate. Overall, we train CabiNet for two weeks, considering over 650K scenes and for 60 billion query pairs. We also modified the architecture of [9] by increasing the voxel sizes.

Waypoint Prediction: Object rearrangement in more constrained environments such as shelves imposes new challenges. Some of the approaches that work quite reliably in table top settings fail in navigating the tight spaces between shelves. One such approach is the work of [9] that finds the collision free path by rolling out multiple trajectories in configuration space (C-space) between the current configuration of the robot and the closest goal $\mathcal{G}$ in the C-space. These rollouts are sampled around a straight line that connects the current robot configuration to $\mathcal{G}$. Given the nominal line between the robot configuration and $\mathcal{G}$, different lines are sampled where the slope of the lines are gaussian distribution centered at the slope of nominal trajectory with a predefined variance. Each rollout is trimmed at the point of collision using SceneCollisionNet and ranked based on the distance of the rollout's final point to $\mathcal{G}$. The success of this approach hinges on the quality of the sampled trajectory. The simple sampling explained above, works quite well for table top scenes where there is more free space. However, it fails to sample promising trajectories for shelves and more constrained environments, such as when the robot needs to move from one compartment of the shelf to another. The majority of the sampled trajectories around the nominal trajectory would go through the divider between shelves.

To address this shortcoming, we propose to use CabiNet to sample waypoints with a larger signed distance value. Given the current end-effector position of the robot and the scene point cloud, it samples goals that gets the robot out of tight spaces. More formally, these waypoints $w \in \mathbb{R}^3$ are defined as to be in the set $\{w | \tau_{min} \leq SDF(S, w) \leq \tau_{max} \cap \|w - p_{gripper}\| \leq D\}$, where S is the scene mesh, $p_{gripper}$ is the end-effector position. The CabiNet waypoint sampler is modelled as a conditional generator $\hat{w} = f_\theta(\Psi_S, p_{gripper}, z)$ as in Generative Adversarial Networks [23]. Instead of using adversarial training, we train the sampler with Implicit Maximum Likelihood Estimation (IMLE) [37] which attempts to make each generated sample similar to a ground truth sample. We empirically found that making the loss bidirectional improved the generated samples

- enforcing that ground truth samples are also similar to the set of nearest predicted samples. Let $z_1, \ldots, z_m \sim \mathcal{N}(0, I)$ denote randomly sampled latent input noise vectors and w_i the ground truth waypoints. The IMLE loss is as follows:

$$\mathcal{L}_{IMLE} = \frac{1}{n}\sum_{i=1}^{n} \min_j |\hat{w}_j - w_i| + \frac{1}{m}\sum_{j=1}^{m} \min_i |\hat{w}_j - w_i| \tag{1}$$

In our experiment, we let $\tau_{min} = 0.40$, $\tau_{max} = 0.45$, $D = 0.40$ and both m and n are 70. We let z have two dimensions and train with SGD with a constant learning rate.

Object Rearrangement: We use contact graspnet [51] to sample 6-DOF grasps for the picking action. For placing objects, we first sample potential object positions based on the scene point cloud and the support surface. The placement orientation is set to the current object pose while it is being grasped. The poses are filtered by CabiNet for collisions with the environment and followed by whether a kinematic solution can be found for the manipulator.

5 Experimental Evaluation

5.1 Evaluation on Collision Benchmark

We evaluate CabiNet on a collision benchmark against four baseline point cloud-based collision detection algorithms. We want to emphasize that our setting only requires a point cloud observation from a single view. We sample synthetic scene/object point cloud pair where the objects move in 64 linear trajectories in a scene. The collision ground truth information is computed with FCL [46] in simulation. For each experiment, we have a balanced set of 256K collision and collision-free queries for a total of 512K queries/experiment. We evaluated on five environments (1000 scenes each) from Fig 2 and the results are averaged across them in Table 1.

Quantitative Metrics: We report the following 1) mean Average Precision (mAP) score for the classifier, averaged across the five environment test sets 2) collision prediction accuracy and 3) time/query in s. Our baselines are as follows:

- **SceneCollisionNet** [9]: We directly evaluated the pretrained model from prior work. This approach was just trained on a single environment (Tabletop) with a fixed robot-to-tabletop transformation, and directly infers collision from point clouds without any preprocessing.
- **Occupancy Mapping** [7]: This is one of the most commonly used geometric collision checking heuristic representation used in the robotics community [1, 4, 7]. We use the open source implementation from Open3D [61] to convert the sensed point cloud to a occupancy map representation. Voxels are specified to be of $1cm$ in size and are labelled to be either collision free or occupied.
- **Marching Cubes + FCL** [46]: We first convert the scene and object point clouds to a mesh with the marching cubes algorithm and use FCL to compute the collision between the scene and object meshes. We parallelize this baseline across 10 processes for a fairer comparison.
- **Marching Cubes + SDF** [18]: The scene point cloud is first converted to a mesh and we fit a SDF to it using the GPU implementation from [18]. Each point in the object point cloud is computed for its SDF value from the scene. If any of the points have a negative distance (due to a penetration), the entire scene/object point cloud is considered to be in collision.

Baseline Comparisons: Overall CabiNet outperforms the baselines methods in terms of both accuracy and inference speed. It has the highest mAP across the five environment test sets. It is nearly 24% and 15% higher mAP and accuracy respectively compared to OccupancyMap [7] which is a popular method in the community, while being nearly 25x faster with a 7μs inference time. OccupancyMap performance has direct correlation with the coverage of point cloud over occluded part of the scenes. The more occluded areas in the scene, the less accurate it becomes. CabiNet, on the other hand, does not suffer from the occlusion issue since it is trained with single camera and has been learned to extrapolate to occluded parts in order to solve collision queries. CabiNet also generalizes to more diverse environments and point data compared to the pretrained SceneCollisionNet [9] which shows the importance of training on diverse set of scenes and objects. Our approach is also about 4X faster than the parallelized FCL baseline with a 20.4% higher mAP score.

Table 1: Results on Collision Benchmark

Collision Model	**mAP**	**Accuracy** (%)	**Time/Query** (μs)
CabiNet (Ours)	**0.971**	**89.0**	**6.41 ± 3.58**
SceneCollisionNet [9]	0.706	69.9	7.03 ± 3.89
OccupancyMap [7]	0.732	74.1	174.5 ± 61.9
MC + FCL [46]	0.767	78.8	27.5 ± 6.82
MC + SDF [18]	0.773	80.0	168.6 ± 46.4

Table 2: CabiNet Generalization to Environments

Train Set	**Test Set (AP)**					
	Tabletop	Shelf	Cubby	Drawers	Cabinet	**mAP**
Tabletop	**0.989**	0.910	0.855	0.861	0.855	0.894
Shelf	0.924	**0.985**	**0.978**	0.956	0.972	0.963
Cubby	0.930	0.974	0.977	0.961	0.990	0.966
Drawers	0.923	0.856	0.848	**0.972**	0.914	0.903
Cabinet	0.924	0.971	0.961	0.961	**0.990**	0.961
All Envs	0.971	0.969	0.965	0.971	0.979	**0.971**

Ablation on Environments: We show in Table 2 that CabiNet generalizes to diverse environments by training with more in-distribution data. We notice that the model generalizes to similar environments even without any training, such as cabinets and shelves. The model trained on all the environments performed the best on all the test sets.

5.2 Object Rearrangement Evaluation in Simulation

We evaluate our collision model in simulated rearrangement trials in IssacGym [54] as shown in Table 3. To focus more on the rearrangement aspect of the problem, we used the ground truth grasp poses from [43]. We adopt a standard state-machine for rearrangement tasks used in prior works [8, 25, 47]. The objects are chosen from bowl, box, and cylinder categories of [43] and are held out from training data. For the environment assets, we use the seven shelves (from ShapeNet) in our test set to construct 30 scenes. We only use one fixed scene camera for all experiments and each scene gets two rearrangement trials, for a total of 60 experiments for each method.

Quantitative Metrics: We report three metrics on this task: 1) *overall success rate* is the success rate for the whole pick and place operation where the robot picks the target object and place it in the designated shelf without any failures. 2) *individual success rate* is the success rate of each state given the number of times the policy state machine reaches to each particular state and 3) *total time* taken for each experiment. There are three stages in rearrangement: pick, transitioning to a placing pose and place. A pick is considered a success if the object is grasped after lifting the object from the support surface. The same is true for transition. The placement is considered a success if the object is detected to be resting on the place support surface, regardless of its final orientation. We emphasize that rearrangement is a long-horizon task and conditional success rates for each state are based on the performance of the previous state. As a result, even if the performance of individual state success rates are high, errors accumulate over time leading to a lower overall success rate.

Collision Representation for Planning: We compared to Occupancy Mapping since it a popular heuristic collision representation used in the community[7]. The performance of all planners using this collision model significantly deteriorated compared to our CabiNet model. This shows the benefit of data driven approaches where they reason beyond the part of the point cloud that is visible and

Table 3: Simulated Rearrangement Experiments

Planner	**Collision Model**	**Waypoint Type**		
		CabiNet (Ours)	Reverse Approach	No Waypoints
MPPI	CabiNet (Ours)	**36.6%/159s**	16.7%/158s	4.3%/201s
MPPI	OccupancyMap [7]	15.0%/289s	11.0%/234s	7.5%/307s
AIT* [20, 26]	OccupancyMap [7]	21.0%/808s	18.5%/380s	5.0%/451s
RRTConnect [34]	OccupancyMap [7]	26.4%/357s	18.1%/380s	5.8%/389s

Table 4: Success Rate by States

Waypoint Strategy	States			
	Pick	Transition	Place	Overall
CabiNet (Ours)	**61.0%**	**80.0%**	**75.0%**	**36.6%**
Reverse Approach	54.8%	43.5%	70.0%	16.7%
No Waypoints	60.9%	21.4%	33.3%	4.3%

implicitly reason about occlusions as well. Occupancy maps is also significantly slower, increasing the rearrangement time by about 80% for the MPPI planner using the CabiNet waypoint sampler.

Comparison to Global Planning pipeline: We compare to the standard off-the-shelf motion planning pipeline used in the community with a Occupancy Mapping [7] representation. Specifically, we compare to the state-of-the-art configuration space planner [20], which is an almost-surely asymptotically optimal planner. We also compare to RRTConnect[34], which commonly is used to find feasible, though not necessarily optimal, paths. We give a timeout of $60s$ to find a solution for both planners. After planning, we apply spline-based, collision-aware trajectory smoothing [26] to the solutions. If the planner fails to find a valid solution, we simply execute the greedy solution to the goal to continue with the rearrangement process. Overall, the MPPI planner with our CabiNet model outperforms RRTConnect and AIT* by about 10% and 15% respectively and is also significantly faster.

Importance of Waypoints: We demonstrate that learning waypoints are crucial for rearrangement to navigate out of tight spaces. We compare to a common heuristic used in the motion generation literature to move robots out of tight spaces [53], which is to move the gripper in the reverse of the approach direction. We noticed that this approach, while proficient for primitive shapes (e.g. cylinders) it does not scale to more complex shapes like bowls, which have more complicated 6-DOF grasps. Hence, retracting in the reverse of the approach direction with an object in hand, the grasped object could collide with neighbouring support surfaces while transitioning from the pick to the place poses. As shown in Table 4, our CabiNet waypoint sampler improves the transition success rate by nearly 60% compared to when having no waypoints and about 35% when compared to the strategy of retracting in the opposite of the approach direction.

5.3 Real Robot Experiments

We run experiments to show that our model transfers to a real robot despite only being trained in simulation.

Hardware Setup: Experiments are done on a 7-DOF Franka Panda Robot with a parallel-jaw gripper. The system is equipped with two cameras: 1) Wrist mounted camera, which is an Intel Realsense D415 RGB-D camera, that is used for grasping 2) External camera, which is an Intel L515 RGB-D camera, that is used for generating the point cloud for CabiNet. Grasps are generated using Contact-GraspNet [51] and placement shelf is manually annotated for each problem by labeling the region that belongs to the desired placement shelf. Instance segmentation is run on both cameras. The user selects the target object by clicking on the external camera image. Upon user selection of target object, the robot takes a closer look at the object and find the object in the wrist camera through relative camera pose between wrist camera and external camera. This step is crucial for grasp success since the wrist camera provides denser points on the object and it mitigates the effect of calibration imperfections. CabiNet has access to only the external camera point cloud and the inference is run on NVIDA Titan RTX gpu.

Experiment Setup: We test our approach in novel environments, with unseen shelf and objects assets in unknown poses. We experiment with three objects from different object categories and two tasks. In the vertical transport task the robot has to pick an object from the top shelf and place it in the bottom one or vice versa. Similarly for horizontal transport task the shelf compartments are horizontally next to each other. For each task-object pair, we attempt four trials, two of which go from one support compartment to the next and two in the opposite direction. In total we have 24 experiments and each task-object pair has a unique environment, where the object has to be picked and placed amidst clutter.

Discussion: Results are reported in Table 5 and the vertical and horizontal transfer tasks have 75% and 58% success rates respectively. There were 2/24 pick failures due to incorrect grasps. One

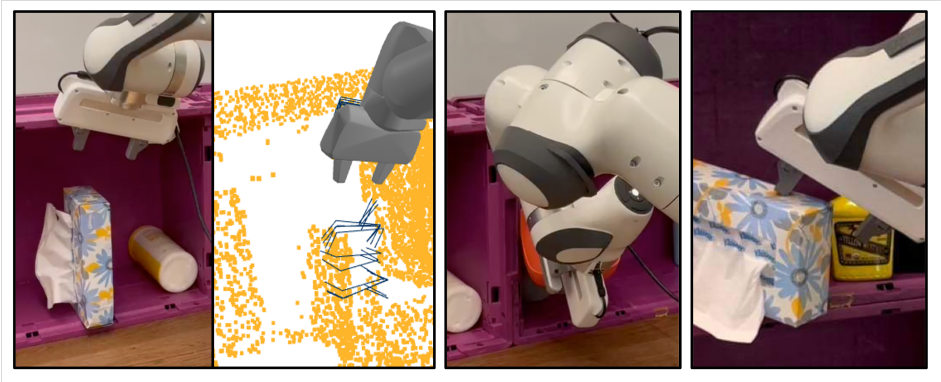

Figure 5: Examples of failure cases, left: roof partially occluded leading to collision with wrist camera during grasping, middle: grasped object collided with barrier, right: pick failure.

Table 5: Real Robot Experiments

Task	**Object Category**			
	Bowl	Box	Bottle	Overall
Vertical Transport	75.0%	50.0%	100.0%	75.0%
Horizontal Transport	50.0%	25.0%	100.0%	58.0%

attempt failed during placing and 5/24 attempts failed when transitioning from pick to place states. The horizontal transfer task was relatively more challenging due to two reasons - the transitioning was over a longer distance leading to a greater chance of collision and the leftmost shelf compartment was not entirely visible from our static scene camera. Three failures were specifically due to occlusion and the CabiNet model not seeing enough of the leftmost cubby geometry from our camera setup, as shown in Fig 5. The real robot executions are included in the supplementary video.

6 Conclusion

We present an effort in scaling up neural rearrangement in clutter. We train our CabiNet model to predict collisions and motion waypoints from point cloud observations. It outperforms baseline approaches in terms of collision predicted, simulated experiments and also transfers well to real world clutter despite being only trained in simulation. A limitation of our architecture is that the 3D voxelization enforces queries to be within the model workspace which can sometimes be out of bounds during manipulation. Potential extensions could also explore more complicated scenes or learning to generate synthetic scenes [31]. One could also explore hybrid architectures leveraging recent learned motion policies [16] that are faster than traditional planners, along with CabiNet.

References

[1] M. Bennewitz C. Stachniss A. Hornung, K. M. Wurm and W. Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. In *Autonomous Robots*, 2013.

[2] Dhruv Batra, Angel X Chang, Sonia Chernova, Andrew J Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, Manolis Savva, and Hao Su. Rearrangement: A challenge for embodied ai. 2020.

[3] Mathieu Belanger-Barrette. What is an average price for a collaborative robot? 2021.

[4] Constantinos Chamzas, Carlos Quintero-Pena, Zachary Kingston, Andreas Orthey, Daniel Rakita, Michael Gleicher, Marc Toussaint, and Lydia E. Kavraki. Motionbenchmaker: A tool to generate and benchmark motion planning datasets. In *IEEE Robotics and Automation Letters*. IEEE, 2021.

[5] A.X. Chang, T. Funkhouser, L. Guibas, Pat Hanrahan, Q. Huang, Z Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository.

Technical report, Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[6] Hao Su Charles R Qi, Li Yi and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Neural Information Processing Systems (NeurIPS)*, 2017.

[7] Sachin Chitta, Ioan Sucan, and Steve Cousins. Moveit![ros topics]. *IEEE Robotics & Automation Magazine*, 19(1):18–19, 2012.

[8] Michael Danielczuk, Matthew Matl, Saurabh Gupta, Andrew Li, Andrew Lee, Jeff Mahler, and Ken Goldberg. Segmenting unknown 3d objects from real depth images using mask r-cnn trained on synthetic data. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2019.

[9] Michael Danielczuk, Arsalan Mousavian, Clemens Eppner, and Dieter Fox. Object rearrangement using learned implicit collision functions. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6010–6017, 2021.

[10] Nikhil Das and Michael Yip. Learning-based proxy collision detection for robot motion planning applications. 2021.

[11] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Jordi Salvador, Kiana Ehsani, Winson Han, Eric Kolve, Ali Farhadi, Aniruddha Kembhavi, et al. Procthor: Large-scale embodied ai using procedural generation. *arXiv preprint arXiv:2206.06994*, 2022.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, and Jakob Uszkoreit andNeil Houlsby. Attention is all you need. In *In International Conference on Learning Representations*, 2021.

[13] Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Manipulathor: A framework for visual object manipulation. In *CVPR*, 2021.

[14] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. ACRONYM: A large-scale grasp dataset based on simulation. In *2021 IEEE Int. Conf. on Robotics and Automation, ICRA*, 2020.

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

[16] Adam Fishman, Adithyavairavan Murali, Clemens Eppner, Bryan Peele, Byron Boots, and Dieter Fox. Motion policy networks. 2022.

[17] 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.

[18] Clement Fuji Tsang, Maria Shugrina, Jean Francois Lafleche, Towaki Takikawa, Jiehan Wang, Charles Loop, Wenzheng Chen, Krishna Murthy Jatavallabhula, Edward Smith, Artem Rozantsev, Or Perel, Tianchang Shen, Jun Gao, Sanja Fidler, Gavriel State, Jason Gorski, Tommy Xiang, Jianing Li, Michael Li, and Rev Lebaredian. Kaolin: A pytorch library for accelerating 3d deep learning research. https://github.com/NVIDIAGameWorks/kaolin, 2022.

[19] A. Aldrich G. Williams and E. A. Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 2017.

[20] Jonathan D. Gammell, Siddhartha S. Srinivasa, and Tim D. Barfoot. Batch informed trees (bit*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3067–3074, 2015.

[21] Caelan Reed Garrett, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated task and motion planning. *Annual Review of Control, Robotics, and Autonomous Systems*, 2021.

[22] Caelan Reed Garrett, Chris Paxton, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Dieter Fox. Online replanning in belief space for partially observable task and motion problems. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

[23] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Neural Information Processing Systems*, 2014.

[24] Walter Goodwin, Sagar Vaze, Ioannis Havoutis, and Ingmar Posner. Semantically grounded object matching for robust robotic scene rearrangement. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 11138–11144. IEEE, 2022.

[25] Ankit Goyal, Arsalan Mousavian, Chris Paxton, Yu-Wei Chao, Brian Okorn, Jia Deng, and Dieter Fox. Ifor: Iterative flow minimization for robotic object rearrangement. 2022.

[26] Kris K. Hauser and Victor Ng-Thow-Hing. Fast smoothing of manipulator trajectories using optimal bounded-acceleration shortcuts. *2010 IEEE International Conference on Robotics and Automation*, pages 2493–2498, 2010.

[27] John Horst, Jeremy Marvel, and Elena Messina. Best practices for the integration of collaborative robots into workcells within small and medium-sized manufacturing operations. *NIST Advanced Manufacturing Series*, 100(41), 2021.

[28] Haojie Huang, Dian Wang, Robin Walter, and Robert Platt. Equivariant transporter network. *arXiv preprint arXiv:2202.09400*, 2022.

[29] Philip M Hubbard. Approximating polyhedra with spheres for time-critical collision detection. *ACM Transactions on Graphics (TOG)*, 15(3):179–210, 1996.

[30] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020.

[31] Amlan Kar, Aayush Prakash, Ming-Yu Liu, Eric Cameracci, Justin Yuan, Matt Rusiniak, David Acuna, Antonio Torralba, and Sanja Fidler. Meta-sim: Learning to generate synthetic datasets. In *ICCV*, 2019.

[32] J. Chase Kew, Brian Ichter, Maryam Bandari, Tsang-Wei Edward Lee, and Aleksandra Faust. Neural collision clearance estimator for batched motion planning. 2021.

[33] Matthew Klingensmith, Siddartha Sirinivasa, and Michael Kaess. Articulated robot motion for simultaneous localization and mapping (arm-slam). In *Robotics and Automation Letters*. IEEE, 2016.

[34] James Kuffner and Steven M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *International Conference on Robotics and Automation (ICRA)*. IEEE, 2000.

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

[36] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Elliott Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, Andrey Kurenkov, Karen Liu, Hyowon Gweon, Jiajun Wu, Li Fei-Fei, and Silvio Savarese. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. In *5th Annual Conference on Robot Learning*, 2021.

[37] Ke Li and Jitendra Malik. Implicit maximum likelihood estimation. *arXiv preprint arXiv:1809.09087*, 2018.

[38] Jeffrey Mahler and Ken Goldberg. Learning deep policies for robot bin picking by simulating robust grasping sequences. *Conference on Robot Learning*, 2017.

[39] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *RSS*, 2017.

[40] Lucas Majerowicz, Ariel Shamir, Alla Sheffer, and Holger H Hoos. Filling your shelves: Synthesizing diverse style-preserving artifact arrangements. *IEEE transactions on visualization and computer graphics*, 20(11):1507–1518, 2013.

[41] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[42] Kaichun Mo, Shilin Zhu, Angel X. Chang, Li Yi, Subarna Tripathi, Leonidas J. Guibas, and Hao Su. PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[43] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. 6-DOF GraspNet: Variational grasp generation for object manipulation. *International Conference on Computer Vision*, 2019.

[44] Adithyavairavan Murali, Arsalan Mousavian, Clemens Eppner, Chris Paxton, and Dieter Fox. 6-dof grasping for target-driven object manipulation in clutter. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[45] Richard Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. *2011 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.

[46] Jia Pan, Sachin Chitta, and Dinesh Manocha. Fcl: A general purpose library for collision and proximity queries. *2012 IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[47] Ahmed H Qureshi, Arsalan Mousavian, Chris Paxton, Michael Yip, and Dieter Fox. Nerp: Neural rearrangement planning for unknown objects. *RSS*, 2021.

[48] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *arXiv:2103.00020*, 2021.

[49] Siddharth Srivastava, Eugene Fang, Lorenzo Riano, Rohan Chitnis, Stuart Russell, and Pieter Abbeel. Combined task and motion planning through an extensible planner-independent interface layer. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[50] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

[51] Martin Sundermeyer, Arsalan Mousavian, Rudolph Triebel, and Dieter Fox. Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes. 2021.

[52] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[53] Karl Van Wyk, Mandy Xie, Anqi Li, Muhammad Asif Rana, Buck Babich, Bryan Peele, Qian Wan, Iretiayo Akinola, Balakumar Sundaralingam, Dieter Fox, Byron Boots, and Nathan D. Ratliff. Geometric fabrics: Generalizing classical mechanics to capture the physics of behavior. *IEEE Robotics and Automation Letters*, 7(2):3202–3209, 2022.

[54] Yunrong Guo Michelle Lu Kier Storey Miles Macklin David Hoeller Nikita Rudin Arthur Allshire Ankur Handa Gavriel State Viktor Makoviychuk, Lukasz Wawrzyniak. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv:2108.10470*, 2021.

[55] Kentaro Wada, Stephen James, and Andrew J. Davison. ReorientBot: Learning object reorientation for specific-posed placement. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[56] Hongtao Wu, Jikai Ye, Xin Meng, Chris Paxton, and Gregory Chirikjian. Transporters with visual foresight for solving unseen rearrangement tasks. *arXiv preprint arXiv:2202.10765*, 2022.

[57] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao

Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[58] Lap-Fai Yu, Sai-Kit Yeung, and Demetri Terzopoulos. The clutterpalette: An interactive tool for detailing indoor scenes. *IEEE transactions on visualization and computer graphics*, 22(2):1138–1148, 2015.

[59] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019.

[60] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. *arXiv preprint arXiv:2010.14406*, 2020.

[61] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.