Learning Gradient Fields for Object Rearrangement

Anonymous Author(s)
Affiliation
Address
email

Abstract

Object Rearrangement is to move objects from an initial state to a goal state. Here, we focus on a more practical setting in object rearrangement, *i.e.*, rearranging objects from shuffled layouts to a normative target distribution without explicit goal specification. However, it remains challenging for AI agents, as it is hard to describe the target distribution (goal state) for reward engineering or collect expert trajectories as demonstrations. Hence, it is infeasible to directly employ reinforcement learning or imitation learning algorithms to address the task. This paper aims to search for a policy only with a set of examples from a target distribution instead of a handcrafted reward function. We employ the score-matching objective to train a *target gradient field (TarGF)*, indicating a direction on each object to increase the likelihood of the target distribution. For object rearrangement, the TarGF can be used in two ways: 1) For model-based planning, we can cast the target gradient into a reference control and output actions with a distributed path planner; 2) For model-free reinforcement learning, the TarGF is not only used for estimating the delta likelihood as a reward but also provides suggested actions in residual policy learning. Experimental results in ball rearrangement and room rearrangement demonstrate that our method significantly outperforms the state-of-the-art methods in the quality of the terminal state, the efficiency of the control process, and scalability.

1 Introduction

Object rearrangement [1, 2] is to manipulate a set of objects from an initial layout to a normative distribution, as shown in Fig. 1. This task is taken for granted by humans [3] and widely exists in our daily life, such as tidying up a table [4], placing furniture [5], and sorting parcels [6].

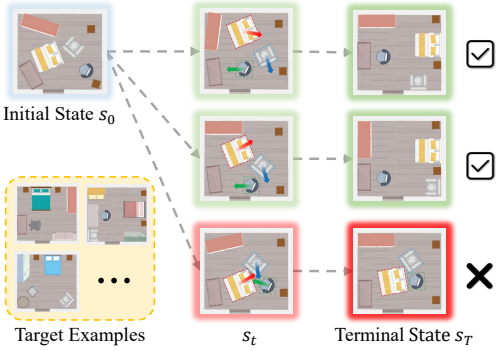

Figure 1: Example-based Object Rearrangement: Given a set of target examples, the agents need to move objects to meet a normative distribution.

However, compared with conventional robot tasks [7, 8, 9, 10], there are two critical challenges in rearrangement tasks: 1) Hard to clearly define the target state (goal), as the target distribution is diverse and the patterns are difficult to describe in the program, *e.g.*, how to evaluate the *cleanness* quantitatively? As a result, it is difficult to manually design a reward/objective function, which is essential to modern control methods, *e.g.*, deep reinforcement learning (DRL) [11]. 2) In the physical world, the agents also need to find an executable path to reach the target state safely and efficiently, *i.e.*, finding a short yet collision-free path to reach the target distribution. To this end, the agents are required to jointly learn to evaluate the quality of the state (reward) and take actions to move objects.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

In other words, the agents should answer the two questions: *What kind of layouts does the user prefer*, and *How to manipulate objects to meet the target distribution*?

Thus, it is necessary to find a way to build an agent without the dependence on the explicit goal state and shaped reward. Imitation learning (IL) can implicitly learn the reward function from expert trajectories [12, 13, 14]. However, collecting a large number of expert trajectories is expensive. Recently, example-based reinforcement learning (RL) [15, 16] has tried to reduce the reliance on expert trajectories, *i.e.*, learning a policy only with examples of successful outcomes. However, it is still difficult to learn a classifier to provide accurate reward signals for learning in the case of high-dimensional state and action space, as there are many objects to control in object rearrangements.

In this work, we separate the task into two stages: 1) learning a score function to qualify the state, and 2) learning/building a policy to rearrange objects with the score function. Inspired by the recent advances in score-based generative models [17, 18, 19, 20, 21], we employ the score-matching objective to learn a *target gradient field (TarGF)* by perturbing the well-arranged examples. The *target gradient field* can be viewed as object-wise guidance that provides a pseudo velocity on each object, pointing to regions with a higher density of the target distribution, *e.g.*, cleaner configurations. To manipulate objects in the physical world, we further demonstrate two ways to leverage the target gradient field in control. 1) We integrate the guidance with a model-based path planner to rearrange objects without collision. 2) We further derive the target gradient to a delta log-likelihood as a reward function, key ingredients to reinforcement learning. With the TarGF-based reward and a TarGF-guided residual policy network [22], we can train an effective RL-based policy from interactions.

In the experiments, we evaluate the effectiveness of our methods on two typical rearrangement tasks: *Ball Rearrangement* and *Room Rearrangement*. First, we introduce three metrics to analyse the quality of the terminal state and the control process. In *Ball Rearrangement*, we demonstrate that both of our control methods can efficiently reach a target distribution with high diversity and realism. We also observe that our methods can find a path with fewer collisions and fewer steps to converge compared with learning-based baselines [13, 16] and planning-based baselines [23]. In *Room Rearrangement*, we further evaluate the feasibility of our methods in complex scenes where there are multiple heterogeneous objects.

Our contributions are summarised as follows:

- We introduce a score-based method to learn a *target gradient field (TarGF)* by perturbing the state of well-arranged target examples.
- We point out two usages of the target gradient field in object rearrangement: 1) providing guidance in moving direction; 2) deriving a reward for learning.
- We conduct experiments to demonstrate that both traditional planning and RL methods can benefit from the target gradient field and significantly outperform the state-of-the-art methods in sample diversity, realism and safety.

2 Related Works

2.1 Object and Scene (Re-)Arrangements

The object arrangement has been previously studied by researchers from the robotics community [24, 2, 25] and the graphics community [26, 27, 28, 29, 30]. However, most of them focus on manually designing rules/ energy functions to find a goal [24, 2, 25] or synthesising a scene configuration [27, 28, 29, 30] that satisfies the preference of the user. The most recent work [4] tries to learn a GNN to output an arrangement tailored to human preferences. However, they neglect the physical constraints during the process. Hence, the accessibility and transition cost from the initial and target states are not guaranteed. Considering the physical interaction, the recent works on scene rearrangement [31, 32] mitigate the goal specification problem by providing *a goal scene* to the agent, making the reward shaping feasible. Concurrent works [33, 34] also notice the necessity of automatic goal inference for tying rooms and exploit the commonsense knowledge from Large Language Model (LLM) or memex graph to infer rearrangements goals when the goal is unspecified. Another concurrent work [35] aims at discovering generalisable spatial goal representations via graph-based active reward learning for 2D object rearrangement. In this paper, we focus on a more fundamental problem in rearrangement: *how to manipulate objects to maximise the similarity (i.e., the target likelihood) between the current state and the example sets*. Here, only a set of positive target examples are required to learn the gradient field, rather than the prior knowledge about specific scenarios.

2.2 Learning without Reward Engineering

Without reward, previous works explore algorithms to learn a control policy from expert demonstrations via imitation learning [36, 37, 38, 12] or inverse reinforcement learning [39, 14, 40, 41]. Imitation learning (IL) [36, 37, 38] aims to directly optimise the policy by cloning the behaviour from the expert trajectories. The inverse reinforcement learning (IRL) [39, 14, 40, 41] tries to learn a reward function from data for reinforcement learning. Considering the difficulty in collecting expert trajectories, there are some example-based methods proposed to learn policy only with a set of successful examples [15, 39, 13, 16]. The most recent work is RCE [16] which directly learns a recursive classifier from successful outcomes and interactions for policy optimisation. We also try to find an example-based control method for rearrangement with a set of examples. Our method can learn the target gradient fields from the examples, without any additional efforts in reward engineering. The target gradient fields can provide meaningful reward signals and action guidance for reinforcement learning and traditional planners. To the best of our knowledge, such usage of gradients fields is not explored in previous works.

3 Problem Statement

We aim to learn a policy that moves objects to reach states close to a target distribution without relying on reward engineering. Similar to the *example-based control* [16], the grounded *target distribution* $p_{tar}(\mathbf{s})$ is unknown to the agent, and the agent is only given a set of *target examples* $S^* = \{\mathbf{s}^*\}$ where $\mathbf{s}^* \sim p_{tar}(\mathbf{s})$. In practice, these examples can be provided by the user without accessing the dynamics of the robot. The agent starts from an initial state $\mathbf{s}_0 \sim p_0(\mathbf{s}_0)$, where N objects are randomly placed in the environment. At time step t, the agent takes action $\pi(\mathbf{a}_t|\mathbf{s}_t)$ imposed on the objects with dynamics $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$. Here, the goal of the agent is to search for a policy $\pi^*(\mathbf{a}_t|\mathbf{s}_t)$ that maximises the *discounted log-target-likelihood* of the future states:

$$\pi^* = \arg\max_{\pi} E_{\rho(\mathbf{s}_0),\tau\sim\pi}\left[\sum_{\mathbf{s}_t\in\tau} \gamma^t \log p_{tar}(\mathbf{s}_t)\right] \tag{1}$$

where $\gamma \in (0, 1]$ denotes the discount factor. Notably, we assume that $p_{tar}(\mathbf{s}) > 0$ everywhere, since we can perturb the original data with a tiny noise (*e.g.*, $\mathcal{N}(0, 0.00001)$) to ensure the perturbed density is positive everywhere. [17] also used this trick to tackle manifold hypothesis issue.

This objective reveals three challenges: **1) Inaccessibility problem:** The grounded target distribution p_{tar} in Eq. 1 is inaccessible. Thus, we need to learn a function to approximate the *log-target-likelihood* for policy search. **2) Sparsity problem:** The log-target-likelihood is sparse in the state space, as mentioned in [17]. Hence, even if we have access to the target distribution p_{tar}, it is still difficult to explore the high-density region due to the *Manifold Hypothesis* mentioned in [17]. **3) Adaptation problem:** The dynamics $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ is also inaccessible. To maximise the cumulative sum in Eq. 1, the agent is required to adapt to the dynamics to efficiently increase the target likelihood.

To address these problems, we partition the task into 1) learning to estimate the log-target-likelihood (reward) of the state and 2) learning/building a policy to rearrange objects to adapt to the dynamics.

4 Method

4.1 Motivation

We argue that estimating the *gradients of the log-target-likelihood* $\nabla_{\mathbf{s}} \log p_{tar}(\mathbf{s})$ can tackle the first two difficulties mentioned in Sec. 3: For the *inaccessibility* problem, we can approximate the reward increment between two adjacent states by the first-order Taylor expansion using the gradient field $\log p_{tar}(\mathbf{s}_{t+1}) - \log p_{tar}(\mathbf{s}_t) \approx \langle \nabla_{\mathbf{s}} \log p_{tar}(\mathbf{s}_t), \mathbf{s}_{t+1} - \mathbf{s}_t \rangle$. As shown in Sec. 4.4, this helps to derive a surrogate objective of Eq. 1.

For the *sparsity* problem, we can leverage the gradient field to help with exploration since the gradient $\nabla_{\mathbf{s}} \log p_{tar}(\mathbf{s})$ indicates the direction that points to the higher density region of the target distribution p_{tar}. To estimate the gradients of the log-target-likelihood $\nabla_{\mathbf{s}} \log p_{tar}(\mathbf{s})$, noted as the *target gradient field (TarGF)* $\mathbf{\Phi}_{tar}$, we adopt the deep score-matching technique [18] that has achieved impressive results in many research areas recently[19, 21, 20].

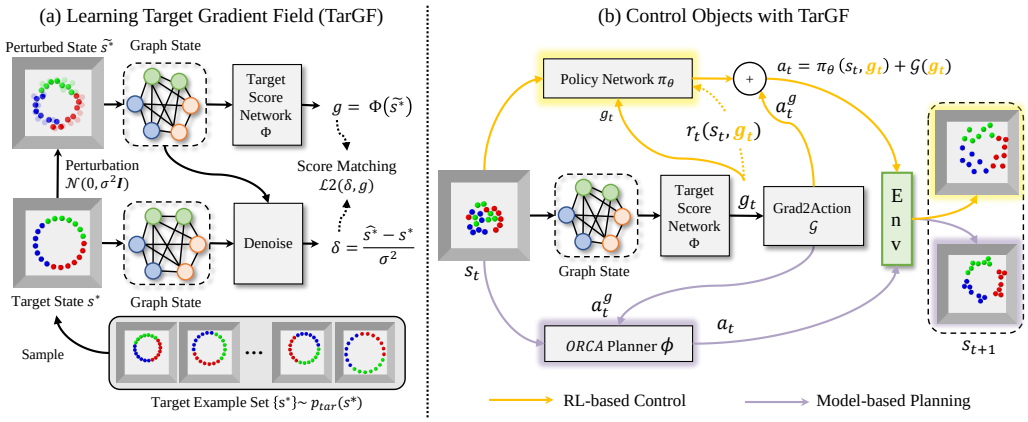

Figure 2: An overview of our method. (a) We train the target score network via the score-matching objective. The target examples from the target distribution are first disturbed by a Gaussian kernel. The target score network is forced to match the denoising direction of the perturbation. (b) Our framework is based on the trained target score network for rearrangement tasks. In the model-free setting, the TarGF provides exploration guidance and reward estimation for RL. In the model-based setting, the TarGF provides reference control for a model-based planner based on ORCA. The planner then outputs a collision-free action based on the reference control.

To address the *adaptation* problem, we further demonstrate two ways to incorporate the trained target gradient field $\mathbf{\Phi}_{tar}$ with control algorithms for object rearrangement: 1) For the model-based setting, our framework casts the target gradient into a reference control and outputs actions with a distributed path planner. 2) For the model-free setting (reinforcement learning), we leverage the target gradient to estimate reward and provide suggested actions in residual policy learning [22].

In the followings, we will first introduce how to train the target gradient field $\mathbf{\Phi}_{tar}$ via score-matching in Sec. 4.2 and then introduce our rearrangement framework under the model-based and model-free settings in Sec. 4.3 and Sec. 4.4 respectively.

4.2 Learning the Target Gradient Field from Examples

The *target score network* $\mathbf{\Phi}_{tar} : \mathcal{S} \to \mathcal{S}$ is the **core module** of our framework, which aims at estimating the *score function*(*i.e.*the gradient of log-density) of the target distribution $\nabla_{\mathbf{s}} \log p_{tar}(\mathbf{s})$.

To train the target score network, we adopt the Denoising Score-Matching (DSM) objective proposed by [42], which can guarantee a reasonable estimation of the $\nabla_{\mathbf{s}} \log p_{tar}(\mathbf{s})$. In training, we pre-specify a noise distribution $q_\sigma(\widetilde{\mathbf{s}}|\mathbf{s})$ on the collected target state $\mathbf{s}$. Then, we match the output of the target gradient field with the score of the perturbed target distribution $q_\sigma(\widetilde{\mathbf{s}}) = \int q_\sigma(\widetilde{\mathbf{s}}|\mathbf{s}) p_{tar}(\mathbf{s}) d\mathbf{s}$:

$$\frac{1}{2}\mathbb{E}_{q_\sigma(\widetilde{\mathbf{s}}|\mathbf{s}), p_{tar}(\mathbf{s})}\left[||\mathbf{\Phi}_{tar}(\widetilde{\mathbf{s}}) - \nabla_{\widetilde{\mathbf{s}}} \log q_\sigma(\widetilde{\mathbf{s}}|\mathbf{s})||_2^2\right] = \frac{1}{2}\mathbb{E}_{q_\sigma(\widetilde{\mathbf{s}}|\mathbf{s}), p_{tar}(\mathbf{s})}\left[||\mathbf{\Phi}_{tar}(\widetilde{\mathbf{s}}) - \frac{\mathbf{s} - \widetilde{\mathbf{s}}}{\sigma^2}||_2^2\right] \quad (2)$$

The DSM objective guarantees the optimal score network satisfies $\mathbf{\Phi}^*_{tar}(\mathbf{s}) = \nabla_{\mathbf{s}} q_\sigma(\mathbf{s})$ almost surely. When σ is small enough, we have $\nabla_{\mathbf{s}} q_\sigma(\mathbf{s}) \approx \nabla_{\mathbf{s}} \log p_{tar}(\mathbf{s})$, so that $\mathbf{\Phi}^*_{tar}(\mathbf{s}) \approx \nabla_{\mathbf{s}} \log p_{tar}(\mathbf{s})$.

In practice, we adopt a variant of DSM proposed by [18], which conducts DSM under different noise scales simultaneously. We implement the target score network as a graph-based network $\mathbf{\Phi}_{tar}(\mathbf{s}, t)$ that is conditioned on a noise level $t \in (0, 1]$. The input states $= [\mathbf{s}^1, \mathbf{s}^2, ..., \mathbf{s}^N]$ is constructed as a fully-connected graph that takes each object's state(*e.g.*position, orientation) $\mathbf{s}^i \in \mathbb{R}^{f_s}$ and static properties (*e.g.*category, bounding-box) $\mathbf{p}^i \in \mathbb{R}^{f_p}$ as the node feature $[\mathbf{s}^i, \mathbf{p}^i] \in \mathbb{R}^{f_s + f_p}$. The input graph is pre-processed by linear initial feature extraction layers and then passed through two Edge-Convolution layers. After message-passing, the output on each node serves as the component of the target gradient on the i-th object's state $\mathbf{\Phi}^i_{tar}(\mathbf{s}) \in \mathbb{R}^{f_s}$. We defer structure details to supplementary.

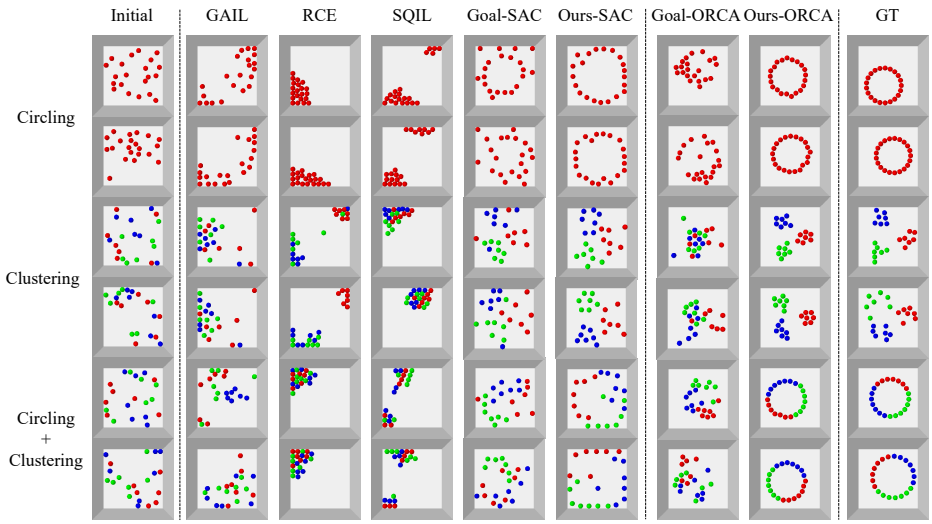

Figure 3: Qualitative results of each method on ball rearrangement tasks. We selected the two most representative results for each method.

4.3 Model-based Planning with the Target Gradient Field

The target gradient $\mathbf{g} = \mathbf{\Phi}_{tar}(\mathbf{s})$ can be translated into a *gradient-based action* $\mathbf{a}^g = \mathscr{G}(\mathbf{g})$, where $\mathscr{G}$ denotes the gradient-to-action translation. A gradient over the state space can be viewed as velocities imposed on each object. For instance, if the state of each object is a 2-dimensional position $\mathbf{s}_i = [x, y]$, then the target gradient component on the i-th object can be viewed as linear velocities on two axes $\mathbf{\Phi}_{tar}(\mathbf{s})^i = [v_x, v_y]$. When the action space is also velocities, we can construct $\mathbf{a}^g$ by simply projecting the target gradient $\mathbf{a}^g = \mathscr{G}(\mathbf{g}) = \frac{v_{max}}{||\mathbf{g}||_2} \cdot \mathbf{g}$ into the action space, where the v_{max} denotes the speed limit. Following the gradient-based action, objects will move towards directions that may increase the likelihood of the next state, which meets the need of the rearrangement task.

However, the gradient-based action *is imperfect*(*e.g.*leading to collisions between the objects), as the target gradient has no access to the information of environment dynamics. This may be inefficient and unsafe due to the severe collisions between objects.

To adapt the gradient-based action to the environment dynamics, we incorporate the target gradient field with a model-based off-the-shelf collision-avoidance algorithm *ORCA*. The ORCA planner ϕ_θ takes the gradient-based action $\mathbf{a}^g$ as reference velocities and then outputs the collision-free velocities $\mathbf{a} = \phi_\theta(\mathbf{a}^g)$ with the least modification to the reference velocities. In this way, the adapted action can lead objects to move towards a higher likelihood region efficiently and safely. For more details of the ORCA planner, we defer to supplementaries.

4.4 Learning Policy with the Target Gradient Field

We further propose a framework based on the model-free reinforcement learning (RL) algorithm to accomplish the rearrangement tasks in the model-free scenario, where the agent needs to adapt the dynamics via interacting with the environment.

To tackle the inaccessibility problem mentioned in Sec. 3, we first derive an equivalent $J^*(\pi)$ of the original objective in Eq. 1 and further obtain a surrogate objective $\hat{J}(\pi)$ to approximate this equivalent:

$$\begin{aligned} J(\pi) \iff & \mathbb{E}_{\rho(\mathbf{s}_0),\tau\sim\pi}[\sum_{\mathbf{s}_t\in\tau} \gamma^t \log p_{tar}(\mathbf{s}_t)] - \underbrace{\frac{\mathbb{E}_{\rho(\mathbf{s}_0)}[\log p_{tar}(\mathbf{s}_0)]}{1-\gamma}}_{constant \quad C} \\ = & \mathbb{E}_{\rho(\mathbf{s}_0),\tau\sim\pi}[\sum_{1\le t\le T} \gamma^t \sum_{1\le k\le t} [\log p_{tar}(\mathbf{s}_k) - \log p_{tar}(\mathbf{s}_{k-1})]] \overset{def}{=} J^*(\pi) \end{aligned} \tag{3}$$

Further, we derive a surrogate objective $\hat{J}(\pi)$ to approximate the equivalent objective $J^*(\pi)$. We notice that in most physical simulated tasks, the distance between two adjacent states $||\mathbf{s}_t - \mathbf{s}_{t-1}||_2$ is quite small, which inspires us to approximate the delta log-likelihood using the first order Taylor expansion:

$$J^*(\pi) \approx \mathbb{E}_{\rho(\mathbf{s}_0),\tau\sim\pi}[\sum_{1\leq t\leq T} \gamma^t \underbrace{\sum_{1\leq k\leq t} \langle \mathbf{\Phi}_{tar}(\mathbf{s}_{k-1}), \mathbf{s}_k - \mathbf{s}_{k-1}\rangle}_{r_t}] \stackrel{def}{=} \hat{J}(\pi) \tag{4}$$

In this way, we can optimise the surrogate objective $\hat{J}(\pi)$ by assigning $\sum_{1\leq k\leq t} \langle \mathbf{\Phi}_{tar}(\mathbf{s}_t), \mathbf{s}_k - \mathbf{s}_{k-1}\rangle$ as the immediate reward. In practice, we simply assign the last term of the cumulative summation as the immediate reward $r_t = \langle \mathbf{\Phi}_{tar}(\mathbf{s}_t), \mathbf{s}_t - \mathbf{s}_{t-1}\rangle$, and this version is shown to be the most effective.

To tackle the sparsity problem mentioned in Sec. 3, our idea is to build our policy π_θ upon the gradient-based action $\mathbf{a}_t^g$ mentioned in Sec. 4.3, as $\mathbf{a}_t^g$ is the best direction to increase the likelihood:

$$\underbrace{\mathbf{a}_t = \pi(\mathbf{s}_t, \mathbf{g}_t) + \mathbf{a}_t^g}_{output\,action}, \quad \underbrace{\mathbf{a}_t^g = \mathscr{G}(\mathbf{g}) = \frac{v_{max}}{||\mathbf{g}_t||_2} \cdot \mathbf{g}_t}_{gradient-based\,action}, \quad \underbrace{\mathbf{g}_t = \mathbf{\Phi}_{tar}(\mathbf{s}_t)}_{target\,gradient} \tag{5}$$

This approach is similar to the residual policy learning proposed by [22]. The $\mathbf{a}_t^g$ serves as an initial policy, and the agent's policy network π_θ takes the target gradient $\mathbf{g}_t$ and the current state $\mathbf{s}_t$ as input and outputs $\pi(\mathbf{s}_t, \mathbf{g}_t)$ to 'polish' the $\mathbf{a}_t^g$. The agent finally outputs 'polished' action $\mathbf{a}_t = \pi(\mathbf{s}_t, \mathbf{g}_t) + \mathbf{a}_t^g$. In this way, the agent can not only benefit from the efficient exploration but also have less training burden as it 'stands on the giant's shoulder' (*i.e.*the gradient-based action $\mathbf{a}_t^g$).

We implement the multi-agent Soft-Actor-Critic(SAC) as the reinforcement learning algorithm backbone. Each object is regarded as an agent, and each agent can communicate with all the other agents. At each time step t, the reward for the i-th agent is $r_t = r_t^{tar} + \lambda * r_t^i$, where $r_t^{tar} = \sum_{1\leq k\leq t} \langle \mathbf{\Phi}_{tar}(\mathbf{s}_t), \mathbf{s}_k - \mathbf{s}_{k-1}\rangle$ is a centralised target likelihood reward, $\lambda > 0$ is a hyperparameter and $r_t^i = \sum_{j\neq i} c_{i,j}$ is a decentralised collision-aware reward where $c_{i,j}$ is a collision penalty. Specifically, when a collision is detected between object i and j, $c_{i,j} = -1$, otherwise $c_{i,j} = 0$.

5 Experiment Setups

5.1 Environment

We design two object rearrangement tasks for evaluation: *Ball Rearrangement* and *Room Rearrangement*, where the former uses controlled environments for numerical analysis and the latter uses real data to demonstrate the potential of our method on a more challenging task.

Ball Rearrangement includes three environments with increasing difficulty of rearrangement: *Circling*, *Clustering*, and the hybrid of the first two, *Circling + Clustering*. There are 21 balls with the same geometry bounded in a square in each environment. In *Clustering* and *Circling+Clustering*, the balls are divided equally into three colours. The goal of each environment is as follows: *Circling* requires all balls to form into a circle. The centre of the circle can be anywhere in the environment; *Clustering* requires all balls to form into three clusters by colour. The joint centre of each cluster has two choices; *Circling + Clustering* requires all balls to form into a circle where balls of the same type are adjacent. To slightly increase the complexity of the physical dynamics, we enlarge the lateral friction coefficient of each ball and make each ball with different physical properties, *e.g.*, friction coefficient and mass. The target examples of each task are sampled from an explicit process. Thus, we can define a 'pseudo likelihood' on a given state according to the sampling program to measure the similarity between the state and the target distribution.

Room Rearrangement is built on a more realistic dataset, 3D-FRONT [43]. After cleaning the data, we get a dataset with 839 rooms, 756 for training(*e.g.*target examples), and 83 for testing. In each episode, we sample a room from the test dataset and shuffle the room with a Brownian Process at the beginning to get an initial state. The agent can assign velocities for each object in the room at each time step. The agent has to learn implicit knowledge learned from the target examples to tidy up the room. More details are in the supplementary.

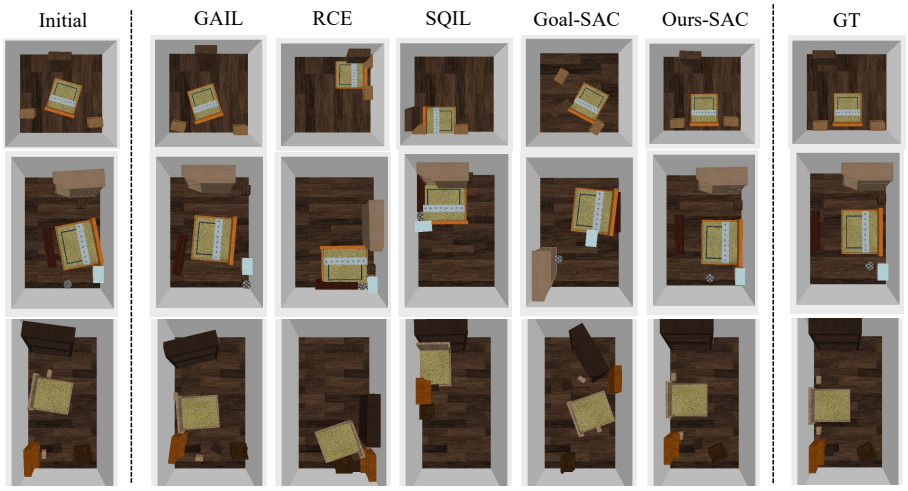

Figure 4: Qualitative results on planar room rearrangement tasks. We visualised the results of each method closest to the ground truth state (the nearest target example to the initial state) in the eight rearrangement results of a given room.

5.2 Evaluation

For both baseline comparison and ablation studies, we collect a fixed number of trajectories $T = \{\tau_i\}$ starting from the same initial states for different methods. Then we calculate the following metrics using the collected trajectories, where the PL is only reported in ball sorting tasks:

Pseudo-Likelihood (PL) measures the efficiency of the rearrangement process and the sample quality of the rearrangement results. We specify a *pseudo-likelihood function* $\mathbf{F}_{proxy}$ for each environment that measures the similarity between a given state and the target examples. For each time step t, the PL is defined as $\mathbf{F}_{proxy}(\mathbf{s}_t)$. We report the mean-variance curve of PL with respect to the time step t, where the mean and variance are calculated over the trajectories T. We do not report PL on room rearrangement as it is hard to program the human preferences.

Coverage Score (CS): measures the diversity and fidelity of the rearrangement results (*i.e.*the terminal states $S_T = \{\mathbf{s}_T\}$ of the trajectories) by calculating the Minimal-Matching-Distance(MMD) [44] between S_T and a fixed set of examples S_{gt} from p_{tar}: $\sum_{\mathbf{s}_{gt} \in S_{gt}} \min_{\mathbf{s}_T \in S_T} ||\mathbf{s}_{gt} - \mathbf{s}_T||$.

Averaged Collision Number (ACN) reflects the efficiency of the rearrangement process since object collision will lead to object blocking and deceleration. ACN is calculated as $\sum_{\tau \in T} \sum_{\mathbf{s}_t \in \tau} \mathbf{c}_t$, where $\mathbf{c}_t$ denotes the total collision number at time step t. More details are in the supplementary.

5.3 Baselines

We compare our framework with planning-based baselines under the model-based setting and with learning-based baselines under the model-free setting. Besides, we design heuristic baselines serving as the upper bounds for ball rearrangement tasks.

Leaning-based Baselines: *GAIL:* A classical inverse RL method that trains a discriminator as the reward function. *RCE:* The SOTA example-based reinforcement learning method. *SQIL:* A competitive example-based imitation learning method. *Goal-RL:* An 'open-loop' method that first generates a goal at the beginning of the episode via a VAE trained from the target examples and then trained to reach the goal using goal-conditioned RL.

Planning-based Baselines: *Goal-ORCA:* This method first generates a goal via the same VAE as in *Goal-RL*. Then the agent assigns an action pointing to the goal as reference control and adjusts the preference velocity using ORCA, similar to Sec. 4.3.

Heuristic-based Baseline: *Oracle:* We slightly perturb the samples from the target distribution and take these samples as the rearrangement results.

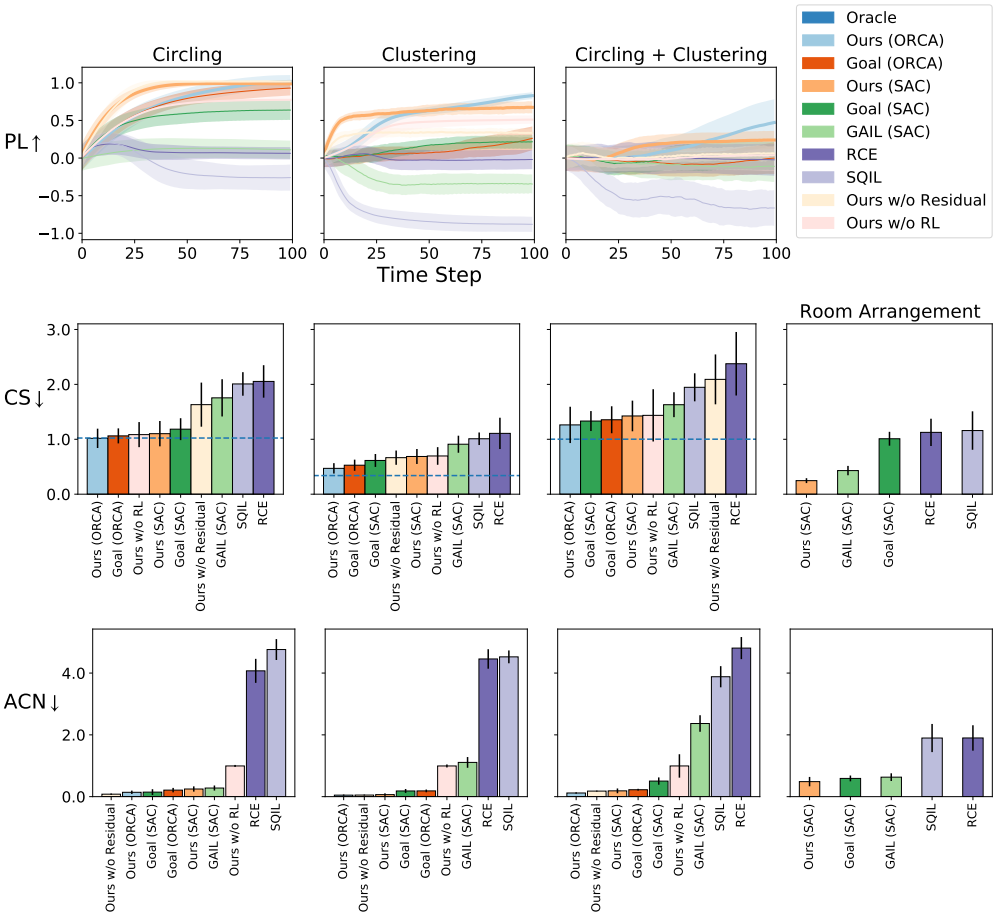

Figure 5: Comparative results of all the methods and ablated versions of our method in ball rearrangement. From top to down, we report pseudo-likelihood (PL), coverage score (CS), and averaged collision number (ACN). The PL curves of each task are normalised by the oracle's performance of this task. For CS and ACN, we report the mean and confidence intervals for five random seeds.

6 Result Analysis

6.1 Baseline Comparison

Ball Rearrangement: We present the quantitative results on ball rearrangement tasks in the first three columns of Fig. 5. As seen from the top row of Fig. 5, the PL curves of our model-free framework(Ours-SAC) and model-based framework (Ours-ORCA) are significantly higher and steeper than the baselines in all three tasks, which shows the effectiveness of our method in arranging objects towards the target distribution efficiently. In the middle row of Fig. 5, Ours-ORCA achieves the best performance across all three tasks, while Goal-SAC outperforms Ours-SAC on *Clustering* and *Circling+Clustering* since the goal is proposed by a generative model that can naturally generate diverse and realistic goals. From the qualitative results shown in Fig. 3, our methods also significantly outperform the baselines in realism. Our methods also achieve the lowest ACN in most cases, except that Goal-SAC is slightly better than Ours-SAC in *Circling*.

The Goal-SAC is the strongest baseline yet loses to our framework in efficiency since the generated goal may be far away from the initial state. As shown in Fig. 3, the Goal-SAC and Goal-ORCA can rearrange objects in the right trend, but the objects may block each other's way. Besides, in the supplementary, we show the VAE used in Goal-SAC is not guaranteed to generate a legal state(*e.g.*the balls overlap in goal). Therefore, the balls may have unexpected behaviour during the process.

The classifier-based reward learning method, GAIL, RCE, and SQIL, basically fails in most tasks, as shown in Fig. 3 and Fig. 5. This is due to the over-exploitation problem that commonly exists in classifier-based reward learning methods. We defer the experiments to supplementary.

Room Rearrangement: The dynamics of the room rearrangement task are too complex to apply explicit planning-based methods, so we only compare our model-free framework with learning-based baselines. As shown in Fig. 4 and the last column of Fig. 5, our method significantly outperforms other baselines in sample realism of rearrangement results and the safety of the rearrangement process, which proves our method can also work on a more complex task.

6.2 Ablation Studies and Analysis

We conduct ablation studies on ball rearrangement tasks to investigate: 1) The effectiveness of the *exploration guidance* provided from the target gradient field and 2) The necessity of combining the target gradient field with *control algorithms* to adapt to the environment dynamics. To this end, we construct an ablated version named *Ours w/o Residual* that drops the residual policy learning and another one named *Ours w/o RL* that only takes gradient-based action $\mathbf{a}^g$ as policy.

Ours w/o Residual: The CS-bar in Fig. 5 shows that Ours significantly outperforms the Ours w/o Residual in CS, which means the agent is easy to over-exploit few modes with high density. We also show qualitative results in supplementary that without the residual policy learning, the objects are usually arranged into a single pattern.

Ours w/o RL: The PL-curve in Fig. 5 shows that without the control algorithm serving as the backbone, the rearrangement efficiency of Ours w/o RL is significantly lower than Ours. Meanwhile, the ACN of Ours w/o RL is significantly larger than Ours, which indicates the efficiency drop may be due to the collisions caused by the gradient-based action.

Generalisation: We further evaluate the generalisation of our methods to different numbers of objects. To be specific, we train all the policies in the ball environment with $3 \times 7 = 21$ balls. During the testing phase, we directly transfer each policy to environments with increased ball number, i.e., 3×8, 3×9, and 3×10, without any fine-tuning. As shown in Fig. 6.2, we report the averaged PL increment from initial to target states of each policy in each configuration. The results indicate that our framework generalises well to different numbers of balls with the least performance drop compared with baselines in all three ball rearrangement tasks.

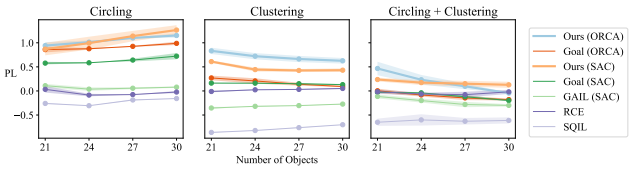

Figure 6: **Generalisation Anaslysis**: Each policy is trained with 3×7 balls and is zero-shot transferred to environments with different ball numbers.

7 Conclusion

In this study, we first analyse the challenge of object rearrangement and then incorporate a gradient field method with either model-based planning or model-free learning approaches to achieve object rearrangement tasks. Experiments demonstrate our effectiveness by comparison with learning-based and planning-based baselines.

Limitation and Future Work. This work only considers the 2-D state-based rearrangement with simplified dynamics that the objects can directly control their own velocities. Future works may extend our framework to more realistic scenarios: **1) Visual Observation:** We may explore how to conduct efficient and accurate score-based reward estimation from image-based examples and leverage the recent progress on visual state representation learning [45, 46, 47] to develop the visual-based gradient-based action. **2) Complex Dynamics:** When the agent can only move one object at a time, we may explore hierarchical methods where high-level policy chooses which object to move and low-level policy leverage the gradient-based action for motion planning. When we can only apply force to objects instead of velocity, we may explore a hierarchical policy where a high-level policy outputs expected velocities and a low-level employ a velocity controller, such as PID controller, to minimises the velocity errors.

Ethics Statement. Our method has the potential to build home-assistant robots. We evaluate our method in simulated environments, which may introduce data bias. However, similar studies also have such general concerns. We do not see any possible major harm of our study.

References

[1] Yun Jiang, Marcus Lim, and Ashutosh Saxena. Learning object arrangements in 3d scenes using human context. *arXiv preprint arXiv:1206.6462*, 2012.

[2] Nichola Abdo, Cyrill Stachniss, Luciano Spinello, and Wolfram Burgard. Robot, organize my shelves! tidying up objects by predicting user preferences. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 1557–1564. IEEE, 2015.

[3] Henry N Ricciuti. Object grouping and selective ordering behavior in infants 12 to 24 months old. *Merrill-Palmer Quarterly of Behavior and Development*, 11(2):129–148, 1965.

[4] Ivan Kapelyukh and Edward Johns. My house, my rules: Learning tidying preferences with graph neural networks. In *5th Annual Conference on Robot Learning*, 2021.

[5] Hanqing Wang, Wei Liang, and Lap-Fai Yu. Scene mover: Automatic move planning for scene arrangement by deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020.

[6] Ju-Bong Kim, Ho-Bin Choi, Gyu-Young Hwang, Kwihoon Kim, Yong-Geun Hong, and Youn-Hee Han. Sortation control using multi-agent deep reinforcement learning in n-grid sortation system. *Sensors*, 20(12):3401, 2020.

[7] 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*, pages 1094–1100. PMLR, 2020.

[8] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.

[9] Wenhan Luo, Peng Sun, Fangwei Zhong, Wei Liu, Tong Zhang, and Yizhou Wang. End-to-end active object tracking and its real-world deployment via reinforcement learning. *IEEE transactions on pattern analysis and machine intelligence*, 42(6):1317–1332, 2019.

[10] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[11] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[12] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29:4565–4573, 2016.

[13] Siddharth Reddy, Anca D Dragan, and Sergey Levine. Sqil: Imitation learning via reinforcement learning with sparse rewards. *arXiv preprint arXiv:1905.11108*, 2019.

[14] Minghuan Liu, Tairan He, Minkai Xu, and Weinan Zhang. Energy-based imitation learning. *arXiv preprint arXiv:2004.09395*, 2020.

[15] Justin Fu, Avi Singh, Dibya Ghosh, Larry Yang, and Sergey Levine. Variational inverse control with events: A general framework for data-driven reward definition. *arXiv preprint arXiv:1805.11686*, 2018.

[16] Benjamin Eysenbach, Sergey Levine, and Ruslan Salakhutdinov. Replacing rewards with examples: Example-based policy search via recursive classification. *arXiv preprint arXiv:2103.12656*, 2021.

[17] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *arXiv preprint arXiv:1907.05600*, 2019.

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

[19] Chence Shi, Shitong Luo, Minkai Xu, and Jian Tang. Learning gradient fields for molecular conformation generation. *arXiv preprint arXiv:2105.03902*, 2021.

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

[21] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[22] Tom Silver, Kelsey Allen, Josh Tenenbaum, and Leslie Kaelbling. Residual policy learning. *arXiv preprint arXiv:1812.06298*, 2018.

[23] Jur van den Berg, Stephen J Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In *Robotics research*, pages 3–19. Springer, 2011.

[24] Martin J Schuster, Dominik Jain, Moritz Tenorth, and Michael Beetz. Learning organizational principles in human environments. In *2012 IEEE International Conference on Robotics and Automation*, pages 3867–3874. IEEE, 2012.

[25] Nichola Abdo, Cyrill Stachniss, Luciano Spinello, and Wolfram Burgard. Organizing objects by predicting user preferences through collaborative filtering. *The International Journal of Robotics Research*, 35(13):1587–1608, 2016.

[26] 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.

[27] Lap Fai Yu, Sai Kit Yeung, Chi Keung Tang, Demetri Terzopoulos, Tony F Chan, and Stanley J Osher. Make it home: automatic optimization of furniture arrangement. *ACM Transactions on Graphics (TOG)-Proceedings of ACM SIGGRAPH 2011, v. 30,(4), July 2011, article no. 86*, 30(4), 2011.

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

[29] Ankur Handa, Viorica Pătrăucean, Simon Stent, and Roberto Cipolla. Scenenet: An annotated model generator for indoor scene understanding. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5737–5743. IEEE, 2016.

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

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

[32] Luca Weihs, Matt Deitke, Aniruddha Kembhavi, and Roozbeh Mottaghi. Visual room rearrangement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5922–5931, 2021.

[33] Yash Kant, Arun Ramachandran, Sriram Yenamandra, Igor Gilitschenski, Dhruv Batra, Andrew Szot, and Harsh Agrawal. Housekeep: Tidying virtual households using commonsense reasoning. *arXiv preprint arXiv:2205.10712*, 2022.

[34] Gabriel Sarch, Zhaoyuan Fang, Adam W Harley, Paul Schydlo, Michael J Tarr, Saurabh Gupta, and Katerina Fragkiadaki. Tidee: Tidying up novel rooms using visuo-semantic commonsense priors. *arXiv preprint arXiv:2207.10761*, 2022.

[35] Aviv Netanyahu, Tianmin Shu, Joshua Tenenbaum, and Pulkit Agrawal. Discovering generalizable spatial goal representations via graph-based active reward learning. In *International Conference on Machine Learning*, pages 16480–16495. PMLR, 2022.

[36] Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

[37] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 661–668. JMLR Workshop and Conference Proceedings, 2010.

[38] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[39] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[40] Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. *arXiv preprint arXiv:1809.02925*, 2018.

[41] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

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

[43] 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.

[44] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018.

[45] Drew Linsley, Girik Malik, Junkyung Kim, Lakshmi Narasimhan Govindarajan, Ennio Mingolla, and Thomas Serre. Tracking without re-recognition in humans and machines. *Advances in Neural Information Processing Systems*, 34:19473–19486, 2021.

[46] Thomas Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. *arXiv preprint arXiv:1911.12247*, 2019.

[47] Zhixuan Lin, Yi-Fu Wu, Skand Peri, Bofeng Fu, Jindong Jiang, and Sungjin Ahn. Improving generative imagination in object-centric world models. In *International Conference on Machine Learning*, pages 6140–6149. PMLR, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] see Sec. 1
   (b) Did you describe the limitations of your work? [Yes] see Sec. 7
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] see supplementary
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] see Sec. 7 and supplementary
2. If you are including theoretical results...

(a) Did you state the full set of assumptions of all theoretical results? [N/A]
(b) Did you include complete proofs of all theoretical results? [N/A]

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] please see the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] more details in the supplemental material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] more details in the supplemental material.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] more details in the supplemental material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]