IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Goal-Conditioned Reinforcement Learning with Disentanglement-based Reachability Planning

Zhifeng Qian, Mingyu You*, Hongjun Zhou, Xuanhui Xu and Bin He

***Abstract*—Goal-Conditioned Reinforcement Learning (GCRL) can enable agents to spontaneously set diverse goals to learn a set of skills. Despite the excellent works proposed in various fields, reaching distant goals in temporally extended tasks remains a challenge for GCRL. Current works tackled this problem by leveraging planning algorithms to plan intermediate subgoals to augment GCRL. Their methods need two crucial requirements: (i) a state representation space to search valid subgoals, and (ii) a distance function to measure the reachability of subgoals. However, they struggle to scale to high-dimensional state space due to their non-compact representations. Moreover, they cannot collect high-quality training data through standard GC policies, which results in an inaccurate distance function. Both affect the efficiency and performance of planning and policy learning. In the paper, we propose a goal-conditioned RL algorithm combined with Disentanglement-based Reachability Planning (REPlan) to solve temporally extended tasks. In REPlan, a Disentangled Representation Module (DRM) is proposed to learn compact representations which disentangle robot poses and object positions from high-dimensional observations in a self-supervised manner. A simple REachability discrimination Module (REM) is also designed to determine the temporal distance of subgoals. Moreover, REM computes intrinsic bonuses to encourage the collection of novel states for training. We evaluate our REPlan in three vision-based simulation tasks and one real-world task. The experiments demonstrate that our REPlan significantly outperforms the prior state-of-the-art methods in solving temporally extended tasks.**

***Index Terms*—Goal-Conditioned Reinforcement Learning, Disentangled Representation, Reachability measure.**

I. Introduction

INTELLIGENT agents are expected to master general-purpose skills in open environments. Therefore, Goal-Conditioned Reinforcement Learning (GCRL) is proposed to enable agents to spontaneously set diverse goals to learn a wide range of skills. The Goal-Conditioned (GC) policies can generalize skills to different situations and tasks. Many excellent works on GCRL have been proposed in various fields, such as robot navigation [1], [2] and manipulation [3], [4].

However, GCRL often struggles to reach long-term goals in temporally extended tasks [5], since the space to explore becomes large over the horizon, which makes agents difficult to explore high-quality states. In complex environments with high-dimensional observations, e.g. images, such problems are aggravated. To address this challenge, recent works [6], [7] leverage planning algorithms to enhance GCRL in temporally extended tasks. These methods plan a series of subgoals from the state space, and optimize them based on the distance measure of subgoals, which can serve as a curriculum to improve the learning efficiency of GCRL.

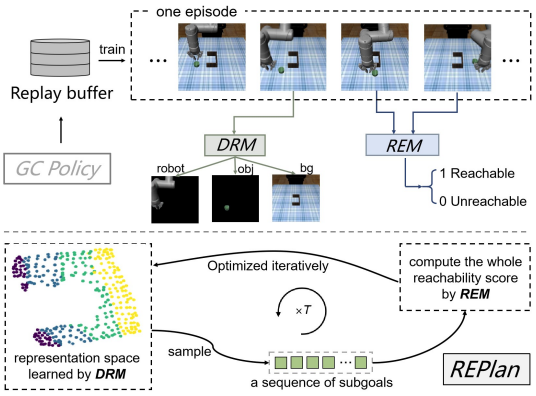

Fig. 1. Illustration of our proposed REPlan. In REPlan, DRM aims to learn compact representations to disentangle the robot poses and object positions, while REM is designed to measure the reachability of subgoals. To plan appropriate subgoals efficiently, valid subgoals are sampled from the representation space learned by DRM. Then REPlan computes the whole reachability score by REM. Finally, REPlan optimizes the objective iteratively and outputs an appropriate sequence of subgoals.

One crucial requirement of such methods [8] is to learn a semantically valid and compact representation space, where suitable subgoals can be efficiently searched. The directly planned subgoals in the high-dimensional space may be random noises or invalid images, e.g. unfeasible robot poses. To ensure valid subgoals, some works [2], [6] learn a representation space by a variational autoencoder (VAE) [9]. However, such representations are non-compact, and inconsistent with the physical properties of entities, as is shown in Section IV-D. Such semantic inconsistency may reduce the efficiency of planning and GC policy learning.

Another key requirement is to build an accurate distance measure of subgoals, which is used to determine whether a subgoal is reachable from another subgoal. The planner can further compute and optimize the distance of the entire subgoal sequence based on the distance measure. Recently, the value function of the GC policy is used as the distance measure in [2], [8]. However, these methods only construct plans during testing, which means they cannot use subgoals to guide the GC policy to explore high-quality samples during training. While the accuracy of the value function depends on the quality of the

Manuscript received February, 23, 2023; Revised April, 15, 2023; Accepted June, 13, 2023. This paper was recommended for publication by Editor Jens Kober upon evaluation of the Associate Editor and Reviewers' comments.

This work was supported in part by the National Natural Science Foundation of China under Grant No. 62073244 and 61825303, NSFC 62088101 Autonomous Intelligent Unmanned Systems, the Science and Technology Commission of Shanghai Municipality (No. 2021SHZDZX0100).

Z. Qian, M. You, H. Zhou, X. Xu and B. He are with the College of Electronic and Information Engineering, Frontiers Science Center for Intelligent Autonomous Systems, Tongji University, Shanghai 201800, China. (*Mingyu You is the corresponding author with e-mail: myyou@tongji.edu.cn)

Digital Object Identifier (DOI): see top of this page.

training samples [7], such methods may result in the estimation deviation of the value function, which is not beneficial to planning and GC policy learning.

To this end, we propose a goal-conditioned RL algorithm combined with Disentanglement-based **RE**achability **Plan**ning (**REPlan**) to solve temporally extended tasks. The illustration of REPlan is shown in Fig. 1. REPlan is designed to learn a compact representation space where subgoals can be sampled efficiently, and to learn an accurate distance measure that can be used to measure the total quality of subgoals. A self-supervised **D**isentangled **R**epresentation and probabilistic generation **M**odule (**DRM**) is proposed to learn a compact representation space, which can disentangle the robot, objects and the background without corresponding mask labels. Through the proposed inter-branch and intra-branch entropy loss, the manifold structure of the learned representations can be consistent with the robot pose and object position, which greatly improves the planning performance. In addition, a **RE**eachability discrimination **M**odule (**REM**) is designed to measure the reachability between subgoals. After sampling a sequence of subgoals from the learned disentangled representation space, REPlan iteratively minimizes the whole reachability score computed by REM. REM converges quickly and can generalize to the unseen distribution. Moreover, REM can compute intrinsic bonuses to encourage exploration and improve the quality of training data, which in turn makes REM more accurate. We show the data efficiency and advanced performance of REPlan in three simulation tasks and one real-world task, which need temporally extended reasoning.

Our main contributions are as follows:

- We propose REPlan to plan subgoals efficiently to augment GCRL to reach distant goals in temporally extended tasks.
- We design DRM to learn representations which can disentangle robot poses and object positions in a self-supervised manner. Such compact representations can improve the efficiency of planning and GC policy learning.
- We propose REM to measure the reachability of subgoals, which is easy to train and can generalize to the unseen distribution. In addition, REM provides intrinsic bonuses to encourage exploration.
- We empirically demonstrate that our REPlan outperforms other state-of-the-art methods in three vision-based simulation tasks and one real-world task.

II. Related works

A. Goal-conditioned RL for Temporally Extended Tasks

Many works on GCRL [10], [11] aim to learn GC policies which enable agents to achieve various goals. Hindsight Experience Replay (HER) [12], [13] is often employed to improve the sample efficiency and robustness by relabeling the transition tuples in the replay buffer. While such methods can learn greedy policies to reach nearby goals, they often struggle to solve temporally extended tasks.

To solve this problem, some prior methods [8], [14] combine subgoal planning with GCRL. In particular, the value function of GCRL is often used to measure the dynamic distances between states for planning. SoRB [8] searches a sequence of subgoals in the replay buffer to guide GCRL to lead to the final goals. LEAP [2] employs Temporal Difference Models (TDMs) [15], a goal-conditioned value function, to plan subgoals in the latent space. However, the accurate estimation of the value function depends on the quality of the interactive data. These methods only plan subgoals during testing, and can't improve the underlying GC policy to sample higher-quality training data. To this end, RIS [6] performs subgoal search by the high-level policy during training, and uses the Kullback-Leibler (KL) regularization to improve the exploration efficiency. However, the value function is easy to overestimate and unstable during training, which may affect the algorithm performance. C-planning [7] employs variational inference to perform a subgoal curriculum to improve the quality of the training data. However, the method excels at dealing with state-based tasks, without additional representation methods designed for vision-based tasks.

B. Representation Learning for Vision-based Control

Many previous works [3], [16] have proved that learning an expressive representation can improve the sample efficiency of planning algorithms and RL. Srinivas et al. [16] combine contrastive learning with model-free RL. Many works [17], [18] on vision-based control use VAE [9] to learn the latent codes of images. However, the learned representations contain redundant information, such as background, and cannot be consistent with the physical attributes of the entity. Later, inspired by compositional scene representation learning in computer vision, recent works learn object-oriented representations in an unsupervised [19], [20] or weakly supervised [3] manner for downstream control. However, they only decompose the objects from the background and fail to decompose the complex robots in the manipulation scenes. In contrast, our DRM can not only disentangle the position, texture and mask of the objects but also disentangle the texture and mask of the robot in a self-supervised manner. Our experiments show that our disentangled representations can improve the efficiency and performance of planning.

C. Curiosity-Motivated Exploration

Curiosity-motivated exploration for RL has been studied in the literature. Task-irrelevant intrinsic rewards are computed based on curiosity to encourage agents to visit a wide range of states. Prediction-based methods [21], [22] use the prediction error as the intrinsic motivation. Count-based methods [23], [24] count visited states and reward infrequently visited states. However, the above methods usually don't excel at high-dimensional observation spaces with long episodic time. We take inspiration from the memory-based method [25], which computes a curiosity bonus by training a network to compare the current observation with those in memory. The difference is that our REM is not only used to encourage exploration but also can measure the subgoal quality for planning.

III. Methods

We propose a goal-conditioned RL algorithm augmented by Disentanglement-based Reachability Planning (REPlan)

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

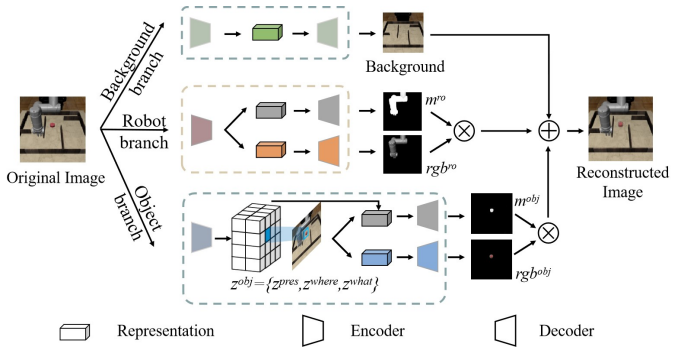

Fig. 2. Schematic of DRM. Taking the original image as input, DRM extracts the independent representations of the background, robot and objects through three branches. The representation of the robot or objects can be decoded into masks and textures. Finally, DRM uses a pixel-wise mixture model to reconstruct the complete image.

to solve temporally extended tasks with high-dimensional observations. In REPlan, a self-supervised Disentangled Representation and probabilistic generation Module (DRM) is proposed to learn a compact representation space which can disentangle the physical attributes of robots and objects for efficient planning. After sampling a sequence of subgoals from the representation space, REPlan computes and optimizes iteratively the reachability score of these subgoals, which is measured by a REachability discrimination Module(REM). To encourage GC policies to collect novel training data, REM can also provide an intrinsic bonus.

In the following, we start by introducing the self-supervised learning of DRM in Section III-A. Then Section III-B presents the architecture and role of REM. The algorithm summary and details of REPlan are shown in Section III-C.

A. Self-supervised Disentangled Representation for Visual Planning

The goal of DRM is to learn a compact representation space, which can ensure that the sampled subgoals are valid. To efficiently plan, the representation should not contain task-irrelevant information. Inspired by previous works [26], [27] in computer vision, our previous work [3] decomposes object representations from the background for robot control. However, in solving temporally extended tasks, object representations cannot model the process of robot movement and object immobility. To this end, DRM can not only disentangle the position, texture and mask of the objects but also disentangle the texture and mask of the robot. Furthermore, our DRM can learn disentangled representations in a self-supervised manner, i.e. without any explicit mask labels.

The schematic of DRM is shown in Fig. 2. DRM decomposes the manipulation scene into three independent representations: robot z^{ro}, object z^{obj} and background z^{bg}. On top of them, DRM uses a pixel-wise mixture model to reconstruct the complete image:

$$\begin{aligned} p\left(s \mid z^{ro}, z^{obj}, z^{bg}\right) &= \underbrace{m^{ro}\left(z^{ro,m}\right) p\left(s \mid z^{ro,rgb}\right)}_{\text{Robot}} \\ &+ \underbrace{m^{obj}\left(z^{obj,m}\right) p\left(s \mid z^{obj,rgb}\right)}_{\text{Foreground}} + \underbrace{m^{bg} p\left(s \mid z^{bg}\right)}_{\text{Background}} \end{aligned} \tag{1}$$

where the mixing probability $m\left(z\right)$ is the mask computed based on z, and $m^{bg} = 1 - m^{ro}\left(z^{ro,m}\right) - m^{obj}\left(z^{obj,m}\right)$. DRM models the RGB distribution $p\left(s \mid z^{ro,rgb}\right)$, $p\left(s \mid z^{obj,rgb}\right)$ and $p\left(s \mid z^{bg,rgb}\right)$ as a Gaussian $\mathcal{N}(\mu^{ro}, \sigma^{ro})$, $\mathcal{N}(\mu^{obj}, \sigma^{obj})$and $\mathcal{N}(\mu^{bg}, \sigma^{bg})$. σ is fixed while μ can be learned.

To model the objects in the scene, DRM divides the observation $s \in \mathbb{R}^{3\times128\times128}$ into $H \times W$ cells. Each cell models a structured representation $z^{obj,rgb} = \{z^{pres}, z^{what}, z^{where}\}$, which is similar to [26]. z^{pres} is a binary value to indicate whether the cell has any object. z^{what} is used to reconstruct the object glimpse and its mask. Then z^{where} is used as the affine transformation parameter of Spatial Transformer [28] to transform the glimpse to the original resolution. We model z^{pres} as a Bernoulli distribution while other representations as a Gaussian distribution. So DRM implements z_{obj} as

$$p\left(z^{obj,rgb}\right) = \coprod_{i=1}^{H*W} p\left(z_i^{pres}\right)\left(p\left(z_i^{where}\right) p\left(z_i^{what}\right)\right)^{z_i^{pres}} \tag{2}$$

We train DRM by optimizing an evidence lower bound(ELBO) [9], which is given by

$$\begin{aligned} \mathcal{L}_{elbo}(x) =& \mathbb{E}_{q(z^{ro},z^{obj},z^{bg}|x)}[logp(x|z^{ro},z^{obj},z^{bg}) \\ &- \mathbb{KL}(q(z^{bg}|x)||p(z^{bg}|x)) \\ &- \sum_{i=1}^{H*W} \mathbb{KL}(q(z_i^{obj}|x)||p(z_i^{obj}|x)) \\ &- \mathbb{KL}(q(z^{bg}|x)||p(z^{bg}|x))] \end{aligned} \tag{3}$$

We reparameterize the continuous variables and model the discrete variables using the Gumbel-Softmax distribution.

Different from previous works in computer vision, which only disentangle the object and the background, DRM also disentangles the robot. Since it is difficult to model robots with huge changes by the cell of the object branch, DRM adds an additional robot branch. To prevent confusion between the robot branch and the object branch learning, we introduce an inter-branch entropy loss, which is denoted as

$$\mathcal{L}_{H_{inter}} = || \sum_{i=1}^{H*W} m_i^{obj} m^{ro} ||_1 \tag{4}$$

To penalize multiple cells for detecting the same object, we introduce an intra-branch entropy loss, which is denoted as

$$\mathcal{L}_{H_{intra}} = \sum_{i=1}^{H*W} \sum_{k}^{K} -m_{i,k}^{obj} log m_{i,k}^{obj} \tag{5}$$

where K denotes the detected objects. To summarize, the total loss function is the sum of each loss weighted by the relevant hyperparameters, which is denoted as

$$\mathcal{L}_{DRM} = \mathcal{L}_{elbo}(x) + \alpha_{inter}\mathcal{L}_{H_{inter}} + \alpha_{intra}\mathcal{L}_{H_{intra}} \tag{6}$$

B. Planning with Reachability Discrimination Module

Planning by reachability. Appropriate subgoals should be reachable between neighbors. If the GC policy can reach the next subgoal given the current state, the agent can reach the long-term goal by reaching each short-term goal. So how can we measure whether those adjacent subgoals are reachable?

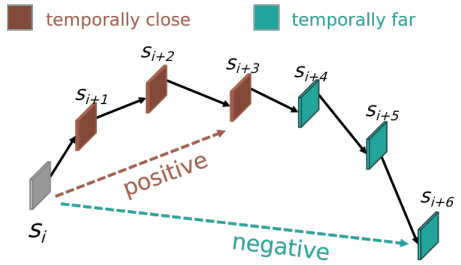

Fig. 3. REM is trained based on the interaction trajectories of the GC policy. We regard the observations temporally close to each other in the trajectory as positive pairs, while temporally far observations as negative pairs.

Accordingly, REM is proposed to measure the reachability of the adjacent subgoals.

REM regards predicting whether two states are reachable as a binary classification task, and there are two categories of reachability and unreachability. Given two observations s_i and s_j, REM predicts the reachable probabilities of the current policy from s_i to s_j.

More generally, given a high-level trajectory composed of N sequential intermediate subgoals $z_{sg_{1:N}} = z_{sg_1}, z_{sg_2}, ..., z_{sg_N}$, we define the episodic reachability as

$$\overrightarrow{R}(z_{s_0}, z_{sg_{1:N}}, z_g) = \sum_{n=0}^{N} R(z_{sg_n}, z_{sg_{n+1}}) \tag{7}$$

where z_{sg_0} denotes the initial observation z_{s_0} and $z_{sg_{N+1}}$ denotes the true goal for simplicity of notation. R denotes the REM. The episodic reachability $\overrightarrow{R}$ provides a measure of a plan's overall reachability. So REPlan is optimized by minimizing the negative norm of the episodic reachability, which is denoted as

$$L(z_{sg_{0:N+1}}) = -||\overrightarrow{R}(z_{sg_{0:N+1}})||_2 \tag{8}$$

Architecture and Training. REM is regarded as a classifier consisting of three fully connected layers, which are simple and easy to train. Given two observations s_i and s_j, we first use the siamese DRM to extract the disentangled representations z_i and z_j. Note that we only use the corresponding robot mask representation $z^{ro,m}$ and the object position representation $z^{obj,where}$, which contain task-relevant information. Then REM predicts the reachability probability between two observations.

The training method is shown in Fig. 3. Two states are sampled in a trajectory from the replay buffer and a fixed step distance is used to distinguish whether they are positive or negative samples. In our setting, the step distance is set to 80. In addition, we select a binary cross entropy loss function, which is commonly used in binary classification tasks, to optimize REM:

$$L_R = -y \log \hat{y} + (1-y) \log(1-\hat{y}) \tag{9}$$

where y is the ground truth and $\hat{y}$ is the prediction.

Note that REM is trained on interaction samples of the GC policy. As the capability of the policy is gradually improved, the training distribution of REM continues to change. It requires REM to be simple and easy to train. Therefore, the siamese DRM extracts compact and expressive representations and fixes the gradient during training REM.

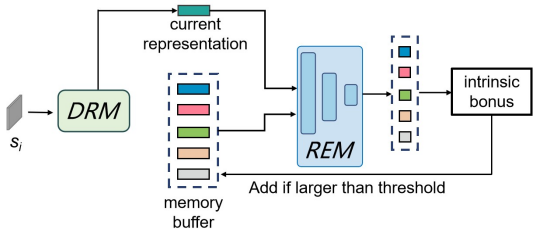

Fig. 4. The curiosity module is used to compute the intrinsic bonus to encourage exploration. CM gives positive rewards to novel states by comparing the reachability between the current representation and the representations in the memory buffer.

Curiosity Module (CM). Another design of REM is to provide an intrinsic bonus to improve the quality of the collected data. We draw inspiration from curiosity-based exploration for RL [25]. We use a REM-based curiosity module (CM) to reward novel states, which is shown in Fig. 10. CM provides positive rewards when the reachability scores of the current state and states in the memory buffer are lower than the novelty threshold.

Given a current observation S_i, DRM extracts the corresponding representation z_i. REM compares z_i with those in the memory buffer to obtain reachability scores. We denote the intrinsic bonus b^i as

$$b^i = \beta - \alpha MAX[R(z_i, z_m)], m = 1:M \tag{10}$$

Empirically, we choose $\beta = 0.2$ and $\alpha = 0.8$. The memory buffer has a limited capacity M. After computing the intrinsic bonus, the representation is added to the memory buffer if the bonus is larger than the novelty threshold $b_{novelty} = 0$. When the memory capacity overflows, the new representation replaces a random old one with a random probability.

We use the intrinsic bonus b^i_t to enhance the extrinsic reward t^e_t and improve the efficiency of GCRL. External rewards t^e_t are computed by the representation distance between the current state and the goal. The rewards are computed as

$$r_t = t^e_t + b^i_t \tag{11}$$

$$t^e_t = ||z_t - z_g||_1 \tag{12}$$

C. Summary of Reachability Enhanced Planning with GCRL

We summarize the whole algorithm of REPlan in Algorithm. 1. We first collect data through random exploration and train DRM by Equ. 6. Then given the initial state s_0 and one goal g, we use Cross-Entropy Method [29], a gradient-free optimization algorithm, to optimize Equ. 8 to choose N subgoals with maximum reachability. Once the subgoals are obtained, we only take the first subgoal as the input of the GC policy. After step t_n, we repeat the above procedure: we replan the $N-1$ subgoals and give the first subgoal to the GC policy to interact with the environment t_n times.

In principle, GCRL can model the variable goal g into any standard reinforcement learning algorithm. We choose the twin delayed deep deterministic policy gradient algorithm (TD3) [30] which is an effective off-policy RL algorithm. We also use Hindsight Experience Replay (HER) [12] to relabel the tuples with the future and generated goals to accelerate policy

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Algorithm 1 REPlan with GCRL

Train DRM with randomly explored samples by Equ. 6.
Initialize REM R_θ, GC policy π_ϕ.
Initialize the max episodes E, the number of subgoals N.
Sample the initial state $s_0 \sim \rho_0$, and the goal $g \sim \rho_g$.
for $i = 1 : E$ episodes **do**
  for $n = 1 : N$ **do**
    Optimize Equ. 8 to choose subgoal representations $z_{sg_n}, ..., z_{sg_N}$ using REM if $n < N$.
    for $t = 1 : t_N$ **do**
      Execute a_t using GC policy $\pi(\cdot|z_t, z_{sg_n})$.
      Augmenting tuple $[z_t, a_t, z_{t+1}, r_t,]$ by Equ. 11.
      Store the tuple in the replay buffer.
    end for
  end for
  Sample trajectories to train REM by Equ. 9.
  Sample tuples to train the GC policy π_ϕ.
end for

learning. In this work, we set the number of subgoals N as a hyperparameter of the task. We compute the maximum horizon between adjacent subgoals $t_n = T_{max}/(N+1)$.

IV. Experiments

To verify the effectiveness of our REPlan, we perform experiments in three vision-based simulation tasks and one real-world task. All these tasks require agents to achieve the long-term goal with temporally extended reasoning. We first introduce our experimental setup in Section IV-A. In the following sections, our experiments study the following questions: (1) How does our REPlan compare with prior state-of-the-art works in goal-conditioned reinforcement learning? (2) Does DRM in REPlan disentangle the components of the observation in a self-supervised manner? (3) Can the learned disentangled representations maintain consistency with the physical attributes of entities such as object positions and robot poses? (4) Can REM in REPlan reflect the reachability of two states? (5) Is each design of our REPlan really effective for solving vision-based temporally extended tasks? (6) Can our REPlan transfer efficiently from simulation to the real world?

A. Experiment Setup

Vision-based Robotic Manipulation Tasks. We perform experiments on three complex robotic manipulation tasks in simulation and one real-world task. For the sake of clarity, we denote them as *UR-Pusher-1*, *UR-Pusher-2*, *UR-Pick-Place* and *Real-Pusher*. The tasks are based on Robosuite [31] powered by the MuJoCo physics engine [32]. The simulation tasks include two tasks of pushing the puck around obstacles and one task of continuously picking up and placing two cubes, while *Real-Pusher* task is similar to *UR-Pusher-1*. Therefore a greedy goal-reaching policy can hardly reach the goal directly from the initial state. It requires the agent to perform temporally extended reasoning.

Baselines. We compare our approach to prior state-of-the-art GCRL methods. **RIG** [17]: combine TD3 [30] with HER [12] for GCRL. RIG also trains a VAE [9], which is used to encode observations, generate unseen goals to relabel samples, and compute the negative Euclidean distance in the representation space as rewards. We regard RIG as a baseline since RIG is a greedy goal-reaching policy without planning subgoals. **DR-GRL** [3]: further learns compact representations capable of disentangling different attributes of objects, which are then utilized for state abstraction and reward function calculation of the GC policies. **LEAP** [2]: uses VAE [9] and Cross-Entropy Method [29] to plan a series of subgoals, which the low-level GC policy need to reach one by one. In addition, LEAP takes the temporal difference models (TDMs) [15], a time-varying goal-conditioned value function, as the reward. **RIS** [6]: predicts subgoals through the high-level policy, and use the Kullback-Leibler(KL) regularization to help the underlying GC policy learning. For a fair comparison, we select TD3 as the RL algorithm to train REPlan, RIG, LEAP and RIS. All the methods are trained on 4 NVIDIA TITAN X GPUs.

B. Comparison with Prior GCRL Methods

To answer the first question, we compare our REPlan to several state-of-the-art methods in solving vision-based temporally extended tasks. To clearly and fairly measure the task completion, we use the European distance between the object and the goal on the table at the end of an episode. We train each method for 1500 epochs on task 1 and 2000 epochs on task 2. Each method is trained with 5 seeds. In each epoch, the agent must interact with the environment for 5 episodes, and the max steps in an episode are 500.

The experiment results are shown in Fig. 5. Dark lines represent the mean value and light areas represent the confidence interval. As is shown, RIG struggles to reach the goal, since RIG is a greedy goal-reaching policy, and uses the Euclidean distance in the VAE feature space as the reward function, which is not consistent with the physical attributes of the robot and the object. RIG cannot perform long-term reasoning, and always pushes the object into obstacle traps. DR-GRL is slightly improved compared to RIG, due to the learned object-centric representations, but still hardly reaches distant goals. LEAP and RIS make progress on three tasks since both of them plan over subgoals to reach distant goals. However, we find that their inefficient exploration in such complex environments leads to inaccurate distance measures, which affects the performance of the algorithm. Compared to the RIG, DR-GRL, LEAP and RIS, our REPlan converges faster and can reach long-term goals within a short distance.

We visualize the subgoals generated by our REPlan in Fig. 6. We use DRM to decode the generated goal representations (including robot and object) with the background. Take task 1 as an example. Given the initial state S_0 and the final goal $Goal$, the agent plans a meaningful and reachable subgoal sequence: at $Subgoal_1$, the agent reaches near the object. At $Subgoal_2$, it pushes the object downward. Note that since it is going to push the object to the right next, the agent will move to the left of the object to make it easier to reach the next subgoal. At $Subgoal_3$, it pushes the object to the right below the goal. At $Subgoal_4$, it goes under the object to prepare for the final push to the goal.

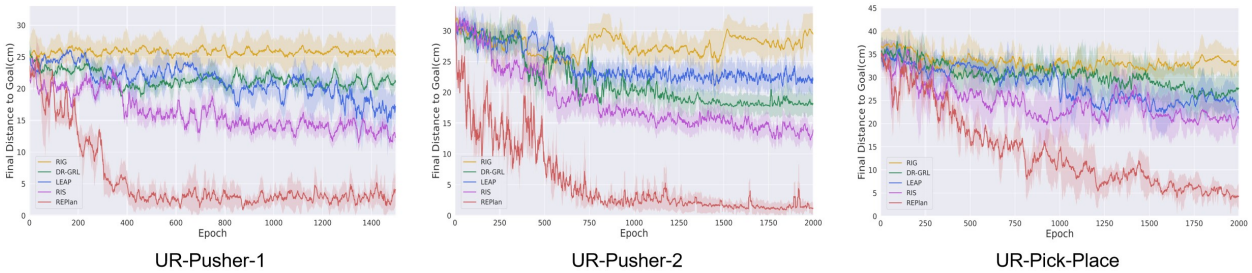

Fig. 5. Comparison with several state-of-the-art methods on three vision-based temporally extended tasks *UR-Pusher-1*, *UR-Pusher-2* and *UR-Pick-Place*. The goals of the first two tasks are to push one puck to the goal position with different complex obstacles, while the goal of the last task is to pick up the two cubes and place them on the corresponding cubes. As is shown, our REPlan outperforms other methods in both sample efficiency and performance on three tasks.

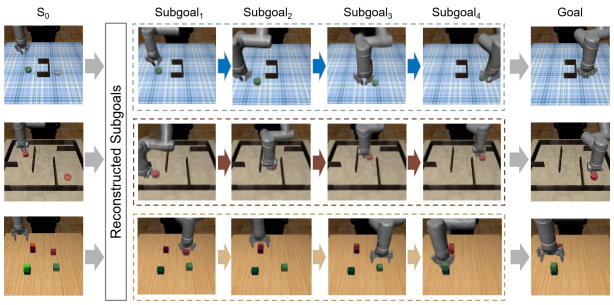

Fig. 6. Visualization of the reconstructed subgoals generated by our method. We use DRM to decode the generated goal representations (including robot and object) with the background into the images.

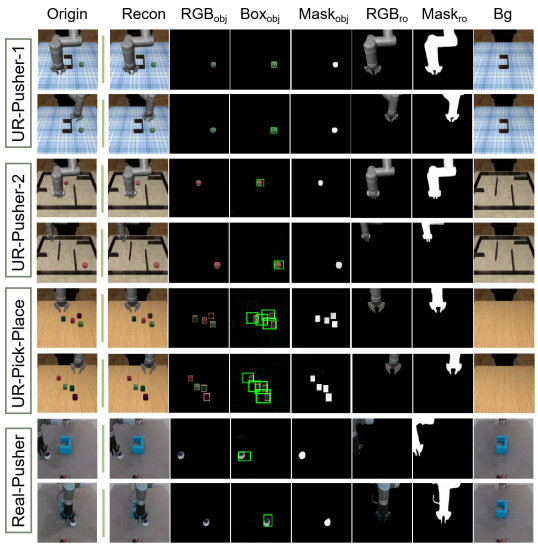

Fig. 7. Visualization of the reconstructed observations by DRM. Two examples of each task are shown here. As is shown, DRM can disentangle the robot, the object and the background from images in a self-supervised manner.

C. Disentanglement and Reconstruction of DRM

To answer the second question, we train DRM on the dataset collected by the random policy in three simulation tasks and one real-world task. The dataset contains different object positions and robot arm poses, as well as different obstacles and different table background textures. In particular, the dataset does not contain any annotation of object coordinates or robot poses. DRM is optimized only by the reconstruction loss and other auxiliary loss functions 6 in a self-supervised manner.

The qualitative results are given in Fig. 7. The first column is the original observations, and the following columns are the overall reconstructed images and the glimpse and the mask of each component. As is shown, DRM can clearly disentangle the object and robot from the complex background in both simulation and the real world. Through the spatial transformer module, DRM can accurately predict the object bounding box and generate clear textures and masks in the bounding box.

D. Consistency of Disentangled Representation and Entity Physical Attribute

To answer the third question of whether the learned representations are consistent with the object positions and the robot poses, we visualize the representations learned by different methods in Fig. 8. RIG and LEAP use VAE to encode the observation, while RIS optimizes the image encoder and the control policy in an end-to-end manner through reinforcement learning. In addition, DR-GRL learns object-centric representations similar to those of our REPlan, but DR-GRL cannot learn robot representations. T-SNE [33] is used to reduce the representation dimension to 2.

We use observations in *UR-Pusher-1* as examples. In the left sub-figure, we use points to represent 20*20 different object positions, while we sample 20*20 different robot poses in the right sub-figure. For clarity, we do not sample the object positions in the middle of obstacles. We use colors to represent changes in object positions and robot poses. It is worth noting that our method can disentangle the object and the robot, while other methods can only encode the overall image. So to reduce the influence of the entity's physical attributes on the representation of other methods, we fix the robot arm at the upper left corner in the left experiment and remove objects in the right experiment. As is shown, the representations of VAE and RIS are relatively messier and overlap. In addition, these representation distances do not change monotonously with the entity attributes. For example, when the distance between the object and the goal gets smaller, the distance between features may become larger.

In contrast, the disentangled representations learned by DRM are consistent with object positions both horizontally and vertically. As is shown, our DRM can well learn the manifold structure of object positions and robot poses in the representation space. As further proved by our experiments in section IV-F, such efficient representations can improve the performance of REPlan. In addition, the representations can effectively help reduce the training difficulty of REM.

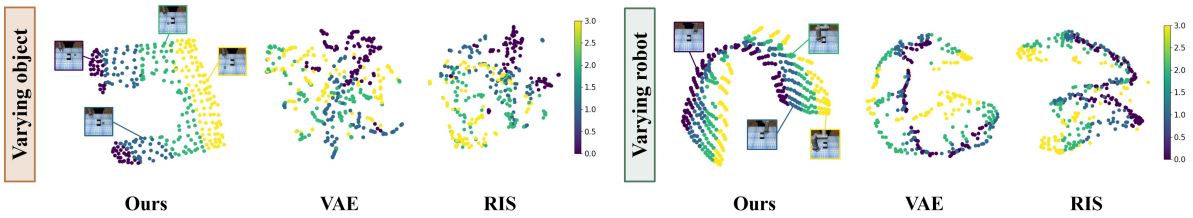

Fig. 8. Visualizations of the observation representations learned by different methods after the dimensionality reduction. T-SNE is used to reduce the dimension of these representations to two. We expect to show whether the learned representations are consistent with the object positions and the robot poses in the observations. Therefore, we uniformly sample about 10 * 10 positions of the cups **(Left)** and about 10 * 10 poses of the robot **(Right)** to show the different types of observation changes. We use different colors to indicate the changes, where label 0 indicates the upper right of the table and label 3 indicates the lower left.

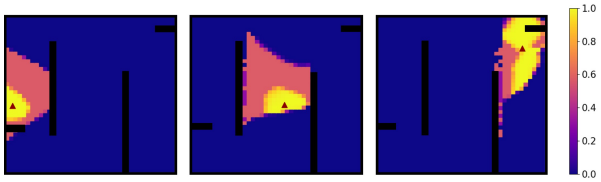

Fig. 9. Heatmap of REM. The heatmap is based on the top view of *UR-Pusher-2*. The red triangle represents the current state and different colors represent the reachability score between the two states. Warmer colors represent higher reachability scores. As is shown, REM can infer that the state behind the obstacle is difficult to reach within a small number of steps by the current policy, although the states are close in physical space.

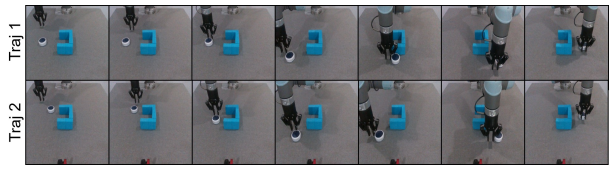

Fig. 10. Some trajectories to successfully push the puck to the goal position in the real world.

E. Performance of REM

To answer the fourth question, we visualize the heatmap of REM, which shows the reachability score between two observations. Some examples are shown in Fig. 9. The heatmap is based on the top view of *UR-Pusher-2*. The red triangle represents the object position in the current state. Warmer colors represent higher reachability scores of the two states. To visualize the REM performance in the 2D plane, we keep the robotic end effector close to the object and uniformly sample the object position on the table.

As is shown in Fig. 9, the reachability of temporally close pairs is high, while that of temporally far pairs is low. In particular, when two states are separated by obstacles, even though they are close in physical space and representation space, REM can infer that the current GC policy cannot reach them within a small number of steps, and tend to predict these reachability scores to be zero.

F. Ablation Study of Each Component in REPlan

To answer the fifth question, we perform several ablation experiments to analyze the role of different components in REPlan, which is shown in Table. I. The components include DRM, REM and Curiosity Module(CM). REPlan w/o REM&CM replaces the subgoal metric with TDMs, which is similar to LEAP. REPlan w/o DRM uses VAE as the encoder. We sample 30 goals from the same distribution and test each model 30 times with 5 seeds. In one episode, when the Euclidean distance between the object and the goal in the plane is smaller than 0.03 meters, we define it as a success.

TABLE I
ABLATION EXPERIMENTS FOR EACH COMPONENT

Method	Success rate(%)		
	UR-Pusher-1	*UR-Pusher-2*	*UR-Pick-Place*
REPlan w/o REM&CM	0.0	0.0	0.0
REPlan w/o CM	56.7	36.7	30.0
REPlan w/o DRM	66.7	60.0	53.3
REPlan	90.0	93.3	83.3

As is shown, REPlan w/o REM&CM cannot complete the tasks, since the model cannot explore high-quality samples in complex environments, leading to inaccurate distance measure and poorly planned subgoals. REPlan w/o CM can achieve some goals successfully, which shows that REM can help the model to plan better subgoals to guide the agent toward the final goal. However, the sample efficiency of the algorithm decreases in certain training epochs. We note that the success rate of REPlan w/o CM dropped faster in the more complex *UR-Pusher-2* and *UR-Pick-Place*. It shows that CM can further help agents to explore more high-quality samples in complex environments. REPlan w/o DRM makes certain progress in 2 tasks. With all components, REPlan can reach the distant goals with a success rate of 90.0% in *UR-Pusher-1*, 93.3% in *UR-Pusher-2* and 83.3% in *UR-Pick-Place*. The further improvement of the success rate shows the compact disentangled representation extracted by DRM can significantly improve the performance of the algorithm.

G. Real-world Experiments

Our REPlan can effectively transfer control policies from simulation to the real world by separating scene representation and control. We transfer the GC policies learned in *UR-Pusher-1* to the real-world task, named *Real-Pusher*. We train DRM of REPlan with 500 real images, including different robot poses and object positions. The qualitative results are given in Fig. 7. As is shown, our DRM can learn the disentangled representations of robots and objects from real observations.

To align disentangled representations under different domains, we learn a simple linear mapping of robot representations with the same pose and object representations with the same position in simulated and real images. We test on UR5 30 times to push the puck to the goal position. Some successful trajectories are shown in Fig. 10. Despite some deviations in

the representation mapping of different domains, our REPlan can reach the distant goals with a success rate of 76.67% and exhibit transferability to the real world.

V. Conclusion

We propose a goal-conditioned algorithm combined with planning, denoted as disentanglement-based REachability Planning with Goal-Conditioned RL (REPlan). REPlan decomposes temporally extended tasks into a sequence of subgoals to guide the GC policy to reach distant goals. To explicitly extract task-relevant information, a self-supervised Disentangled Representation probabilistic Module (DRM) is proposed to disentangle the physical attributes of the entities (object positions and robot poses). On top of the representations, a REachability discrimination Module (REM) is designed to measure the reachability of subgoal sequences, which can further be optimized by a gradient-free planning algorithm. In addition, REM provides an intrinsic bonus to encourage exploration. We empirically demonstrate that REPlan can significantly outperform prior state-of-the-art methods in three challenging vision-based simulated tasks and exhibit transferability to the real world.

Our work suggests several directions for further research. While REPlan needs to collect offline datasets to learn the entity representations in the observations, how to simultaneously explore environments and learn representations remains challenging. Self-supervised entity discovery from more cluttered real-world scenarios is another open issue. In addition, future work can encourage the exploration of GCRL by using self-generated subgoals or intrinsic bonuses to obtain high-quality samples for efficient planning and policy learning.

References

[1] D. Shah, B. Eysenbach, G. Kahn, N. Rhinehart, and S. Levine, "Ving: Learning open-world navigation with visual goals," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 13 215–13 222.

[2] S. Nasiriany, V. Pong, S. Lin, and S. Levine, "Planning with goal-conditioned policies," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[3] Z. Qian, M. You, H. Zhou, and B. He, "Weakly supervised disentangled representation for goal-conditioned reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2202–2209, 2022.

[4] A. Goyal, A. Mousavian, C. Paxton, Y.-W. Chao, B. Okorn, J. Deng, and D. Fox, "Ifor: Iterative flow minimization for robotic object rearrangement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 787–14 797.

[5] M. Liu, M. Zhu, and W. Zhang, "Goal-conditioned reinforcement learning: Problems and solutions," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, L. D. Raedt, Ed. International Joint Conferences on Artificial Intelligence Organization, 7 2022, pp. 5502–5511, survey Track. [Online]. Available: https://doi.org/10.24963/ijcai.2022/770

[6] E. Chane-Sane, C. Schmid, and I. Laptev, "Goal-conditioned reinforcement learning with imagined subgoals," in *International Conference on Machine Learning*. PMLR, 2021, pp. 1430–1440.

[7] T. Zhang, B. Eysenbach, R. Salakhutdinov, S. Levine, and J. E. Gonzalez, "C-planning: An automatic curriculum for learning goal-reaching tasks," in *The Tenth International Conference on Learning Representations, ICLR 2022*. OpenReview.net, 2022, pp. 1–17.

[8] B. Eysenbach, R. R. Salakhutdinov, and S. Levine, "Search on the replay buffer: Bridging planning and reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[9] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *International Conference on Learning Representations (ICLR)*, pp. 1–14, 2014.

[10] L. P. Kaelbling, "Learning to achieve goals," in *International Joint Conference on Artificial Intelligence*, vol. 2. Citeseer, 1993, pp. 1094–8.

[11] C. Colas, T. Karch, O. Sigaud, and P.-Y. Oudeyer, "Autotelic agents with intrinsically motivated goal-conditioned reinforcement learning: a short survey," *Journal of Artificial Intelligence Research*, vol. 74, pp. 1159–1199, 2022.

[12] M. Andrychowicz, D. Crow, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *NIPS*, 2017.

[13] C. Packer, P. Abbeel, and J. E. Gonzalez, "Hindsight task relabelling: Experience replay for sparse reward meta-rl," *Advances in Neural Information Processing Systems*, vol. 34, pp. 2466–2477, 2021.

[14] L. Zhang, G. Yang, and B. C. Stadie, "World model as a graph: Learning latent landmarks for planning," in *International Conference on Machine Learning*. PMLR, 2021, pp. 12 611–12 620.

[15] V. Pong, S. Gu, M. Dalal, and S. Levine, "Temporal difference models: Model-free deep rl for model-based control," in *International Conference on Learning Representations*, 2018, pp. 1–14.

[16] A. Srinivas, M. Laskin, and P. Abbeel, "Curl: Contrastive unsupervised representations for reinforcement learning," in *ICML*, 2020.

[17] A. V. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine, "Visual reinforcement learning with imagined goals," *Advances in neural information processing systems*, vol. 31, 2018.

[18] A. Khazatsky, A. Nair, D. Jing, and S. Levine, "What can i do here? learning new skills by imagining visual affordances," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 14 291–14 297.

[19] W. Yuan, C. Paxton, K. Desingh, and D. Fox, "Sornet: Spatial object-centric representations for sequential manipulation," in *Conference on Robot Learning*. PMLR, 2022, pp. 148–157.

[20] Y. Wang, N. Gautham, X. Lin, B. Okorn, and D. Held, "Roll: Visual self-supervised reinforcement learning with object reasoning," in *Conference on Robot Learning*. PMLR, 2021, pp. 1030–1048.

[21] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International conference on machine learning*. PMLR, 2017, pp. 2778–2787.

[22] Y. Burda, H. Edwards, A. Storkey, and O. Klimov, "Exploration by random network distillation," in *International Conference on Learning Representations*, 2018, pp. 1–17.

[23] H. Tang, R. Houthooft, D. Foote, A. Stooke, O. Xi Chen, Y. Duan, J. Schulman, F. DeTurck, and P. Abbeel, "# exploration: A study of count-based exploration for deep reinforcement learning," *Advances in neural information processing systems*, vol. 30, 2017.

[24] D. Jo, S. Kim, D. W. Nam, T. Kwon, S. Rho, J. Kim, and D. Lee, "Leco: Learnable episodic count for task-specific intrinsic reward," in *Advances in Neural Information Processing Systems*, 2022.

[25] N. Savinov, A. Raichuk, R. Marinier, D. Vincent, M. Pollefeys, T. Lillicrap, and S. Gelly, "Episodic curiosity through reachability," in *International Conference on Learning Representations (ICLR)*, 2019, pp. 1–20.

[26] Z. Lin, Y.-F. Wu, S. V. Peri, W. Sun, G. Singh, F. Deng, J. Jiang, and S. Ahn, "Space: Unsupervised object-oriented scene representation via spatial attention and decomposition," in *International Conference on Learning Representations*, 2020, pp. 1–22.

[27] J. Jiang, S. Janghorbani, G. De Melo, and S. Ahn, "Scalor: Generative world models with scalable object representations," in *International Conference on Learning Representations*, 2020, pp. 1–22.

[28] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, p. 2017–2025.

[29] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of operations research*, vol. 134, no. 1, pp. 19–67, 2005.

[30] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[31] Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín, "robosuite: A modular simulation framework and benchmark for robot learning," *arXiv preprint arXiv:2009.12293*, 2020.

[32] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[33] L. V. D. Maaten and G. E. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.