Solving Robotic Manipulation with Sparse Reward Reinforcement Learning via Graph-Based Diversity and Proximity

Zhenshan Bing, Hongkuan Zhou, Rui Li, Xiaojie Su, *Senior Member, IEEE*
Fabrice O. Morin, Kai Huang, Alois Knoll, *Senior Member, IEEE*

***Abstract*—In multi-goal reinforcement learning (RL), algorithms usually suffer from inefficiency in the collection of successful experiences in tasks with sparse rewards. By utilizing the ideas of relabeling hindsight experience and curriculum learning, some prior works have greatly improved the sample efficiency in robotic manipulation tasks, such as hindsight experience replay (HER), hindsight goal generation (HGG), graph-based hindsight goal generation (G-HGG), and curriculum-guided hindsight experience replay (CHER). However, none of these can learn efficiently for solving challenging manipulation tasks with distant goals and obstacles, since they rely either on heuristic or simple distance-guided exploration. In this work, we introduce graph-curriculum-guided hindsight goal generation (GC-HGG), an extension of CHER and G-HGG, that works by selecting hindsight goals on the basis of graph-based proximity and diversity. We evaluated GC-HGG in four challenging manipulation tasks involving obstacles in both simulations and real-world experiments, in which significant enhancements in both sample efficiency and overall success rates over prior works were demonstrated. Videos and codes can be viewed at this link: https://videoviewsite.wixsite.com/gc-hgg.**

***Index Terms*—Reinforcement learning, hindsight experience replay, robotic arm manipulation, path planning.**

I. Introduction

DEEP reinforcement learning has successfully revolutionized the process of solving decision-making problems in many areas ranging from robotics, for example, solving a Rubik's cube or enabling autonomous driving, to mind games such as AlphaGo, Atari games, and Starcraft [1]–[6]. A recurrent problem of RL, however, is that it requires handcrafted reward functions that are tailored to individual tasks, which usually feature complex and as yet unknown behaviors in most real-world applications. Therefore, the design of a proper reward is challenging and a major impediment to the widespread adoption of RL for use in real-world applications.

On the one hand, recent work has shown that learning with sparse rewards, such as binary signals indicating a task's completion, can enable better applicability to multi-goal RL tasks than engineered dense rewards, since sparse rewards can easily be derived from the task definition without any further manual engineering effort. On the other hand, the sparsity of such rewards hinders the ability to collect sufficient successful experience, usually resulting in a long learning period and a low success rate. To tackle this issue, researchers have proposed several ideas to improve the sample efficiency, such as relabeling hindsight experience with different tasks [7] or learning on a designed sequence of training samples, such as a curriculum, to progress from easy to difficult [8].

One such relabeling approach is hindsight experience replay (HER) [9], which greatly improves the success rate and sample efficiency of RL algorithms in multi-goal RL tasks with sparse rewards. HER first learns with hand-crafted heuristic hindsight goals from previously achieved states that are easy to reach and then continues with difficult goals. However, it is limited in that it is only applicable when goals can be easily reached by heuristic explorations and fails in environments with distant goals. In such environments, the agent only explores around the initial states and will never be able to reach real goals, since it experiences no positive reward during random explorations.

Curriculum-guided HER (CHER) [8] extended the idea of HER by adaptively prioritizing the replay buffer entries according to the diversities with respect to other replay goals and the proximities to target goals. The diversity metric is formulated as the intra-distances among hindsight goals selected for training and is maximized to encourage exploration. The proximity metric is modeled as the Euclidean distance between hindsight goals and target goals and is minimized to encourage exploitation. However, CHER is not applicable to tasks with obstacles that are able to mislead the Euclidean distance. Moreover, it fails to provide a mechanism for designing a proper trade-off with which to balance the diversity and proximity, which leads to low sample efficiency in complex tasks where delicate exploration strategies are required.

Another idea for improving the sample efficiency is based on curriculum learning, which aims to design a proper curriculum for guiding the exploration step by step towards final goals. Hindsight goal generation (HGG) [10] is an automatic curriculum generation approach that selects proper intermediate goals that can lead the agent to the target goals and are also easy to reach at the same time. HGG is better able to solve tasks with long-distant goals than HER or CHER, since it can experience positive rewards when reaching intermediate goals generated by the curriculum. Similar to CHER, HGG uses the Euclidean distance to define the Wasserstein distance

Z. Bing, H. Zhou, F. Morin, and A. Knoll are with the Department of Informatics, Technical University of Munich, Germany. E-mail: bing@in.tum.de. K. Huang is with the School of Computer Science, Sun Yat-sen University, China. R. Li and X. Su are with the School of Automation, Chongqing University, China.

to measure the distance between a set of intermediate goals and the final goal distribution. Therefore, HGG is also not applicable in environments with obstacles, in which the shortest obstacle-avoiding distance between two goals cannot be computed using the Euclidean metric. On the basis of HGG, our prior work, graph-based hindsight goal generation (G-HGG), solves this problem by selecting hindsight goals based on shortest distances in an obstacle-avoiding graph, which is a discrete representation of the environment [11]. Although G-HGG is applicable to environments with obstacles, it only learns from randomly sampled hindsight goals from replay-buffer, in which not all the experiences are equally useful to different learning stages. Furthermore, the sample efficiency of G-HGG highly depends on the gird size. A larger grid-size means a higher error in the shortest obstacle-avoiding distance and therefore miss-lead the selection of intermediate goals.

To overcome the aforementioned limitations, we propose graph-curriculum-guided hindsight goal generation (GC-HGG), an extension of CHER and G-HGG, to improve the sampling efficiency of solving complex robotic manipulation tasks with obstacles within the framework of sparse-reward RL. First, we create an obstacle-aware graph representation of the environment as pre-training steps. Second, we define a graph-based diversity that is lightweight to compute and a graph-based proximity that can guide the object through obstacles. Third, we design a trade-off mechanism that automatically balances diversity and proximity exploration. Finally, by comparing performances in four challenging manipulation environments, we show that GC-HGG provides a significant enhancement in both sample efficiency and success rate over HER, CHER, HGG, and G-HGG. By transferring learned policies from simulations, we were able to successfully perform GC-HGG on the four tasks in the real world.

Our main contribution to the literature is an algorithm that bridges graph-based planning, diversity and proximity-based exploration, and automatic curriculum generation for solving complex manipulation tasks. First, we upgrade the design of the graph-based representation of an environment proposed in G-HGG [11] by eliminating the possible interference between the obstacles and the body of the robotic arm. Second, the new lightweight graph-based diversity and objective optimization approach can greatly reduce the computation burden, while the graph-based proximity can guide the agent through environments with obstacles. By introducing this graph-based diversity, the negative effect of large gird-size on G-HGG can be largely reduced because the agent can still explore the environment, even intermediate goals are not selected appropriately. Third, our automatic mechanism for balancing proximity and diversity is a general solution that can be applied to different environments at different training stages.

II. Related Work

There are a number of previous studies that aim to develop informative and effective exploration strategies for solving goal-conditioned RL tasks with sparse rewards. We briefly introduce them on the basis of two main ideas.

A. Hindsight Experience Relabeling and Prioritization

Hindsight experience relabeling is a data enrichment method, and its intuitive idea is that some trajectory that is less informative for the current task is likely to be a rich information source for other tasks. By relabeling one trajectory with a different task that the behavior is better suited to, the knowledge gained can be used for different tasks and thus improves the sample efficiency. HER achieves its success by relabeling past experience with a heuristic choice of hindsight goals from achieved states, but it suffers from its inefficient random replay of experience. Prioritized sampling is a method of enforcing exploration on valuable experiences, and some of its ideas are based on the temporal-difference error [12], reward-weighted entropy [13], transition energy [14], and diversity-proximity of achieved goals [8].

B. Curriculum Learning and Automatic Goal Generation

Curriculum learning is another method of tackling the exploration problem in sparse-reward multi-goal RL. It aims to create a curriculum that enforces the agent to learn skills that are suitable for the current learning stage, according to the current ability of the agent. There are several ways of creating such a curriculum, for example, using an intrinsic motivation to improve exploration [15]–[20]. Another way of constructing a meaningful curriculum is to predict high-reward states and to generate goals close to these meaningful states [8], [21]–[23]. Another idea for improving exploration that is similar to curriculum learning is automatic goal generation, which aims to first solve some sub-problems (easy problems) that will be helpful later on for solving the complex problems. The generation of these sub-problems (which are often intermediate goals) is expected to be automatic. One approach for selecting appropriate goals for the current training stage is to use a goal generative adversarial network (Goal GAN) [24]. A goal discriminator is trained to evaluate whether a goal is at the appropriate level of difficulty for the current policy. Some other ideas are based on making some certain characteristics of a goal [25], inverse dynamics [26], and imitation learning [7].

Although all of the above methods have demonstrated improved exploration, they share one significant drawback: exploration is not well guided towards distant target goals. In other words, when the goals are "hidden" from the agent, e.g., when the target goals are far away from the initial goals or blocked by obstacles, the unguided exploration process may take too long to learn the task or may even fail to do so.

III. Preliminary

A. Curriculum-Guided Hindsight Experience Replay

C-HER [8] samples hindsight goals guided by the diversity and proximity. The diversity demonstrates an agent's curiosity to explore an environment. A set of goals with high diversity means that there is more exploration in different states and different areas within the environment. The proximity denotes how close these goals are to the desired goals. A large proximity enforces training towards the desired goals.

Consider that we have already sampled all the achieved trajectories in a replay buffer B. A set of B contains all goals

achieved in previous exploration. In each iteration, a mini-batch needs to be sampled to train the policy. In contrast to uniform sampling, C-HER proposes selecting a subset A of B ($A \subset B$), according to the diversity and proximity of A. The proximity of A can be defined as,

$$F_{prox} \triangleq C_1 - \sum_{g_i \in A} (dis(g_i, g)) \tag{1}$$

where $dis(g_i, g)$ is the Euclidean distance between a hindsight goal g_i and the target goal g. C_1 is a constant to ensure $F_{prox} \geq 0$. Apparently, as the distance $dis(g_i, g)$ decreases, the proximity of set A increases. The diversity is defined as

$$F_{div}(A) \triangleq C_2 - \sum_{j \in B/A} \min_{i \in A} dis(g_i, g_j), \tag{2}$$

where C_2 is a constant to ensure $F_{div} \geq 0$. It should be noted that the values of C_1 and C_2 do not require any prior knowledge, since they can simply be set with large values in practice. Then the selection of A from B can be solved by following combinatorial optimization that maximizes both proximity and diversity:

$$\max_{A \subset B} F(A) \triangleq \lambda F_{prox}(A) + F_{div}(A) \tag{3}$$

where λ is a trade-off weight that balances the proximity and diversity, which controls the proportion of the diversity and proximity during training. It increase gradually in line with

$$\lambda_k = (1 + \tau)^k \lambda_0 \tag{4}$$

where τ determines the increasing rate of the proximity.

B. Hindsight Goal Generation (HGG)

HGG [10] extends HER to tasks with distant goal distributions that are far away from the initial state distribution and cannot be solved by heuristic exploration. These target goals $\mathcal{G}_T$ belong to a goal space $\mathcal{G}$ and the initial states $\mathcal{S}_0$ belong to the state space $\mathcal{S}$. The distribution $\mathcal{T}^* : \mathcal{G} \times \mathcal{S} \to \mathbb{R}$ determines how they are sampled. Instead of optimizing V^π with the difficult target goal-initial state distribution $\mathcal{T}^*$, which carries the risk of being too far from the know goals, HGG tries to optimize with a set of intermediate goals sampled from $\mathcal{T}$. On the one hand, the goals contained in $\mathcal{T}$ should be easy to reach, which requires a high $V^\pi(\mathcal{T})$. On the other hand, goals in $\mathcal{T}$ should be close enough to $\mathcal{T}^*$ to be challenging for the agent. This trade-off can be formalized as

$$\max_{\mathcal{T}, \pi} V^\pi(\mathcal{T}) - L \cdot \mathcal{D}(\mathcal{T}^*, \mathcal{T}). \tag{5}$$

The Lipschitz constant L is treated as a hyper-parameter. In practice, to select these goals, HGG first approximates $\mathcal{T}^*$ by taking K samples from $\mathcal{T}^*$ and storing them in $\hat{\mathcal{T}}^*$. Then, for an initial state and goal $(\hat{s}_0^i, \hat{g}^i) \in \hat{\mathcal{T}}^*$, HGG selects a trajectory $\tau = \{s_t\}_{t=1}^T$ that minimizes the following function:

$$w(\hat{s}_0^i, \hat{g}^i, \tau) := c\|m(\hat{s}_0^i) - m(s_0)\| + \min_{s_t \in \tau} \left(\|\hat{g}^i - m(s_t)\| - \frac{1}{L} V^\pi((s_0 || m(s_t)) \right). \tag{6}$$

$m(\cdot)$ is a state abstraction that maps from the state space to the goal space. $||$ is a symbol of concatenation. $c > 0$ provides a trade-off between 1) the distance between target goals and 2) the distance between the goal representation of the initial states. Finally, from each of the K selected trajectories τ^i, the hindsight goal g^i is selected from the state $s_t^i \in \tau^i$, that minimized (6).

$$g^i := \arg\min_{s_t \in \tau} \left(\|\hat{g}^i - m(s_t)\| - \frac{1}{L} V^\pi((s_0 \| m(s_t)) \right). \tag{7}$$

C. Graph-based Hindsight Goal Generation (G-HGG)

Our prior work G-HGG replaces the Euclidean-distance with Graph-based distance to get the appropriate obstacle-avoiding shortest distance to guide the exploration. To do that, we create a bounded goal space $\mathcal{G}_\mathcal{A}$ containing all the goals that the object can reach. Then we create a representation of $\mathcal{G}_\mathcal{A}$ with a graph $G = (P, E)$, which consists of a set of vertices P, weighted edges E, and an assigned weight w.

$$E \subset \{(p_1, p_2, w) \mid (p_1, p_2) \in P^2, p_1 \neq p_2, w \in \mathbb{R}\}, \tag{8}$$

where p_1 and p_2 are two possible vertices.

In environments with obstacles, any goals $g_{obs} \in \mathcal{G}$ lying within an obstacle that are blocked from being reached are not elements of the accessible goal space $g_{obs} \notin \mathcal{G}_A$. Since $\mathcal{G}_A$ is bounded, it can be enclosed in a parallelepipedic bounding box defined by values x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max} $\in \mathbb{R}$, which describes the span of the box in each coordinate direction. We then use this box to generate a finite set of vertices $\hat{P}$, spatially arranged in an orthorhombic lattice. $\hat{P}$ is defined by the total number of vertices $n = n_x \cdot n_y \cdot n_z$, with $n_x, n_y, n_z \in \mathbb{N}$ in each direction of $\mathcal{G}_A$, or alternatively by the distance between two adjacent grid-points in each coordinate direction given by

$$\Delta_x = \tfrac{x_{max} - x_{min}}{n_x - 1},\ \Delta_y = \tfrac{y_{max} - y_{min}}{n_y - 1},\ \Delta_z = \tfrac{z_{max} - z_{min}}{n_z - 1}. \tag{9}$$

Finally, the set of vertices are defined as $P = \hat{P} \cap \mathcal{G}_A$, where

$$\hat{P} := \{(x_{min} + \Delta_x \cdot i,\ y_{min} + \Delta_y \cdot j,\ z_{min} + \Delta_z \cdot k) \mid i \in [0, n_x - 1], j \in [0, n_y - 1], k \in [0, n_z - 1]\}. \tag{10}$$

In the next step, we connect two adjacent vertices $p_1 = (\hat{x}_1, \hat{y}_1, \hat{z}_1) \in P$, $p_2 = (\hat{x}_2, \hat{y}_2, \hat{z}_2) \in P$ with an edge of weight w considering the following:

$$(p_1, p_2, w) \in E \iff |\hat{x}_2 - \hat{x}_1| \leq \Delta_x, |\hat{y}_2 - \hat{y}_1| \leq \Delta_y \text{ and } |\hat{z}_2 - \hat{z}_1| \leq \Delta_z, \tag{11}$$

where $w := \sqrt{(\hat{x}_2 - \hat{x}_1)^2 + (\hat{y}_2 - \hat{y}_1)^2 + \cdot(\hat{z}_2 - \hat{z}_1)^2}$.

In environments with obstacles, it is important to ensure that no edge in the graph cuts through an obstacle. To achieve that, we define the space of one of the cuboids with the edges α, β, γ. The graph must therefore satisfy the graph density criterion (12) for every convex sub-obstacle, i.e., the continuous set goals not included in $\mathcal{G}_\mathcal{A}$:

$$\Delta_x < \alpha_{min}^{obs};\ \Delta_y < \beta_{min}^{obs};\ \Delta_z < \gamma_{min}^{obs}, \tag{12}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIE.2022.3172754, IEEE Transactions on Industrial Electronics

where α_{min}^{obs}, β_{min}^{obs}, $\gamma_{min}^{obs} \in \mathbb{R}$, describing the infimum length of edges of all the obstacles.

Considering the graph that represents the environment, we employ a shortest path algorithm, such as Dijkstra's algorithm [27], to calculate the shortest paths and shortest distances $\hat{d}_G$ between every possible pair of vertices $(p_1, p_2) = ((\hat{x}_1, \hat{y}_1, \hat{z}_1), (\hat{x}_2, \hat{y}_2, \hat{z}_2)) \in P^2$ in a graph $G = (P, E)$. All possible combinations of the resulting shortest distance function $\hat{d}_G$ can be efficiently pre-computed and stored in an $n \times n$ table, where n denotes the number of vertices in P.

Given two goals $g_1 = (x_1, y_1, z_1) \in \mathcal{G}$, $g_2 = (x_2, y_2, z_2) \in \mathcal{G}$ and a graph $G = (P, E)$ representing the approximate goal space $\mathcal{G}_A \subset \mathcal{G}$ where x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}, Δ_x, Δ_y, and Δ_z, the graph-based distance $d : \mathcal{G}^2 \to \mathbb{R}$ is defined such that

$$d_G(g_1, g_2) = \begin{cases} \hat{d}_G\Big(\nu(g_1), \nu(g_2)\Big), & \text{if } g_1 \in \mathcal{G}_A \wedge g_2 \in \mathcal{G}_A \\ \infty, & \text{otherwise} \end{cases} \tag{13}$$

where ν maps goals in $\mathcal{G}_A$ to the closest vertex in P:

$$\nu(g) = \nu(x, y, z) = (\hat{x}, \hat{y}, \hat{z}) = \Big(x_{min} + \Delta_x \cdot \Big\lfloor \frac{x - x_{min}}{\Delta_x} \Big\rceil, \\ y_{min} + \Delta_y \cdot \Big\lfloor \frac{y - y_{min}}{\Delta_y} \Big\rceil,\ z_{min} + \Delta_x \cdot \Big\lfloor \frac{z - z_{min}}{\Delta_z} \Big\rceil\Big) \tag{14}$$

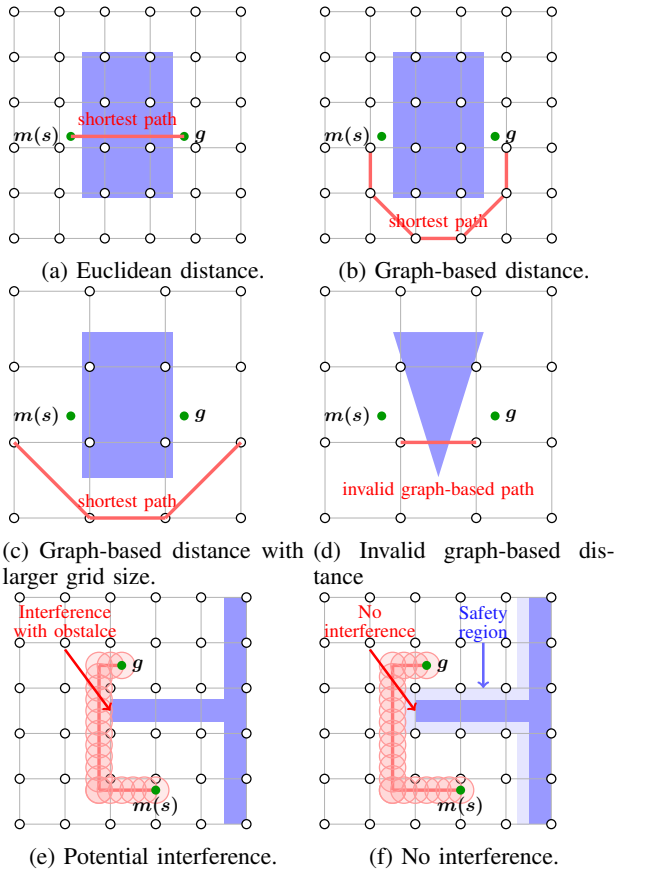
$\lfloor a \rceil$ rounds any $a \in \mathbb{R}$ to the closest integer value. Mathematically, $d(g_1, g_2) = \hat{d}_G(\nu(g_1), \nu(g_2))$.

IV. Methodology

A. Problem Statement

Our method aims to solve robotic manipulation tasks with sparse rewards by making it applicable to environments with obstacles and improving its sample efficiency on the basis of G-HGG.

First, even G-HGG is capable of guiding exploration in environments with obstacle, the sample efficiency of G-HGG is sensitive to the grid size. In Fig. 1, we show an example in a 2D environment, in which the task is to push an object from its initial position $m(s)$ to the goal g. Figure 1a shows a case that Euclidean distance may select an invalid path that cut through an obstacle, while Fig. 1b shows a properly designed graph to generate a valid graph-based distance to bypass the obstacle. As shown in Fig. 1c and 1d, a larger grid size will lead to great distant error or even totally failure, since it might mislead the selection of intermediate goals. However, a much fine-tuned grid will lead to great computation burden or manual parameter tuning. One way to reduce the influence of large distance error is to ensure the diversity of hindsight goals selection. Second, G-HGG uniformly selects hindsight goals from all achieved goals with different significance for a success training, but it is not properly designed to control the adaptive exploration-exploitation trade-off in selecting suitable experiences for different environments, which is extremely important when there is a lack of distance-based guidance from the environment. Third, G-HGG simply considers the movement of the robotic arm as segments of lines, while it should be modeled as a region of space due to its body size.



(a) Euclidean distance. (b) Graph-based distance. (c) Graph-based distance with larger grid size. (d) Invalid graph-based distance (e) Potential interference. (f) No interference.

Fig. 1: The shortest Euclidean distance and graph-based distance between two points in a 2D environment with obstacles.

As illustrated in Fig. 1e, the space covered by the body of the robot is represented as light red circles. Although the trajectory (red lines) of the robotic arm has no intersection with the obstacle, its body still has interference with the obstacle during the movement, which may cause collision in the real world.

In this paper, we propose an algorithm named graph-curriculum-based hindsight goal generation (GC-HGG), in which graph-based distance is used to select suitable hindsight goals using graph-based diversity and proximity metrics. The architecture of GC-HGG is shown in Fig. 2. We first create a graph representation of the environment that can be used to calculate graph-based distances. We then reformulate (6) and (7) to select intermediate goals by replacing the Euclidean metric $||\hat{g}^2 - m(s_t^i)||$ with the graph-based distance d_G. Finally, we reformulate (1), (2), and (4) to prioritize hindsight experiences so as to improve the sample efficiency, replacing them with graph-based proximity and diversity, as well as a dynamic trade-off mechanism.

B. Graph Creation With Collision Tolerance

As illustrated in Fig. 1f, we improve the graph creation by incorporating the idea of the safety region, which is designed to avoid the potential collision between the obstacle and the body of the robotic arm. In theory, the best way to avoid the collision is to calculate any interference at every time step,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIE.2022.3172754, IEEE Transactions on Industrial Electronics

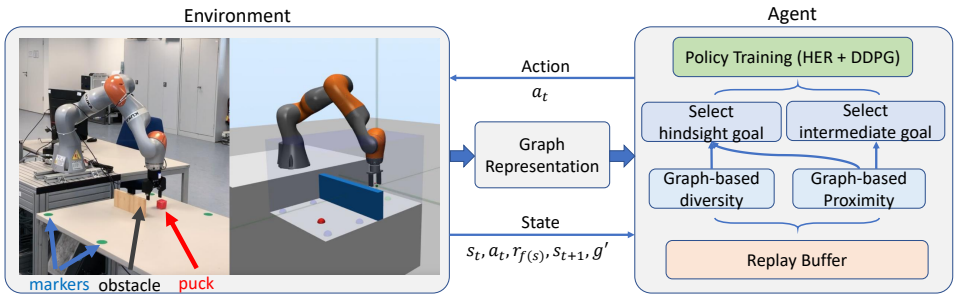


Fig. 2: Architecture of GC-HGG. GC-HGG creates a graph representation to define two metrics: graph-based diversity for selecting proper hindsight goals and graph-based proximity for selecting proper intermediate goals.

which consumes a great amount of computation. In this work, we simply tackle this by virtually increasing the size of the obstacle as a safety region, which is not accessible to the agent. The increased size can be determined by the radius ϵ of the space occupied by the robotic arm (the flange). Therefore, x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max} used in Section III-C are updated as $x_{min}-\epsilon$, $x_{max}+\epsilon$, $y_{min}-\epsilon$, $y_{max}+\epsilon$, $z_{min}-\epsilon$, $z_{max}+\epsilon$, respectively. Then, the graph can be created by following the steps in Section III-C.

C. Graph-Based Proximity

we can replace the Euclidean distance (1) in CHER by our graph-based distance given as

$$F_{prox} \triangleq C_1 - \sum_{g_i \in A} (d_G(g_i, g)), \tag{15}$$

where d_G is the graph-based distance as defined in (13). An illustration of such graph-based distance is given in Fig. 1b.

D. Graph-Based Diversity

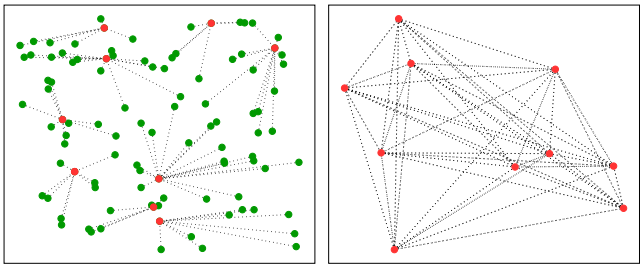
The diversity (2) in CHER is a metric with the evaluation time complexity around $O(|A||B|)$, where $|A|$ and $|B|$ are the sizes of the mini-batch and replay buffer, respectively. This large time complexity makes it infeasible when the task requires a large batch size for training. In this work, we redesign it by proposing a graph-based diversity that can sufficiently encourage the exploration and yet with a much smaller complexity of $O(|A|log_2|A|)$.

As illustrated in Fig. 3a, we first define $B_1, B_2, ..., B_{|A|} \in B$, and B_i contains those goals that are close to $g_i \in A$. Then the diversity in (2) can be approximated as

$$F_{div} \triangleq C_2 - \sum_{i=1}^{|A|} \sum_{g \in B_i} d_G(g, g_i), \tag{16}$$

where $B_1 ... B_{|A|}$ are $|A|$ clusters of goals which have relatively small intra-class distances. Since F_{div} is designed to encourage the diversity of goal distributions and we know that a good clustering should have both large inter-class distances and small intra-class distances, we can redesign the diversity on the basis of the inter-class distance, which is formulated as

$$\sum_{g_i \in A} \sum_{g_j \in A} d_G(g_i, g_j) \tag{17}$$



(a) Points (green) in B connects with the closest point(red) in A.

(b) the distances between any two points in A.

Fig. 3: Inter-class distance and intra-class distance.

To compute (17), we need 2 nested loops to iterate over all possible combinations of (g_i, g_j) in subset A and sum up the distances of each goal pair. Therefore the time complexity is $O(|A|^2)$. To reduce this time complexity, we derive the lower bound as

$$\begin{aligned} \sum_{g_i \in A} \sum_{g_j \in A} d_G(g_i, g_j) &\geq \sum_{g_i \in A} (|A| - 1) \cdot \min_{g_j \in A} d_G(g_i, g_j) \\ &\geq (|A| - 1) \cdot \sum_{g_i \in A} \min_{g_j \in A} d_G(g_i, g_j) \end{aligned}, \tag{18}$$

where $\forall g_i, g_j \in A$, $g_i \neq g_j$, $d_G(g_i, g_j) \geq \min_{g \in A} d_G(g_i, g)$.

The advantage of using this lower bound as a metric for diversity is that we only need to calculate the distance between each goal in the subset A and its nearest neighbor rather than all other goals in A. By leveraging the k-nearest-neighbor algorithm with k-d tree structure, we can complete the calculation of diversity within time complexity of $O(|A|log|A|)$, because for each goal in A, querying its nearest neighbor only needs $O(log|A|)$ and there exists $|A|$ goals in the set A. Finally, after normalization, the diversity $F_{div}(A)$ from (2) can be reformulated as

$$F_{div}(A) = \sum_{g_i \in A} \min_{g_j \in A} d_G(g_i, g_j). \tag{19}$$

E. Trade-off Between Curiosity and Proximity

To increase the sample efficiency, an agent has to sample from a large diversity to explore the environment and gradually focus on specific goals. In CHER, the trade-off λ was defined

Algorithm 1 Curriculum-guided sampling

Given: the number of random samples K, the trade-off λ calculated as (4)
$F_{max} = -inf$, $A_{select} = \emptyset$
for 1 to K **do**
    randomly sample A from replay buffer B using EBP
    find K-nearest-neighbour of A
    calculate $F(A) = F_{div}(A) + \lambda F_{prox}(A)$
    if $F(A) > F_{max}$ **then**
        $A_{select} = A$, $F_{max} = F(A)$
return A_{select}

only as a simple exponential function in (4). This naive design cannot adapt to tasks with different difficulties and results in poor sample efficiency.

Based on the Wasserstein distance of the desired goal distribution $\mathcal{T}^*$ and the current intermediate goal distribution $\mathcal{T}$, we propose a new and adaptive trade-off as

$$\lambda = \eta exp(\frac{-D_G(\mathcal{T}, \mathcal{T}^*)}{\sigma^2}), \tag{20}$$

where η and σ are hyper-parameters and $D_G(\cdot, \cdot)$ is the graph-based Wasserstein distance, given as

$$\mathcal{D}_G(\mathcal{T}^{(1)}, \mathcal{T}^{(2)}) := \inf_{\mu \in \Gamma(\mathcal{T}^{(1)}, \mathcal{T}^{(2)})} \left(\mathbb{E}_\mu \left[d_G(s_0^{(1)} || g^{(1)}, s_0^{(2)} || g^{(2)}) \right] \right). \tag{21}$$

This trade-off mechanism will encourage the agent to explore the environment with a large degree of curiosity in the initial learning state or at a stage at which the intermediate goals (achieved hindsight goals) are far from the desired goals. Also, as $D_G(\cdot, \cdot)$ decreases (intermediate goals approach the desired goals), λ will gradually and adaptively increase, so as to enlarge the weight of the proximity.

F. Objective Optimization

Optimizing the final objective (3) can be regarded as the facility location problem, which is NP-hardness with high computational complexity. CHER uses a greedy algorithm to obtain an approximate solution, which largely reduces the complexity of this NP-hard problem. Meanwhile, it still needs $O(|B||A|)$ times F_{div} and F_{prox} evaluations, where $|A|$ and $|B|$ are the sizes of mini-batch and replay buffer, respectively. However, considering the millions of samples throughout the whole training process, it is not necessary to calculate the maximum value of $F(A)$ from the entire replay buffer. A sampled minibatch A with relatively higher $F(A)$ would have a positive impact on our training. In this work, we reduce this computation complexity by randomly sampling $A_1, A_2, ..., A_K$ from B and only selecting $A_i = \arg\max_{A_i \in \{A_1, A_2, ..., A_K\}} F(A_i)$ as the mini-batch (See Algorithm 1). Therefore, the computation time of the sampling is reduced to $O(K|A| \log_2 |A|)$, since the computation time of $F_{prox}(A)$ is $O(|A|)$ and $F_{div}(A)$ is $O(|A| \log_2 |A|)$, using the k-nearest-neighbor algorithm. With such a complexity reduction, each iteration in the training process costs only about 180 seconds with these sizes of minibatch (256) and replay buffer (10000) in our experiment, while CHER takes around 5 hours with the same setting. The entire algorithm is provided as Algorithm 2.

Algorithm 2 GC-HGG

Given: an off-policy RL algorithm $\mathbb{A}$, a strategy $\mathbb{S}$ for sampling goals for replay, a sparse reward function r_g.
Construct a graph representation G ▷ Sec. III-C
Pre-compute the shortest distance $\hat{d}_G$ between every pair of vertices $(p_1, p_2) \in P^2$ with Dijkstra
Initialize $\mathbb{A}$ and replay buffer R
for $iteration$ **do**
    Construct a set of M intermediate tasks $\{(\hat{s}_0^i, g^i)\}_{i=1}^M$:
    - Sample target tasks $\{(\hat{s}_0^i, \hat{g}^i)\}_{i=1}^K \sim \mathcal{T}^*$
    - Find K distinct trajectories $\{\tau^i\}_{i=1}^K$ that together minimize (6) ▷ weighted bipartite matching
    - Find M intermediate tasks $(\hat{s}_0^i, g^i)$ by selecting an intermediate goal g^i from each τ^i
    - Calculate the balance of proximity and diversity based on the Wasserstein distance between the intermediate and the target goal distribution ▷ (20)
    for $episode = 1, M$ **do**
        $(s_0, g) \leftarrow (\hat{s}_0^i, g^i)$
        for $t = 0, T-1$ **do**
            Sample a_t using the policy from $\mathbb{A}$ with noise:

$$a_t \leftarrow \pi(s_t \parallel g) + \mathcal{N}_t \tag{22}$$

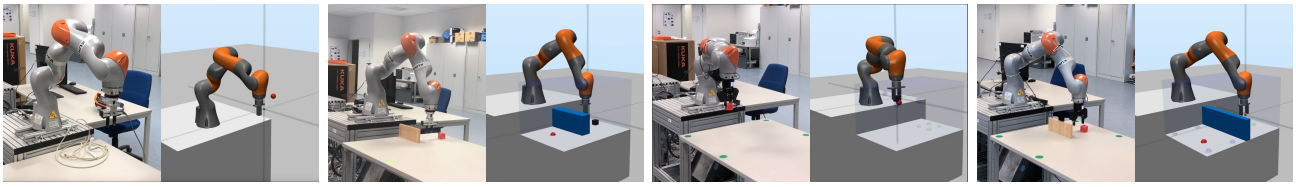
            Execute a_t and observe a new state s_{t+1}
        for $t = 0, T-1$ **do**
            $r_t := r_g(s_t, a_t)$
            Store transition $(s_t||g,\ a_t,\ r_t,\ s_{t+1}||g)$ in R
            Sample a set of additional goals for replay $G := \mathbb{S}(current\ episode)$
            for $g' \in G$ **do**
                $r' := r_{g'}(s_t, a_t)$; Store the transition $(s_t||g',\ a_t,\ r',\ s_{t+1}||g')$ in R ▷ HER
    for $t = 1, N$ **do**
        B = curriculum-guided sampling ▷ Alg. 1
        One optimization step using $\mathbb{A}$ and B ▷ DDPG

V. Experiments

A. Environments

To show the advantages of GC-HGG over HGG, HER, and CHER, we create new experimental environments inspired by the widely adopted Fetch-gripper benchmark [9]. All our environments are MuJoCo environments, which feature a modeled Kuka robot with a gripper. They all adopt the following control strategy. The state space $\mathcal{S}$ contains positions and velocities for all the joints of the robotic arm, the position of the end-effector, and the position and orientation of the object. Depending on whether gripper control is enabled or disabled, the action space is three- or four-dimensional. An action consists of the end effector's position for the next time step and the gripper's opening control parameter.

KukaReach (Figure 4a): The goal is simply to control the arm to reach a goal position. The gripper remains permanently



(a) KukaReachEnv (b) KukaPushNewEnv (c) KukaPickNoObstacleEnv (d) KukaPickObstacleEnv

Fig. 4: Robotic manipulation environments in simulations and the real world setup.

(a) KukaReach (b) KukaPushNew (c) KukaPickNoObstacle (d) KukaPickObstacle

Fig. 5: Median success rate (line) and interquartile range (shaded) in different environments. One iteration contains 50 episodes.

closed and the gripper control is disabled, leading to a 3D action space. This environment is the easiest task that is designed as a benchmark task to examine the usability of each algorithm. **KukaPushNew** (Figure 4b): The goal is to push the puck from its initial position around the blue obstacle towards a goal. The gripper remains permanently closed and the gripper control for picking up the object is disabled, leading to a three-dimensional action space. **KukaPickNoObstacle** (Figure 4c): The goal is to pick up the cube from its initial position, lift it up, and place it at a goal position. No obstacle is present in this scenario, but the target goals are located in the air. Gripper control is enabled, and the gripper can be symmetrically opened and closed by a single actuator, leading to a 4D action space. This task is designed to compare the exploration capability of different algorithms in an obstacle-free environment that involves grasping. **KukaPickObstacle** (Figure 4d): The goal is to pick up the cube from its initial position, lift it over the obstacle, and place it at a goal position. Gripper control is enabled, the gripper can be symmetrically opened and closed leading to a 4-D action space. Note that all four scenarios are multi-goal tasks and therefore the goal is randomly picked in each episode within the predefined range. The sparse reward used in each scenario is defined as

$$r_g(s,a) := \begin{cases} 0 & \text{, if } ||g - m(s)|| \leq 0.05 \\ -1 & \text{, otherwise} \end{cases}, \qquad (23)$$

where $||g - m(s)||$ represents the Euclidean distance between the gripper or the puck to the goal. 0.05 is a distance threshold.

B. Results

We tested GC-HGG in the four environments with that of compare its performance to HGG, HER, and CHER. Since we know that EBP [14] enhances the performance of both HER and HGG, we used EBP in all training of HER and HGG. The results clearly show that GC-HGG far outperforms the other algorithms in all complex environments, in terms of both sample efficiency and maximum success rate.

KukaReach: In KukaReach, there exists no much difference in performance among HER, HGG, CHER, GC-HGG (See Fig. 5a). Since no complex exploration is required and no obstacle is present, FetchReach can be easily solved by all algorithms within 10 iterations. In most instances, the success rate peaks after around 10 iterations. It should be noted that, in a scenario that no obstacle is present, the graph-based distance can simply be replaced with the Euclidean distance.

KukaPushNew: In KukaPushNew, GC-HGG performs far better than the other algorithms (Fig. 5b) in terms of their success rates. While CHER, HGG, and HER are not be able to solve this task over 200 training iterations, GC-HGG achieves a success rate over 90% after 70 iterations. The reasons are obvious. First, HGG and CHER repeatedly choose the hindsight goals which are close to the target position, guided by the sampled intermediate goals using the Euclidean metric. These hindsight goals will be blocked by an obstacle, which would have limited effects on exploration. Second, due to a poor trade-off between exploration and exploitation, some hindsight goals can even bypass the obstacle, they are rarely selected as suitable intermediate goals, and resulting the failure of the training. Since KukaPushNew is not as challenging as the KukaPickNoObstalce and KukaPickObstacle, in which the agent has to explore to learn to grasp the object, G-HGG is also able to solve the task with slightly worse sample efficiency and much higher variance than GC-HGG.

KukaPickNoObstacle: In this task, GC-HGG displays a remarkable performance. Within 70 iterations, the success rate of GC-HGG reaches over 90%. In terms of HGG and G-HGG, they need more iterations to complete this task. This shows that a diverse exploration at the beginning and a gradual focus on the target have distinctly positive effects on the training.

KukaPickObstacle: In this task, GC-HGG is able to complete the task in 250 iterations. Although G-HGG can also

complete the task, GC-HGG demonstrates better sample efficiency and robustness. The other methods HER, CHER and HGG fail to finish the task. We can clearly see from the figure that a large diversity at the beginning of the training process and increasing the proximity based on the current learning stage is helpful to both exploration efficiency and the median success rate.

The computation time of each algorithm is demonstrated in Table I. All experiments are performed on the Intel Core i9 9880H CPU and the AMD Radeon Pro 5300 GPU. The training time of HER for each iteration is the shortest because of its simple structure. HGG requires more time compared to HER due to the selection of intermediate goals. G-HGG spends more computation time than HGG, while most of the time is used on reading out the graph-based distance metric during the training process. The computational time for GC-HGG is slightly higher than G-HGG due to the K time random sampling algorithm to select the mini-batch. In our set up, we set K equals 8 and implemented parallel computation. CHER faces time issue, especially when the size of the reply-buffer is large. It costs approximately 5 hours to finish one iteration. The table also indicates that the total computational cost of GC-HGG is even less than G-HGG in the environments KukaPickNoObstacle and KukaPickObstacle, because we implemented the same stop condition check [11] of HGG or G-HGG in our GC-HGG. The algorithm will continue with HER if the selected intermediate goals are close enough to the original target goals. Since GC-HGG has a better sample efficiency, the agent can learn faster compared with G-HGG so that it takes fewer iterations to switch to HER. Note that Table I only shows the training time, while all the tasks are performed in real time when deployed in the real world.

TABLE I: Computation time of experimental runs (* indicates that the algorithm can not complete the task in 400 iterations)

Environment	HER	HGG	G-HGG	GC-HGG	CHER
KukaReach	7(h)	7(h)	7(h)	7(h)	8(h)
KukaPushNew	7(h)*	9(h)*	11(h)	11.5(h)	>10(d)*
KukaPickNoObstacle	7(h)*	8.5(h)	15.5(h)	8(h)	>10(d)*
KukaPickObstacle	7(h)*	7(h)*	15.5(h)	12.5(h)	>10(d)*
Single Iteration	52(s)	87(s)	153(s)	179(s)	5(h)

C. Ablation Study

We present the ablation studies of η and σ for the trade-off coefficient λ, the trade-off method (ours and CHER's), and distance metrics (graph-based distance and Euclidean distance). Due to the page limit, we only show the ablation study results in the environment KukaPickNoObstacle. The full ablation study in other scenarios can be found at here[1]. Figure 6a illustrates the success rates for various values of η in (20) in the environment KukaPickNoObstacle. We set $\sigma = 0.3$ and keep it unchanged during the experiments. The plot shows that GC-HGG performs best when $\eta = 1000$. Other choices of η only show a slight degradation in sample efficiency. Figure 6b demonstrates the success rates for various of σ, where $\sigma = 0.3$ gives the best performance. We set $\eta = 1000$ and it remains the same during the experiments. Figure 6c shows the success rates of different trade-off methods. The blue curve indicates our trade-off based on (20). The other three demonstrate the success rates of GC-HGG with CHER's trade-off which based on (4). As we can seen from the figure, our trade-off has a better sample efficiency and more robust than CHER's, since our trade-off method is based on the current learning progress rather a naive designed exponential function. Figure 6d is an ablation study of distance metrics. There is no much difference between graph-based distance and euclidean distance since the environment KukaPickNoObstacle does not contain an obstacle. The ablation study of distance metrics for other environments can be found on our website.

[1]https://videoviewsite.wixsite.com/gc-hgg

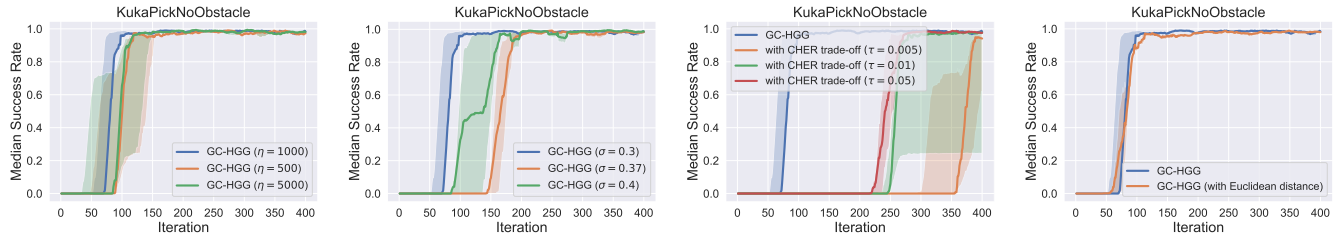
D. Real-World Experiments

The real-world experiment setup comprised with a KUKA LBR iiwa R800 robotic arm, a Robotiq 2F85 gripper, and an XBOX 360 Kinect, used to obtain the coordinates of the manipulatable object. Since the robotic arm is controlled via its default Java interface and our RL controller is programmed with Python, we use JPype, a python module that provides full access to Java, to exchange data between the robot and the RL controller. The control rate has the same value as the simulation, which is 20 Hz across all experiments.

As in the four environments in the simulation, we also create four environments featuring the KUKA robotic arm with the gripper (See Fig. 4). It should be noted that unlike the simulation that obtains the coordinates of the object directly, we use a camera to gather this information when the object is on the table in the real world. Specifically, we use four green markers to create a global coordinate and obtain the coordinates of the object by tracking its red color. However, when the object is picked up, we can directly calculate its position from the kinematics of the robotic arm.

We took the policy directly from each of the tasks trained in the simulation and deployed it in the real world without any fine-tuning. Inspired by the experiment performed by HER, we also add Gaussian noise to the observed object's position during policy training to compensate for the small errors introduced by the camera, which can increase the success rate of these tasks. The performances of these four tasks demonstrate that the policy can be successfully transferred to the corresponding tasks in the real world.

As shown in the video, the robot arm can always successfully approach the target position in most trajectories. KukaReach has the highest success rate among these four tasks since it is the easiest one and the position of the end effector obtained from the Kuka interface is accurate. Even the position of the object is obtained using images, the robotic arm can still solve KuakPush, KukaPickNoObstacle, and KukaPickObstacle with no significant drop in the success rate compared to the performances in the simulation. The median success of the real-world experiments are listed in Table II. KukaReach has the highest success rate among these four tasks, since it is the easiest one and the position of the end effector obtained from the Kuka interface is accurate. For KukaPushNew, there exists only two cases that the robotic arm can not push the cube to the right position. For one case, it pushed the cube



(a) Ablation study of η. (b) Ablation study of σ. (c) Ablation study of trade-off methods. (d) Ablation study of distance metrics.

Fig. 6: Ablation studies in the environment of KukaPickNoObstacle.

out of its reaching range by accident. For the other, the end effector blocked the view of the camera so that the camera could not detect the position of the cube. These unexpected cases sometimes also happened in KukaPickObstacle and KukaPickNoObestacle. The median success rates for those two tasks are therefore decreased compared to the performance in corresponding simulation environments. Videos of these failure cases on the project page.

TABLE II: The median success rates of real-world environments and simulation environments (- indicates that the algorithm can not solve the task.)

	KukaReach		KukaPushNew	
	sim.	real	sim.	real
HER	1.0	1.0(15/15)	-	-
HGG	1.0	1.0(15/15)	-	-
CHER	1.0	1.0(15/15)	-	-
G-HGG	1.0	1.0(15/15)	0.97 $\pm$ 0.03	0.80(12/15)
GC-HGG	1.0	1.0(15/15)	**0.99 $\pm$ 0.01**	**0.86(13/15)**
	KukaPickNoObstacle		KukaPickObstacle	
	sim.	real	sim.	real
HER	-	-	-	-
HGG	0.99 $\pm$ 0.01	0.80(12/15)	-	-
CHER	-	-	-	-
G-HGG	0.99 $\pm$ 0.01	0.73(11/15)	0.82 $\pm$ 0.18	0.60(10/15)
GC-HGG	**0.99 $\pm$ 0.01**	**0.8(12/15)**	**0.99 $\pm$ 0.01**	**0.80(12/15)**)

E. Discussion

To summary the main differences between GC-HGG and the other algorithms, we provide a brief discussion. The main difference between GC-HGG and HGG lies in the intermediate goal selection strategy, with which those intermediate goals are selected and used for the optimization of the policy. GC-HGG enables the agent to select intermediate goals only from the accessible goal space, in which all the obstacles are eliminated. Therefore, GC-HGG is applicable in scenarios with obstacles. However, HGG only selects intermediate goals by replying on the Euclidean distance, which is not applicable in scenarios with obstacles. There is no mechanism for CHER or HER to select proper intermediate goals. GC-HGG proposes an automated trade-off mechanism to balance the diversity and the proximity during training, which is designed on the basis of graph-based distance and leads to superior performance.

To discuss the potential collision between the body of the arm and the obstacle, a push task is illustrated in Figure 7, in which the arm is controlled to push one object from its left side to its right side by bypassing the obstacle. As explained

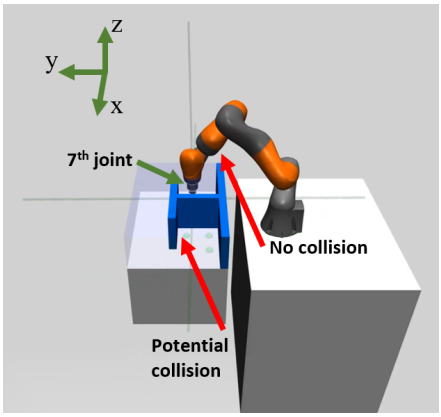


Fig. 7: Illustration of potential collision between the robotic arm and the obstacle.

before, the action space for this task is 3 dimensional, namely, the position of the gripper in x, y, z direction. Therefore, the gripper can only have translational movement in the $x-y$ plane in this task. Since the obstacle is not higher than the 7^{th} joint of the robotic arm in the goal space, there is no possibility that joint 1-5 will have potential collision with the obstacle. The only possible collision can happen between the gripper and the labyrinth obstacle. As explained in Figure 1.e and Figure 1.f in the revision paper, the safety region can make sure that there is no collision between the gripper and the obstacle in the $x-y$ plane by setting a safety threshold.

VI. Conclusion

We introduced a novel automatic hindsight goal generation and exploration algorithm GC-HGG on the basis of G-HGG and CHER for complex object manipulation in environments with obstacles, in which the selection of valuable hindsight goals is generated by balancing the graph-based diversity and proximity metrics. We schemed GC-HGG as a graph construction as pretraining steps, and the graph-based diversity and proximity computations as critical steps, during the training. Simulations and real-world experiments on four different challenging object manipulation tasks demonstrated superior performance by GC-HGG over HER, CHER, or HGG in terms of both the maximum success rate and sample efficiency.

Acknowledgment

This project/research has received funding from the European Union's Horizon 2020 Framework Programme for

Research and Innovation under the Specific Grant Agreement No.945539 (Human Brain Project SGA3). We thank Prof. Alin Albu-Schäffer and Mr. Daniel Seidel from the German Aerospace Center for providing the robotic arm.

REFERENCES

[1] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[2] Z. Hou, J. Fei, Y. Deng, and J. Xu, "Data-efficient hierarchical reinforcement learning for robotic assembly control applications," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 11, pp. 11 565–11 575, 2021.

[3] L. Wen, X. Li, and L. Gao, "A new reinforcement learning based learning rate scheduler for convolutional neural network in fault classification," *IEEE Transactions on Industrial Electronics*, 2020.

[4] W. Bai, T. Li, Y. Long, and C. L. P. Chen, "Event-triggered multigradient recursive reinforcement learning tracking control for multiagent systems," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2021.

[5] T. Li, W. Bai, Q. Liu, Y. Long, and C. L. P. Chen, "Distributed fault-tolerant containment control protocols for the discrete-time multiagent systems via reinforcement learning method," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2021.

[6] Y. Li, J. Zhang, W. Liu, and S. Tong, "Observer-based adaptive optimized control for stochastic nonlinear systems with input and state constraints," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2021.

[7] B. Eysenbach, X. GENG, S. Levine, and R. R. Salakhutdinov, "Rewriting history with inverse rl: Hindsight inference for policy improvement," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33, pp. 14 783–14 795. Curran Associates, Inc., 2020.

[8] M. Fang, T. Zhou, Y. Du, L. Han, and Z. Zhengyou, "Curriculum-guided Hindsight Experience Replay," *NeurIPS 2019*, 2019.

[9] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *Neurips*, pp. 5049–5059, 2017.

[10] Z. Ren, K. Dong, Y. Zhou, Q. Liu, and J. Peng, "Exploration via Hindsight Goal Generation," *33rd Conference on Neural Information Processing Systems, NeurIPS 2019*, 2019.

[11] Z. Bing, M. Brucker, F. O. Morin, R. Li, X. Su, K. Huang, and A. Knoll, "Complex robotic manipulation via graph-based hindsight goal generation," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2021.

[12] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *International Conference on Learning Representations, 2016*.

[13] R. Zhao, X. Sun, and V. Tresp, "Maximum entropy-regularized multi-goal reinforcement learning," in *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 13 022–13 035, 2019.

[14] R. Zhao and V. Tresp, "Energy-Based Hindsight Experience Prioritization," *2nd Conference on Robot Learning, CoRL 2018*, 2018.

[15] S. Singh, A. G. Barto, and N. Chentanez, "Intrinsically motivated reinforcement learning," *Neurips*, 2005.

[16] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," *34th International Conference on Machine Learning, ICML 2017*, vol. 6, 2017.

[17] S. Forestier, Y. Mollard, and P.-Y. Oudeyer, "Intrinsically Motivated Goal Exploration Processes with Automatic Curriculum Learning," 2017. [Online]. Available: http://arxiv.org/abs/1708.02190

[18] A. Péré, S. Forestier, O. Sigaud, and P. Y. Oudeyer, "Unsupervised learning of goal spaces for intrinsically motivated goal exploration," *6th International Conference on Learning Representations, ICLR 2018*.

[19] C. Colas, P. Founder, O. Sigaud, M. Chetouani, and P. Y. Oudeyer, "CURIOUS: Intrinsically motivated modular multi-goal reinforcement learning," in *International Conference on Machine Learning*, 2019.

[20] A. Aubret, L. Matignon, and S. Hassas, "A survey on intrinsic motivation in reinforcement learning," 2019.

[21] A. Goyal, P. Brakel, W. Fedus, S. Singhal, T. Lillicrap, S. Levine, H. Larochelle, and Y. Bengio, "Recall traces: Backtracking models for efficient reinforcement learning," *7th International Conference on Learning Representations, ICLR 2019*, pp. 1–19, 2019.

[22] C. Florensa, D. Held, X. Geng, and P. Abbeel, "Automatic Goal Generation for Reinforcement Learning Agents," *Proceedings of the 35th International Conference on Machine Learning*, 2018.

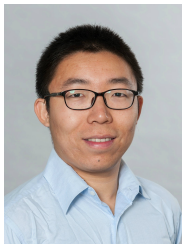
[23] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, and P. Abbeel, "Reverse Curriculum Generation for Reinforcement Learning," *1st Conference on Robot Learning, CoRL*, 2017.

[24] C. Florensa, D. Held, X. Geng, and P. Abbeel, "Automatic goal generation for reinforcement learning agents," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80, pp. 1515–1528, 10–15 Jul 2018. [Online]. Available: http://proceedings.mlr.press/v80/florensa18a.html

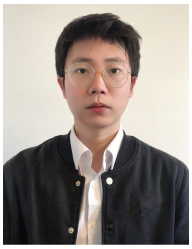
[25] M. Eppe, S. Magg, and S. Wermter, "Curriculum goal masking for continuous deep reinforcement learning," *2019 Joint IEEE 9th International Conference on Development and Learning and Epigenetic Robotics, 2019*, pp. 183–188, 2019.

[26] H. Sun, Z. Li, X. Liu, B. Zhou, and D. Lin, "Policy continuation with hindsight inverse dynamics," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.

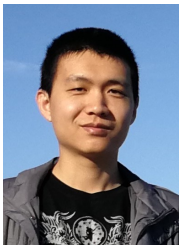
[27] E. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik 1*, 1959.



Zhenshan Bing received his doctorate degree in Computer Science from the Technical University of Munich, Germany, in 2019. He received his B.S degree in Mechanical Design Manufacturing and Automation from Harbin Institute of Technology, China, in 2013, and his M.Eng degree in Mechanical Engineering in 2015, at the same university. Dr. Bing is currently a post-doctroal researcher with Informatics 6, Technical University of Munich, Munich, Germany. His research investigates the snake-like robot which is controlled by artificial neural networks and its related applications.



Hongkuan Zhou received his B. Sc. Degree in Games 2Engineering at Technical University of Munich, Germany, in 2021. His research interest is artificial intelligence on robotics implementations, especially with reinforcement learning algorithms.



Rui Li received the B.Eng degree in automation engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2013 and the Ph.D. degree from the Institute of Automation, Chinese Academy of Science (CASIA), Beijing, China in 2018, respectively. He visited Informatics 6, Technical University of Munich as a guest post-doc researcher in 2019. Currently he is a assitant researcher with School of Automation, Chongqing University. His research interests include intelligent robot system and high-precision assembly.

Fabrice O. Morin received an engineering degree from the Ecole Nationale Supérieure des Mines de Nancy (Nancy, France) in 1999, a Master's Degree in Bioengineering from the University of Strathclyde (Glasgow, United Kingdom) in 2000, and a Ph.D. in Materials Science from the Japanese Advanced Institute of Science and Technology (Nomi-Shi, Japan) in 2004. After several post-docs at the University of Tokyo (Japan) and the IMS laboratory (University of Bordeaux, France), in 2008 he joined Tecnalia, a nonprofit RTO in San Sebastián (Spain), first as a senior researcher, then as a group leader. There, he worked on various projects in Neurotechnology and Biomaterials, funded both by public programs and private research contracts. Since 2017, he has worked as a scientific coordinator at the Technical University of Munich (Germany) where, in the framework of the Human Brain Project, he oversees the development of software tools for embodied simulation applied to Neuroscience and Artificial Intelligence.

Kai Huang joined Sun Yat-Sen University as a Professor in 2015. He was appointed as the director of the Institute of Unmanned Systems of School of Data and Computer Science in 2016. He was a senior researcher in the Computer Science Department, the Technical University of Munich, Germany from 2012 to 2015 and a research group leader at fortiss GmbH in Munich, Germany, in 2011. He earned his Ph.D. degree at ETH Zurich, Switzerland, in 2010, his MSc from University of Leiden, the Netherlands, in 2005, and his BSc from Fudan University, China, in 1999. His research interests include techniques for the analysis, design, and optimization of embedded systems, particularly in the automotive and robotic domains. He was awarded the Program of Chinese Global Youth Experts 2014 and was granted the Chinese Government Award for Outstanding Self-Financed Students Abroad 2010. He was the recipient of Best Paper Awards ESTC 2017, ESTIMedia 2013, SAMOS 2009, Best Paper Candidate ROBIO 2017, ESTMedia 2009, and General Chairs' Recognition Award for Interactive Papers in CDC 2009. He has served as a member of the technical committee on Cybernetics for Cyber-Physical Systems of IEEE SMC Society since 2015.

Xiaojie Su (SM'18) received the PhD degree in Control Theory and Control Engineering from Harbin Institute of Technology, China in 2013. He is currently a professor and the associate dean with the College of Automation, Chongqing University, Chongqing, China. He has published 2 research monographs and more than 50 research papers in international referred journals.

His current research interests include intelligent control systems, advanced control, and unmanned system control. He currently serves as an Associate Editor for a number of journals, including IEEE Systems Journal, IEEE Control Systems Letters, Information Sciences, and Signal Processing. He is also an Associate Editor for the Conference Editorial Board, IEEE Control Systems Society. He was named to the 2017, 2018 and 2019 Highly Cited Researchers list, Clarivate Analytics.

Alois Knoll (Senior Member) received his diploma (M.Sc.) degree in Electrical/Communications Engineering from the University of Stuttgart, Germany, in 1985 and his Ph.D. (*summa cum laude*) in Computer Science from Technical University of Berlin, Germany, in 1988. He served on the faculty of the Computer Science department at TU Berlin until 1993. He joined the University of Bielefeld, Germany as a full professor and served as the director of the Technical Informatics research group until 2001. Since 2001, he has been a professor at the Department of Informatics, Technical University of Munich (TUM), Germany. He was also on the board of directors of the Central Institute of Medical Technology at TUM (IMETUM). From 2004 to 2006, he was Executive Director of the Institute of Computer Science at TUM. Between 2007 and 2009, he was a member of the EU's highest advisory board on information technology, ISTAG, the Information Society Technology Advisory Group, and a member of its subgroup on Future and Emerging Technologies (FET). In this capacity, he was actively involved in developing the concept of the EU's FET Flagship projects. His research interests include cognitive, medical and sensor-based robotics, multi-agent systems, data fusion, adaptive systems, multimedia information retrieval, model-driven development of embedded systems with applications to automotive software and electric transportation, as well as simulation systems for robotics and traffic.