Learning to Solve Tasks with Exploring Prior Behaviours

Ruiqi Zhu[1], Siyuan Li[2], Tianhong Dai[3], Chongjie Zhang[4], Oya Celiktutan[1]

***Abstract*— Demonstrations are widely used in Deep Reinforcement Learning (DRL) for facilitating solving tasks with sparse rewards. However, the tasks in real-world scenarios can often have varied initial conditions from the demonstration, which would require additional prior behaviours. For example, consider we are given the demonstration for the task of *picking up an object from an open drawer*, but the drawer is closed in the training. Without acquiring the prior behaviours of opening the drawer, the robot is unlikely to solve the task. To address this, in this paper we propose an Intrinsic Reward Driven Example-based Control (IRDEC). Our method can endow agents with the ability to explore and acquire the required prior behaviours and then connect to the task-specific behaviours in the demonstration to solve sparse-reward tasks without requiring additional demonstration of the prior behaviours. The performance of our method outperforms other baselines on three navigation tasks and one robotic manipulation task with sparse rewards. Codes are available at `https://github.com/Ricky-Zhu/IRDEC`.**

I. Introduction

Deep reinforcement learning (DRL) has demonstrated impressive performance in sequential decision-making problems, such as video games [1], [2], robotics manipulation [3], [4], and autonomous driving [5]. However, for tasks with sparse rewards, the lack of learning signals can hamper the learning process. To address this, demonstrations are often leveraged to facilitate the learning process.

In real-world applications, we can often encounter situations under which the initial conditions vary from the demonstration, termed as the *task-specific behaviour demonstration* in this paper, which therefore requires additional prior behaviours to complete the tasks. Prior works require collecting additional demonstration of prior behaviours, which have overlapped states with task-specific behaviour demonstration to overcome the problem [6]. However, as the initial conditions can be various, instead of collecting the demonstration of prior behaviours, the robots should be able to adapt to different initial conditions and leverage the available task-specific behaviour demonstration to learn the required prior behaviours. For example, consider we are given the demonstration for the task of *picking up the object from a closed drawer*, but in the training, there are obstacles in the way. Therefore, the robots should be able to explore and learn the essential prior behaviours of *removing the obstacles* first and then conduct the task-specific behaviours of *picking up the object from a closed drawer* to complete the task.

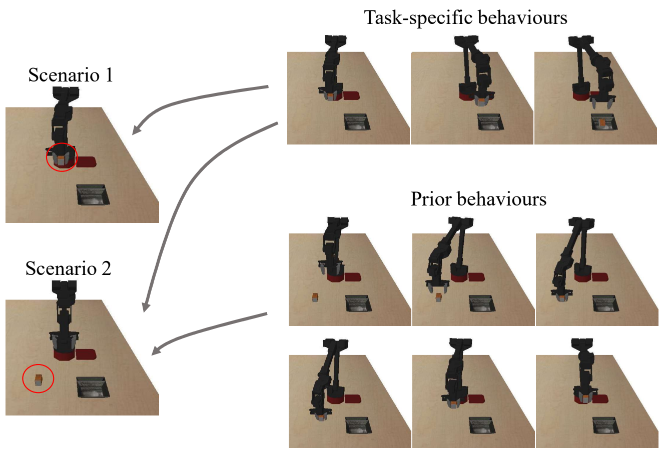

Fig. 1: Illustration of the problem definition. Scenario 1: the task of placing the in-hand object into the tray. Scenario 2: the task of placing the object, which is on the table, into the tray. task-specific behaviour demonstration: the demonstration of placing behaviours. Prior behaviours: the behaviours of grasping (top) and lifting (bottom). Scenario 1 can be solved by mimicing the task-specific behaviour demonstration while solving scenario 2 requires the prior behaviours that are absent in the task-specific behaviour demonstration. We attempt to solve scenario 2 given only the task-specific behaviour demonstration without the demonstration of the prior behaviours.

In this paper, we aim to utilize task-specific behaviour demonstrations to facilitate the learning of tasks with varied initial conditions without requiring additional demonstration of the prior behaviours. The absence of the prior behaviour demonstration indicates that the agent needs to acquire the essential prior behaviours to reach the demonstrated states in the task-specific behaviour demonstration. For example, as illustrated in Fig. 1, the task-specific behaviour is placing the in-hand object into the tray. The agent is required to learn the prior behaviours of grasping and lifting the object from the table, and then mimic the demonstrated actions of the task-specific behaviours to complete the *pick-and-place* task.

When children are shown the task-specific behaviour demonstration and then put in a different initial condition, they would intuitively explore the environment to return to the demonstrated states, which are termed as *familiar states*. Therefore, they can follow the demonstrated actions to complete the tasks. Inspired by that, we propose an intrinsic reward module to encourage the agents to explore, while the exploration direction is biased towards the familiar

[1] R. Zhu and O. Celiktutan are with the Department of Engineering, King's College London.
[2] S. Li is with the School of Computer Science and Technology, Harbin Institute of Technology.
[3] T. Dai is with the Department of Computing Science, University of Aberdeen.
[4] C. Zhang is with the Institute for Interdisciplinary Information Sciences (IIIS), Tsinghua University.

states for acquiring the required prior behaviours. Following that, the agents follow the demonstrated actions to complete the tasks, which is achieved with an adaptive behaviour regularizer. The whole framework is trained in an end-to-end manner and can be implemented with off-policy actor-critic RL algorithms, such as soft actor-critic (SAC) [7] and deep deterministic policy gradient (DDPG) [8]. Our method was evaluated on challenging long-horizon sparse-reward navigation [9] and robotic manipulation tasks [6]. The empirical results show that the proposed method can enable the agent to leverage the task-specific behaviour demonstration to learn the essential prior behaviours to solve tasks with sparse rewards efficiently.

The main contributions of this paper are: (i) We propose an Intrinsic Reward Driven Example-based Control **(IRDEC)** learning framework, which enables the agents to acquire prior behaviours and connects to the task-specific behaviours adaptively given only the task-specific behaviour demonstration. (ii) We compare our method with several baselines on challenging navigation and robotic manipulation tasks with sparse rewards and our method achieves the best results. (iii) We carry out an ablation analysis to investigate the importance of each component in our method and the results show that both components, the intrinsic reward module and the example-guided exploration, are necessary for effectively learning the required prior behaviours.

II. Related Work

Reinforcement Learning (RL) with Demonstration. Demonstrations are usually utilized in RL methods to facilitate learning in complex environments or mitigate the problem of sparse rewards. The utilization can often be categorized into: (1) imitation learning in which supervised learning is used to enforce the agent to mimic the demonstrated actions [10] or a reward function is inferred from the demonstration and the policy is trained to optimize it [11]–[13]; (2) behaviour extraction in which temporal abstracted behaviours are encoded from a large-scale offline dataset and policy is trained to optimize the extrinsic reward function by acting on the behaviour space to facilitate the exploration [14], [15]; (3) the demonstration is used to initialize the RL actor or regularize the RL actor during the training [16]. However, these works require that the demonstration contains all the required behaviours for the target tasks. But in our proposed method, only the task-specific behaviour demonstration is provided, while the essential prior behaviours are acquired during the training by leveraging the familiar states.

Goal-conditioned RL. Goal-conditioned RL has demonstrated competitive performance on tasks with sparse rewards. By augmenting the state with the goal sampled from the behavioural goal space, the sparse rewards are relabelled with the rewards computed via evaluating the Euclidean distance between the achieved goal and the sampled behaviour goal [9], [17]. In our proposed method, the task-specific behaviour demonstration contains the goal states of the target tasks, which implicitly implies the goal information. However, goal-conditioned RL requires a mapping function

Algorithm 1 Intrinsic Reward Driven Example-Based Control (IRDEC)

```
Input: task-specific behaviour demonstration $\mathcal{D}$
 1: Initialize: policy $\pi_\psi$, forward dynamics model $f_{\theta_{fw}}$, inverse dynamics model $g_{\theta_{inv}}$, state representation model $\phi_s$ , action representation model $\phi_a$, classifier $C_\omega(s,a)$, intrinsic value head $Q_\varphi(s,a)$, online replay buffer $\mathcal{B} \leftarrow \emptyset$, regularization weighting $\lambda_{reg} \leftarrow \lambda_0$
 2: for $k = 1, M$ do
 3:     Collect a new trajectory: $\mathcal{B} \leftarrow \mathcal{B} \cup \{\tau \sim \pi_\psi\}$
 4:     Sample transitions $\mathcal{MB} = \{s_j, a_j, r^e_j, s'_j\}^K_{j=1}$ from $\mathcal{B}$
 5:     Update models within the intrinsic module with samples $\mathcal{MB}$ using Eq. 5
 6:     Compute the intrinsic reward $r^i_j$ using Eq. 8
 7:     Replace the rewards $r^e_j$ in $\mathcal{MB}$ with $r_j = r^e_j + r^i_j$
 8:     Optimize the intrinsic value head by minimizing the loss in Eq. 9
 9:     Sample *familiar states* and corresponding demonstrated actions $(s^*_i, a^*_i)$ from $\mathcal{D}$
10:     Optimize the classifier by minimizing the loss in Eq. 12
11:     Update the policy using Eq. 17
12: end for
```

to transform the state space into goal space. In contrast, our method does not require the mapping function and is therefore more applicable.

Exploration with Intrinsic Reward. In environments with sparse rewards, intrinsic reward is usually leveraged to encourage the exploration of novel states before any extrinsic rewards are obtained. A large body of works focuses on using forward dynamic model prediction error as intrinsic reward [18]–[20]. However, intrinsic reward generated in this way can vanish with the agent being familiar with the environment, and therefore the prediction error tends to converge to zero [21], [22]. In [23], [24], a count-based exploration bonus is used to incentivize the agent to explore. However, the measure of the counts in large continuous state spaces is non-trivial. Our method incorporates curiosity intrinsic rewards, and impact intrinsic rewards, which do not vanish as the training progresses to encourage the agent to aggressively expand its explored state space so that the familiar states can more easily be encountered.

III. Background

The learning problem is formulated as a Markov Decision Process (MDP) characterized by a tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho, \gamma\}$ of state space, action space, transition probability mapping from current state s and action a to the next state s', reward function, initial state distribution, and discount factor. In each episode, the initial state $s_0 \in \mathcal{S}$ is sampled from the initial state distribution $s_0 \sim \rho(s_0)$, the agent chooses its action $a_t \in \mathcal{A}$ according to the policy $a_t \sim \pi(\cdot|s_t)$, and then the environment will generate the next state $s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t)$ and the reward $r = \mathcal{R}(s_t, a_t, s_{t+1})$. The objective of the agent is to maximize the sum of discounted rewards, $\mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{t=\infty} \gamma^t r_t\right]$, where trajectory τ is sampled from the policy π.

Example-based Control. Unlike the goal-conditioned RL, which requires the mapping from the state space to the goal space, example-based control provides another way of reaching the goal over the future state distribution, defined as Eq. 1 where s_{t+} is the future state. Given examples of goal states, the actor policy is optimized to maximize the probability of reaching these states [25], [26].

$$p^{\pi}(s_{t+}|s_t, a_t) \triangleq (1-\gamma) \sum_{\Delta=0}^{\infty} \gamma^{\Delta} p^{\pi}(s_{t+\Delta} = s_{t+}|s_t, a_t) \tag{1}$$

The probability as shown in Eq. 2 is defined over the future state distribution.

$$p^{\pi}(e_{t+}|s_t, a_t) = \mathbb{E}_{p^{\pi}(s_{t+|s_t, a_t})}[p(e_{t+}|s_{t+})] \tag{2}$$

where e_{t+} is referred to as the event of reaching the example states conditioned on the future state from time step t. In our method, we leverage the classifier in [25] to estimate the probability of reaching the demonstrated states in the task-specific behaviour demonstration in the future. However, the exploration introduced by maximizing the classifier values could often encounter danger areas in tasks, which can result in failure to solve the tasks. Please refer to Section IV for further details.

Off-policy Actor-critic RL. Our method can be implemented with off-policy actor-critic algorithms. In these algorithms, the critic learns an off-policy estimate of the value function for the current actor policy with the samples collected from behavioural actor policies [27]. The value function in our method estimates the weighted sum of the future return of the extrinsic and intrinsic rewards and the classifier values under the target actor policy. The actor policy in turn is optimized to maximize the estimates using the off-policy samples which are collected online.

IV. Intrinsic Reward Driven Example-Based Control Framework

The proposed method consists of two main components, namely, a curiosity-impact driven intrinsic reward module that encourages the agent to expand explored areas, and example-guided exploration provided by a classifier that predicts the probability of reaching familiar states in the future. To reach the familiar states and connect the task-specific behaviours, proper exploration over the state space is essential. The exploration direction introduced by directly maximizing the classifier values without the intrinsic reward module can potentially encounter danger areas. The exploration introduced by the intrinsic reward module without the guidance provided by the classifier can cover less promising areas related to the target tasks, which can be inefficient and could hinder the agent from solving the tasks. By expanding the explored areas towards the familiar states with leveraging the guidance from the task-specific behaviour demonstration, our proposed method can learn the essential prior behaviours and connect them to the behaviours in the task-specific behaviour demonstration for solving the target tasks as illustrated in Fig. 2.

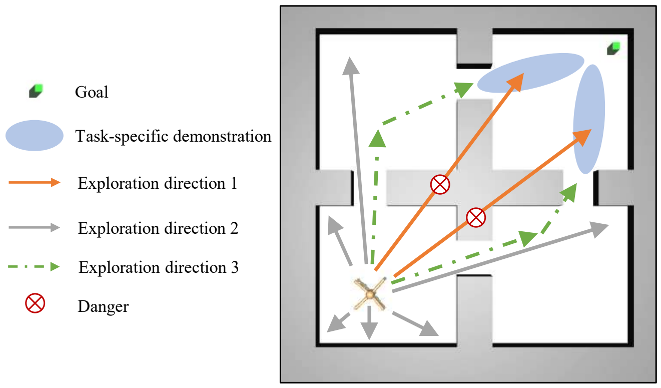

Fig. 2: Illustration of the proposed method. Exploration direction 1: the exploration direction introduced by the classifier. Exploration direction 2: the exploration direction introduced by the intrinsic reward module. Exploration direction 3: the exploration direction introduced by our method.

A. *Curiosity-Impact Driven Intrinsic Reward Module*

Our intrinsic reward module combines curiosity intrinsic rewards that encourage the agent to visit novel states [18] and impact intrinsic rewards that incentivize the agents to explore local states with large differences from current states, which can help to reach critical stages during the exploration [28]. We train a forward and an inverse dynamics models to learn the state representation $\phi_s(s)$ and the action representation $\phi_a(a)$. The forward dynamics model parameterized by θ_{fw} is used to predict the representation of the next state $\phi_s(s_{t+1})$ given the current state representation $\phi_s(s_t)$ and action representation $\phi_a(a_t)$. The loss for the forward dynamics model is given in Eq. 3:

$$L_{fw}(s_t, a_t, s_{t+1}) = \frac{1}{2}\|f_{\theta_{fw}}(\phi_s(s_t), \phi_a(a_t)) - \phi_s(s_{t+1})\|_2^2 \tag{3}$$

The inverse dynamics model parameterized by θ_{inv} is used to predict the action representation $\phi_a(a_t)$ given the consecutive state representations $\phi_s(s_t)$ and $\phi_s(s_{t+1})$. The loss for the inverse dynamics model is shown in Eq. 4. With the inverse dynamics model, the adverse impact on state representation learning, caused by the inherent noise of the environment where transition might not be affected by the agent's actions, i.e. noisy TV [19], [23], can be mitigated by retrieving the action leading to the transition.

$$L_{inv}(s_t, a_t, s_{t+1}) = -\log(g_{\theta_{inv}}(\phi_a(a_t)|\phi_s(s_t), \phi_s(s_{t+1})) \tag{4}$$

Thus, the total loss for the intrinsic module can be written as follows:

$$L_{icm} = L_{fw} + L_{inv} \tag{5}$$

The curiosity intrinsic reward is defined in Eq. 6. A larger curiosity intrinsic reward, namely a larger prediction error, indicates the novel states being visited as the forward dynamics model is unfamiliar with the transition.

$$r_t^{curiosity} = \|f_{\theta_{fw}}(\phi_s(s_t), \phi_a(a_t)) - \phi_s(s_{t+1})\|_2 \tag{6}$$

However, as the training progresses, the curiosity intrinsic rewards could vanish as the agent is being familiar with the environment. Thus, the impact intrinsic rewards defined as the squared Euclidean distance between consecutive state representations, as shown in Eq. 7, are utilized to encourage the agent to aggressively change its states to accelerate the exploration.

$$r_t^{impact} = \frac{\|\phi_s(s_{t+1}) - \phi_s(s_t)\|_2}{d_m^2}, \tag{7}$$

where d_m is the running average of the numerator for scaling the bonus. The impact intrinsic rewards will not vanish as the training progresses. Overall, the intrinsic reward can be written as:

$$r_t^i = \eta r_t^{curiosity} + (1-\eta) r_t^{impact}, \tag{8}$$

where $\eta > 0$ is a scalar weighing the two types of intrinsic rewards. Thus, the overall reward r_t at step t is defined as the addition of intrinsic reward r_t^i and the sparse extrinsic reward r_t^e. A value head parameterized by φ is trained to approximate the action value based on the overall reward. We optimize the value head by minimizing the loss:

$$L_Q = (r_t + \gamma \mathbb{E}_{a_{t+1}\sim\pi} Q^{\pi}_{\varphi'}(s_{t+1}, a_{t+1}) - Q^{\pi}_{\varphi}(s_t, a_t))^2, \tag{9}$$

where $Q^{\pi}_{\varphi'}$ is the target network of the value head Q^{π}_{φ} for stabilizing the training [1].

B. Example-guided Exploration

To learn the essential prior behaviours and connect them to the task-specific behaviours, visiting the overlapped state distribution of these behaviours are necessary [6]. Here we encourage the agent to visit the familiar states in the task-specific behaviour demonstration to construct such overlapped state distribution. To enable the agent to reach the familiar states, we utilize the classifier in [25] to discriminate between the state-action pairs which lead to reaching the familiar states in the future or not. The positive state-action pair is defined as the pair of familiar states sampled from task-specific behaviour demonstration $\mathcal{D}$ and the action given by the current policy conditioned on the sampled familiar states. The negative state-action pairs are those sampled from the online buffer $\mathcal{B}$. Thus, the relation between the optimal classifier and $p^{\pi}(e_{t+}|s_t, a_t)$ can be written as:

$$C_\omega(s_t, a_t) = \frac{p^{\pi}(s_t, a_t|e_{t+}=1)p(e_{t+}=1)}{p^{\pi}(s_t, a_t|e_{t+}=1)p(e_{t+}=1) + p(s,a)}. \tag{10}$$

Thus, the objective to optimize can be derived as:

$$p^{\pi}(e_{t+}=1|s_t, a_t) = \frac{C_\omega(s_t, a_t)}{1 - C_\omega(s_t, a_t)}. \tag{11}$$

The loss for the classifier is shown as Eq. 12. In the equation, $\mathcal{CE}$ denotes the cross entropy loss.

$$\begin{aligned}\mathcal{L}(\theta) =& (1-\gamma)\mathbb{E}_{s\sim\mathcal{D}, a\sim\pi(\cdot|s)} \mathcal{CE}(C_\omega(s,a); y_{pos}) \\ &+ (1+\gamma w)\mathbb{E}_{(s,a,s')\sim B} \mathcal{CE}(C_\omega(s,a); y_{neg})\end{aligned} \tag{12}$$

The positive target y_{pos} is 1, while the negative target y_{neg} is computed as follows:

$$y_{neg} = \frac{\gamma w(s')}{1 + \gamma w(s')}, \tag{13}$$

where the $w(s)$ is computed as shown in Eq. 14. The detailed derivation can be referred to [25].

$$w(s) = \frac{C^{\pi}_{\omega}(s, \pi(\cdot|s))}{1 - C^{\pi}_{\omega}(s, \pi(\cdot|s))} \tag{14}$$

C. Adaptive Behaviour Regularization

During the training, the agent should maintain the capacity of executing task-specific behaviours after encountering the familiar states, i.e., after grasping the object the robot needs to know how to place the object into the tray by mimicing the demonstrated actions in the task-specific behaviour demonstration. Here, we utilized Eq. 15 as the regularization loss to the update of the actor.

$$L_{reg}(s^*, a^*) = -\log(\pi_\psi(a^*|s^*)), \tag{15}$$

where (s^*, a^*) are the state-action pairs sampled from the task-specific behaviour demonstration. However, the weighting of the regularization in the actor update should vary in different stages of the training. In the early stage of the training, the agent focuses on exploring and reaching the familiar states; therefore, the weighting should be relatively small to avoid the adverse impact on the exploration of familiar states. When the agent acquires the prior behaviours to reach the familiar states, the agent should focus more on exploiting the task-specific behaviour demonstration and mimicing the demonstrated actions, where the weighting should be relatively large. The similarity between the sampled online collected states and familiar states increases as the agent is more capable of reaching the familiar states. Thus, we leverage a kernel density estimator fitted with the familiar states in the task-specific behaviour demonstration to estimate the similarity and therefore adjust the regularization of loss weighting. However, as the range of estimation scores is unknown, the effective way to adjust the weighting is to compare the scores of consecutive sampled batches. The weighting is computed as:

$$\lambda_{reg} = \text{clip}(\lambda_{i-1} + \frac{m(b_i) - m(b_{i-1})}{\max(m(b))} \times r, \lambda_{min}, \lambda_{max}) \tag{16}$$

where the λ_{i-1} is the weighting value at update step $i-1$, and $m(b_i)$ is the density estimation score of the sampled batch of states at update step i. r is the pre-defined rate. $\max(m(b))$ and λ_0 are the recorded maximum estimation score and the initial value of λ_{reg}. Meanwhile, we clip the λ_{reg} which is out of the pre-defined range $[\lambda_{min}, \lambda_{max}]$.

Overall, the parameters of the actor policy network can be updated as follows:

$$\begin{aligned}\psi \leftarrow \psi &+ \lambda \nabla_\psi \mathbb{E}_{a_t\sim\pi_\psi, s_t\sim\mathcal{B}}[(C^{\pi}(s_t, a_t) + Q^{\pi}(s_t, a_t))] \\ &+ \lambda_{reg} \nabla_\psi \mathbb{E}_{(s^*, a^*)\sim\mathcal{D}} L_{reg}(s^*, a^*).\end{aligned} \tag{17}$$

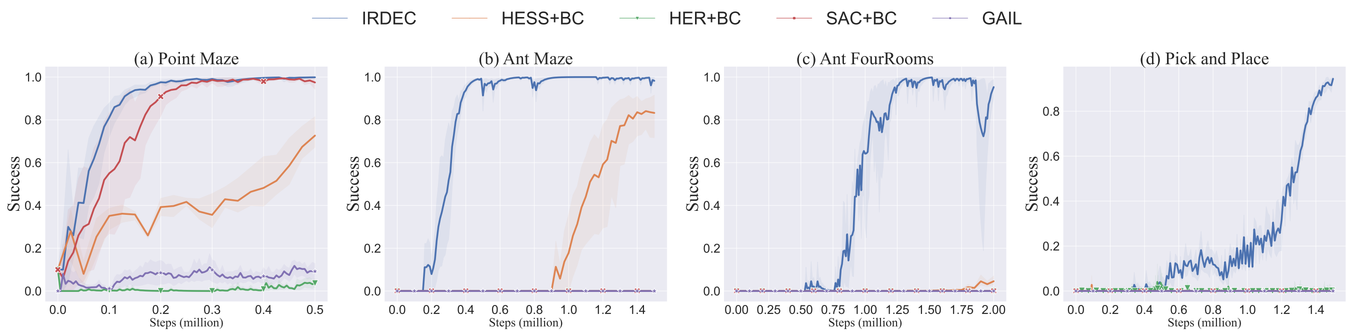

Fig. 3: Learning curves of the proposed method and baselines on all tasks. All the curves are smoothed equally for visual clarity.

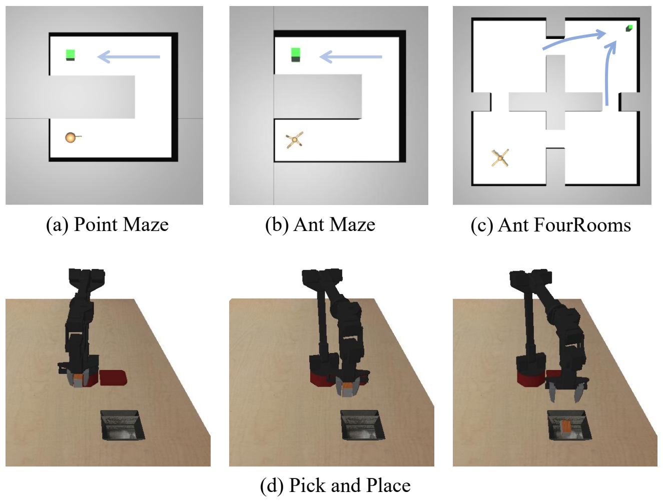

Fig. 4: Illustration of settings for all the tasks. (Top) Three navigation tasks: (a) Point Maze, (b) Ant Maze, and (c) Ant FourRooms. The arrows represent the trajectories of the task-specific behaviour demonstration. (Bottom) Pick and Place task. The sequence of the three frames represents the task-specific behaviour demonstration (placing behaviours).

V. EXPERIMENT

We aim to answer the following questions through our experiments: **(1)** Can our proposed method effectively and efficiently leverage the task-specific behaviour demonstration to learn the essential prior behaviours and further develop a complete policy for solving the target tasks with sparse rewards? **(2)** How does our method compare to alternative methods? **(3)** What is the importance of each component of our method?

A. *Experiment setup*

We evaluate the proposed framework on three long-horizon navigation tasks and one robotic manipulation task as illustrated in Fig. 4. In our problem settings, we assume the access to the task-specific behaviour demonstration $\mathcal{D}$ in the form of state-action trajectories $\tau_i = \{(s_0^*, a_0^*, ..., s_{T_i}^*, a_{T_i}^*)\}$. In our experiments, the task-specific behaviour demonstration for each task contains 100 trajectories collected with a sub-optimal policy. The trajectory lengths are around 150 and 25 for the navigation tasks and the robotic manipulation task respectively.

Navigation Tasks. We evaluated our method on three simulated long-horizon navigation tasks with sparse rewards based on Mujoco [29]. In these tasks, the agent should learn to control the robot (point or ant) to navigate through the maze to reach the goal as illustrated in Fig. 4 (a-c). For the point robot, the observation space $\mathcal{O} \in \mathbb{R}^6$ consists of the positions and velocities of the mass centre, and action space $\mathcal{A} \in \mathbb{R}^2$. For the ant robot, the observation space $\mathcal{O} \in \mathbb{R}^{29}$ consists of the positions and velocities of its torso, and action space $\mathcal{A} \in \mathbb{R}^8$. The extrinsic reward is given as $+1$ when the robot reaches the goal and 0 otherwise. The task-specific behaviour demonstration for each task is illustrated as arrows in Fig. 4 (a-c). The agent is supposed to learn the essential prior behaviours of navigating the robot to the familiar states and then conduct the behaviours of the task-specific behaviour demonstration to solve the tasks.

Pick and Place Task. To evaluate our method on robotic manipulation tasks, we utilized the Pick and Place environment in [6]. The simulated environment consists of a 6-DoF Widow X robot in front of a tray. The robot is supposed to learn how to control its arm to grasp the object from the table and place it in the tray. The observation space $\mathcal{O} \in \mathbb{R}^{17}$, includes the state of the end-effector and the gripper. The action space $\mathcal{A} \in \mathbb{R}^8$, includes the Cartesian coordinate changes, orientation changes of the end-effectors, and the gripper open degree. In the task, the agent needs to learn how to *grasp*, *lift* the object, and *place* the object in the tray. The extrinsic reward is $+1$, when the objective is placed in the tray and otherwise 0. For this task, the task-specific behaviour demonstration consists of the trajectories of controlling the robot to place the in-hand object into the tray as illustrated in Fig. 4 (d). The robot needs to learn the essential prior behaviours of grasping and lifting the object from the table and connect them to the task-specific behaviour to solve the task.

Baselines. We compare our method with: **(1) HER+BC** [17], a goal-conditioned method that utilizes a goal relabelling method to augment sufficient positive samples in goal-conditional tasks, while applying behaviour cloning loss to regularize the actor update with the task-specific behaviour demonstration. **(2) HESS+BC** [9], a goal-conditioned hierarchical method which utilized the high-level policy to assign sub-goal for the low-level policy to facilitate exploration, which has demonstrated competitive performance in long-horizon tasks with sparse rewards. Meanwhile, behaviour cloning loss is applied to regularize the actor update with

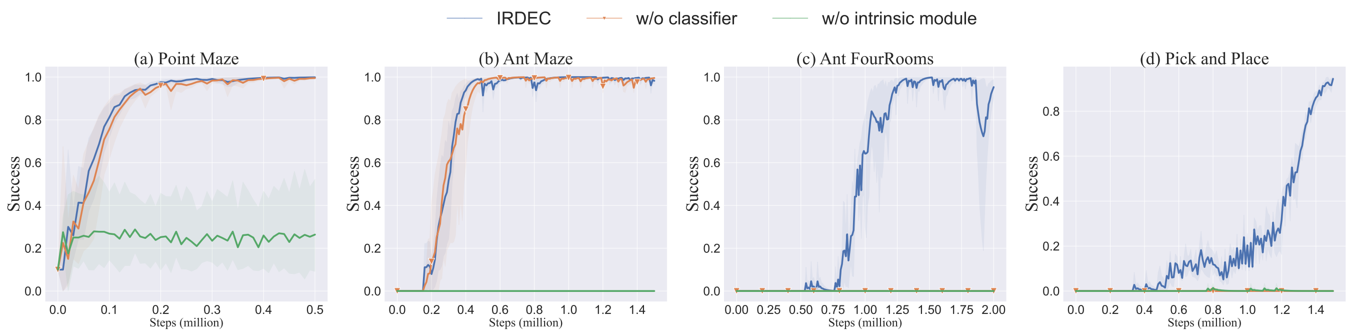

Fig. 5: Ablation studies of our proposed method.

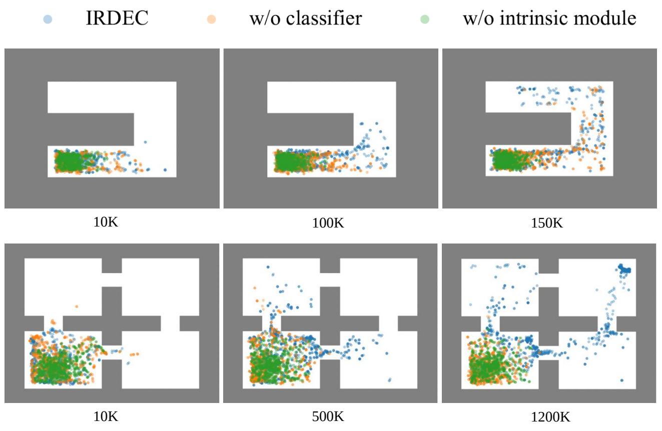

Fig. 6: Visulization of explored areas. 1K points are sampled uniformly from the online buffer to represent the explored areas. Top: the explored areas after training steps of 10K, 100K, and 150K for the Ant Maze task. Bottom: the explored areas after training steps of 10K, 500K, and 1200K for the Ant FourRooms task.

the task-specific behaviour demonstration. **(3) SAC+BC** [7], which trains the agent with SAC while applying behaviour cloning loss to regularize the actor update with the task-specific behaviour demonstration. **(4) GAIL** [11], an imitation learning method which attempts to match the state distribution between the training data and the demonstration. The hyperparameters used in these baseline methods were adopted from the original papers.

B. *Experimental results*

During the training, we evaluated the performance every 10K training steps with 10 episodes and recorded the average test returns. For each task, we trained our method and baselines with 5 different seeds and reported the results in Fig. 3. As shown in the figure, the proposed method outperforms all the baselines in all tasks. The outperformance is more significant in the Ant FourRooms task and the Pick and Place task as the exploration problems are harder. Without an effective incentive for expanding explored area, the sampled behavioural goals of HER+BC are insufficient with respect to diversity, which accounts for its poor performance in these long-horizon tasks [30]. By leveraging the hierarchical structure, HESS+BC assigns informative behavioural subgoals to encourage the low-level policy to visit promising areas for solving the tasks. However, it presents less efficiency compared to our proposed method in all the long-horizon navigation tasks and it fails in the Pick and Place task as the subgoal representation is not well learned in the robotic manipulation task. SAC+BC performs poorly in all tasks except for the Point Maze task as the action space and observation space are relatively low-dimensional so the entropy term in the actor update is sufficient for addressing the exploration problem. GAIL failed in all tasks because the policy cannot generate the training data that matches the demonstration.

C. *Ablation analysis*

To understand the role and importance of each component in our method, we conducted an ablation analysis on the curiosity-impact driven intrinsic reward module and the example-guided exploration separately. We compare IRDEC, IRDEC without the intrinsic reward module, and IRDEC without the classifier. As shown in Fig. 5, our method without the intrinsic reward module failed in all the tasks while our method without the classifier can solve the Point Maze and the Ant Maze tasks but failed in the hard Ant FourRooms and Pick and Place tasks. To understand the underlying causes, we visualize the explored areas for the tasks of the Ant Maze and the Ant FourRooms as shown in Fig. 6. We sampled 1K points from the online buffer to represent the explored areas after specific training steps. IRDEC without the intrinsic reward module failed both tasks, as the exploration attempts to pass through the wall directly towards the familiar states. And in robotic manipulation tasks, Pick and Place, the ”wall” could be the unachievable robot configurations due to the singularity. For the Ant Maze task, IRDEC and IRDEC without the classifier can expand their explored areas towards the goal while IRDEC is slightly more efficient as the exploration direction is biased towards familiar states. For the Ant FourRooms task, only IRDEC succeed in reaching the goal. IRDEC without the classifier lacks guidance towards the goal and therefore the exploration directions are random. When the state spaces are large, the random exploration direactions could lead the failure in passing the critical points, such as the narrow doors connecting each room in Ant FourRooms. Moreover, as the training progresses, the curiosity intrinsic reward part in the intrinsic reward module vanishes, which makes the exploration harder. The ablation analysis demonstrates that IRDEC can effectively leverage the familiar states in the task-specific behaviour demonstration to bias the exploration direction introduced by the intrinsic reward module for learning the required prior behaviours and connecting them to the task-specific behaviour to solve the tasks.

VI. Conclusion

In this paper, we present IRDEC, a method that incorporates an intrinsic reward module to proactively expand explored areas while biasing the exploration towards familiar states in the task-specific behaviour demonstration, for endowing the agent with the capabilities to adapt to initial conditions that are unseen from the demonstration. Our proposed method shows its capability of automatically learning the essential prior behaviours and connecting them to the behaviours in the task-specific behaviour demonstration for solving long-horizon tasks with sparse rewards. With our method, agents can adapt to tasks with varied initial conditions from the task-specific behaviour demonstration without requiring additional demonstration of the required prior behaviours. Additionally, the empirical results show the efficiency and viability of IRDEC compared to all the baselines. An exciting direction for future work would be applying the method to high-dimensional pixel-based control.

VII. Acknowledgement

The work of Ruiqi Zhu has been supported by the King's China Scholarship Council (K-CSC) PhD Scholarship programme.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[2] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[3] A. Singh, L. Yang, K. Hartikainen, C. Finn, and S. Levine, "End-to-end robotic reinforcement learning without reward engineering," in *Robotics: Science and Systems*, 2019.

[4] R. Zhu, D. Zhang, and B. Lo, "Deep reinforcement learning based semi-autonomous control for robotic surgery," *arXiv preprint arXiv:2204.05433*, 2022.

[5] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[6] A. Singh, A. Yu, J. Yang, J. Zhang, A. Kumar, and S. Levine, "Cog: Connecting new skills to past experience with offline reinforcement learning," *arXiv preprint arXiv:2010.14500*, 2020.

[7] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning*, 2018, pp. 1861–1870.

[8] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning." in *International Conference on Learning Representations*, 2016.

[9] S. Li, J. Zhang, J. Wang, Y. Yu, and C. Zhang, "Active hierarchical exploration with stable subgoal representation learning," in *International Conference on Learning Representations*, 2022.

[10] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 627–635.

[11] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[12] Y. Li, J. Song, and S. Ermon, "Infogail: Interpretable imitation learning from visual demonstrations," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[13] K. Zakka, A. Zeng, P. Florence, J. Tompson, J. Bohg, and D. Dwibedi, "Xirl: Cross-embodiment inverse reinforcement learning," in *Conference on Robot Learning*, 2022, pp. 537–546.

[14] K. Pertsch, Y. Lee, Y. Wu, and J. J. Lim, "Demonstration-guided reinforcement learning with learned skills," in *Conference on Robot Learning*, 2021.

[15] A. Singh, H. Liu, G. Zhou, A. Yu, N. Rhinehart, and S. Levine, "Parrot: Data-driven behavioral priors for reinforcement learning," in *International Conference on Learning Representations*, 2021.

[16] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," in *Robotics: Science and Systems*, 2017.

[17] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[18] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International Conference on Machine Learning*, 2017, pp. 2778–2787.

[19] Y. Burda, H. Edwards, A. Storkey, and O. Klimov, "Exploration by random network distillation," in *International Conference on Learning Representations*, 2019.

[20] T. Nguyen, T. M. Luu, T. Vu, and C. D. Yoo, "Sample-efficient reinforcement learning representation learning with curiosity contrastive forward dynamics model," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 3471–3477.

[21] A. P. Badia, P. Sprechmann, A. Vitvitskyi, D. Guo, B. Piot, S. Kapturowski, O. Tieleman, M. Arjovsky, A. Pritzel, A. Bolt, and C. Blundell, "Never give up: Learning directed exploration strategies," in *International Conference on Learning Representations*, 2020.

[22] R. Raileanu and T. Rocktäschel, "Ride: Rewarding impact-driven exploration for procedurally-generated environments," in *International Conference on Learning Representations*, 2020.

[23] G. Ostrovski, M. G. Bellemare, A. Oord, and R. Munos, "Count-based exploration with neural density models," in *International Conference on Machine Learning*, 2017, pp. 2721–2730.

[24] M. Seurin, F. Strub, P. Preux, and O. Pietquin, "Don't do what doesn't matter: Intrinsic motivation with action usefulness," in *Internationnal Joint Conference on Artificial Intelligence*, 2021.

[25] B. Eysenbach, S. Levine, and R. R. Salakhutdinov, "Replacing rewards with examples: Example-based policy search via recursive classification," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[26] B. Eysenbach, R. Salakhutdinov, and S. Levine, "C-learning: Learning to achieve goals via recursive classification," in *International Conference on Learning Representations*, 2021.

[27] T. Degris, M. White, and R. S. Sutton, "Off-policy actor-critic," in *International Conference on Machine Learning*, 2012.

[28] T. Zhang, H. Xu, X. Wang, Y. Wu, K. Keutzer, J. E. Gonzalez, and Y. Tian, "Bebold: Exploration beyond the boundary of explored regions," *arXiv preprint arXiv:2012.08621*, 2020.

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

[30] S. Pitis, H. Chan, S. Zhao, B. Stadie, and J. Ba, "Maximum entropy gain exploration for long horizon multi-goal reinforcement learning," in *International Conference on Machine Learning*, 2020, pp. 7750–7761.