
Interpretable Reward Redistribution in Reinforcement Learning: A Causal Approach

Yudi Zhang

Eindhoven University of Technology
y.zhang5@tue.nl

Yali Du

King's College London
yali.du@kcl.ac.uk

Biwei Huang

University of California San Diego
bih007@ucsd.edu

Ziyan Wang

King's College London
ziyan.wang@kcl.ac.uk

Jun Wang

University College London
jun.wang@cs.ucl.ac.uk

Meng Fang

University of Liverpool;
Eindhoven University of Technology
Meng.Fang@liverpool.ac.uk

Mykola Pechenizkiy

Eindhoven University of Technology
m.pechenizkiy@tue.nl

Abstract

A major challenge in reinforcement learning is to determine which state-action pairs are responsible for future rewards that are delayed. Reward redistribution serves as a solution to re-assign credits for each time step from observed sequences. While the majority of current approaches construct the reward redistribution in an uninterpretable manner, we propose to explicitly model the contributions of state and action from a causal perspective, resulting in an interpretable reward redistribution and preserving policy invariance. In this paper, we start by studying the role of causal generative models in reward redistribution by characterizing the generation of Markovian rewards and trajectory-wise long-term return and further propose a framework, called *Generative Return Decomposition* (GRD), for policy optimization in delayed reward scenarios. Specifically, GRD first identifies the unobservable Markovian rewards and causal relations in the generative process. Then, GRD makes use of the identified causal generative model to form a compact representation to train policy over the most favorable subspace of the state space of the agent. Theoretically, we show that the unobservable Markovian reward function is identifiable, as well as the underlying causal structure and causal models. Experimental results show that our method outperforms state-of-the-art methods and the provided visualization further demonstrates the interpretability of our method. The source code will be released at https://github.com/ReedZyd/GRD_NeurIPS2023.

1 Introduction

Reinforcement Learning (RL) has achieved significant success in a variety of applications, such as autonomous driving [Kiran et al., 2021, Chen et al., 2019], robot [Christiano et al., 2017, Zhang et al., 2019], games [Silver et al., 2018, Han et al., 2019, Du et al., 2019], financial trading [Zhang et al., 2020b, Yang et al., 2020], and healthcare [Yu et al., 2021]. A challenge in real-world RL is the delay of reward feedback [Ke et al., 2018, Fang et al., 2018, 2019b, Han et al., 2022]. In such a delayed reward setting, the learning process may suffer from instability due to the lack of proper guidance from the reward sequences [Arjona-Medina et al., 2019, Patil et al., 2022, Ren et al., 2022]. Methods such as reward shaping [Hu et al., 2020, Tambwekar et al., 2019], curiosity-driven

intrinsic reward [Rajeswar et al., 2022, Pathak et al., 2017, Zheng et al., 2021, Colas et al., 2020] and hindsight relabeling [Andrychowicz et al., 2017, Eysenbach et al., 2020, Fang et al., 2019a,b] have been proposed to provide proxy rewards in RL.

How to compute the contribution of each state-action pair affected by a delayed reward and explain the reasons behind such contribution are equally important, as they can provide insights into the underlying dynamics and the decision process, guiding the development of more effective RL algorithms. Recent return decomposition methods employ the return equivalent hypothesis to produce proxy rewards for the state-action pair at each time step [Arjona-Medina et al., 2019, Patil et al., 2022, Ren et al., 2022]. These methods allow for the decomposition of the trajectory-wise return into Markovian rewards, preserving the policy invariance. However, previous methods construct reward redistribution by hand-designed rules [Arjona-Medina et al., 2019, Patil et al., 2022], or an uninterpretable model [Ren et al., 2022], where the explanation of how the state and action contribute to the proxy reward is unclear. On the other hand, recently, causal modeling has shown promise in environmental model estimation and follow-up policy learning for RL [Huang et al., 2021, Pitis et al., 2022]. Works along this direction investigate and characterize the significant causes with respect to their expected outcomes, which help improve the efficiency of exploration by making use of structural constraints [Guestrin et al., 2001, Huang et al., 2022, Feng et al., 2022], resulting in an impressive performance improvement. Therefore, causal modeling serves as a natural tool to investigate the contributions of states and actions toward the Markovian rewards.

In this paper, we propose a novel algorithm for reward redistribution with causal treatment, called Generative Return Decomposition (GRD). GRD tackles reward redistribution with causal modeling that enjoys the following advantages. First, instead of a flat representation, GRD specifies each state and action as a combination of values of several constituent variables relevant to the problem and accounts for the causal relationships among variables in the system. Such a structural factored representation, relating to Factored MDP [Hallak et al., 2015, Kearns and Koller, 1999], provides favorability to form and identify the Markovian reward function from the perspective of causality. Unlike previous approaches, GRD utilizes such a parsimonious graphical representation to discover how each dimension of state and action contributes to the Markovian reward. Moreover, within the generative process of the MDP environment, we can naturally explain and model the observed delayed return as a causal effect of the unobserved Markovian reward sequence, which provides insights into the underlying dynamics of the environment, preserves the policy invariance as well. Figure 1 shows the framework of GRD, involving the causal relationship among environmental variables. The nodes denote different variables in the MDP environment, *i.e.*, all dimensions of state $s_{.,t}$ and action $a_{.,t}$, Markovian rewards r_t for $t \in [1, T]$, as well as the long-term return R . For sparse reward settings in RL, the Markovian rewards r_t are unobservable, which are represented by nodes with blue filling. While considering the return-equivalent assumption in return decomposition, we can observe the trajectory-wise long-term return, R , which equals the discounted sum of delayed reward o_t and evaluates the performance of the agent within the whole episode. A special case of delayed rewards is in episodic RL, where $o_{1:T-1} = 0$ and $o_T \neq 0$. Theoretically, we prove that the underlying generative process, including the unknown causal structure, the Markovian reward function, and the dynamics function, are identifiable. GRD learns the Markovian reward function and dynamics function in a component-wise way to recover the underlying causal generative process, resulting in an explainable and principled reward redistribution. Furthermore, we identify a minimal sufficient representation for policy training from the learned parameters, consisting of the dimensions of the state that have an impact on the generated Markovian rewards, both directly and indirectly. Such a compact representation has a matching causal structure with the generation of the Markovian reward, aiding in the effectiveness and stability of policy learning.

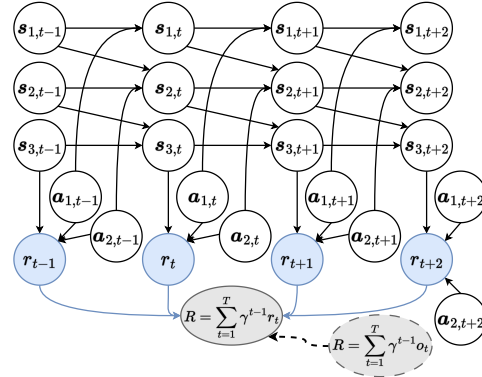


Figure 1: A graphical illustration of causal structure in our Generative Return decomposition. See the main text for its interpretation.

We summarize the main contributions of this paper as follows. First, we reformulate the reward redistribution by introducing a graphical representation to describe the causal relationship among the dimensions of state, action, the Markovian reward, and the long-term return. The causal structure over the environmental variables, the unknown Markovian reward function, and the dynamics function are identifiable, which is supported by theoretical proof. Second, we propose GRD to learn the underlying causal generative process. Based on the learned model, we construct interpretable reward redistribution and identify a compact representation to facilitate policy learning. Furthermore, empirical experiments on robot tasks from the MuJoCo environment demonstrate that our method outperforms the state-of-the-art methods and the provided visualization shows the interpretability of the redistributed rewards, indicating the usefulness of causal modeling for the sparse reward settings.

2 Related Work

Below we review the related work on reward redistribution and causality-facilitated RL methods.

Previous work redistributes rewards to deal with the delayed rewards in RL, such as reward shaping [Hu et al., 2020, Tambwekar et al., 2019] and curiosity-driven intrinsic reward [Rajeswar et al., 2022, Pathak et al., 2017, Zheng et al., 2021, Colas et al., 2020]. Return decomposition draws attention since RUDDER [Arjona-Medina et al., 2019] rethinks the return-equivalent condition for reward shaping [Ng et al., 1999] to decompose the long-term return into proxy rewards for each time step through an LSTM-based long-term return predictor and manually designed assignment. Align-RUDDER [Patil et al., 2022] introduces the bioinformatical alignment method to align successful demonstration sequences, then manually scores new sequences according to the aligned demonstrations. However, in many reinforcement learning (RL) tasks, obtaining a sufficient number of successful demonstrations can be challenging. Liu et al. [2019] and Widrich et al. [2021] explore to improve RUDDER by the replacement of expressive language models [Vaswani et al., 2017] and the continuous modern Hopfield network [Ramsauer et al., 2020] to decompose delayed rewards. Apart from them, RRD [Ren et al., 2022] proposes an upper bound for the common return-equivalent assumption to serve as a surrogate optimization objective, bridging the return decomposition and uniform reward redistribution in IRCR [Gangwani et al., 2020]. However, those methods lack the explanation of how the contributions derives [Ren et al., 2022, Liu et al., 2019], or applying unflexible manual design [Arjona-Medina et al., 2019, Patil et al., 2022] to decompose the long-term returns into dense proxy rewards for the collected sequences. By contrast, we study the role of the causal model, investigate the relationships within the generation of the Markovian reward and exploit them to guide interpretable return decomposition and efficient policy learning.

Plenty of work explores solving diverse RL problems with *causal treatment*. Most conduct research on the transfer ability of RL agents. For instance, Huang et al. [2021] learns factored representation and an individual change factor for different domains, and Feng et al. [2022] extends it to cope with non-stationary changes. More recently, Pitis et al. [2022], Wang et al. [2022] remove unnecessary dependencies between states and actions variables in the causal dynamics model to improve the generalizing capability in the unseen state. Also, causal modeling is introduced to multi-agent task [Grimbly et al., 2021, Jaques et al., 2019], model-based RL [Zhang and Bareinboim, 2016], imitation learning [Zhang et al., 2020a] and so on. However, most of them do not consider the cases of MDP with observed delayed and sparse rewards explicitly. As an exception, Mesnard et al. [2021] distinguishes the impact of the actions from one-time step and future time step to the delayed feedback by a policy gradient estimator, but still suffering from very delayed sparse rewards. Compared with the previous work, we investigate the causes for the generation of sequences of Markovian rewards to address very delayed rewards, even episodic delayed ones.

3 Preliminaries

In this section, we review the Markov Decision Process [Sutton and Barto, 2018] and return decomposition [Arjona-Medina et al., 2019, Patil et al., 2022, Ren et al., 2022].

Markov Decision Process (MDP) is represented by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \gamma, P \rangle$, where \mathcal{S} , \mathcal{A} , \mathcal{R} denote the state space, action space, and reward function, separately. The state transition probability of the environment is denoted as $P(s_{t+1} | s_t, a_t)$. The goal of reinforcement learning is to find an optimal policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ to maximize the expected long-term return, *i.e.*, a

discounted sum of the rewards with the predefined discounted factor γ and episode length T , $J(\pi) = \mathbb{E}_{s_t \sim P(s_t | s_{t-1}, a_{t-1}), a_t \sim \pi(a_t | s_t)} \left[\sum_{t=1}^T \gamma^{t-1} \mathcal{R}(s_t, a_t) \right]$.

Return decomposition is proposed to decompose the long-term return feedback into a sequence of Markovian rewards in RL with delayed rewards [Arjona-Medina et al., 2019, Patil et al., 2022] while maintaining the invariance of optimal policy [Arjona-Medina et al., 2019]. In the case of delayed reward setting, the agent can only observe some sparse delayed rewards. An extreme case is that the agent can only get an episodic non-zero reward o_T at the end of each episode, *i.e.*, $o_{1:T-1} = 0$ and $o_T \neq 0$, called episodic reward [Ren et al., 2022]. In general, the observed rewards o_t are delayed and usually harm policy learning, since the contributions of the state-action pairs are not clear. Therefore, return decomposition is proposed to generate proxy rewards \hat{r}_t for each time step, which is expected to be non-delayed, dense, Markovian, and able to clarify the contributions of the state-action pairs to the delayed feedback. The work along this line shares a common assumption,

$$R = \sum_{t=1}^T \gamma^{t-1} o_t = \sum_{t=1}^T \gamma^{t-1} \hat{r}_t, \quad (1)$$

where R is the long-term return and o_t, \hat{r}_t denote the observed delayed reward and the decomposed proxy reward for each time step, separately.

4 Causal Reformulation of Reward Redistribution

As a foundation, below we introduce a Dynamic Bayesian Network (DBN) [Murphy, 2002] to reformulate reward redistribution and characterize the underlying generative process, leading to a natural interpretation of the explicit contribution of each dimension of state and action towards the Markovian rewards.

Causal Modeling. We describe the MDP process with a trajectory-wise long-term return, $\mathcal{P} = \langle \mathcal{S}, \mathcal{A}, g, P_f, \gamma, \mathcal{G} \rangle$. \mathcal{S}, \mathcal{A} represent the sets of states s and actions a , respectively. \mathcal{G} denotes a DBN that describes the causal relationship within the MDP environment, constructed over a finite number of random variables, $\{s_{1,t}, \dots, s_{d^s,t}, a_{1,t}, \dots, a_{d^a,t}, r_t\}_{t=1}^T \cup R$ where d^s and d^a are the number of dimensions of s and a . r_t is the unobservable Markovian reward that serves as the objective of return decomposition. The state transition P_f can be represented as, $P_f(s_{i,t+1} | s_t, a_t) = \prod_{i=1}^{d^s} P(s_{i,t+1} | \text{pa}(s_{i,t+1}))$, where i is the index of dimension of s_t . Here $\text{pa}(s_{i,t+1})$ denotes the causal parents of $s_{i,t+1}$ and usually is a subset of the dimensions of s_t and a_t . We assume a given initial state distribution $P(s_1)$. Similarly, we define the functions, g , which generates unobservable Markovian reward, $r_t = g(\text{pa}(r_t))$, where $\text{pa}(r_t)$ is a subset of the dimensions of s_t and a_t . The trajectory-wise long-term return R is the causal effect of all the Markovian rewards. An example of the causal structure denoted by \mathcal{G} is given in Figure 1. For simplicity, we reformalize the generative process as,

$$\begin{cases} s_{i,t+1} = f(C_{\cdot,i}^{s \rightarrow s} \odot s_t, C_{\cdot,i}^{a \rightarrow s} \odot a_t, \epsilon_{s,i,t}) \\ r_t = g(C^{s \rightarrow r} \odot s_t, C^{a \rightarrow r} \odot a_t, \epsilon_{r,t}) \\ R = \sum_{t=1}^T \gamma^{t-1} r_t \end{cases} \quad (2)$$

for $i = (1, 2, \dots, d^s)$. \odot is the element-wise product. f and g stand for the dynamics function and reward function, separately. $C^{\cdot \rightarrow \cdot}$ are categorical masks organizing the causal structure in \mathcal{G} . $\epsilon_{s,i,t}$ and $\epsilon_{r,t}$ are i.i.d random noises. Specially, $C^{s \rightarrow r} \in \{0, 1\}^{d^s}$ and $C^{a \rightarrow r} \in \{0, 1\}^{d^a}$ control if a specific dimension of state s_t and action a_t impact the Markovian reward r_t , separately. For example, if there is an edge from $s_{i,t}$ to r_t , then the $C_i^{s \rightarrow r} = 1$. Similarly, $C^{s \rightarrow s} \in \{0, 1\}^{d^s \times d^s}$ and $C^{a \rightarrow s} \in \{0, 1\}^{d^a \times d^s}$ indicate the causal relationship between s_t, a_t and s_{t+1} , separately. In particular, we assume that \mathcal{G} is time-invariant, *i.e.*, $f, g, C^{\cdot \rightarrow \cdot}$ are invariant. In the system, s_t, a_t , and R are observable, while r_t are unobservable and there are no unobserved confounders and instantaneous causal effects.

Identifiability result. Below we give the identifiability result of learning the latent causal structure and unknown functions in the above causal generative model.

Proposition 1 (Identifiability) *Suppose the state s_t , action a_t , trajectory-wise long-term return R are observable while Markovian rewards r_t are unobservable, and they form an MDP, as described in Eq. 2. Then, under the global Markov condition and faithfulness assumption, the reward function g and the Markovian rewards r_t are identifiable, as well as the causal structure that is characterized by binary masks $C^{\cdot \rightarrow \cdot}$ and the transition dynamics f .*

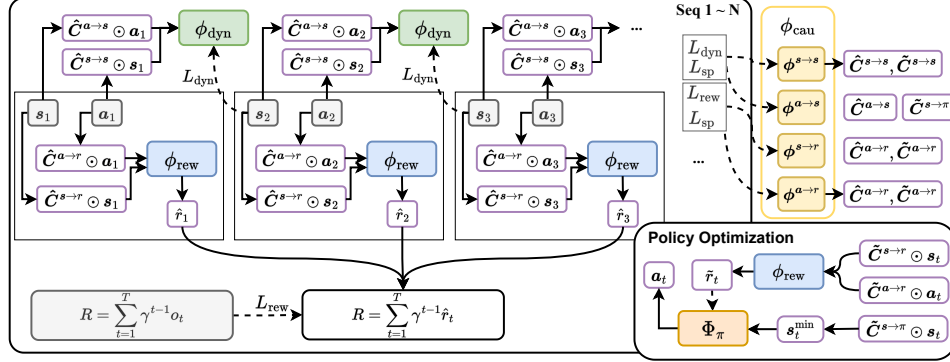


Figure 2: The neural network architecture of the proposed Generative Return Decomposition (GRD). ϕ_{cau} , ϕ_{rew} , ϕ_{dyn} in generative model Φ_m are marked as yellow, blue and green, while Policy Model Φ_π are marked as orange. The observable variables, state s_t , action a_t , and the delayed reward o_t , are marked as gray. The mediate results, binary masks, $C^{\cdot \rightarrow \cdot}$, outputs of policy, and the predicted Markovian rewards r_t are denoted as purple squares. GRD produces a compact representation based on $\tilde{C}^{s \rightarrow \pi}$, which is derived from $\tilde{C}^{s \rightarrow s}$ and $\tilde{C}^{s \rightarrow r}$. The Policy Model takes as input $s_t^{\text{min}} = \tilde{C}^{s \rightarrow \pi} \odot s_t$ and its training is supervised by the predicted Markovian reward \tilde{r}_t . The dotted lines represent the supervision signals, *i.e.*, losses, and predicted Markovian rewards.

Details of the proof for Proposition 1 are deferred to Appendix D. Proposition 1 provides the foundation for us to identify the causal structure $C^{\cdot \rightarrow \cdot}$ and the unknown functions f , g in the generative process from the observed data. As a result, with the identified structures and functions in Eq. 2, we can naturally construct an interpretable return decomposition. Here we clarify the difference between our work and the previous work for a better understanding: 1) compared with Ren et al. [2022], Liu et al. [2019], g in Eq. 2 is a general description to characterize the Markovian reward as the causal effect of a subset of the dimensions of state and action, 2) compared with Arjona-Medina et al. [2019], Patil et al. [2022], treating long-term reward R as the causal effect of all the Markovian rewards in a sequence is an interpretation of return-equivalent assumption from the causal view, allowing a reasonable and flexible return decomposition.

5 Generative Return decomposition

In this section, we propose a principled framework for reward redistribution from the causal perspective, named Generative Return Decomposition (GRD). Specifically, we first introduce how GRD recovers a generative model within the MDP environment in Sec. 5.1, and then show how GRD deploys the learned parameters of the generative model to facilitate efficient policy learning in Sec. 5.2. The GRD consists of two parts, Φ_m for the parameterized generative process and Φ_π for the policy. Therefore, the corresponding loss function includes $L_m(\Phi_m)$ and $J_\pi(\Phi_\pi)$, for causal generative model estimation and policy optimization, respectively. Hence, the overall objective can be formulated as

$$\min_{\Phi_m, \Phi_\pi} L(\Phi_m, \Phi_\pi) = L_m(\Phi_m) + J_\pi(\Phi_\pi). \quad (3)$$

In the following subsections, we will present each component of the objective function.

5.1 Generative Model Estimation

In this subsection, we present how our proposed GRD recovers the underlying generative process. This includes the identification of binary masks ($C^{\cdot \rightarrow \cdot}$), as well as the estimation of unknown functions (f and g). An overview of our method is illustrated in Figure 2. The parameterized model Φ_m that incorporates the structures and functions in Eq. 2 is used to approximate the causal generative process. The optimization is carried out by minimizing L_m over the replay buffer \mathcal{D} , which consists of trajectories $\{\tau\}$ with each trajectory $\tau = \{(s_t, a_t) |_{t=1}^T, R\}$, where R is the discounted sum of observed delayed rewards o_t , as given in Eq. 1.

Generative Model. The parameterized model Φ_m consists of three parts, ϕ_{cau} , ϕ_{dyn} , ϕ_{rew} , which are described below. (1) ϕ_{cau} is used to identify the causal structure in \mathcal{G} by predicting the values of binary masks in Eq. 2. It consists of four parts, $\phi_{\text{cau}}^{s \rightarrow s} \in \mathbb{R}^{d_s \times d_s \times 2}$, $\phi_{\text{cau}}^{a \rightarrow s} \in \mathbb{R}^{d_a \times d_s \times 2}$, $\phi_{\text{cau}}^{s \rightarrow r} \in \mathbb{R}^{d_s \times 2}$ and

$\phi_{cau}^{a \rightarrow r} \in \mathbb{R}^{d_a \times 2}$. We take the prediction of $C^{s \rightarrow s}$ with $\phi_{cau}^{s \rightarrow s}$ as an example to explain the learned causal relationships. $\phi_{cau}^{s \rightarrow s}$ characterizes $d^s \times d^s$ i.i.d Bernoulli distributions for the existence of the edges organized in the matrix, $C^{s \rightarrow s}$. We denote the probability of the existence of an edge from i -th dimension of the state to j -th dimension of the next state in the causal graph as $P(C_{i,j}^{s \rightarrow s})$. Binary prediction of $C^{s \rightarrow s}$ is sampled by applying element-wise gumbel-softmax [Jang et al., 2016] during training and denoted as $\hat{C}^{s \rightarrow s}$. Similarly, we can obtain $P(C_{\cdot,i}^{a \rightarrow s})$, $P(C_{\cdot,i}^{s \rightarrow r})$, $P(C_{\cdot,i}^{a \rightarrow r})$, $\hat{C}_{\cdot,i}^{a \rightarrow s}$, $\hat{C}_{\cdot,i}^{s \rightarrow r}$ and $\hat{C}_{\cdot,i}^{a \rightarrow r}$. (2) ϕ_{rew} is constructed with fully-connected layers and approximates the reward function g in Eq. 2, which takes as input the state s_t , action a_t , as well as the predictions of $C^{s \rightarrow r}$ and $C^{a \rightarrow r}$ to obtain the prediction of Markovian rewards. (3) ϕ_{dyn} is constructed on a Mixture Density Network [Reynolds et al., 2009], and is used to approximate the dynamics function f in Eq. 2, which takes as inputs the state s_t , action a_t , the predictions of causal structures, $\hat{C}^{s \rightarrow s}$ and $\hat{C}^{a \rightarrow s}$. As an example, the predicted distribution for the i -th dimension of the next state is $P(s_{i,t+1} | s_t, a_t, \hat{C}_{\cdot,i}^{s \rightarrow s}, \hat{C}_{\cdot,i}^{a \rightarrow s}; \phi_{dyn})$. More details for Φ_m can be found in the Appendix E.2.

Loss Terms. Accordingly, the loss term L_m contains three components with $L_m(\Phi_m) = L_{sp} + L_{dyn} + L_{rew}$. Considering the long-term return R as the causal effect of the Markovian rewards r_t , we optimize ϕ_{rew} , $\phi_{cau}^{s \rightarrow r}$ and $\phi_{cau}^{a \rightarrow r}$ by

$$L_{rew}(\phi_{rew}, \phi_{cau}^{s \rightarrow r}, \phi_{cau}^{a \rightarrow r}) = \mathbb{E}_{\tau \sim \mathcal{D}} \left[\|R - \sum_{t=1}^T \gamma^{t-1} \hat{r}_t\|^2 \right] = \mathbb{E}_{\tau \sim \mathcal{D}} \left[\left\| \sum_{t=1}^T \gamma^{t-1} o_t - \sum_{t=1}^T \gamma^{t-1} \hat{r}_t \right\|^2 \right], \quad (4)$$

where \hat{r}_t is predicted by ϕ_{rew} , i.e., $\hat{r}_t = \phi_{rew}(s_t, a_t, \hat{C}^{s \rightarrow r}, \hat{C}^{a \rightarrow r})$. To optimize ϕ_{dyn} , $\phi_{cau}^{s \rightarrow s}$ and $\phi_{cau}^{a \rightarrow s}$, we minimize

$$L_{dyn}(\phi_{dyn}, \phi_{cau}^{s \rightarrow s}, \phi_{cau}^{a \rightarrow s}) = \mathbb{E}_{s_t, a_t, s_{t+1} \sim \mathcal{D}} \left[-\sum_{i=1}^{d^s} \log P(s_{i,t+1} | s_t, a_t, \hat{C}_{\cdot,i}^{s \rightarrow s}, \hat{C}_{\cdot,i}^{a \rightarrow s}; \phi_{dyn}) \right]. \quad (5)$$

Additionally, we minimize the following cross-entropy terms to regulate the sparsity of learned causal structure to avoid trial solutions. It is achieved by force to optimize the parameters towards the direction of the nonexistence of the causal edge. Let $D_i(x) = \log P(x_i)$, where $P(x_i)$ is the possibility that the edge x_i exists. The regularizer term is,

$$L_{sp}(\phi_{cau}) = \underbrace{\lambda_1 \sum_i D_i(C^{s \rightarrow r})}_{\text{state-to-reward}} + \underbrace{\lambda_2 \sum_i D_i(C^{a \rightarrow r})}_{\text{action-to-reward}} + \underbrace{\lambda_3 \sum_{j \neq i} D_{i,j}(C^{s \rightarrow s})}_{\text{state-to-state (excluding self-connections)}} + \underbrace{\lambda_4 \sum_{j=i} D_{i,j}(C^{s \rightarrow s})}_{\text{state-to-state (self-connections)}} + \underbrace{\lambda_5 \sum_{j,i} D_{i,j}(C^{a \rightarrow s})}_{\text{action-to-state}}. \quad (6)$$

where $\mathbb{1}(\cdot)$ is the indicator function and hyper-parameters $\lambda_{(\cdot)}$ are listed in appendix E.5.

5.2 Policy Optimisation with Generative Models

Here we explain how the learned generative model aids policy optimization with delayed rewards.

Compact Representation as Inputs. Inspired by Huang et al. [2022], we identify a minimal and sufficient state set, s^{\min} , for policy learning as the input of the policy model, Φ_π , called *compact representation*. It contains the dimensions of states that have a direct or indirect impact on the reward. To be more specific, it includes each dimension of state $s_{i,t} \in s_t$, which either 1) has an edge to the reward r_t i.e., $C_{i,j}^{s \rightarrow r} = 1$, or 2) has an edge to another state component in the next time-step, $s_{j,t+1} \in s_t$, i.e., $C_{i,j}^{s \rightarrow s} = 1$, such that the same component at time t is in the compact representation, i.e., $s_{j,t} \in s_t^{\min}$. We first sample $\tilde{C}^{s \rightarrow s}$ and $\tilde{C}^{s \rightarrow r}$ greedily from the i.i.d Bernoulli distributions characterized by $\phi^{s \rightarrow s}$ and $\phi^{s \rightarrow r}$ (as illustrated in Appendix E.2) and organize the state dimension which exists in s_t^{\min} as $\tilde{C}^{s \rightarrow \pi}$. We define $\mathbb{C} := \{i, \forall \tilde{C}_{i,j}^{s \rightarrow r} = 1\}$ to denote the dimensions of s_t which have impact on the Markovian reward r_t directly. Then we have,

$$\tilde{C}^{s \rightarrow \pi} = \{\tilde{C}^{s \rightarrow r}\} \vee \{\tilde{C}_{\cdot, C_1}^{s \rightarrow s} \vee \dots \vee \tilde{C}_{\cdot, C_d^C}^{s \rightarrow s}\}, \quad (7)$$

where d^C denotes the size of \mathbb{C} and \vee denotes element-wise logical or operation. Then, the compact representation which serves as the input of policy is defined as, $s_t^{\min} = \tilde{C}^{s \rightarrow \pi} \odot s_t$. In this way, GRD takes into consideration the dimensions that contribute to the current reward provider (the learned Markovian reward function) to choose action, thus leading to efficient policy learning.

Markovian Rewards as Guidance. We adopt soft actor-critic (SAC) [Haarnoja et al., 2018] for policy optimization, with the predicted Markovian rewards. Specifically, given s_t and a_t , we replace the observed delayed reward o_t by $\tilde{r}_t = \phi_{rew}(s_t, a_t, \tilde{C}^{s \rightarrow r}, \tilde{C}^{a \rightarrow r})$, where $\tilde{C}^{s \rightarrow r}$ and $\tilde{C}^{a \rightarrow r}$ are sampled greedily from $\phi_{cau}^{s \rightarrow r}$ and $\phi_{cau}^{a \rightarrow r}$, as illustrated in Appendix E.2. The policy model Φ_π contains two parts, a critic ϕ_v and an actor ϕ_π due to the applied SAC algorithm. SAC alternates between a policy evaluation step, which trains a critic $\phi_v(s_t^{\min}, a_t)$ to estimate $Q^\pi(s_t^{\min}, a_t) = \mathbb{E}_\pi[\sum_{t=1}^T \gamma^{t-1} \tilde{r}(s_t^{\min}, a_t) \mid s_1 = s, a_1 = a]$ using the Bellman backup operator, and a policy improvement step, which trains an actor $\phi_\pi(s_t^{\min})$ by minimizing the expected KL-divergence,

$$J_\pi(\Phi_\pi) = \mathbb{E}_{s_t \sim \mathcal{D}} [D_{KL}(\phi_\pi \parallel \exp(Q^\pi - V^\pi))], \quad (8)$$

where V^π is the value function of state-action pairs [Haarnoja et al., 2018]. More detailed implementation of Φ_π can be founded in Appendix E.3.

We optimize the generative model and the policy model simultaneously, cutting the need for collecting additional diverse data for causal discovery. It is worth noticing that SAC can be replaced by any policy learning backbones [Mnih et al., 2016, Schulman et al., 2017, Janner et al., 2019, Clavera et al., 2018]. Besides, for those model-based ones [Janner et al., 2019, Clavera et al., 2018], our learned generative model serves as a natural alternative to the used environment model.

6 Experiments

In this section, we begin by performing experiments on a collection of MuJoCo locomotion benchmark tasks to highlight the remarkable performance of our methods. We then conduct an ablation study to demonstrate the effectiveness of the compact representation. Finally, we investigate the interpretability of our GRD through the visualizations of learned causal structure and decomposed rewards.

6.1 Setup

Tasks. We evaluate our method on eight widely used classical robot control tasks in the MuJoCo environment [Todorov et al., 2012], including *Half-Cheetah*, *Ant*, *Walker2d*, *Humanoid*, *Swimmer*, *Hopper*, *HumanoidStandup*, and *Reacher* tasks. All robots are limited to observing only one episode reward at the end of each episode, which is equivalent to the accumulative Markov reward that describes their performance throughout the episode. The maximum episode length is shared among all tasks and set to be 1000. In these tasks, the contributions across dimensions in the state to the Markovian reward can differ significantly due to the agents’ different objectives, even in *Humanoid* and *HumanoidStandup*, where the robots hold the same dynamic physics. A common feature is that the control costs are designed to penalize the robots if they take overly large actions. Therefore, the grounded Markovian reward functions consider different dimensions of state and all dimensions of action with varying weights.

Baselines. We compare our GRD with RRD (biased) [Ren et al., 2022], RRD (unbiased) [Ren et al., 2022], and IRCR [Gangwani et al., 2020], whose implementation details are provided in Appendix E.1. For a fair comparison, we use the same hyper-parameters for policy learning as RRD. While RUDDER and Align-RUDDER are also popular methods for return decomposition, RRD and IRCR demonstrate superior performance compared to them. Moreover, Align-RUDDER requires successful trajectories to form a scoring rule for state-action pairs, which are not available in MuJoCo. Therefore, we do not compare our performance with theirs.

Evaluation Metric. We report the average accumulative reward across 5 seeds with random initialization to demonstrate the performance of evaluated methods. Intuitively, a method that earns higher accumulative rewards in evaluation indicates better reward redistribution.

6.2 Main Results

Figure 3 provides an overview of the performance comparison among the full version of our method, GRD, an ablation version without compact representation, GRD w/o CR, and the baseline methods. The ablation version GRD w/o CR uses all state dimensions instead of the compact representation as the input of policy model. For better visualization, we scale the time step axis to highlight the different converging speeds for each task. Our method GRD outperforms the baseline methods in all tasks, as shown by higher average accumulative rewards and faster convergence, especially in the tasks with high-dimensional states. For instance, in *HalfCheetah* and *Ant*, GRD obtains the highest

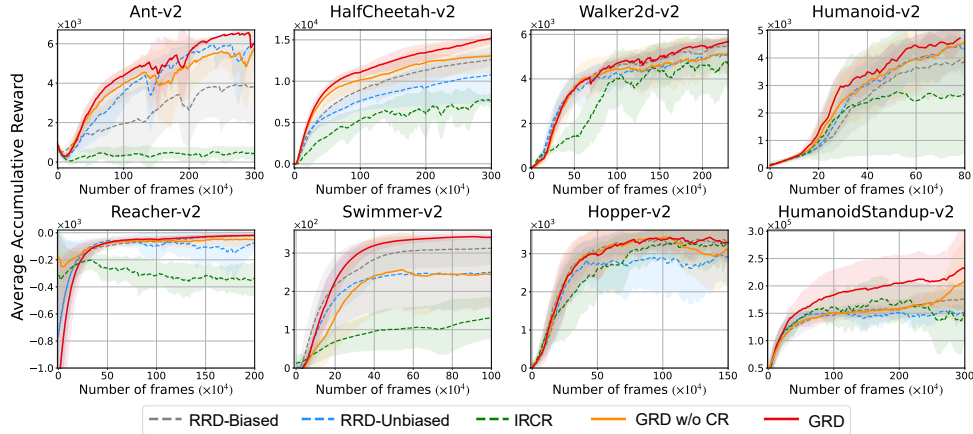


Figure 3: Learning curves on a suite of MuJoCo benchmark tasks with episodic rewards, based on 5 independent runs with random initialization. The shaded region indicates the standard deviation and the curves were smoothed by averaging the 10 most recent evaluation points using an exponential moving average. An evaluation point was established every 10^4 time steps.

Table 1: The mean and the standard variance of average accumulative reward and sparsity rate S regarding diverse λ_1 at different time step t in *Swimmer*.

λ_1 / t	$1e5$	$2e5$	$3e5$	$5e5$	$8e5$	$1e6$
0	$87 \pm 41(0.77)$	$153 \pm 43(0.80)$	$151 \pm 32(0.80)$	$131 \pm 55(0.80)$	$170 \pm 36(0.77)$	$159 \pm 25(0.78)$
$3e-6$	$182 \pm 103(0.72)$	$201 \pm 109(0.71)$	$217 \pm 108(0.72)$	$217 \pm 100(0.68)$	$229 \pm 95(0.52)$	$220 \pm 102(0.50)$
$5e-6$	$130 \pm 103(0.56)$	$245 \pm 104(0.69)$	$261 \pm 100(0.74)$	$286 \pm 77(0.77)$	$272 \pm 78(0.75)$	$262 \pm 94(0.66)$
$5e-5$	$118 \pm 100(0.46)$	$109 \pm 107(0.41)$	$123 \pm 103(0.34)$	$141 \pm 96(0.25)$	$152 \pm 93(0.19)$	$158 \pm 90(0.17)$

average accumulative reward of 1.5×10^4 and 6.5×10^3 , separately, while the baseline methods achieve the best value of 1.25×10^4 and 6×10^3 , respectively. In the task where the agents are equipped with high-dimension states, such as *HumanoidStandup* of 376-dimension states and *Ant* of 111-dimension states, GRD gains significant improvement. Taking *HumanoidStandup* as an example, GRD achieves a higher average accumulative reward of 2.3×10^5 at the end of the training and always demonstrates better performance at a certain time step during the training. The possible explanation is that GRD can quickly learn the grounded causal structure and filter out the nearly useless dimensions for approximating the real reward function. The visualization of the learned causal structure and the related discussion can be found in Sec. 6.4.

6.3 Ablation Study

In this section, we first demonstrate that the compact representation improves policy learning and then investigate the potential impact of varying causal structures on policy learning.

Compact representation for policy learning. According to Figure 3, although the GRD w/o CR has access to more information (it takes all the dimensions of the state as the input of policy model), GRD earns higher accumulative rewards on all the tasks. This is because only the dimensions tied to the predicted Markovian rewards reflect the goals of the agents. That is, the agent is supervised by the learned reward function while choosing an action based on the associated state dimensions, ensuring the input of the policy model aligns with its supervision signals. This constrains the policy model to be optimized over a smaller state space, leading to a more succinct and efficient training process.

The impact of learned causal structure. Define sparsity of causal structure of the state to reward, $S = \frac{1}{d^s} \sum_{i=1}^{d^s} \hat{C}_i^{s \rightarrow r}$. We control the value of λ_1 to obtain different sparsities of learned causal structure. The average accumulative reward and the sparsity of causal structure during the training process in *Swimmer* are presented in Table 1. With the increase of λ_1 (like from $5e-6$ to $5e-5$), the causal structure generally gets more sparse (the sparsity S decreases), leading to less policy improvement. The most possible explanation is, GRD does not take enough states to predict Markovian rewards and thus can not reflect the true goal of the agent, leading to misguided policy learning. By contrast, if we set a relatively low λ_1 to form a denser structure, GRD may take into consideration redundant dimensions that harm policy learning. Therefore, a reasonable causal structure for the reward function can improve both the convergence speed and the performance of policy training.

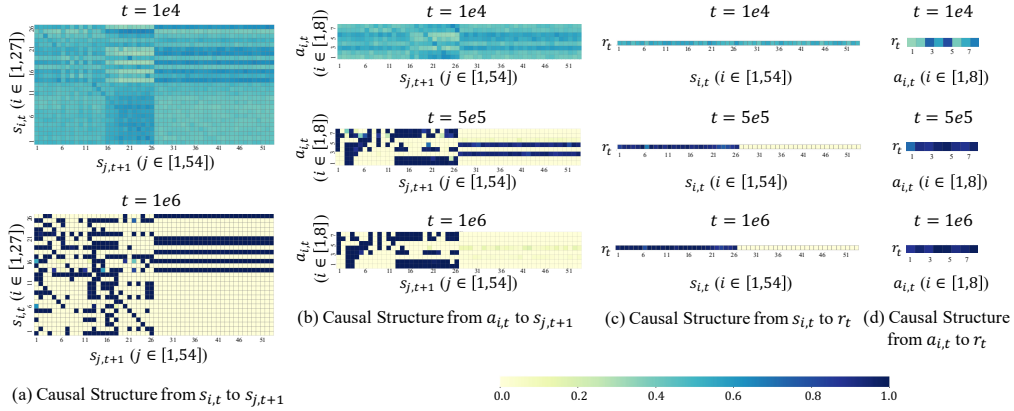


Figure 4: The visualization of learned causal structure for *Ant* when $t \in [1e4, 5e5, 1e6]$. The color indicates the probability of the existence of causal edges, whereas darker colors represent higher probabilities. There are 111 dimensions in the state variables but only the first 27 ones are used. (a) The learned causal structure among the first 27 dimensions of the state variable s_t to the first 54 dimensions of the next state variable s_{t+1} . Due to the limited space, we only visualize the structure at $t = 1e4$ and $t = 1e6$. (b) The learned causal structure among all dimensions of the action variable a_t to the first 54 dimensions of the next state variable s_{t+1} . (c) The learned causal structure among the first 54 dimensions of the state variable s_t to the Markovian reward variable r_t . (d) The learned causal structure among all dimensions of action variable a_t to the Markovian reward variable r_t .

Robustness. To evaluate the robustness of GRD, we provide results under noisy states in *Ant*. During policy evaluation, the independent Gaussian noises with the mean of 0 and standard deviation of $0 \sim 1$ are introduced into the insignificant dimensions (28 ~ 111). As shown in Figure 5, GRD is unaffected by the injected noises, while the performance of the baseline methods decreases. That is because the insignificant dimensions are not in the compact representation, which serves as the policy input.

Consistent improvement. We provide results to demonstrate the consistent improvement of our method over different RL backbones (DDPG [Silver et al., 2014] and TD3 [Fujimoto et al., 2018]) and manipulation tasks in MetaWorld [Yu et al., 2020]. Please refer to Appendix F.

6.4 Visualization

We visualize the learned causal structure and redistributed rewards in *Ant*, where the robot observes 111-dimension states and 84 dimensions are not used [Todorov et al., 2012]. The actions are 8-dimensional.

Causal Structure. As Figure 4, the existence of each element in the causal structure becomes more deterministic (indicated by the more distinguishable color) and further filters out more elements with the training. At the $1e4$ time step, GRD regards 75% elements in $C^{s \rightarrow r} \in \{0, 1\}^{376}$, 70% dimensions in $C^{a \rightarrow s} \in \{0, 1\}^{8 \times 376}$, 98% dimensions in $C^{s \rightarrow s} \in \{0, 1\}^{376 \times 376}$ to be 0 with high probabilities, indicating that the causal edge does not exist. The learned structure is nearly consistent with the fact that there should not be an edge from the unused dimensions of the state variable to the other variables. Although some redundant edges have been learned here (as shown in Figure 4 (a) the grids for the causal structure

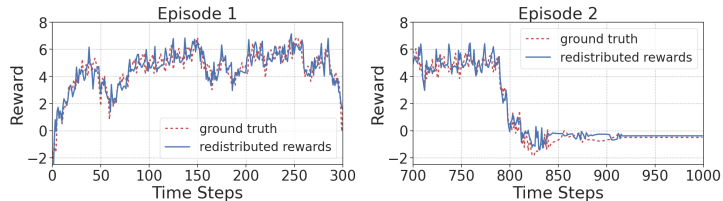


Figure 6: The visualization of decomposed rewards (blue solid lines) and the grounded rewards (red dotted lines).

Figure 5: Evaluation with Gaussian Noise in the State. The plot shows Average Accumulative Reward (y-axis, scaled by $1e3$) versus Standard Deviation of Noise (x-axis, 0.0 to 1.0). GRD (red line) maintains a high reward around 6.5, while RRD-Bias (black line), RRD-Unbias (blue line), and IRCR (green line) show a significant decrease in reward as noise increases, with IRCR dropping to near 0.

from some dimensions of state to the 28 \sim 54 dimensions of the next state are of dark color), these edges do not affect our policy learning, since there is no edge from the 28 \sim 54 dimensions of s_t to r_t , *i.e.*, these dimensions do not exist in the identified compact representation. Additionally, according to Figure 4(d), the edges from different dimensions of a to r always exist. It corresponds to the reward design in *Ant* since the robots are expected to act with the lowest control cost, defined as $r_{t,cost} = \sum_{i=1}^{d^a} a_i^2$. Therefore, the learned causal structure captures some causal characteristics of the environment, which can explain the generation of the Markovian rewards.

Decomposed Rewards. We visualize the comparison of decomposed rewards and the ground truth rewards to demonstrate the accuracy of predicting Markovian reward by GRD. According to Figure 6, the redistributed rewards (blue lines) consistently align with the ground truth (red lines), indicating that our method indeed distinguishes the state-action pairs with less contribution to the long-term return from those with more contributions. More cases are provided in Appendix F.3.

7 Conclusion

In this paper, we propose Generative Return Decomposition (GRD) to address reward redistribution for the RL tasks with delayed rewards in an interpretable and principled way. GRD reformulates reward redistribution from a causal view and recovers the generative process within the MDPs with trajectory-wise return, supported by theoretical evidence. Furthermore, GRD forms a compact representation for efficient policy learning. Experimental results show that, in GRD, not only does the explainable reward redistribution produce a qualified supervision signal for policy learning, but also the identified compact representation promotes policy optimization.

Appendix organization:

- Appendix A: Limitation and future work;
- Appendix B: Boarder impacts of GRD;
- Appendix C: Causal background: Markov and faithfulness assumptions;
- Appendix D: Proofs of identifiability result;
- Appendix E: Implementation details;
- Appendix F: Additional results.

A Limitation and Future Work

The limitations of our work arise from the made assumptions. Firstly, we assumed that the reward function is stationary, which means that our method may not be able to handle dynamic changes in the reward function. This limitation may prevent our method from being used in some non-stationary scenarios, such as online RL. Additionally, the static causal graph and causal discovery methods we have employed may not be suitable for scenarios with potential confounding factors and dynamic changes in causal structure. Although such scenarios are common in real-world problems, our method may not be able to address them effectively. Lastly, an important direction for future work is to extend our proposed method in the case of partial observable MDP and to discuss the identifiability in such scenarios.

B Broader Impacts

Our motivation is to strive to make decisions that are both understood and trusted by humans. By increasing the credibility and transparency of the decision-making process, human users develop a better understanding, validate and enhance the decisions made by algorithms. As a result, it is possible to bridge the gap between humans and AI, fostering a symbiotic relationship that leverages the strengths of both to enable more reliable and responsible decision-making in various fields. For example, in finance, transparent algorithms can enhance risk assessment, investment strategies, and fraud detection, while in transportation, trustworthy decision-making algorithms can contribute to safe and efficient navigation and logistics. To advance this mission, our work on the GRD algorithm represents a significant step forward. GRD explores facilitating policy learning in the presence of delayed rewards. By decomposing the overall return into Markovian rewards, we provide a clearer understanding of the contribution made by each state-action pair. Furthermore, we go beyond simply explaining the rewards and delve into the causal view of reward generation. This approach allows us to provide interpretable explanations of how Markovian rewards are generated, enabling a more transparent decision-making process. This interpretability is vital for building trust with human users who may need to understand and validate the decisions made by the algorithm. Moreover, with an interpretable reward function, we can readily incorporate additional restrictions, such as security constraints, into the decision-making process. This flexibility allows us to tailor the algorithm to specific needs and requirements, further enhancing its trustworthiness. Additionally, since the principles and techniques we are exploring can be applied across a wide range of domains and industries, our collaboration with GRD not only contributes to the field of reinforcement learning but also has broader implications beyond robotics.

C Causal Background: Markov and faithfulness assumptions

A directed acyclic graph (DAG), $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, can be deployed to represent a graphical criterion carrying out a set of conditions on the paths, where \mathbf{V} and \mathbf{E} denote the set of nodes and the set of directed edges, separately.

Definition 1 (d-separation [Pearl, 2000]) *A set of nodes $\mathbf{Z} \subseteq \mathbf{V}$ blocks the path p if and only if (1) p contains a chain $i \rightarrow m \rightarrow j$ or a fork $i \leftarrow m \rightarrow j$ such that the middle node m is in \mathbf{Z} , or (2) p contains a collider $i \rightarrow m \leftarrow j$ such that the middle node m is not in \mathbf{Z} and such that no descendant of m is in \mathbf{Z} . Let \mathbf{X} , \mathbf{Y} and \mathbf{Z} be disjoint sets of nodes. If and only if the set \mathbf{Z} blocks*

all paths from one node in \mathbf{X} to one node in \mathbf{Y} , \mathbf{Z} is considered to d -separate \mathbf{X} from \mathbf{Y} , denoting as $(\mathbf{X} \perp_d \mathbf{Y} \mid \mathbf{Z})$.

Definition 2 (Global Markov Condition [Spirtes et al., 2000, Pearl, 2000]) *If, for any partition $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$, \mathbf{X} is d -separated from \mathbf{Y} given \mathbf{Z} , i.e., $\mathbf{X} \perp_d \mathbf{Y} \mid \mathbf{Z}$. Then the distribution P over \mathbf{V} satisfies the global Markov condition on graph G , and can be factorizes as, $P(\mathbf{X}, \mathbf{Y} \mid \mathbf{Z}) = P(\mathbf{X} \mid \mathbf{Z})P(\mathbf{Y} \mid \mathbf{Z})$. That is, \mathbf{X} is conditionally independent of \mathbf{Y} given \mathbf{Z} , writing as $\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z}$.*

Definition 3 (Faithfulness Assumption [Spirtes et al., 2000, Pearl, 2000]) *The variables, which are not entailed by the Markov Condition, are not independent of each other.*

Under the above assumptions, we can apply d -separation as a criterion to understand the conditional independencies from a given DAG \mathcal{G} . That is, for any disjoint subset of nodes $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathbf{V}$, $(\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z})$ and $\mathbf{X} \perp_d \mathbf{Y} \mid \mathbf{Z}$ are the necessary and sufficient condition of each other.

D Proofs

Proposition 1 (Identifiability) *Suppose the state \mathbf{s}_t , action \mathbf{a}_t , trajectory-wise long-term return R are observable while Markovian rewards r_t are unobservable, and they form an MDP, as described in Eq. 2. Then under the global Markov condition and faithfulness assumption, the reward function g and the Markovian rewards r_t are identifiable, as well as the causal structure that is characterized by binary masks $\mathbf{C}^{\mathbf{s} \rightarrow \cdot}$ and $\mathbf{C}^{\cdot \rightarrow \cdot}$ and the transition dynamics f .*

Below is the proof of Proposition 1. We begin by clarifying the assumptions we made and then provide the mathematical proof.

Assumption We assume that, $\epsilon_{s,i,t}$ and $\epsilon_{r,t}$ in Eq. 2 are i.i.d additive noise. From the weight-space view of Gaussian Process [Williams and Rasmussen, 2006], equivalently, the causal models for $\mathbf{s}_{i,t+1}$ and r_t can be represented as follows, respectively,

$$\mathbf{s}_{i,t+1} = f_i(\mathbf{s}_t, \mathbf{a}_t) + \epsilon_{s,i,t} = W_{i,f}^T \phi_f(\mathbf{s}_t, \mathbf{a}_t) + \epsilon_{s,i,t}, \quad (\text{A1})$$

$$r_t = g(\mathbf{s}_t, \mathbf{a}_t) + \epsilon_{r,t} = W_g^T \phi_g(\mathbf{s}_t, \mathbf{a}_t) + \epsilon_{r,t}, \quad (\text{A2})$$

where $\forall i \in [1, d^s]$, and ϕ_f and ϕ_g denote basis function sets.

Then we denote the variable set in the system by \mathbf{V} , with $\mathbf{V} = \{\mathbf{s}_{1,t}, \dots, \mathbf{s}_{d^s,t}, \mathbf{a}_{1,t}, \dots, \mathbf{a}_{d^a,t}, r_t\}_{t=1}^T \cup R$, and the variables form a Bayesian network \mathcal{G} . Following AdaRL [Huang et al., 2021], there are possible edges only from $\mathbf{s}_{i,t-1} \in \mathbf{s}_{t-1}$ to $\mathbf{s}_{i',t} \in \mathbf{s}_t$, from $\mathbf{a}_{j,t-1} \in \mathbf{a}_{t-1}$ to $\mathbf{s}_{i',t} \in \mathbf{s}_t$, from $\mathbf{s}_{i,t} \in \mathbf{s}_t$ to r_t , and from $\mathbf{a}_{j,t} \in \mathbf{a}_t$ to r_t in \mathcal{G} . In particular, the r_t are unobserved, while $R = \sum_{t=1}^T \gamma^{t-1} o_t$ is observed. Thus, there are deterministic edges from each r_t to R .

Below we omit the γ for simplicity.

Proof 1 *Given trajectory-wise long-term return R , the binary masks, $\mathbf{C}^{\mathbf{s} \rightarrow r}$, $\mathbf{C}^{\mathbf{a} \rightarrow r}$ and Markovian reward function g and the rewards r_t are identifiable. Following the above assumption, we first rewrite the function to calculate trajectory-wise long-term return in Eq. 2 as,*

$$\begin{aligned} R &= \sum_{t=1}^T r_t \\ &= \sum_{t=1}^T [W_g^T \phi_g(\mathbf{s}_t, \mathbf{a}_t) + \epsilon_{r,t}] \\ &= W_g^T \sum_{t=1}^T \phi_g(\mathbf{s}_t, \mathbf{a}_t) + \sum_{t=1}^T \epsilon_{r,t}. \end{aligned} \quad (\text{A3})$$

For simplicity, we replace the components in Eq. A3 by,

$$\begin{aligned} \zeta_g(X) &= \sum_{t=1}^T \phi_g(\mathbf{s}_t, \mathbf{a}_t), \\ E_r &= \sum_{t=1}^T \epsilon_{r,t}, \end{aligned} \quad (\text{A4})$$

where $X := [\mathbf{s}_t, \mathbf{a}_t]_{t=1}^T$ representing the concatenation of the covariates \mathbf{s}_t and \mathbf{a}_t from $t = 1$ to T . Consequently, we derive the following equation,

$$R = W_g^T \zeta_g(X) + E_r. \quad (\text{A5})$$

Then we can obtain a closed-form solution of W_g^T in Eq. A5 by modeling the dependencies between the covariates X_τ and response variables R_τ , where both are continuous. One classical approach to finding such a solution involves minimizing the quadratic cost and incorporating a weight-decay regularizer to prevent overfitting. Specifically, we define the cost function as,

$$C(W_g) = \frac{1}{2} \sum_{X_\tau, R_\tau \sim \mathcal{D}} (R_\tau - W_g^T \zeta_g(X_\tau))^2 + \frac{1}{2} \lambda \|W_g\|^2. \quad (\text{A6})$$

where τ represents trajectories consisting of state-action pairs X_τ and long-term returns R_τ , which are sampled from the replay buffer \mathcal{D} . λ is the weight-decay regularization parameter. To find the closed-form solution, we differentiate the cost function with respect to W_g and set the derivative to zero:

$$\frac{\partial C(W_g)}{\partial W_g} = 0. \quad (\text{A7})$$

Solving this equation will yield the closed-form solution for W_g^T , i.e.,

$$W_g = (\lambda I_d + \zeta_g \zeta_g^T)^{-1} \zeta_g R = \zeta_g (\zeta_g^T \zeta_g + \lambda I_n)^{-1} R \quad (\text{A8})$$

Therefore, W_g , which indicates the causal structure and strength of the edge, can be identified from the observed data. In summary, given trajectory-wise long-term return R , the binary masks, $\mathbf{C}^{s \rightarrow r}$, $\mathbf{C}^{\alpha \rightarrow r}$ and Markovian reward function g and the rewards r_t are identifiable.

The binary masks, $\mathbf{C}^{s \rightarrow s}$, $\mathbf{C}^{\alpha \rightarrow s}$ and the transition dynamics f are identifiable In a similar manner, based on the assumption and Eq. 2, we can rewrite Eq. A1 to,

$$\mathbf{s}_{t+1} = W_{i,f}^T \phi_f(\mathbf{s}_t, \mathbf{a}_t) + \epsilon_{s,i,t}. \quad (\text{A9})$$

To obtain a closed-form solution for $W_{i,f}$, f^T in Equation A9, we can model the dependencies between the covariates X_t and the response variables \mathbf{s}_{t+1} , both of which are continuous. The closed-form solution can be represented as:

$$C(W_{i,f}) = \frac{1}{2} \sum_{\mathbf{s}_{i,t}, \mathbf{s}_{i,t+1} \sim \mathcal{D}} (\mathbf{s}_{i,t+1} - W_{i,f}^T \phi_{i,f}(\mathbf{s}_t, \mathbf{a}_t))^2 + \frac{1}{2} \lambda \|W_{i,f}\|^2. \quad (\text{A10})$$

By taking derivatives of the cost function and setting them to zero, we can obtain the closed-form solution,

$$\begin{aligned} W_{i,f} &= (\lambda I_d + \phi_{i,f} \phi_{i,f}^T)^{-1} \phi_{i,f} \mathbf{s}_{i,t+1} \\ &= \phi_{i,f} (\phi_{i,f}^T \phi_{i,f} + \lambda I_n)^{-1} \mathbf{s}_{i,t+1}. \end{aligned} \quad (\text{A11})$$

Therefore, $W_{i,f}$ can be identified from the observed data. This conclusion applies to all dimensions of the state. As a result, the f , which indicates the parent nodes of the i -dimension of the state, as well as the strength of the causal edge, are identifiable. In summary, the binary masks, $\mathbf{C}^{s \rightarrow s}$, $\mathbf{C}^{\alpha \rightarrow s}$ and the transition dynamics f are identifiable.

Considering the Markov condition and faithfulness assumption, we can conclude that for any pair of variables $V_i, V_j \in \mathbf{V}$, V_i and V_j are not adjacent in the causal graph \mathcal{G} if and only if they are conditionally independent given some subset of $\{V_l \mid l \neq i, l \neq j\}$. Additionally, since there are no instantaneous causal relationships and the direction of causality can be determined if an edge exists, the binary structural masks $\mathbf{C}^{s \rightarrow r}$, $\mathbf{C}^{\alpha \rightarrow r}$, $\mathbf{C}^{s \rightarrow s}$, and $\mathbf{C}^{\alpha \rightarrow s}$ defined over the set \mathbf{V} are identifiable with conditional independence relationships [Huang et al., 2022]. Consequently, the functions f and g in Equation 2 are also identifiable.

Algorithm 1 Learning the generative process and policy jointly.

```

1: Initialize: Environment  $\mathcal{E}$ , trajectory  $\tau \leftarrow \emptyset$ , buffer  $\mathcal{D} \leftarrow \emptyset$ 
2: Initialize: Generative Model  $\Phi_m := [\phi_{\text{cau}}, \phi_{\text{dyn}}, \phi_{\text{rew}}]$ ; Policy Model  $\Phi_\pi$ 
3: for  $i = 1, 2, \dots, 3 \times 10^4$  do
4:    $\tau \leftarrow \emptyset$ , reset  $\mathcal{E}$ 
5:   for  $n_{\text{step}} = 1, 2, \dots, 100$  do
6:     sample data  $\langle \mathbf{s}_t, \mathbf{a}_t, o_t \rangle$  from  $\mathcal{E}$ , and store them to trajectory  $\tau$ 
7:     if  $\mathcal{E}$  done then
8:       store trajectory  $\tau = \{\mathbf{s}_{1:T}, \mathbf{a}_{1:T}, R\}$  to buffer  $\mathcal{D}$ , where  $R = \sum_{i=1}^T \gamma^{t-1} o_t$ 
9:        $\tau \leftarrow \emptyset$ , reset  $\mathcal{E}$ 
10:    end if
11:    for  $n_{\text{batch}} = 1, 2, \dots, \text{train\_batches}$  do
12:      Sample  $D_1$  consisting of  $M$  trajectories from  $\mathcal{D}$ :  $D_1 = \{ \langle \mathbf{s}_t^m, \mathbf{a}_t^m \rangle |_{t=1}^T, R^m \} |_{m=1}^M$ 
13:      Sample  $D_2$  consisting of  $N$  samples from  $\mathcal{D}$ :  $D_2 = \{ \mathbf{s}_{t_n}, \mathbf{a}_{t_n}, \mathbf{s}_{t_n+1} \} |_{n=1}^N$ 
14:      Sample binary masks by Gumbel-Softmax from  $\phi_{\text{cau}}$ :  $\hat{\mathbf{C}}^{s \rightarrow r}, \hat{\mathbf{C}}^{a \rightarrow r}, \hat{\mathbf{C}}^{s \rightarrow s}$  and  $\hat{\mathbf{C}}^{a \rightarrow s}$ 
15:      Sample binary masks deterministically from  $\phi_{\text{cau}}$ :  $\tilde{\mathbf{C}}^{s \rightarrow r}, \tilde{\mathbf{C}}^{a \rightarrow r}$ , and  $\tilde{\mathbf{C}}^{s \rightarrow s}$ 
16:      Calculate  $\tilde{\mathbf{C}}^{s \rightarrow \pi}$  based on  $\tilde{\mathbf{C}}^{s \rightarrow r}$  and  $\tilde{\mathbf{C}}^{s \rightarrow s}$ 
17:      Update  $D_2$ :  $D_2 \leftarrow \{ \tilde{\mathbf{C}}^{s \rightarrow \pi} \odot \mathbf{s}_{t_n}, \mathbf{a}_{t_n}, \mathbf{s}_{t_n+1}, \phi_{\text{rew}}(\mathbf{s}_{t_n}, \mathbf{a}_{t_n}, \tilde{\mathbf{C}}^{s \rightarrow r}, \tilde{\mathbf{C}}^{a \rightarrow r}) \} |_{n=1}^N$ 
18:      Optimize  $\phi_{\text{rew}}$  with  $D_1$  (Using  $\hat{\mathbf{C}}^{s \rightarrow r}$  and  $\hat{\mathbf{C}}^{s \rightarrow s}$ ):  $\phi_{\text{rew}} \leftarrow \phi_{\text{rew}} - \alpha \nabla_{\phi_{\text{rew}}} L_{\text{rew}}$  (Eq. 4)
19:      Optimize  $\phi_{\text{dyn}}$  with  $D_2$  (Using  $\hat{\mathbf{C}}^{s \rightarrow s}$  and  $\hat{\mathbf{C}}^{s \rightarrow s}$ ):  $\phi_{\text{dyn}} \leftarrow \phi_{\text{dyn}} - \alpha \nabla_{\phi_{\text{dyn}}} L_{\text{dyn}}$  (Eq. 5)
20:      Optimize  $\phi_{\text{cau}}$ :  $\phi_{\text{cau}} \leftarrow \phi_{\text{cau}} - \alpha \nabla_{\phi_{\text{cau}}} (L_{\text{sp}} + L_{\text{rew}} + L_{\text{dyn}})$  (Eq. 6)
21:      Optimize  $\Phi_\pi$ :  $\Phi_\pi \leftarrow \Phi_\pi - \alpha \nabla_{\Phi_\pi} J_\pi$  (Eq. 8)
22:    end for
23:  end for
24: end for

```

E Implementation Details

E.1 Baselines

We compare our method against the following baselines,

- RRD (biased). This baseline utilizes a surrogate objective called randomized return decomposition loss for reducing the consumption of estimating the Markovian reward function. It applies Monte-Carlo sampling to get a biased estimation of the Mean Square Error (MSE) between the observed episodic reward and the sum of Markovian reward predictions in a sequence. We keep the same setting and hyper-parameters with its official implementation to reproduce the results, in which the policy module is optimized by soft actor-critic (SAC) algorithm [Haarnoja et al., 2018].
- RRD (unbiased). This variant of RRD (biased) provides an unbiased estimation of MSE by sampling short sub-sequences. It offers a computationally efficient approach to optimize MSE. According to Ren et al. [2022], RRD (biased) and RRD (unbiased) achieve state-of-the-art performance in episodic MuJoCo tasks.
- This baseline performs non-parametric uniform reward redistribution. At each time step, the proxy reward is set to the normalized value of the trajectory return. IRCR is a simple and efficient approach, and except for RRD, it achieves state-of-the-art performance in the literature. The implementation is from RRD [Ren et al., 2022].

E.2 Detailed Generative Model

The parametric generative model Φ_m used in the MDP environment consists of three components: ϕ_{cau} , ϕ_{rew} , and ϕ_{dyn} . We provide a detailed description of their model structures below.

ϕ_{cau} for predicting the causal structure ϕ_{cau} comprises a set of free parameters without input. We divide ϕ_{cau} into four parts, each corresponding to the binary masks in Equation 2. Specifically, we have

- $\phi_{\text{cau}}^{s \rightarrow s} \in \mathbb{R}^{d^s \times d^s \times 2}$ for $\mathbf{C}^{s \rightarrow s} \in \{0, 1\}^{d^s \times d^s}$,

Layer#	1	2	3
ϕ_{rew}	FC256	FC256	FC1
ϕ_{dyn}	FC256	FC256	FC9
ϕ_{π}	FC256	FC256	FC $2d^a$
ϕ_v	FC256	FC256	FC1

Table A1: The network structures of ϕ_{rew} , ϕ_{dyn} , ϕ_{π} and ϕ_v . $FC256$ denotes a fully-connected layer with an output size of 256. Each hidden layer is followed by an activation function, ReLU. d^a is the number of dimensions of the action in a specific task.

- $\phi_{\text{cau}}^{\mathbf{a} \rightarrow \mathbf{s}} \in \mathbb{R}^{d^a \times d^s \times 2}$ for $\mathbf{C}^{\mathbf{a} \rightarrow \mathbf{s}} \in \mathbb{R}^{d^a \times d^s}$,
- $\phi_{\text{cau}}^{\mathbf{s} \rightarrow \mathbf{r}} \in \mathbb{R}^{d^s \times 2}$ for $\mathbf{C}^{\mathbf{s} \rightarrow \mathbf{r}} \in \mathbb{R}^{d^s}$,
- $\phi_{\text{cau}}^{\mathbf{a} \rightarrow \mathbf{r}} \in \mathbb{R}^{d^a \times 2}$ for $\mathbf{C}^{\mathbf{a} \rightarrow \mathbf{r}} \in \mathbb{R}^{d^a}$.

Below we explain the shared workflows in ϕ_{cau} using the example of predicting the causal edge from the i -th dimension of state $\mathbf{s}_{i,t}$ to the j -th dimension of the next state $\mathbf{s}_{j,t+1}$, by part of the free parameters, $\phi_{\text{cau},i,j}^{\mathbf{s} \rightarrow \mathbf{s}}$.

For simplicity, we denote $\phi_{\text{cau},i,j}^{\mathbf{s} \rightarrow \mathbf{s}}$ as ψ . The shape of ψ is now easy to be determined. That is $\psi \in \mathbb{R}^2$ and we write it as $\psi = [\psi_0, \psi_1]$. With this 2-element vector, we can characterize a Bernoulli distribution, where each element corresponds to the unnormalized probability of classifying the edge as existing (ψ_0) or not existing (ψ_1), respectively. Therefore, the probability of the causal edge existing from the i -th dimension of state $\mathbf{s}_{i,t}$ to the j -th dimension of the next state $\mathbf{s}_{j,t+1}$ can be calculated as:

$$P(\mathbf{C}_{i,j}^{\mathbf{s} \rightarrow \mathbf{s}}) = \frac{\exp(\psi_0)}{\exp(\psi_0) + \exp(\psi_1)}. \quad (\text{A12})$$

One example of using the estimated mask is given as Figure A1. **Obtain $\hat{\mathbf{C}}_{i,j}^{\mathbf{s} \rightarrow \mathbf{s}}$ through Gumbel-Softmax sampling in the training phases.** During training, it is crucial to maintain the gradient flow for backpropagation. To achieve this, we sample the binary values of $\hat{\mathbf{C}}_{i,j}^{\mathbf{s} \rightarrow \mathbf{s}}$ by applying Gumbel-Softmax [Jang et al., 2016],

$$\hat{\mathbf{C}}_{i,j}^{\mathbf{s} \rightarrow \mathbf{s}} = \text{GS}(\psi) \quad (\text{A13})$$

where GS denotes the Gumbel-Softmax sampling, which allows us to obtain binary discrete samples from the Bernoulli distribution. Applying Gumbel Softmax sampling allows us to randomly sample from the Bernoulli distribution in a stochastic manner, rather than simply selecting the class with the highest probability. This introduces some randomness, enabling the model to explore the balance and uncertainty between different classifications more flexibly.

Obtain $\tilde{\mathbf{C}}_{i,j}^{\mathbf{s} \rightarrow \mathbf{s}}$ by deterministic sampling in the inference phases. During inference, including data sampling and policy learning, we get the prediction of $\mathbf{C}_{i,j}^{\mathbf{s} \rightarrow \mathbf{s}}$ through a deterministic sampling,

$$\tilde{\mathbf{C}}_{i,j}^{\mathbf{s} \rightarrow \mathbf{s}} = \begin{cases} 1, \psi_0 \geq \psi_1 \\ 0, \psi_0 < \psi_1. \end{cases} \quad (\text{A14})$$

This is a greedy sampling to avoid introducing randomness during the Gumbel-Softmax sampling.

The above explanation of the workflow in ϕ_{cau} for predicting a single causal edge provides insight into the overall implementation of the entire module ϕ_{cau} and can be applicable for all the causal edges. Therefore, we can obtain $\hat{\mathbf{C}}^{\mathbf{a} \rightarrow \mathbf{s}}$, $\hat{\mathbf{C}}^{\mathbf{s} \rightarrow \mathbf{r}}$, $\hat{\mathbf{C}}^{\mathbf{a} \rightarrow \mathbf{r}}$, $\tilde{\mathbf{C}}^{\mathbf{s} \rightarrow \mathbf{r}}$ and $\tilde{\mathbf{C}}^{\mathbf{a} \rightarrow \mathbf{r}}$, using similar procedures.

ϕ_{rew} for predicting the Markovian rewards ϕ_{rew} is a stacked fully-connected network, and the details of the network structure are provided in Table A1. During training, the prediction of Markovian reward can be written as,

$$\hat{r} = \phi_{\text{rew}}(\mathbf{s}_t, \mathbf{a}_t, \hat{\mathbf{C}}^{\mathbf{s} \rightarrow \mathbf{r}}, \hat{\mathbf{C}}^{\mathbf{a} \rightarrow \mathbf{r}}) = \text{FCs}([\hat{\mathbf{C}}^{\mathbf{s} \rightarrow \mathbf{r}} \odot \mathbf{s}_t, \hat{\mathbf{C}}^{\mathbf{a} \rightarrow \mathbf{r}} \odot \mathbf{a}_t]), \quad (\text{A15})$$

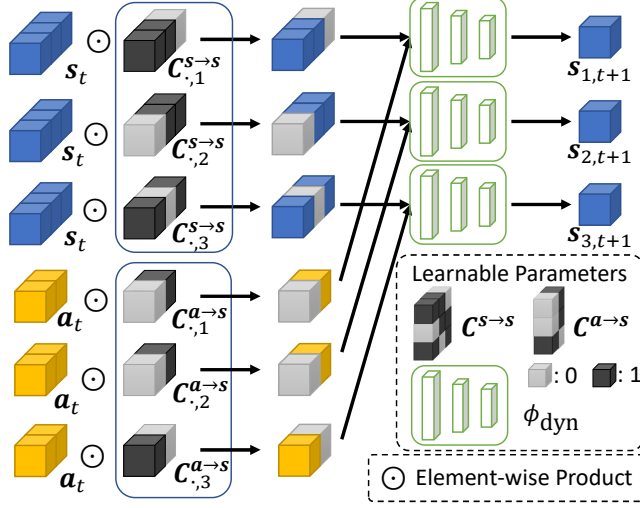


Figure A1: The illustration of using learnable masks to predict the next state.

where $[\cdot, \cdot]$, \odot denotes concatenation and element-wise multiply operations, respectively. FCs denotes the stacked fully-connected network. $\hat{C}^{s \rightarrow r}$ and $\hat{C}^{a \rightarrow r}$ are derived from ϕ_{cau} by Gumbel-Softmax.

During inference, including policy learning and data sampling, the predicted Markovian reward is

$$\tilde{r} = \phi_{\text{rew}}(s_t, \mathbf{a}_t, \tilde{C}^{s \rightarrow r}, \tilde{C}^{a \rightarrow r}) = \text{FCs}([\tilde{C}^{s \rightarrow r} \odot s_t, \tilde{C}^{a \rightarrow r} \odot \mathbf{a}_t]), \quad (\text{A16})$$

where $\hat{C}^{s \rightarrow r}$ and $\hat{C}^{a \rightarrow r}$ are derived from ϕ_{cau} greedily by deterministic sampling.

ϕ_{dyn} for modeling the environment dynamics In our experiment, we do not directly utilize ϕ_{dyn} in policy learning. Instead, this module serves as a bridge to optimize $\phi_{\text{cau}}^{s \rightarrow s}$ and $\phi_{\text{cau}}^{a \rightarrow s}$. Subsequently, $\phi_{\text{cau}}^{s \rightarrow s}$ can be utilized in the calculation of $\tilde{C}^{s \rightarrow \pi}$.

During training, we initially sample $\hat{C}^{s \rightarrow s}$ and $\hat{C}^{a \rightarrow s}$ using Gumbel-Softmax. The prediction for the i -th dimension of the next state can be represented as follows,

$$\hat{s}_{i,t} = \text{MDN}([\hat{C}^{s \rightarrow s}_{\cdot,i} \odot s_t, \hat{C}^{a \rightarrow s}_{\cdot,i} \odot \mathbf{a}_t]), \quad (\text{A17})$$

where $[\cdot, \cdot]$ denotes concatenation and MDN denotes the Mixture Density Network which outputs the means, variances, and probabilities for N_{Gau} Gaussian cores. The parameters of MDN are shared across the predictions of different dimensions of the next state. We set $N_{\text{cau}} = 3$ in our experiments. More details about ϕ_{dyn} can be found in Table A1.

E.3 Detailed Policy Model

Considering the specific requirements of the employed RL algorithm, Soft Actor-Critic (SAC), our Policy Model Φ_{π} comprises two components, the actor ϕ_{π} and the critic ϕ_v . Detailed network structures for both components can be found in Table A1.

E.4 Training Process

We follow the line of joint learning in Ren et al. [2022], which avoids learning a return decomposition model in advance using data sampled by optimal or sub-optimal policies [Patil et al., 2022]. During each mini-batch training iteration, we sample two sets of data separately from the replay buffer \mathcal{D} :

- $D_1 = \{ \langle s_t^m, \mathbf{a}_t^m \rangle \}_{t=1}^T, R^m \}_{m=1}^M$ consists of M trajectories. Provided with the trajectory-wise long-term returns R^m , D_1 is utilized to optimize $\phi_{\text{cau}}^{s \rightarrow r}$, $\phi_{\text{cau}}^{a \rightarrow r}$ and ϕ_{rew} , with L_{rew} .

Table A2: The table of the hyper-parameters used in the experiments for GRD.

Envs	λ_1	λ_2	λ_3	λ_4	λ_5
<i>Ant</i>	10^{-5}	0	10^{-7}	10^{-8}	10^{-8}
<i>HalfCheetah</i>	10^{-5}	10^{-5}	10^{-5}	10^{-6}	10^{-5}
<i>Walker2d</i>	10^{-5}	10^{-5}	10^{-6}	10^{-6}	10^{-7}
<i>Humanoid</i>	10^{-5}	10^{-8}	10^{-5}	10^{-7}	10^{-8}
<i>Reacher</i>	5×10^{-7}	10^{-8}	10^{-8}	10^{-8}	10^{-8}
<i>Swimmer</i>	10^{-7}	10^{-9}	10^{-9}	0	10^{-9}
<i>Hopper</i>	10^{-6}	10^{-6}	10^{-6}	10^{-7}	10^{-6}
<i>HumanStandup</i>	10^{-5}	10^{-4}	10^{-6}	10^{-7}	10^{-7}

Table A3: The hyper-parameters.

hyperparameters	value	hyperparameters	value
epochs	3	optimizer	Adam
cycles	100	learning rate	3×10^{-4}
iteration	100	N	256
train_batches	100	M	4
replay buffer size	10^6	γ	1.00
evaluation episodes	10	Polyak-averaging coefficient	0.0005

- $D_2 = \{s_{t_n}, a_{t_n}, s_{t_{n+1}}\}_{n=1}^N$ consists of N state-action pairs. D_2 are used for policy optimization and optimize the parts of causal structure, $\phi_{\text{cau}}^{s \rightarrow s}$ and $\phi_{\text{cau}}^{a \rightarrow s}$, ϕ_{dyn} . With such a D_2 , GRD breaks the temporal cues in the training data to learn the policy and dynamics function.

Please refer to Algorithm 1 for a detailed training process.

E.5 Hyper-Parameters

The network is trained from scratch using the Adam optimizer, without any pre-training. The initial learning rate for both model estimation and policy learning is set to 3×10^{-4} . The hyperparameters for policy learning are shared across all tasks, with a discount factor of 1.00 and a Polyak-averaging coefficient of 5×10^{-4} . The target entropy is set to the negative value of the dimension of the robot action. To facilitate training, we utilize a replay buffer with a size of 1×10^6 time steps. The warmup size of the buffer for training is set to 1×10^4 . The model is trained for 3 epochs, with each epoch consisting of 100 training cycles. In each cycle, we repeat the process of data collection and model training for 100 iterations. During each iteration, we collect data from 100 time steps of interaction with the MuJoCo simulation, which is then stored in the replay buffer. For training the ϕ_{rew} , we sample 4 episodes, each containing 5×10^3 steps. As for policy learning and the optimization of ϕ_{dyn} , we use data from 256 time steps. ϕ_{cau} is trained together with ϕ_{rew} and ϕ_{dyn} . Validation is performed after every cycle, and the average metric is computed based on 10 test rollouts. The hyperparameters for learning the GRD model can be found in Table A2. All experiments were conducted on an HPC system equipped with 128 Intel Xeon processors operating at a clock speed of 2.2 GHz and 5 terabytes of memory.

F Additional Results

F.1 Results over different RL backbones

We provide the results of training with DDPG [Silver et al., 2014] and TD3 [Fujimoto et al., 2018] in Figure A2. As the experimental result shows, on the tasks of *HalfCheetah*, GRD consistently outperforms the baseline methods, RRD-Bias, RRD-Unbias, and IRCR, which are modified to run based on the same policy optimization algorithm, DDPG and TD3. We also provide results of ‘‘None’’, which utilizes the observed delayed reward for policy learning directly.

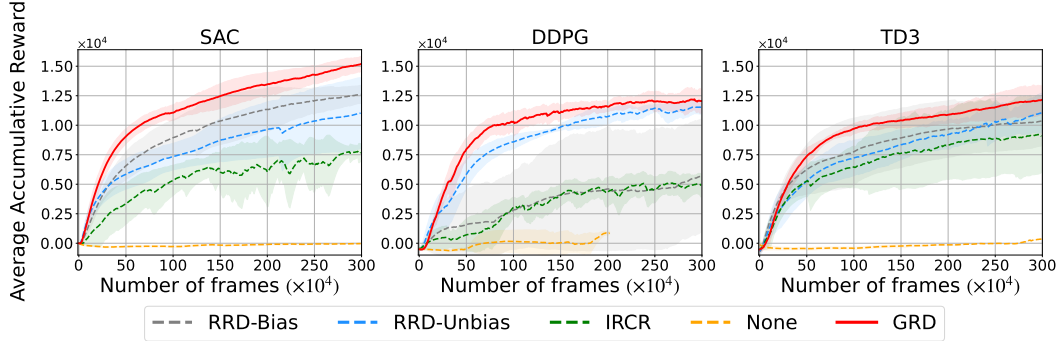


Figure A2: Evaluation with different RL backbones, SAC, DDPG and TD3. “None” is training with the observed delayed rewards.

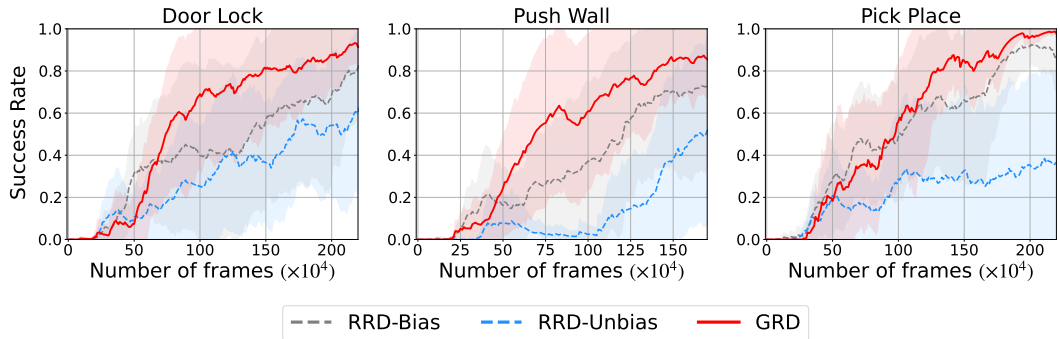


Figure A3: Evaluation on the manipulation tasks of MetaWorld, *Door Lock*, *Push Wall*, *Pick Place*.

F.2 Results over manipulation tasks

We provide the comparison of our method with RRD in the three tasks of MetaWorld, as shown in Figure A3. The evaluation metric is the success rate. As shown in the results, compared with RRD, GRD achieves comparable or better performance.

- Door Lock: The agent must lock the door by rotating the lock clockwise.
- Push Wall: The agent is required to bypass a wall and push a puck to a goal. The puck and goal positions are randomized.
- Pick Place: The agent needs to pick and place a puck to a goal. The puck and goal positions are randomized.

F.3 Visualization of Decomposed Rewards

As shown in Figure A4, we visualize the redistributed rewards in *Ant* by GRD, as well as the grounded rewards provided by the environment.

F.4 Visualization of Learned Causal Structure

We provide the visualization of learned causal structure in *Swimmer*, as shown in A5. Since the grounded causal structure is not accessible, we verify the reasonability of the learned causal structure by some observations:

- All the edges from different dimensions of a to r always exist, as shown in Figure A5 (d): *Swimmer* shares the same characteristic with *Ant* that the edges from different dimensions of

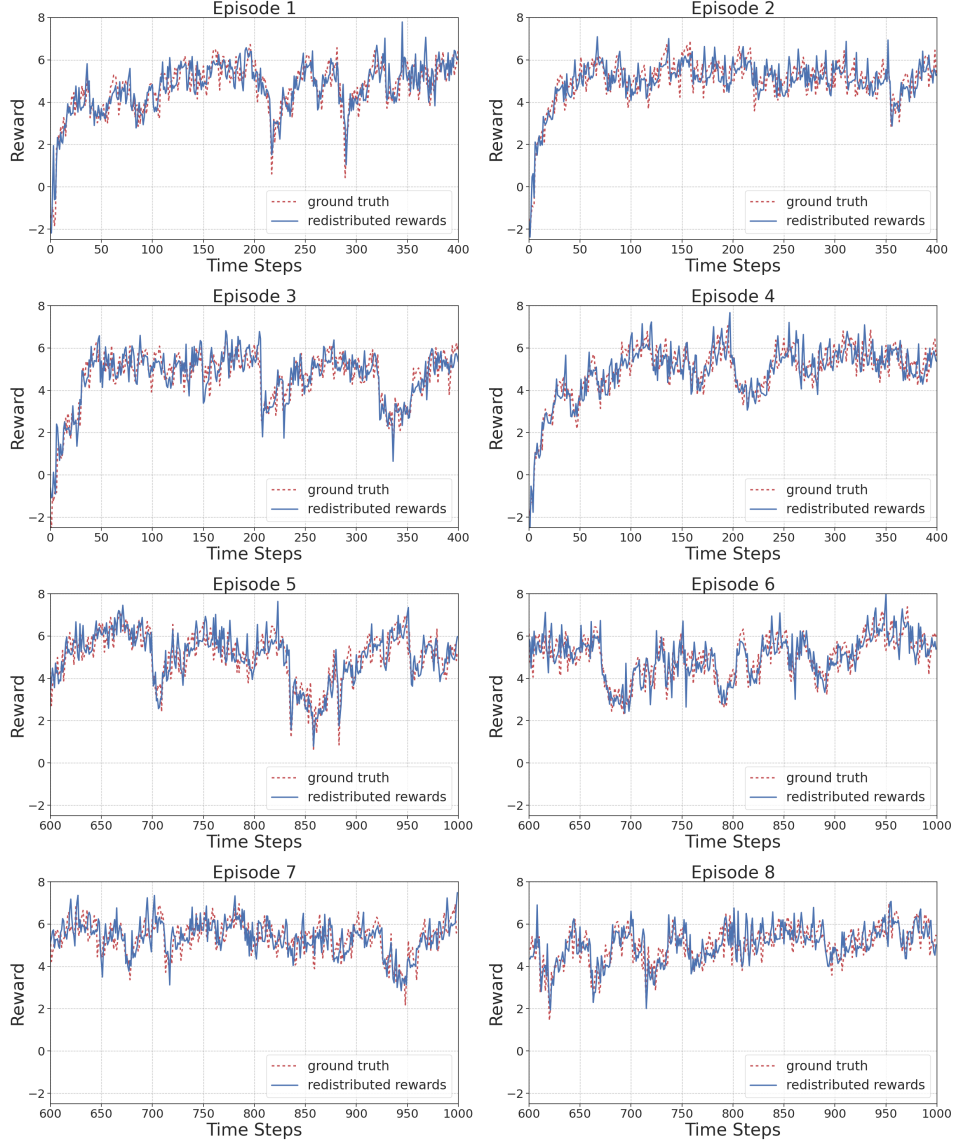


Figure A4: The visualization of redistributed rewards and grounded rewards in *Ant*. The results are produced by the GRD model trained after 1×10^6 steps. The redistributed rewards are shown in red, and the grounded rewards are shown in blue.

- \mathbf{a} to r always exist, corresponding with the reward design of penalizing the agent if it takes actions that are too large, measured by the sum of the values of all the action dimensions.
- According to Figure A5 (b), the first dimension of action (Torque applied on the first rotor) has an impact on the last three dimensions of state (angular velocity of front tip, angular velocity of first rotor, second rotor), which is corresponding with the intuitive that the first dimension of action should impact the part that connects to first rotor. We can get a similar observation for the second action dimension from Figure A5.
 - We can observe that all the state dimensions are learned to be connected to the reward; the possible explanation is that in the swimmer robot, any changes of the front tip, or two rotors will impact the position of the robot, potentially influencing the reward.

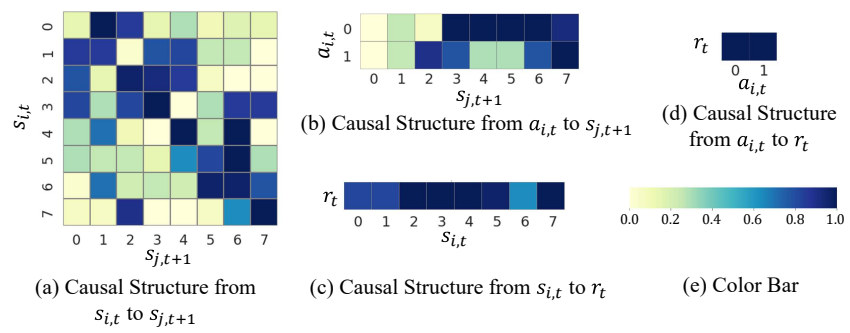


Figure A5: Learned causal structure in *Swimmer-v2*.

References

- Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in Neural Information Processing Systems*, 30, 2017.
- Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. *Advances in Neural Information Processing Systems*, 32, 2019.
- Jianyu Chen, Bodi Yuan, and Masayoshi Tomizuka. Model-free deep reinforcement learning for urban autonomous driving. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2765–2771. IEEE, 2019.
- Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.
- Ignasi Clavera, Jonas Rothfuss, John Schulman, Yasuhiro Fujita, Tamim Asfour, and Pieter Abbeel. Model-based reinforcement learning via meta-policy optimization. In *Conference on Robot Learning*, pages 617–629. PMLR, 2018.
- Cédric Colas, Tristan Karch, Nicolas Lair, Jean-Michel Dussoux, Clément Moulin-Frier, Peter Dominey, and Pierre-Yves Oudeyer. Language as a cognitive tool to imagine goals in curiosity driven exploration. *Advances in Neural Information Processing Systems*, 33:3761–3774, 2020.
- Yali Du, Lei Han, Meng Fang, Tianhong Dai, Ji Liu, and Dacheng Tao. Liir: learning individual intrinsic reward in multi-agent reinforcement learning. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 4403–4414, 2019.
- Ben Eysenbach, Xinyang Geng, Sergey Levine, and Russ R Salakhutdinov. Rewriting history with inverse rl: Hindsight inference for policy improvement. *Advances in neural information processing systems*, 33: 14783–14795, 2020.
- Meng Fang, Cheng Zhou, Bei Shi, Boqing Gong, Jia Xu, and Tong Zhang. Dher: Hindsight experience replay for dynamic goals. In *International Conference on Learning Representations*, 2018.
- Meng Fang, Cheng Zhou, Bei Shi, Boqing Gong, Jia Xu, and Tong Zhang. Dher: Hindsight experience replay for dynamic goals. In *International Conference on Learning Representations*, 2019a.
- Meng Fang, Tianyi Zhou, Yali Du, Lei Han, and Zhengyou Zhang. Curriculum-guided hindsight experience replay. *Advances in neural information processing systems*, 32, 2019b.
- Fan Feng, Biwei Huang, Kun Zhang, and Sara Magliacane. Factored adaptation for non-stationary reinforcement learning. *Advances in Neural Information Processing Systems*, 2022.
- Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.
- Tanmay Gangwani, Yuan Zhou, and Jian Peng. Learning guidance rewards with trajectory-space smoothing. *Advances in Neural Information Processing Systems*, 33, 2020.
- St John Grimbly, Jonathan Shock, and Arnú Pretorius. Causal multi-agent reinforcement learning: Review and open problems. In *Cooperative AI Workshop, Advances in Neural Information Processing Systems*, 2021.
- Carlos Guestrin, Daphne Koller, and Ronald Parr. Multiagent planning with factored mdps. *Advances in Neural Information Processing Systems*, 14, 2001.
- Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870. PMLR, 2018.
- Assaf Hallak, François Schnitzler, Timothy Mann, and Shie Mannor. Off-policy model-based learning under unknown factored dynamics. In *International Conference on Machine Learning*, pages 711–719. PMLR, 2015.
- Beining Han, Zhizhou Ren, Zuofan Wu, Yuan Zhou, and Jian Peng. Off-policy reinforcement learning with delayed rewards. In *International Conference on Machine Learning*, pages 8280–8303. PMLR, 2022.

- Lei Han, Peng Sun, Yali Du, Jiechao Xiong, Qing Wang, Xinghai Sun, Han Liu, and Tong Zhang. Grid-wise control for multi-agent reinforcement learning in video game ai. In *International Conference on Machine Learning (ICML)*, pages 2576–2585. PMLR, 2019.
- Yujing Hu, Weixun Wang, Hangtian Jia, Yixiang Wang, Yingfeng Chen, Jianye Hao, Feng Wu, and Changjie Fan. Learning to utilize shaping rewards: A new approach of reward shaping. *Advances in Neural Information Processing Systems*, 33:15931–15941, 2020.
- Biwei Huang, Fan Feng, Chaochao Lu, Sara Magliacane, and Kun Zhang. Adarl: What, where, and how to adapt in transfer reinforcement learning. In *International Conference on Learning Representations*, 2021.
- Biwei Huang, Chaochao Lu, Liu Leqi, José Miguel Hernández-Lobato, Clark Glymour, Bernhard Schölkopf, and Kun Zhang. Action-sufficient state representation learning for control with structural constraints. In *International Conference on Machine Learning*, pages 9260–9279. PMLR, 2022.
- Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2016.
- Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in neural information processing systems*, 32, 2019.
- Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, DJ Strouse, Joel Z Leibo, and Nando De Freitas. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International Conference on Machine Learning*, pages 3040–3049. PMLR, 2019.
- Nan Rosemary Ke, Anirudh Goyal ALIAS PARTH GOYAL, Olexa Bilaniuk, Jonathan Binas, Michael C Mozer, Chris Pal, and Yoshua Bengio. Sparse attentive backtracking: Temporal credit assignment through reminding. *Advances in neural information processing systems*, 31, 2018.
- Michael J. Kearns and Daphne Koller. Efficient reinforcement learning in factored mdps. In Thomas Dean, editor, *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 740–747. Morgan Kaufmann, 1999.
- B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2021.
- Yang Liu, Yunan Luo, Yuanyi Zhong, Xi Chen, Qiang Liu, and Jian Peng. Sequence modeling of temporal credit assignment for episodic reinforcement learning. *arXiv preprint arXiv:1905.13420*, 2019.
- Thomas Mesnard, Theophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Thomas S Stepleton, Nicolas Heess, Arthur Guez, et al. Counterfactual credit assignment in model-free reinforcement learning. In *International Conference on Machine Learning*, pages 7654–7664. PMLR, 2021.
- Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *CoRR*, abs/1602.01783, 2016.
- Kevin Patrick Murphy. *Dynamic bayesian networks: representation, inference and learning*. University of California, Berkeley, 2002.
- Andrew Y Ng, Daishi Harada, and Stuart J Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 278–287, 1999.
- Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pages 2778–2787. PMLR, 2017.
- Vihang P. Patil, Markus Hofmarcher, Marius-Constantin Dinu, Matthias Dorfer, Patrick M. Blies, Johannes Brandstetter, Jose A. Arjona-Medina, and Sepp Hochreiter. Align-rudder: Learning from few demonstrations by reward redistribution. In *International Conference on Machine Learning*. PMLR, 2022.
- Judea Pearl. *Causality: Models, reasoning, and inference*, 2000.
- Silviu Pitis, Elliot Creager, Ajay Mandlekar, and Animesh Garg. Mocoda: Model-based counterfactual data augmentation. *Advances in Neural Information Processing Systems*, 2022.

- Sai Rajeswar, Cyril Ibrahim, Nitin Surya, Florian Golemo, David Vazquez, Aaron Courville, and Pedro O Pinheiro. Haptics-based curiosity for sparse-reward tasks. In *Conference on Robot Learning*, pages 395–405. PMLR, 2022.
- Hubert Ramsauer, Bernhard Schöfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.
- Zhizhou Ren, Ruihan Guo, Yuan Zhou, and Jian Peng. Learning long-term reward redistribution via randomized return decomposition. In *International Conference on Learning Representations*, 2022.
- Douglas A Reynolds et al. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663), 2009.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.
- David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. Pmlr, 2014.
- David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dhharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018. doi: 10.1126/science.aar6404. URL <https://www.science.org/doi/abs/10.1126/science.aar6404>.
- Peter Spirtes, Clark N Glymour, Richard Scheines, and David Heckerman. *Causation, prediction, and search*. MIT press, 2000.
- Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J Martin, Animesh Mehta, Brent Harrison, and Mark O Riedl. Controllable neural story plot generation via reward shaping. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 5982–5988, 2019.
- Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.
- Zizhao Wang, Xuesu Xiao, Zifan Xu, Yuke Zhu, and Peter Stone. Causal dynamics learning for task-independent state abstraction. *International Conference on Machine Learning*, pages 23151–23180, 2022.
- Michael Widrich, Markus Hofmarcher, Vihang Prakash Patil, Angela Bitto-Nemling, and Sepp Hochreiter. Modern hopfield networks for return decomposition for delayed rewards. In *Deep RL Workshop NeurIPS 2021*, 2021.
- Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.
- Hongyang Yang, Xiao-Yang Liu, Shan Zhong, and Anwar Walid. Deep reinforcement learning for automated stock trading: An ensemble strategy. In *Proceedings of the First ACM International Conference on AI in Finance*, pages 1–8, 2020.
- Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021.
- Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.
- Junzhe Zhang and Elias Bareinboim. Markov decision processes with unobserved confounders: A causal approach. Technical report, Technical report, Technical Report R-23, Purdue AI Lab, 2016.
- Junzhe Zhang, Daniel Kumor, and Elias Bareinboim. Causal imitation learning with unobserved confounders. *Advances in Neural Information Processing Systems*, 33:12263–12274, 2020a.

- Marvin Zhang, Sharad Vikram, Laura Smith, Pieter Abbeel, Matthew Johnson, and Sergey Levine. Solar: Deep structured representations for model-based reinforcement learning. In *International conference on machine learning*, pages 7444–7453. PMLR, 2019.
- Zihao Zhang, Stefan Zohren, and Stephen Roberts. Deep reinforcement learning for trading. *The Journal of Financial Data Science*, 2(2):25–40, 2020b.
- Lulu Zheng, Jiarui Chen, Jianhao Wang, Jiamin He, Yujing Hu, Yingfeng Chen, Changjie Fan, Yang Gao, and Chongjie Zhang. Episodic multi-agent reinforcement learning with curiosity-driven exploration. *Advances in Neural Information Processing Systems*, 34:3757–3769, 2021.