A Supplementary Illustration

A.1 Notations

Symbol	Definition
s_t	The environment state at timestep t
a_t	The action interacting with the environment at timestep t
r_t	The reward feedback from the environment at timestep t
$\tilde{s}_t$	The state observed by the agent at timestep t
$\tilde{a}_t$	The action taken by the agent at timestep t
$\tilde{r}_t$	The reward received by the agent at timestep t
p^a	Action delay distribution
p^o	Observation delay distribution
p^{joint}	The distribution of the sum of action delay and observation delay
δ_a	Maximum value of the action delay distribution
δ_o	Maximum value of the observation delay distribution
δ_{joint}	Maximum value of p^{joint}
d_p	Wasserstein metric
$\bar{d}_p$	The maximal form of the Wasserstein metric

Table 4: Notations.

A.2 Interaction Process in Random Delay Environments

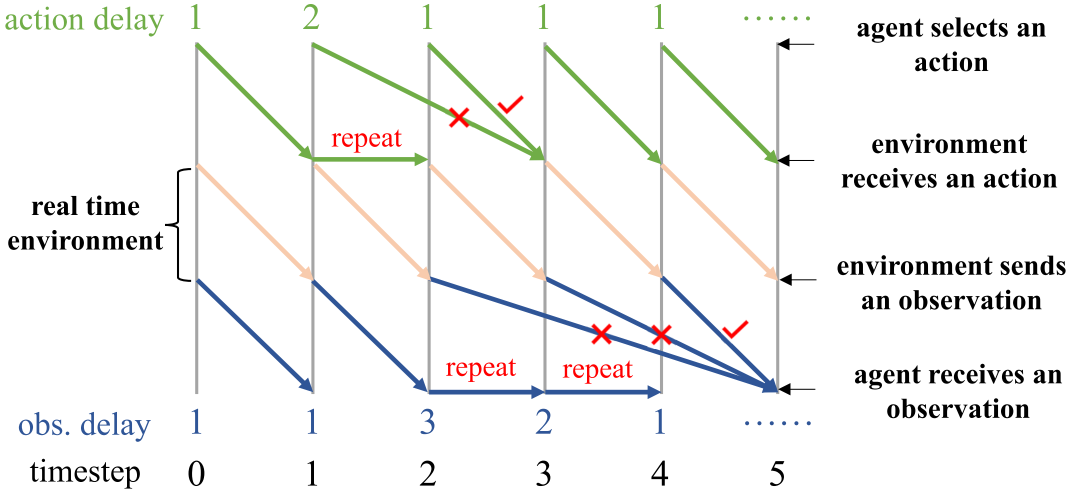


Figure 4: Interaction Process in Random Delay Environments.

Figure 4 illustrates the impact of random delays on agent–environment interactions, including cases of missing or duplicated observations and actions. Specifically, if the agent receives multiple observations and rewards at a given timestep due to random delays, it selects the most recent ones; if none are received, it reuses those from the previous timestep. Similarly, the environment executes the most recent action when multiple are received; otherwise, it reuses the action from the preceding timestep.

B Proof

B.1 Proposition 1 Derivation

Proof. Under pure action delay, the stochastic delay representation $Z(\tilde{s}_t, \tilde{a}_t)$ at timestep t is expressed as:

$$Z(\tilde{s}_t, \tilde{a}_t) = \sum_{i=1}^{\delta_a} p_i^a \cdot (\tilde{r}_{t+i} + \gamma Z(\tilde{s}_{t+i+1}, \tilde{a}_{t+i+1})) \,. \tag{9}$$

Under pure observation delay, it takes the form:

$$Z(\tilde{s}_t, \tilde{a}_t) = \sum_{j=1}^{\delta_o} p_j^o \cdot \left(\tilde{r}_{t+j} + \gamma Z(\tilde{s}_{t+j+1}, \tilde{a}_{t+j+1})\right). \tag{10}$$

In the presence of both delays, assuming independence of i and j as $\mathbb{P}(i,j) = p_i^a \cdot p_j^o$, the combined version of stochastic delay representation becomes:

$$Z(\tilde{s}_t, \tilde{a}_t) = \sum_{i=1}^{\delta_a} \sum_{j=1}^{\delta_o} p_i^a \cdot p_j^o \cdot \left(\tilde{r}_{t+(i+j)} + \gamma Z(\tilde{s}_{t+(i+j)+1}, \tilde{a}_{t+(i+j)+1})\right). \tag{11}$$

Let $k = i + j$ and define the joint delay distribution as:

$$p_k^{joint} = \sum_{\substack{i+j=k \\ i\in[1,\delta_a], j\in[1,\delta_o]}} p_i^a \cdot p_j^o, \tag{12}$$

with the total maximum delay denoted as $\delta_{joint} = \delta_a + \delta_o$. To simplify notation and unify bounds across formulations, we allow k to range from 1 to δ_{joint} and define $p_1^{joint} = 0$ to cover the case where no such (i, j) exists with $i + j = 1$. This does not affect the value of the summation but allows consistent indexing.

Substituting into Eq. (11) yields the unified form:

$$Z(\tilde{s}_t, \tilde{a}_t) = \sum_{k=1}^{\delta_{joint}} p_k^{joint} \cdot \left(\tilde{r}_{t+k} + \gamma Z(\tilde{s}_{t+k+1}, \tilde{a}_{t+k+1})\right). \tag{13}$$

Comparing Eq. (13) with Eqs. (9) and (10), we conclude that $Z(\tilde{s}_t, \tilde{a}_t)$ depends only on the aggregated delay k and its probability p_k^{joint}, and not on whether the delay originates from action or observation. Therefore, $Z(\tilde{s}_t, \tilde{a}_t)$ is invariant under the decomposition of k into i and j, establishing the equivalence of delay types in the stochastic delay representation.

B.2 Theorem 1 Derivation

In section B.2, we provide the proof of Theorem 1, which builds upon existing theoretical results in distributional reinforcement learning (Bellemare, Dabney, and Munos 2017; Duan et al. 2021; Nam, Kim, and Park 2021).

The stochastic delay representation with distributional returns operator derived from the Eq. (4) can be defined as

$$\mathcal{T}^\pi Z(\tilde{s}, \tilde{a}) = \sum_{i=1}^{\delta_a} p_i^a \sum_{j=1}^{\delta_o} p_j^o \left(\tilde{R}_{i+j}(\tilde{s}_{i+j}) + \gamma P^\pi Z(\tilde{s}_{i+j}, \tilde{a}_{i+j})\right) \tag{14}$$

where $P^\pi : \mathcal{Z} \to \mathcal{Z}$ is a state transition operator under policy π, with $\mathcal{Z}$ defined as the distributional value function space, and $P^\pi Z(\tilde{s}, \tilde{a}) :\overset{D}{=} Z(\tilde{s'}, \tilde{a'})$, where

$$\tilde{S'} \sim \mathop{\mathbb{E}}_{\substack{i\sim p_i^a \\ j\sim p_j^o}} P(\cdot \mid s_{-j}, \tilde{a}_{-i-j}) \tag{15}$$

and $\tilde{a}' \sim \pi(\cdot|\tilde{s}')$. The subscripts indicate the temporal relationship of states and actions relative to the current timestep. Specifically, $(\tilde{s}_{i+j}, \tilde{a}_{i+j})$ represents the state-action pair $i+j$ timesteps after the current pair $(\tilde{s}, \tilde{a})$, s_{-j} denotes the environment state i timesteps prior to the current pair, and $\tilde{a}_{-i-j}$ refers to the action taken $i + j$ timesteps prior to the current pair. We view the reward function as a random vector $\tilde{R}_{i+j} \in \mathcal{Z}$.

Lemma 1*: (Stochastic Delay Representation with Distributional Returns Policy Evaluation)*: Consider the stochastic delay representation with distributional returns backup operator $\mathcal{T}^\pi$, and a state–action distribution function $\mathcal{Z}_0(Z_0(\tilde{s}, \tilde{a})|\tilde{s}, \tilde{a}) : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(Z_0(\tilde{s}, \tilde{a}))$, which maps a state–action pair $(\tilde{s}, \tilde{a})$ to a distribution over random state–action returns $Z_0(\tilde{s}, \tilde{a})$. Define $Z_{i+1}(\tilde{s}, \tilde{a}) = \mathcal{T}^\pi Z_i(\tilde{s}, \tilde{a})$, where $Z_{i+1}(\tilde{s}, \tilde{a}) \sim \mathcal{Z}_{i+1}(\cdot|\tilde{s}, \tilde{a})$. Then, the sequence $\mathcal{Z}_i$ will converge to $\mathcal{Z}^\pi$ as $i \to \infty$.

Proof. As proved in (Bellemare, Dabney, and Munos 2017), $\bar{d}_p$ is a metric over value distributions, which can be formulated as

$$\bar{d}_p(Z_1, Z_2) = \sup_{s,a} d_p(Z_1(s,a), Z_2(s,a)) \tag{16}$$

where d_p is the Wasserstein metric, which has the following properties:

$$d_p(aU, aV) \leq |a| d_p(U, V) \tag{P1}$$

$$d_p(A + U, A + V) \leq d_p(U, V) \tag{P2}$$

$$d_p\left(\sum_{i=1}^{n} U_i, \sum_{i=1}^{n} V_i\right) \leq \sum_{i=1}^{n} d_p(U_i, V_i) \tag{P3}$$

where a is a scalar, U and V are random variables, while U_i and V_i are independent.

Consider $Z_1, Z_2 \in \mathcal{Z}$. By definition,

$$\bar{d}_p(\mathcal{T}^\pi Z_1, \mathcal{T}^\pi Z_2) = \sup_{\tilde{s},a} d_p(\mathcal{T}^\pi Z_1(\tilde{s},\tilde{a}), \mathcal{T}^\pi Z_2(\tilde{s},\tilde{a})). \tag{17}$$

By the properties of d_p, we can obtain that

$$\begin{aligned}
& d_p(\mathcal{T}^\pi Z_1(\tilde{s},\tilde{a}), \mathcal{T}^\pi Z_2(\tilde{s},\tilde{a})) \\
& = d_p(\sum_{i=1}^{\delta_a} p_i^a \sum_{j=1}^{\delta_o} p_j^o \left(\tilde{R}_{i+j}(\tilde{s}_{i+j}) + \gamma P^\pi Z_1(\tilde{s}_{i+j}, \tilde{a}_{i+j})\right), \sum_{i=1}^{\delta_a} p_i^a \sum_{j=1}^{\delta_o} p_j^o \left(\tilde{R}_{i+j}(\tilde{s}_{i+j}) + \gamma P^\pi Z_2(\tilde{s}_{i+j}, \tilde{a}_{i+j})\right)) \\
& \leq d_p(\sum_{i=1}^{\delta_a} p_i^a \sum_{j=1}^{\delta_o} p_j^o \gamma P^\pi Z_1(\tilde{s}_{i+j}, \tilde{a}_{i+j}), \sum_{i=1}^{\delta_a} p_i^a \sum_{j=1}^{\delta_o} p_j^o \gamma P^\pi Z_2(\tilde{s}_{i+j}, \tilde{a}_{i+j})) \\
& \leq \sum_{i=1}^{\delta_a} p_i^a \sum_{j=1}^{\delta_o} p_j^o \gamma d_p(P^\pi Z_1(\tilde{s}_{i+j}, \tilde{a}_{i+j}), P^\pi Z_2(\tilde{s}_{i+j}, \tilde{a}_{i+j})) \\
& \leq \sum_{i=1}^{\delta_a} p_i^a \sum_{j=1}^{\delta_o} p_j^o \gamma \sup_{\tilde{s}'_{i+j}, \tilde{a}'_{i+j}} d_p(Z_1(\tilde{s}'_{i+j}, \tilde{a}'_{i+j}), Z_2(\tilde{s}'_{i+j}, \tilde{a}'_{i+j}))
\end{aligned} \tag{18}$$

By utilizing the definition of Eq. (17) and the results of Eq. (18), we obtain

$$\begin{aligned}
\bar{d}_p(\mathcal{T}^\pi Z_1, \mathcal{T}^\pi Z_2) & = \sup_{\tilde{s},\tilde{a}} d_p(\mathcal{T}^\pi Z_1(\tilde{s},\tilde{a}), \mathcal{T}^\pi Z_2(\tilde{s},\tilde{a})) \\
& \leq \sum_{i=1}^{\delta_a} p_i^a \sum_{j=1}^{\delta_o} p_j^o \gamma \sup_{\tilde{s}'_{i+j}, \tilde{a}'_{i+j}} d_p(Z_1(\tilde{s}'_{i+j}, \tilde{a}'_{i+j}), Z_2(\tilde{s}'_{i+j}, \tilde{a}'_{i+j})) \\
& = \sum_{i=1}^{\delta_a} p_i^a \sum_{j=1}^{\delta_o} p_j^o \gamma \bar{d}_p(Z_1, Z_2)
\end{aligned} \tag{19}$$

The weights p_i^o and p_j^a are normalized: $\sum_{i=1}^{\delta_a} p_i^a = 1, \quad \sum_{j=1}^{\delta_o} p_j^o = 1$. Combining these weights with the discount factor, we define an effective discount factor:

$$\gamma_{\text{eff}} = \sum_{i=1}^{\delta_a} p_i^a \sum_{j=1}^{\delta_o} p_j^o \gamma \in (0,1). \tag{20}$$

For any Z_1, Z_2, the operator satisfies:

$$\bar{d}_p(\mathcal{T}^\pi Z_1, \mathcal{T}^\pi Z_2) \leq \gamma_{\text{eff}} \cdot \bar{d}_p(Z_1, Z_2), \tag{21}$$

By Banach's fixed-point theorem, $\mathcal{T}^\pi$ has a unique fixed point Z^π, and the sequence $Z_i = \mathcal{T}^\pi Z_{i-1}$ will converges to it as $i \to \infty$, i.e., $\mathcal{Z}_i$ will converge to $\mathcal{Z}^\pi$ as $i \to \infty$.

***Lemma 2**: (Policy Improvement)*: Let π_{new} be the optimal solution of maximizing cumulative returns. Then, $Q^{\pi_{\text{new}}}(\tilde{s},\tilde{a}) \geq Q^{\pi_{\text{old}}}(\tilde{s},\tilde{a})$ for $\forall(\tilde{s},\tilde{a}) \in \mathcal{S} \times \mathcal{A}$.

Proof. We can update the policy by maximizing the objective in terms of Q value

$$\pi_{new}(\cdot|\tilde{s}) = \arg\max_{\pi} \mathop{\mathbb{E}}_{\tilde{a}\sim\pi} [Q^{\pi_{old}}(\tilde{s},\tilde{a})] \quad \forall \tilde{s} \in \mathcal{S} \tag{22}$$

and the expected stochastic delay representation operator $\mathcal{T}_\pi$ can be written as

$$T^\pi Q^\pi(\tilde{s},\tilde{a}) = \sum_{i=1}^{\delta_a} p_i^a \cdot \sum_{j=1}^{\delta_o} p_j^o \cdot \left[\mathop{\mathbb{E}}_{\tilde{r}_{i+j}\sim\tilde{R}_{i+j}} [\tilde{r}_{i+j}(\tilde{s}_{i+j})] + \gamma \mathop{\mathbb{E}}_{\substack{\tilde{s}_{i+j+1}\sim p \\ \tilde{a}_{i+j+1}\sim\pi}} [Q^\pi(\tilde{s}_{i+j+1}, \tilde{a}_{i+j+1})] \right]. \tag{23}$$

Then, we can obtain that

$$\mathbb{E}_{\tilde{a}\sim\pi_{new}} [Q^{\pi_{old}}(\tilde{s},\tilde{a})] \geq \mathbb{E}_{\tilde{a}\sim\pi_{old}} [Q^{\pi_{old}}(\tilde{s},\tilde{a})] \quad \forall \tilde{s} \in \mathcal{S}. \tag{24}$$

By utilizing Eq. (24) and repeatedly expanding $Q^{\pi_{old}}$ on the right side with Eq. (23), we can derive the following results:

$$
\begin{aligned}
Q^{\pi_{old}}(\tilde{s}, \tilde{a}) &= \sum_{i=1}^{\delta_a} p_i^a \cdot \sum_{j=1}^{\delta_o} p_j^o \cdot \left[\underset{\tilde{r}_{i+j} \sim \tilde{R}_{i+j}}{\mathbb{E}} [\tilde{r}_{i+j}(\tilde{s}_{i+j})] + \gamma \underset{\substack{\tilde{s}_{i+j+1} \sim p \\ \tilde{a}_{i+j+1} \sim \pi_{old}}}{\mathbb{E}} [Q^{\pi_{old}} (\tilde{s}_{i+j+1}, \tilde{a}_{i+j+1})] \right] \\
&\leq \sum_{i=1}^{\delta_a} p_i^a \cdot \sum_{j=1}^{\delta_o} p_j^o \cdot \left[\underset{\tilde{r}_{i+j} \sim \tilde{R}_{i+j}}{\mathbb{E}} [\tilde{r}_{i+j}(\tilde{s}_{i+j})] + \gamma \underset{\substack{\tilde{s}_{i+j+1} \sim p \\ \tilde{a}_{i+j+1} \sim \pi_{new}}}{\mathbb{E}} [Q^{\pi_{old}} (\tilde{s}_{i+j+1}, \tilde{a}_{i+j+1})] \right] \\
&\dots \\
&\leq Q^{\pi_{new}}(\tilde{s}, \tilde{a}) \;\; \forall (\tilde{s}, \tilde{a}) \in \mathcal{S} \times \mathcal{A}
\end{aligned}
\tag{25}
$$

Thus, through the policy improvement, we can obtain $Q^{\pi_{\text{new}}}(\tilde{s}, \tilde{a}) \geq Q^{\pi_{\text{old}}}(\tilde{s}, \tilde{a})$ for $\forall (\tilde{s}, \tilde{a}) \in \mathcal{S} \times \mathcal{A}$.

We define π_k as the policy at iteration k. From Lemma 1, we can obtain the $\mathcal{Z}^{\pi_k}$ for $\forall \pi_k$ through stochastic delay representation with distributional returns policy evaluation process. Since the expectation of distributional value function $Z(\tilde{s}, \tilde{a})$ is the value $Q(\tilde{s}, \tilde{a})$, i.e. $Q(\tilde{s}, \tilde{a}) = \mathbb{E}[Z(\tilde{s}, \tilde{a})]$, we can obtain $Q^{\pi_k}(\tilde{s}, \tilde{a}) = \mathbb{E}[\mathcal{Z}^{\pi_k}(\tilde{s}, \tilde{a})]$. By Lemma 2, $Q^{\pi_k}(\tilde{s}, \tilde{a})$ is monotonically increasing for $\forall (\tilde{s}, \tilde{a}) \in \mathcal{S} \times \mathcal{A}$. Since Q^{π} is bounded everywhere for $\forall \pi$, the policy sequence π_k converges to some $\pi^{\dagger}$ as $k \to \infty$, and it follows that

$$
\mathbb{E}_{\tilde{a} \sim \pi^{\dagger}} \left[Q^{\pi^{\dagger}}(\tilde{s}, \tilde{a}) \right] \geq \mathbb{E}_{\tilde{a} \sim \pi} \left[Q^{\pi^{\dagger}}(\tilde{s}, \tilde{a}) \right] \;\; \forall \pi \, \forall \tilde{s} \in \mathcal{S}. \tag{26}
$$

Utilizing the results of Lemma 2, we can obtain

$$
Q^{\pi^{\dagger}}(\tilde{s}, \tilde{a}) \geq Q^{\pi}(\tilde{s}, \tilde{a}) \quad \forall \pi \, \forall (\tilde{s}, \tilde{a}) \in \mathcal{S} \times \mathcal{A}. \tag{27}
$$

Therefore, $\pi^{\dagger}$ is optimal, i.e., $\pi^{\dagger} = \pi^*$.

It is worth mentioning that Eq. (4) is applicable to environments where both observation and action delays are present and can be simplified to Eq. (5) when only one type of delay is present in the environment. Thus, Eq. (5) exhibits the same convergence.

C Related Work

Delay-aware Reinforcement Learning. Delays in the environment render the standard Markov decision process (MDP) inapplicable, prompting the development of variants such as delay-aware and constant delayed MDPs. To reformulate these as standard MDPs, a widely adopted state augmentation method concatenates the last observed state with intervening actions since that observation (Chen et al. 2021; Derman, Dalal, and Mannor 2021). In this line of research, the Delay-Correcting Actor-Critic (DCAC) algorithm (Bouteiller et al. 2021) is capable of handling both random observation and action delays, which aligns closely with the problem we aim to address. However, this method suffers from extremely low computational efficiency caused by the recursive nature of the partial resampling operator. Furthermore, the augmented state space grows exponentially as the number of delayed timesteps increases.

To address this issue, several state prediction-based methods(Valensi et al. 2024; Yu et al. 2023; Liotet, Venneri, and Restelli 2021; Karamzade et al. 2024) have been proposed to predict the undelayed state by utilizing historical state and action information, for example, by leveraging world models, recurrent neural networks (RNNs), etc. However, in complex random delay environments, the accuracy and generality of these prediction methods significantly limit their broader application. Additionally, (Kim et al. 2023) proposed a novel belief projection method that tackles the state-space explosion problem by projecting the augmented state space into a smaller one. However, this method is only applicable to constant delay environments, which do not reflect real-world practical features.

Recently, several auxiliary-policy-based methods have been proposed to mitigate limitations of state augmentation and prediction-based methods. (Wu et al. 2024a) framed the problem as variational inference and used behavior cloning to approximate undelayed policies, while (Wu et al. 2024b) leveraged auxiliary tasks with short delays to improve policy learning for long-delay tasks. Similarly, (Liotet et al. 2022) utilized imitation learning to teach delayed policies based on undelayed demonstrations. However, they rely on the unrealistic assumption that effective policies can be obtained in undelayed or short-delay environments, which is impractical due to the inherent and patterned nature of delays in real-world applications. Moreover, their focus on constant delays further limits practical applicability. The main methods are categorized by their characteristics, as shown in Table 1.

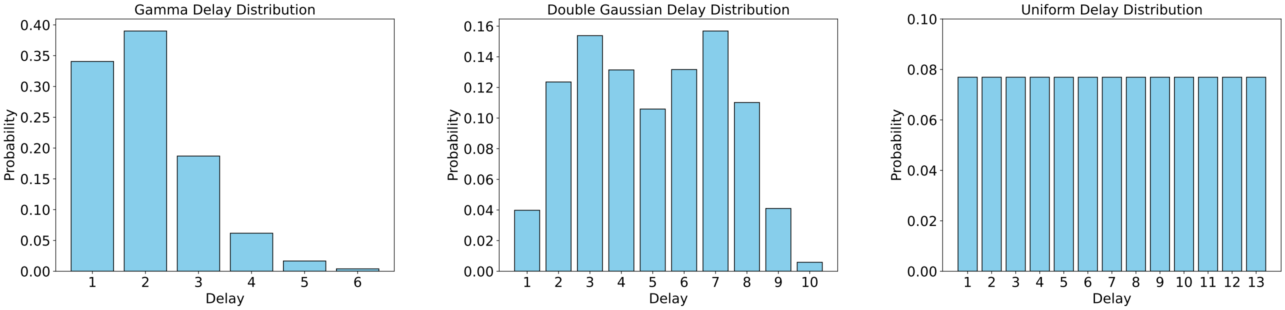
Distributional Reinforcement Learning. Conventional reinforcement learning optimizes the expected return, but randomness between the agent and the environment causes returns to follow a distribution under a policy π. (Bellemare, Dabney, and Munos 2017) introduced the distributional DQN (C51), representing returns as discrete distributions, establishing the foundation for distributional reinforcement learning. Subsequently, several methods have refined distribution modeling, providing robust theoretical and practical advances (Dabney et al. 2018b,a; Rowland et al. 2019; Zhou, Wang, and Feng 2020). The distributional perspective has also been extended to actor-critic frameworks, such as Gaussian Mixture Actor-Critic (GMAC) (Nam, Kim, and Park 2021), which models returns with Gaussian mixtures, and algorithms like D4PG (Barth-Maron et al. 2018) and

DSAC (Duan et al. 2021), improving value estimation in complex scenarios. Building on these advancements, to the best of our knowledge, this work is the first to leverage distributional reinforcement learning to model uncertainty in random delay environments and address challenges in accurate return estimation under delays.

D Experimental Setup

D.1 Random Delay Environments

To better evaluate the algorithm's performance, we design three delay distributions to simulate random delays in real application scenarios, as shown in Figure 5. The gamma delay distribution has a range of 1 to 6 with an expectation of 2, and the double Gaussian delay distribution has a range of 1 to 10 with an expectation of 5, while the uniform delay distribution has a range of 1 to 13 with an expectation of 6.



(a) Gamma Distribution.

(b) Double Gaussian Distribution.

(c) Uniform Distribution.

Figure 5: The different distributions of random delays.

D.2 Hyperparameters

Hyperparameter	**Setting**
Network	[256, 256, 256]
Batch Size	256
Total Timesteps	1,000,000
Learning Rate	0.0001
Learning Rate for α	0.0003
Hidden Activation	GELU
Output Activation	Linear
γ	0.99
Optimizer	Adam
Initial α	0.2

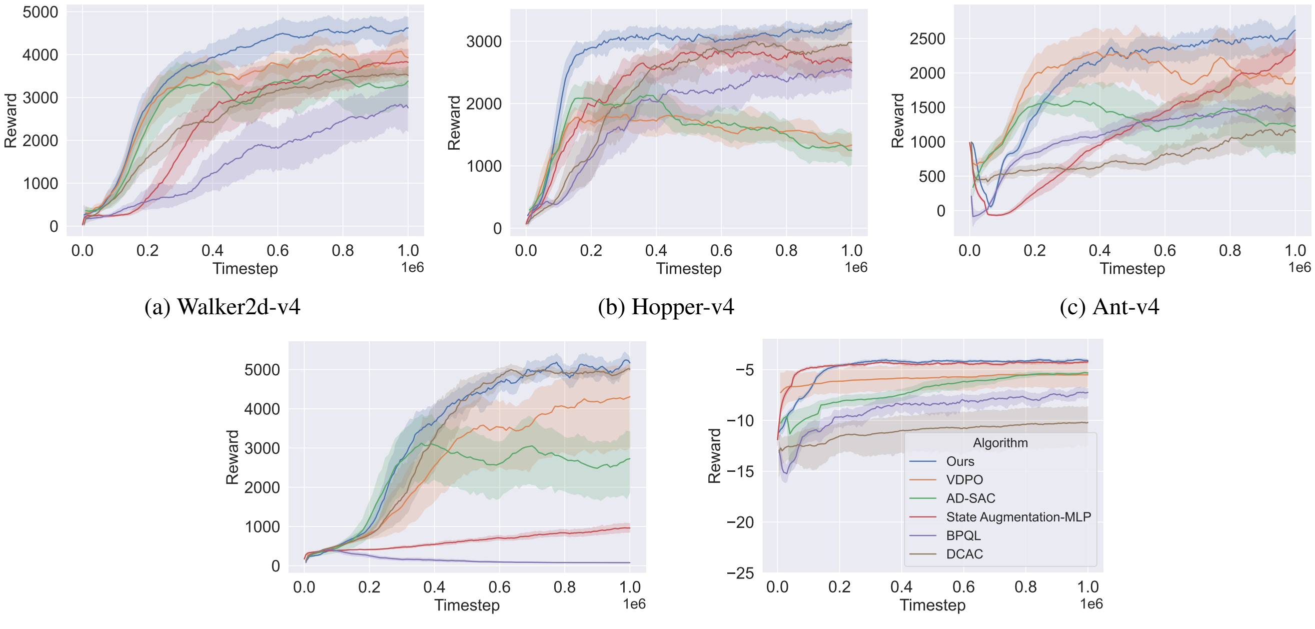
Table 5: Hyperparameter Settings

D.3 Compute Resources

All experimental results across our experiments are obtained on servers equipped with Intel(R) Xeon(R) CPU E5-2678 v3 @ 2.50GHz and NVIDIA GeForce RTX3090.

E Experimental Results

E.1 Comparative Experimental Results in Random Delay Environments



(a) Walker2d-v4

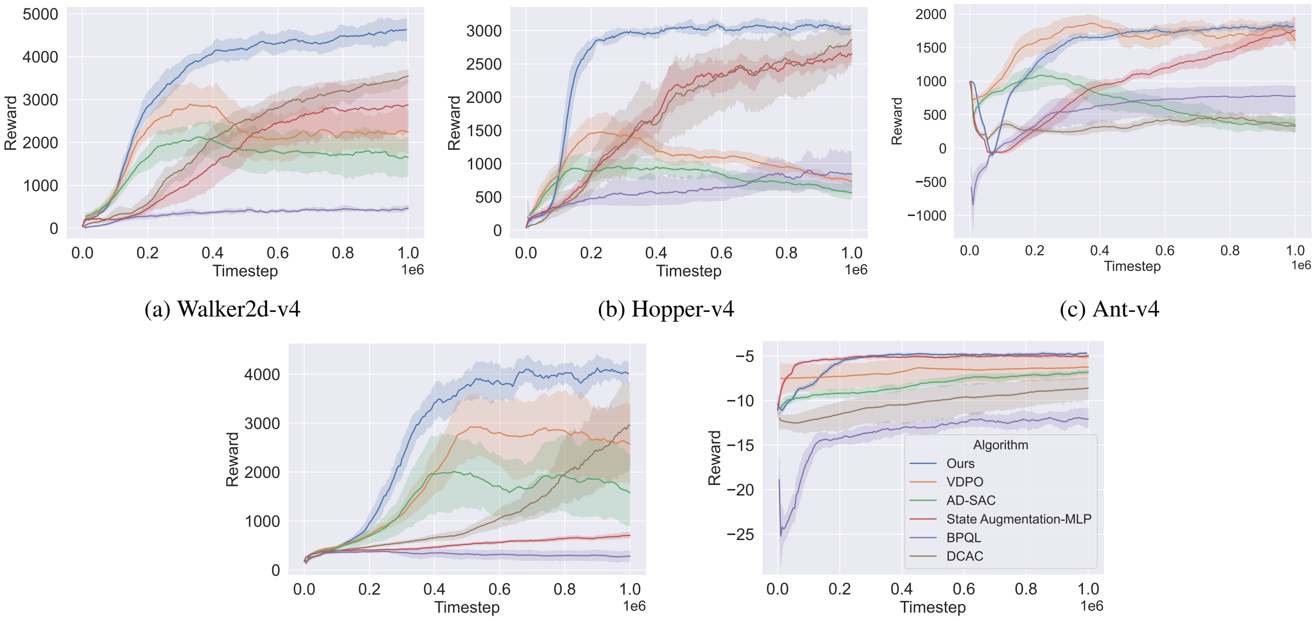
(b) Hopper-v4

(c) Ant-v4

(d) Humanoid-v4

(e) Reacher-v4

Figure 6: Comparison results in the gamma delay environment.



(a) Walker2d-v4

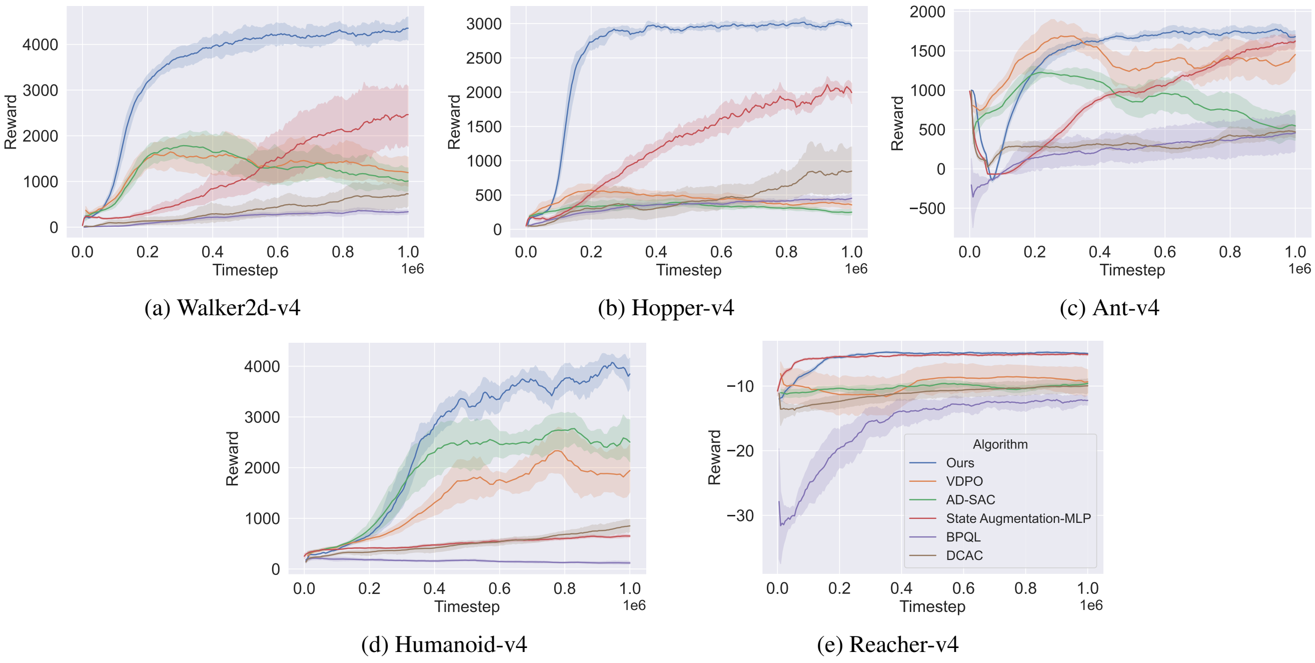
(b) Hopper-v4

(c) Ant-v4

(d) Humanoid-v4

(e) Reacher-v4

Figure 7: Comparison results in the double Gaussian delay environment.



(a) Walker2d-v4

(b) Hopper-v4

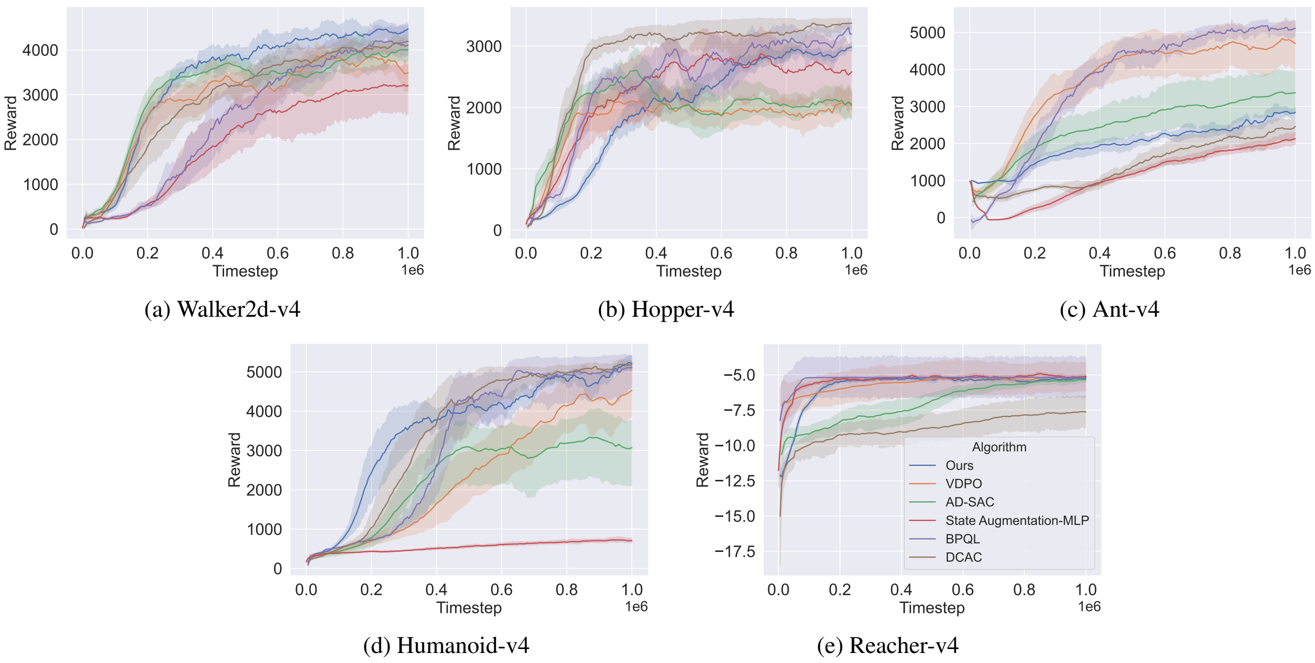
(c) Ant-v4

(d) Humanoid-v4

(e) Reacher-v4

Figure 8: Comparison results in the uniform delay environment.

E.2 Comparative Experimental Results in a Constant Delay Environment



(a) Walker2d-v4

(b) Hopper-v4

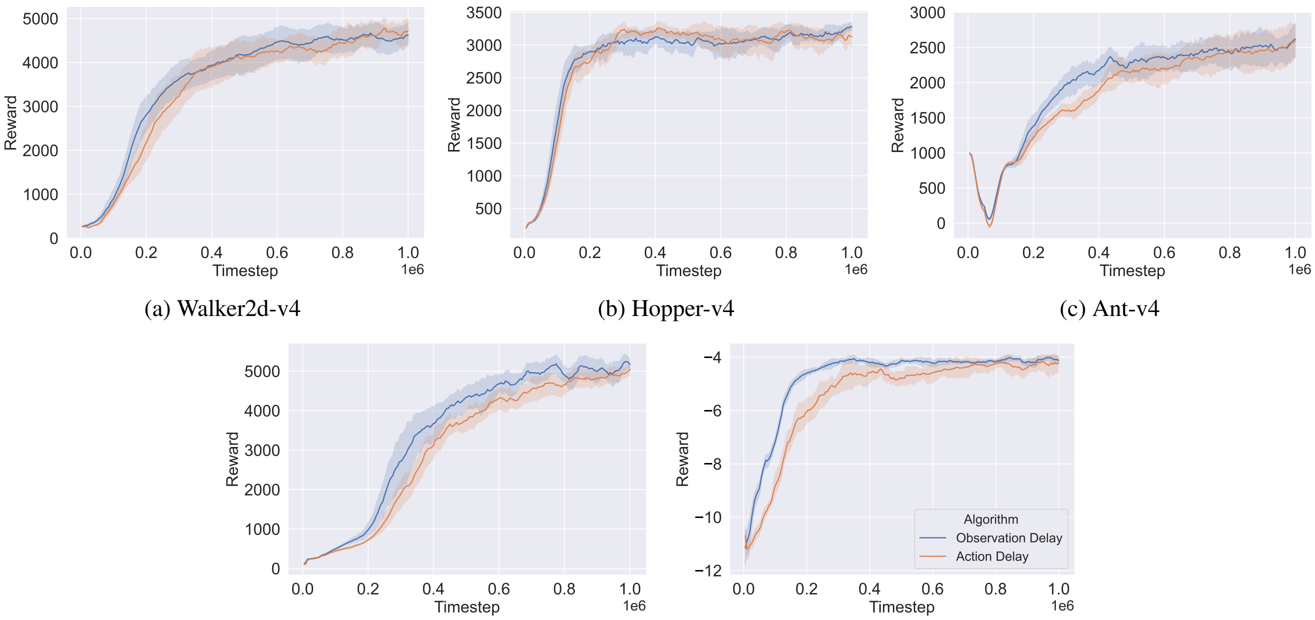
(c) Ant-v4

(d) Humanoid-v4

(e) Reacher-v4

Figure 9: Comparison results in the constant-5 delay environment.

E.3 Evaluation Results of Handling Observation and Action Delays



(a) Walker2d-v4

(b) Hopper-v4

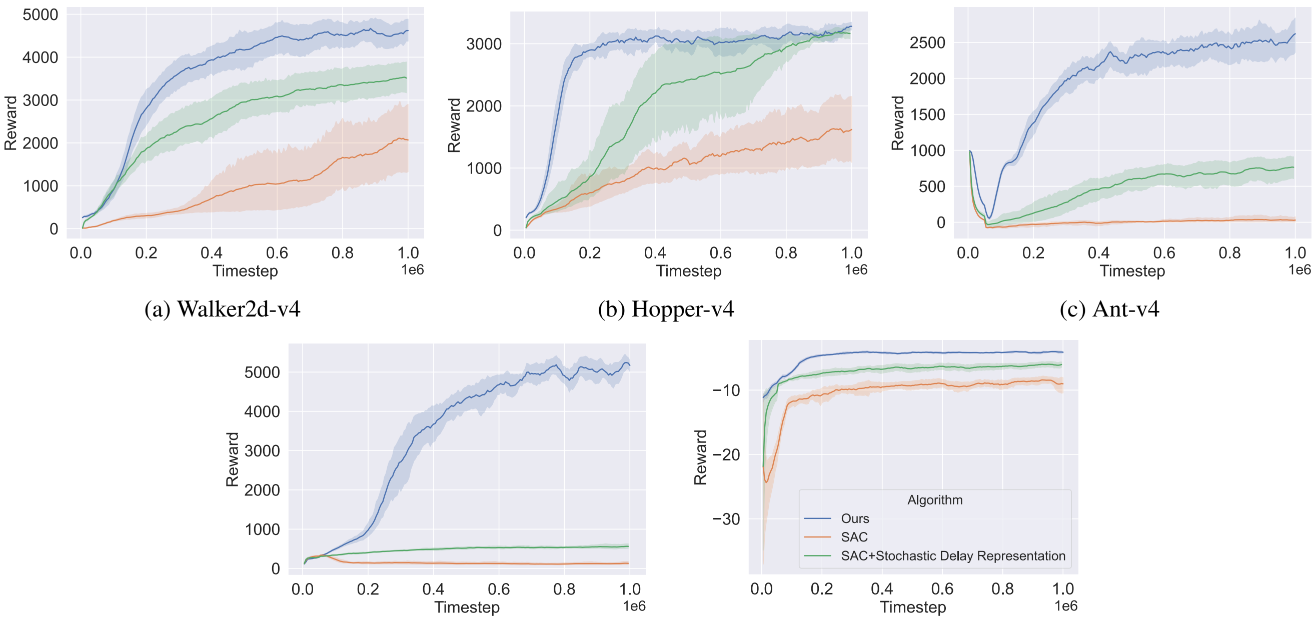
(c) Ant-v4

(d) Humanoid-v4

(e) Reacher-v4

Figure 10: Evaluation Results of Handling Observation and Action Delays.

E.4 Ablation Experimental Results



(a) Walker2d-v4

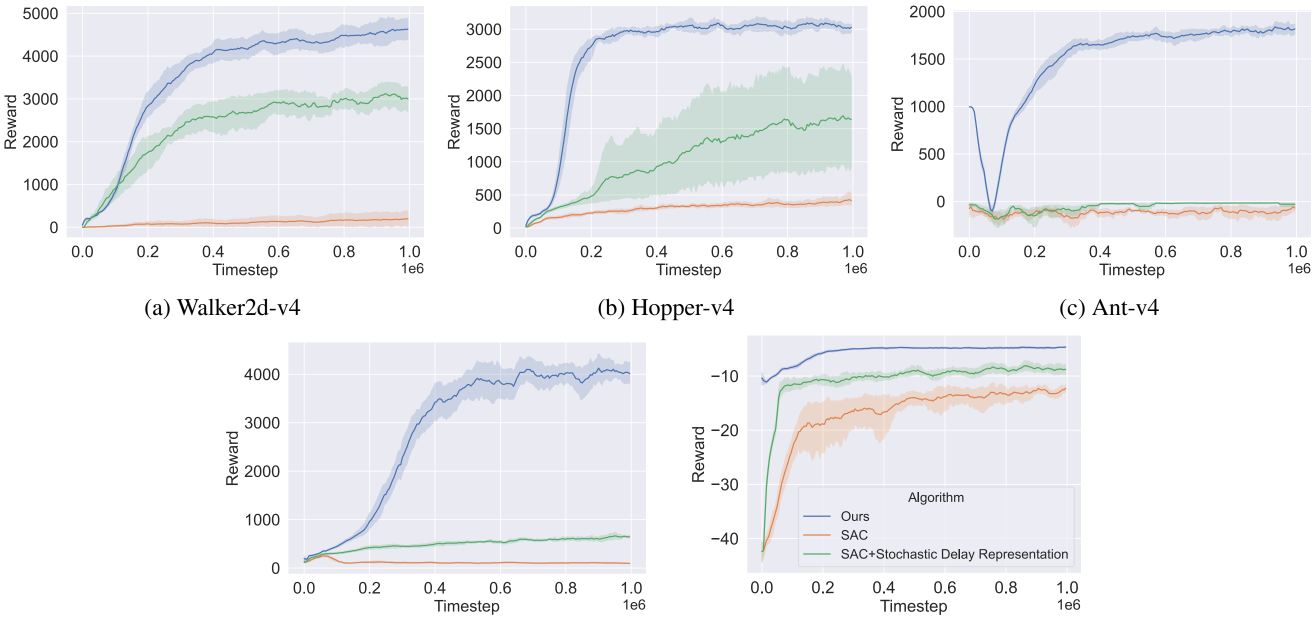
(b) Hopper-v4

(c) Ant-v4

(d) Humanoid-v4

(e) Reacher-v4

Figure 11: Ablation results in the gamma delay environment.



(a) Walker2d-v4

(b) Hopper-v4

(c) Ant-v4

(d) Humanoid-v4

(e) Reacher-v4

Figure 12: Ablation results in the double Gaussian delay environment.