석 사 학 위 논 문
Master's Thesis

향상된 시각적 일반화를 통한 모델기반 강화학습 방법론

Model-based Reinforcement Learning with Improved Observational Generalization

2025

박 민 규 (朴 玟 奎 Park, Mingyu)

한 국 과 학 기 술 원

Korea Advanced Institute of Science and Technology

석 사 학 위 논 문

향상된 시각적 일반화를 통한 모델기반 강화학습 방법론

2025

박 민 규

한 국 과 학 기 술 원

전기및전자공학부 (로봇공학 학제전공)

향상된 시각적 일반화를 통한 모델기반 강화학습 방법론

박 민 규

위 논문은 한국과학기술원 석사학위논문으로
학위논문 심사위원회의 심사를 통과하였음

2024년 12월 19일

심사위원장 이 동 환 (인)

심 사 위 원 성 영 철 (인)

심 사 위 원 이 기 민 (인)

Model-based Reinforcement Learning with Improved Observational Generalization

Mingyu Park

Advisor: Donghwan Lee

A dissertation submitted to the faculty of
Korea Advanced Institute of Science and Technology in
partial fulfillment of the requirements for the degree of
Master of Science in Electrical Engineering (Robotics)

Daejeon, Korea
December 19, 2024

Approved by

Donghwan Lee
Professor of Electrical Engineering

The study was conducted in accordance with Code of Research Ethics[1].

[1] Declaration of Ethical Conduct in Research: I, as a graduate student of Korea Advanced Institute of Science and Technology, hereby declare that I have not committed any act that may damage the credibility of my research. This includes, but is not limited to, falsification, thesis written by someone else, distortion of research findings, and plagiarism. I confirm that my thesis contains honest conclusions based on my own careful research under the guidance of my advisor.

MRE 박민규. 향상된 시각적 일반화를 통한 모델기반 강화학습 방법론. 전기및전자공학부 (로봇공학 학제전공) . 2025년. 32+iii 쪽. 지도교수: 이동환. (영문 논문)
Mingyu Park. Model-based Reinforcement Learning with Improved Observational Generalization. School of Electrical Engineering (Robotics Program) . 2025. 32+iii pages. Advisor: Donghwan Lee. (Text in English)

초 록

학습하는 동안 관찰하지 못했던 이미지에 일반화 가능한 강화 학습 (RL) 에이전트를 학습하는 것은 심층 강화학습을 실제 세계에 더 많이 적용할 수 있게 해준다. 해당 분야에서는 데이터 증강 (augmentation) 및 보조적인 표현 학습 (representation learning) 기법을 활용하여 이전 문헌에서 상당한 진전을 보였다. 그러나 관찰하지 못했던 이미지에 대해 샘플 효율적이고 일반화 가능한 정책을 학습하려면 종종 엄청난 양의 샘플이 필요로 한다. 이 연구에서는 널리 사용되는 모델 기반 RL 구조와 가치 함수 기반 RL의 선행 문헌에서 연구했던 기술들을 혼합하여 뛰어난 샘플 효율로 관찰 일반화를 장려하는 새로운 모델 기반 RL 방법을 제안한다. 우리의 핵심 아이디어는 이미지에 가해진 방해와 관계없이 일관된 표현을 예측하도록 모델을 제한하는 것이다. 해당 논문은 다양한 환경과 작업에서 RL 에이전트의 일반화 능력에 관한 광범위한 결과를 제공한다.

핵 심 낱 말 심층 강화학습, 시각적 강화학습, 모델기반 강화학습, 표현 학습, 강화학습에서의 시각적 일반화

Abstract

Learning a generalizable reinforcement learning (RL) agent to the unseen visual image enables further deployments of deep RL into the real world. The field has witnessed significant progress in the prior literature by leveraging data augmentation and auxiliary representation learning techniques. However, learning a sample-efficient and generalizable policy to unseen visual inputs often requires tremendous samples. In this work, we devise a novel model-based RL method for encouraging observational generalization with superior sample efficiency by blending a popular model-based RL architecture with advanced recipes from prior literature in model-free RL. Our key idea is to constrain the model to predict consistent representation regardless of perturbations. We provide extensive results concerning the generalization ability of RL agents with diverse environments and tasks.

Keywords Deep reinforcement learning, visual reinforcement learning, model-based reinforcement learning, representation learning, and visual generalization in reinforcement learning

Contents

List of Tables

List of Figures

Chapter 1. Introduction

Reinforcement learning (RL), a branch of machine learning interconnected with optimal control, trains an agent to maximize expected return by interacting with the environment. While another branch in machine learning, i.e. supervised learning, typically requires a dataset of pairs (s_t, a_t) to train a decision-making agent that predicts an optimal action, an RL agent can learn which action should be executed based on the value function that predicts future expected return without any supervisions [38]. Furthermore, the RL agent can find an optimal decision rule so-called policy without accurate physical information regarding the environment where the agent interacts. This makes RL remarkable in contrast to optimal control which necessitates accurate dynamics equations for emitting a precise control solution.

Nevertheless, traditional RL exhibits a few limitations regarding more complex decision-making problems. For instance, expensive computing budgets for value and policy learning hamper efficient learning when the state or action space contains high-dimensional information [40], e.g. pixel image state. Recent breakthroughs tackle these problems by combining RL with the neural network to solve diverse decision-making problems [18, 34, 41, 42]. Deep RL becomes more prominent in solving challenging control problems by adopting novel learning techniques [5, 8, 26, 32, 33]. Actually, the agent given eminent representations related to decision-making can easily train the accurate value function, leading to superior policy learning. However, one might consider a different but plausible scenario for the agent: what would happen if the agent is given heterogeneous environments between training and evaluation? To give an example, the autonomous driving agent would encounter a road scene during evaluation similar to the training environment other than the luminosity of view. While humans who have acquired expert driving skills can maneuver well regardless of the background brightness, the agent would face severe difficulty in making correct decisions since trained neural networks would fail to predict trustful output when trained on limited data and examined with a similar but unseen view [6].

Alleviating this generalization issue has induced numerous challenges since deep RL often couples policy learning and representation learning. Previous approaches address learning robust representation learning [24, 30, 43, 45, 49], applying stronger data augmentations [13, 14, 20, 23], and stabilizing value function learning [14, 19, 27]. Regarding the data augmentation, enlarging the limited dataset with *weakly* augmented, i.e. random shift, visual data contributes to the significant sample-efficient RL with visual input [23, 46, 47], whereas employing a relatively *strong* augmentation, e.g. random convolution or overlay, improves generalization capability of the agent over unseen image inputs during training.

Interestingly, a common ground shared across these approaches is that they fall into the model-free RL category where the agent mainly relies on the Q value function for policy learning. Since typical model-based RL exploits samples from the trained model that is responsible for generating extra experiences to learn the policy, the agent of model-based RL may show poor performance if the model tries to predict future trajectories with unseen input [38]. In contrast, model-free RL usually involves policy iterations with temporal difference (TD) learning, where the Q value function is trained to minimize the TD target and predicts the expected future return of the given state and action. However, the nature of model-free RL that updates the policy incrementally and contains weak inductive bias [2] essentially decreases the sample efficiency of the RL agent. Furthermore, the stochasticity of the environment or high-dimensional state and action spaces worsen this problem [48]. Alongside model-free RL, groundbreaking ideas in model-based RL have proven their superior performance and sample efficiency

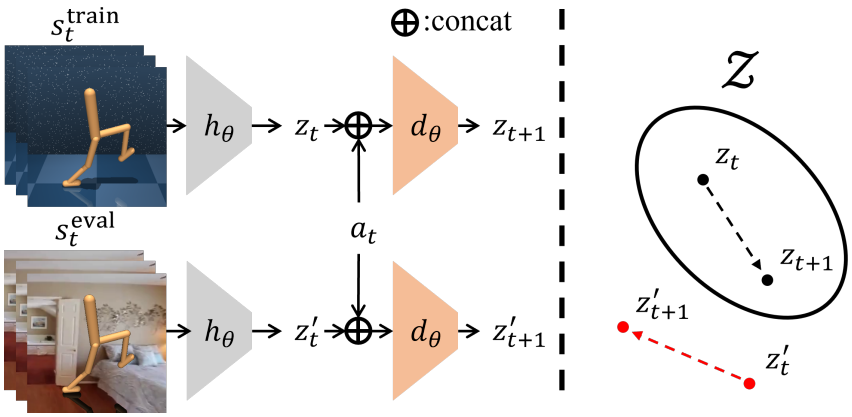


Figure 1.1: **Out-of-distributional representation.** Distribution shift occurs when sampled states between training and evaluation distribution differ. s_t^{train} and s_t^{eval} are example states. h_θ and d_θ are the encoder and transition dynamics, z and z' are extracted representations from in-distributional and out-of-distributional states, respectively. a is an action and $\mathcal{Z}$ is the distribution of z where representations are projected from only the training distribution. Subscript t represents a time step of the environment transition.

in diverse and challenging continuous control suites in recent years [9, 10, 12, 15, 16]. By learning a latent transition dynamics model with additional components regarding the model, current model-based RL has validated scalability to higher dimensions and brilliant performance on more complex domains. Thus, one might throw a question in this context, "Can we derive a model-based RL method that enjoys both sample efficiency and better generalization over unseen input by adopting recipes from model-free RL?"

A model-based RL agent with visual input first obtains corresponding representations using a feature extractor, i.e. the encoder, and afterward, rolls out the (latent) transition dynamics model with given representations. Therefore, the encoder that may predict inaccurate representations given unseen image input could be attributed to the collapse of the model-based RL agent in the observational generalization problem since the transition dynamics model would be conditioned on out-of-distributional representations in Figure 1.1. However, we contend that model-based RL can generalize to unseen image input with surpassing sample efficiency based on the idea of projecting out-of-distribution image samples to in-distribution representations and generating future representations consistent with the in-distribution samples for downstream model learning and planning.

In this paper, we propose **M**odel-**B**ased RL with **O**bservational **G**eneralization (MBOG), a model-based RL that empirically demonstrates strong generalization ability over unseen image input without sacrificing sample efficiency by employing recipes from model-free RL. MBOG consists of three key factors for improving performance: (1) applying weak and strong data augmentations to given image input for sample efficiency and generalization, (2) predicting a consistent latent representation simulated by the latent transition dynamics, and (3) regularizing the encoder to extract consistent representations over differently augmented input. We perform extensive experiments to verify our design choice contributes to superior performance on the generalization benchmark [50] across DM-Control [39] and Robosuite [52] benchmarks. Through a comprehensive ablation study, we prove that the proposed design becomes the best fit for solving observational generalization with model-based RL.

Chapter 2. Related Works

2.1 Observational Generalization in Deep RL

Learning a policy that outputs an action maximizing the expected cumulative return under different observation spaces between training and evaluation produces a unique challenge. Observational generalization refers to how the agent trained with visual input maximizes the return during evaluation where the input images from training and evaluation environments are visually different. Prior approaches often incorporate model-free value-based algorithms with representation learning [1, 24, 30, 43, 45, 49], data augmentation [13, 14, 20, 23, 25], and stabilization of value learning [14, 19, 27]. Since jointly learning low-dimensional compact representation from a high-dimensional raw image while capturing optimal behavior from reward signal in an end-to-end manner usually necessitates a large quantity of dataset [31, 36], learning an encoder that can extract helpful information for RL training from data plays a critical role in observational generalization. In this work, we focus on the observational generalization problem in RL similar to prior works. However, we also address the sample efficiency problem during RL training in addition to generalization performance, where prior works have been overlooked. We contend that considering the sample efficiency problem is as significant as the generalization performance since we are given only a limited set of training images according to problem formulation, which exacerbates when a pool of evaluation images increases.

2.2 Model-based Reinforcement Learning

Expanding previous value-based RL methods with the deep neural network has enabled successful adoptions of conventional RL to challenging domains, including a high-dimensional state or continuous action space. However, a prerequisite of a huge bucket of experience replay to learn a well-performing policy becomes a primary bottleneck for RL practitioners [48]. Model-based RL has been introduced as an alternative approach that trains a proxy of the transition model of the environment and exploits the learned model to generate synthetic data for further policy learning [4, 37], allowing the agent to simulate future states and plan the best action to maximize expected return. Since the proxy model is trained via limited collections of the transition, using the ensembles of the trained model [3, 21, 22] alleviates the uncertainty arising from the imperfect model. Learning a world model that simulates future states usually from high-dimensional observations with a latent sequential transition model [7] demonstrates superior sample efficiency and downstream RL performance. Formally, learning a recurrent transition model while reconstructing future images with encoder-decoder structure [9, 10, 12] or combining the planning with model predictive controller without reconstructions [15, 16] proves successful adoption to continuous control of more complicated domains. In this work, we choose TD-MPC [15] as a backbone model-based RL method for observational generalization problems since recent results have shown superior sample efficiency of TD-MPC compared to another state-of-the-art architecture, Dreamer [16]. We provide further discussions concerning model-based RL in Appendix 7.2.

Chapter 3. Preliminaries

3.1 Problem Formulation

We design the problem an RL agent tries to solve as the Markov Decision Problem (MDP). MDP is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ is the transition dynamics probability, and $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function. The agent receives not the state directly but the high-dimensional image from the observation space $\mathcal{O}$. Likewise in [14, 47], we define the state s_t as a stack of consequent images for simplicity, i.e. $s_t = \{o_t, o_{t-1}, o_{t_2}, \ldots, o_{t-k+1}\}$ where $s_t \in \mathcal{S}$, $o_t \in \mathcal{O}$, t and k is the time-step and the number of image stacks, respectively. The goal of the agent is to find an optimal policy π^* that maximizes the cumulative expected return $\mathbb{E}_{a_t \sim \pi(\cdot|s_t)} \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$ with the discount factor $\gamma \in [0, 1)$.

3.2 Observational Generalization

Following [14, 49, 50], we define the observational generalization problem as a particular problem set where an agent is trained with an MDP $\mathcal{M}$ and evaluated with a set of MDPs $\mathbb{M} = \{\bar{\mathcal{M}}_1, \bar{\mathcal{M}}_2, \ldots, \bar{\mathcal{M}}_n\}$. MDPs in the set share the same tuple with $\mathcal{M}$ other than the perturbed state space $\bar{\mathcal{S}}$ where the state of the perturbed state space $\bar{s} \in \bar{\mathcal{S}}$ is a concatenation of sampled images from the perturbed observation space $\bar{\mathcal{O}}$. The perturbed observation contains partial but essential information about the original observation (e.g., a locomotion agent's body image). During training, an agent receives the state (a stack of images) only from $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma \rangle$ to learn an optimal policy. In contrast, the agent is evaluated with an MDP sampled from $\mathbb{M}$ and given the perturbed state (a stack of perturbed images, $\bar{s}_t = \{\bar{o}_t, \bar{o}_{t-1}, \bar{o}_{t_2}, \ldots, \bar{o}_{t-k+1}\}$) to maximize the expected return, i.e., $\bar{o}_t \in \bar{\mathcal{O}}_i, \bar{\mathcal{M}}_i = \langle \bar{\mathcal{S}}_i, \mathcal{A}, \mathcal{T}, r, \gamma \rangle, \bar{\mathcal{M}}_i \sim \mathbb{M}$. Perturbed images are first sampled from $\mathcal{O}$ and perturbed with a transformation $\nu \sim \mathcal{N}; \nu : \mathcal{O} \times \mathcal{N} \mapsto \mathcal{O}$ that is also sampled from the set of perturbations (e.g., background color change). The goal of an agent is to maximize the expected return during evaluation without any access to the evaluation images during training.

3.3 Temporal Difference learning for MPC

Our method is built upon TD-MPC [15], a model-based RL architecture that combines temporal difference learning [38] for terminal Q value function with the model predictive control (MPC) for planning. TD-MPC is a latent space decoder-free world model that jointly learns parameters of the model: (i) a representation $z = h_\theta(s)$ by encoding a stack of high-dimensional inputs s into a low-dimensional representation z with an encoder h_θ, (ii) a latent dynamics model $z' = d_\theta(z, a)$ that predicts the next latent state z' given current latent state z and action a, (iii) a reward function $\hat{r} = R_\theta(z, a)$ that predicts the one-step reward, (iv) a Q value function $\hat{q} = Q_\theta(z, a)$ that predicts the state-action value function, and (v) a prior policy $\hat{a} \sim \pi_\theta(z)$ that is trained to maximize the Q value function Q_θ and used as a guiding policy for planning. z' and s' are the successor (latent) state while z and s are predecessor (latent) state, respectively.

During online training, the world model is trained via minimizing a weighted loss over the prediction horizon given the experience replay $\mathcal{B}$:

$$\mathcal{L}_{\text{TD-MPC}}(\theta; \mathcal{L}_{rew}, \mathcal{L}_Q, \mathcal{L}_{dyn}, \mathcal{B}) = \mathbb{E}_{\Gamma\sim\mathcal{B}}\left[\sum_{i=t}^{t+H} \lambda^{i-t} \mathcal{L}_{\text{TD-MPC}}(\theta; \mathcal{L}_{rew}, \mathcal{L}_Q, \mathcal{L}_{dyn}, \Gamma)\right]$$
$$= \mathbb{E}_{\Gamma\sim\mathcal{B}}\left[\sum_{i=t}^{t+H} \lambda^{i-t}\Big(c_1\mathcal{L}_{rew}(\theta; z_i, a_i, r_i) + c_2\mathcal{L}_Q(\theta; z_i, a_i, r_i, \tilde{z}_{i+1}) + c_3\mathcal{L}_{dyn}(\theta; z_i, a_i, z_{i+1}^{targ})\Big)\right], \quad (3.1)$$

with each prediction loss:

$$\mathcal{L}_{rew}(\theta; z_t, a_t, r_t) = \|R_\theta(z_t, a_t) - r_t\|_2^2, \quad (3.2)$$
$$\mathcal{L}_Q(\theta; z_t, a_t, r_t, \tilde{z}_{t+1}) = \left\|Q_\theta(z_t, a_t) - \mathtt{sg}\Big(r_t + \gamma Q_{\theta^-}\big(\tilde{z}_{t+1}, \pi_\theta(\tilde{z}_{t+1})\big)\Big)\right\|_2^2, \quad (3.3)$$
$$\mathcal{L}_{dyn}(\theta; z_t, a_t, \tilde{z}_{t+1}^{targ}) = \|d_\theta(z_t, a_t) - \mathtt{sg}\left(\tilde{z}_{t+1}^{targ}\right)\|_2^2, \quad (3.4)$$

where a horizontal trajectory segment $\Gamma = (s_t, a_t, r_t, s_{t+1})_{t:t+H}$ with a horizon H is sampled from the replay buffer $\mathcal{B}$ and $\lambda \in \mathbb{R}^+$ is a constant decaying over the horizon to weight closer predictions higher. $\mathcal{L}_{rew}, \mathcal{L}_Q, \mathcal{L}_{dyn}$ are the reward, Q value, and latent transition dynamics prediction loss, respectively, and $c_i \in \mathbb{R}^+, i = 1, 2, 3$ are the coefficients balancing each loss. θ^- stands for exponentially moving average target parameters of online parameter θ, $\mathtt{sg}$ is the *stop-grad* operator that prevents the computed gradient from influencing the remaining gradient computations. $\tilde{z}_{t+1}^{targ} = h_{\theta^-}(s_{t+1})$ and $\tilde{z}_{t+1} = h_\theta(s_{t+1})$ are directly extracted representation from the target and online encoder, respectively. At each time of the model learning, z_t is encoded from the state s_t first and recursively fed into the latent transition dynamics model d_θ to compute the loss; $z_t = h_\theta(s_t), z_k = d_\theta(z_k, a_k) \forall k \in [t+1, t+H]$. The prior policy π_θ is trained to maximize the Q value function over horizons only with respect to the policy parameters:

$$\mathcal{L}_\pi(\theta) = \mathbb{E}_{\Gamma\sim\mathcal{B}}\left[\sum_{i=t}^{t+H} \lambda^{i-t} Q_\theta\Big(\mathtt{sg}(z_i), \pi_\theta\big(\mathtt{sg}(z_i)\big)\Big)\right],$$

where the objective is commonly used in model-free actor-critic methods.

During inference (planning), a trained world model is used for the model predictive controller, specifically model predictive path integral (MPPI, [44]). The solution of MPPI can be found by iteratively fitting a time-dependent multivariate Gaussian with diagonal covariance over the action space. The objective of iterative fitting is to maximize the expected return:

$$\hat{R} = \sum_{i=t}^{t+H} \gamma^{i-t} R_\theta(z_i, a_i) + \gamma^H Q_\theta(z_H, a_H),$$

where $z_{t+1} = d_\theta(z_t, a_t)$ and action a_t is sampled from the multivariate normal distribution with the mean u_t^j and diagonal standard deviation σ_t^j at the sampling iteration j and time-step t. The parameters are initialized with zero means and unit standard variance in the action space. Since the reward function is trained to predict the 'instantaneous' reward, i.e., $\hat{r} = R_\theta(z_t, a_t)$, the Q value function allows the agent to fit the optimal action sequence based on 'farsighted' return. Following the procedure, an optimal action maximizing the expected return can be obtained via planning; $a_t \sim \Pi(\cdot|h_\theta(s_t))$. We refer to [15] for the additional details.

Chapter 4. Method

In this section, we present MBOG, a model-based RL method that empirically demonstrates strong generalization ability over unseen image input without sacrificing sample efficiency by employing verified recipes from the model-free RL realm. MBOG is built upon TD-MPC has proven strong sample efficiency over continuous control tasks and applies advanced techniques for observational generalization: (1) weak and strong data augmentations to given image input for sample efficiency and generalization, (2) consistent latent representation simulated by the latent transition dynamics and (3) regularization that allows the encoder extract consistent representations over differently augmented input. Our method is compatible with any model-based RL method that learns the latent transition dynamics with a visual feature extractor (the encoder) since we do not constrain any change of the underlying algorithm in principle. In the following, we explain how MBOG tackles the problem by leveraging the core components under the hood.

4.1 Architectural Overview

An overview of MBOG can be found in Figure 4.1. We build our method on top of TD-MPC, a sample-efficient model-based architecture, by fusing the world model learning with data augmentations and representation learning. We employ weak and strong augmentations for latent world model learning by applying weak and strong augmentations to the original image, subsequently. Representations encoded from heterogeneous images are mixed into the latent representation for world model learning (e.g., reward and transition model). Since the world model is trained over the prediction horizon, we regularize the latent dynamics and encoder over the horizon to have consistent representations regardless of an augmentation. We do not enforce constraints or changes in the model planning procedure.

4.2 Weak and Strong Augmentation

We refer to weak augmentation as employing a relatively minor change in an image (e.g., random shift transformation) and strong augmentation as applying a significant change in the image (e.g., random color convolution). While prior works have shown these augmentations boost the generalization performance and sample efficiency [14, 25, 46], the empirical results are limited to the value-based model-free learning approach. Hence, we propose a novel method for adapting data augmentations into model-based RL. Following prior works, we adopt *random-shift* [46] as weak and *random-overlay* [13] as strong augmentation in this work: *random-shift* augmentation applies a fixed amount of padding to a random direction in *top, bottom, right*, and *left* of the image and *random-overlay* augmentation linearly interpolates between a random image and an original image where the random image is sampled from an unrelated data to the task [51]. Likewise in previous works [13, 14, 49], while one can feed both weakly and strongly augmented images to the encoder in principle, we empirically find that dividing the batch randomly in half and augmenting two sub-batches with different augmentations can produce a similar performance with decreased computing budget. Consider a set of indices $\mathcal{I} = \{1, 2, \dots, B\}$ where B is the size of the batch. Let the indices of the batch be weakly and strongly augmented as $\mathcal{I}^w$ and $\mathcal{I}^s$, respectively. Then, the representations from weakly and strongly augmented images at time-step t, i.e.,

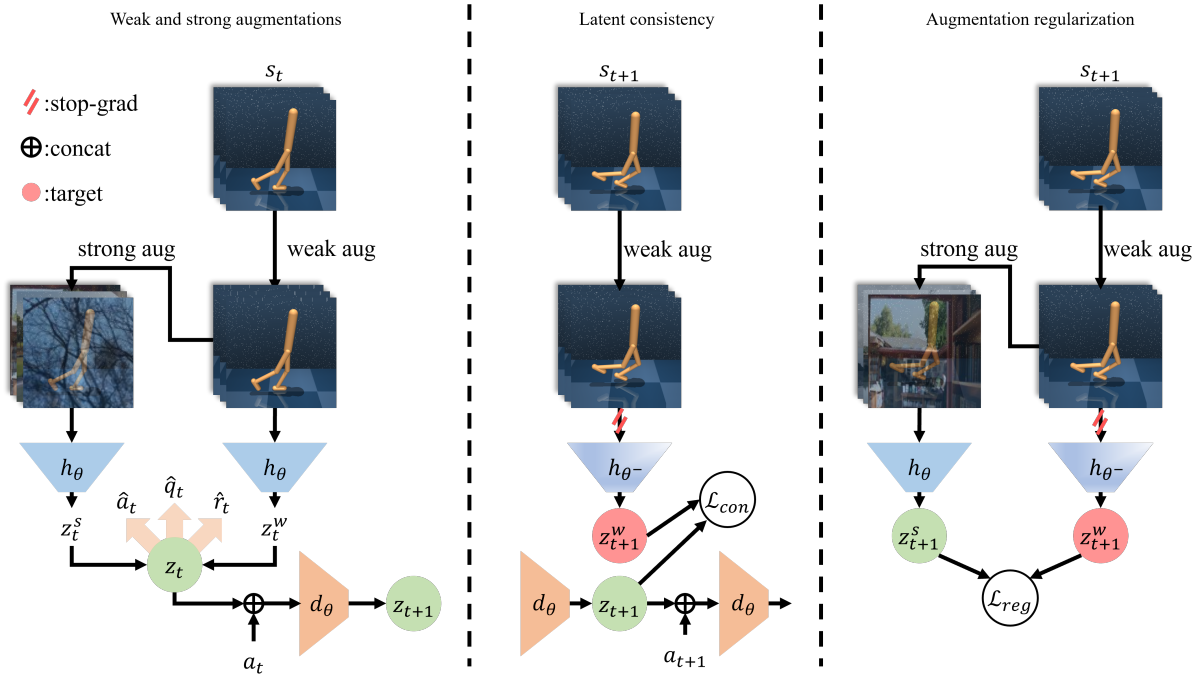


Figure 4.1: **MBOG architecture.** (Left) Weak and strong augmentations are used for observational generalization. (Center) A bias toward weakly augmented representation ensures latent representation consistency. (Right) The encoder is regularized to extract consistent representation regardless of augmentations.

z_t^w and z_t^s, become:

$$z_t^w = h_\theta(\tau^w(s_t, v^w)), \quad s_t = \{s_{t,i} : i \in \mathcal{I}^w\},$$
$$z_t^s = h_\theta(\tau^s(s_t^w, v^s)), \quad s_t^w = \{s_{t,j}^w : j \in \mathcal{I}^s\},$$

where $z_t = z_t^w \oplus z_t^s$ is the total representation at time-step t where $\oplus$ is element-wise concatenation, $\tau : \mathcal{S} \times \Upsilon \mapsto \mathcal{S}$ is a random augmentation function with a parameter $v \sim \Upsilon$ [13], and $s_{t,n}$ corresponds to the state that is collected by choosing elements in s_t of an index n along the batch dimension. Superscripts w and s state weak and strong augmentation, respectively. $\mathcal{I}^w$ and $\mathcal{I}^s$ are subsets of $\mathcal{I}$ where subsets are complementary and disjoint subsets, i.e., $\mathcal{I}^w \sim \text{Uniform}(1, B), \mathcal{I}^s = \mathcal{I}/\mathcal{I}^w, |\mathcal{I}^w|/|\mathcal{I}^s| = \zeta \in \mathbb{R}$. Through all experiments, we set the weak and strong augmentation ratio as $\zeta = 1.0$: divide the original batch in half for weak and strong augmentation. Although the representation z_t is recursively used for world model learning over the horizon, we apply these augmentations only at time-step t.

4.3 Latent Consistency

While strong data augmentation enables better generalization with unseen visual input, employing strong data augmentation to downstream latent model learning can become problematic. As observed in many prior works [14, 17, 29], noisy and high-variance target values might impede the fast convergence of the Q value function. Since the Q value network is conditioned on the representation in TD-MPC and the representation is encoded from the observation images directly over the horizon, the representation encoded from the strongly augmented image may produce a trivial signal for downstream model learning. However, the field has observed that weak data augmentation often encourages sample-efficient RL in high-dimensional observation space configuration [15, 46, 47]. To enable sample-efficient model learning

without sacrificing generalization performance, we constrain the latent representation to have consistency toward weakly augmented representation z_t^w. After the representation z_t is encoded with weak and strong augmentation, the latent transition dynamics model predicts the successor latent representation z_{t+1} given predecessor z_t and action a_t in the equation 3.1. The parameters of the latent transition dynamics model are updated by solving a regression problem: $\mathcal{L}(\theta; d_\theta) = \text{MSE}(d_\theta(z_t, a_t), z_{t+1})$ where $z_{t+1} = \text{sg}(h_{\theta^-}(s_{t+1}))$. We implement the weak augmentation, i.e., *random-shift*, to the target images to generate consistent target representation:

$$\mathcal{L}_{con}(\theta; z_t, a_t, \tilde{z}_{t+1}^{w,targ}) = \|d_\theta(z_t, a_t) - \texttt{sg}(\tilde{z}_{t+1}^{w,targ})\|_2^2,$$

where $\tilde{z}_{t+1}^{w,targ} = h_{\theta^-}(s_{t+1}^w)$ is the representation extracted from a weakly augmented state s_t^w through the target encoder h_{θ^-}.

4.4 Regularization over Augmentation

Following the previous steps, the latent transition model and other components of the world model are trained to predict consistent outputs regardless of whether the state in the batch is weakly augmented or strongly augmented. However, the encoder might predict inconsistent representation between training and evaluation images. Although the latent transition model is trained to predict consistent representations over the horizon, the encoder has no constraint to predict a similar representation whether the training or evaluation image is given. Hence, the model should generate reliable synthetic samples regardless of the training or evaluation phase to enable sample-efficient and generalizable model-based RL. To this end, we bring the auxiliary representation learning task to encoder learning during world model training. By regulating the encoder to preserve similar features (e.g., the physical body of the agent) and discarding irrelevant information (e.g., background and luminosity) between the original image and the augmented image, we can obtain consistent representation in both training and evaluation settings. Following [13], we implement the weak and strong augmentation to the state s_t to generate two different views of the original image. Subsequently, we train the encoder h_θ to extract applied strong augmentation in the weakly augmented image by minimizing regularization loss:

$$\mathcal{L}_{reg}(\theta; z_t^{w,targ}, z_t^s) = \left\| \frac{z_t^s}{\|z_t^s\|_2} - \frac{z_t^{w,targ}}{\|z_t^{w,targ}\|_2} \right\|_2^2,$$

where $z_t^s = h_\theta(s_t^s)$ is the representation extracted from a strong-augmented state with the online encoder while $z_t^{w,targ} = \texttt{sg}(h_{\theta^-}(s_t^w))$ is the representation extracted from a weak-augmented state with the target encoder and *stop-grad* operator.

Aggregated learning objectives. We incorporate three key components to the world model learning procedure of TD-MPC over the horizon in Equation 3.1 as follows:

$$\begin{aligned}\mathcal{L}_{\text{MBOG}}(\theta; \mathcal{B}, \Upsilon^w, \Upsilon^s) &= \mathbb{E}_{\Gamma\sim\mathcal{B}}\left[\mathbb{E}_{\upsilon^w\sim\Upsilon^w, \upsilon^s\sim\Upsilon^s}\left[\sum_{i=t}^{t+H} \mathcal{L}_{\text{MBOG}}(\theta; \Gamma, \upsilon^w, \upsilon^s)\right]\right] \\ &= \mathbb{E}_{\Gamma\sim\mathcal{B}}\left[\mathbb{E}_{\upsilon^w\sim\Upsilon^w, \upsilon^s\sim\Upsilon^s}\left[\sum_{i=t}^{t+H} \lambda^{i-t}\mathcal{L}_{\text{TD-MPC}}(\theta; \mathcal{L}_{rew}, \mathcal{L}_Q, \mathcal{L}_{con}, \Gamma) + \alpha\mathcal{L}_{reg}(\theta; z_i^{w,targ}, z_i^s)\right]\right], \quad (4.1)\end{aligned}$$

where $\mathcal{B}$ is the experience replay and α is a coefficient that balances the gradients of world model learning and regularization. Υ^w and Υ^s are the distribution of weak and strong random augmentation parameters. In principle, our method can be injected into *any* model-based RL method that trains the

latent transition dynamics model. We summarize our method in Algorithm 1. We exclude the learning process of the prior policy π_θ in Algorithm 1 since we employ the same procedure of TDMPC. We provide additional details regarding implementing MBOG and hyperparameters in Appendix 7.1.

Algorithm 1 MBOG: Model-based RL with Observational Generalization

$\tau^w(\cdot|\upsilon^w \sim \Upsilon^w), \tau^s(\cdot|\upsilon^s \sim \Upsilon^s)$: weak and strong augmentation functions

η, δ, α: learning and target network update rate, and regularization coefficient

θ, θ^-: randomly initialized network parameters

while not converged **do**

  // Collect experiences via planning:

  for time-step $t = 1, 2, \ldots, T$ **do**

    $a_t \sim \Pi(\cdot|h_\theta(s_t))$ ▷ Sample an action by planning with the model

    $s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t)$

    $\mathcal{B} \leftarrow \mathcal{B} \cup (s_t, a_t, r(s_t, a_t), s_{t+1})$ ▷ Update the replay buffer

  // Learn the world model:

  for gradient-step $t_g = 1, 2, \ldots, T_g$ per episode **do**

    $(s_t, a_t, r_t, s_{t+1})_{t:t+H} \sim \mathcal{B}$ ▷ Sample horizontal transitions

    $L \leftarrow 0$ ▷ Initialize cumulative loss

    for $i = t, t+1, \ldots, t+H$ **do**

      $\upsilon^w \sim \Upsilon^w, \upsilon^s \sim \Upsilon^s$ ▷ Sample augmentation parameters

      $L \leftarrow L + \mathcal{L}_{\text{MBOG}}(\theta; \Gamma, \upsilon^w, \upsilon^s)$ ▷ Equation 4.1

    $\theta \leftarrow \theta + \eta \frac{1}{H} \nabla_\theta L$ ▷ Update the online parameters

    $\theta^- \leftarrow (1-\delta)\theta^- + \delta\theta$ ▷ Update the target parameters

Chapter 5. Experiments

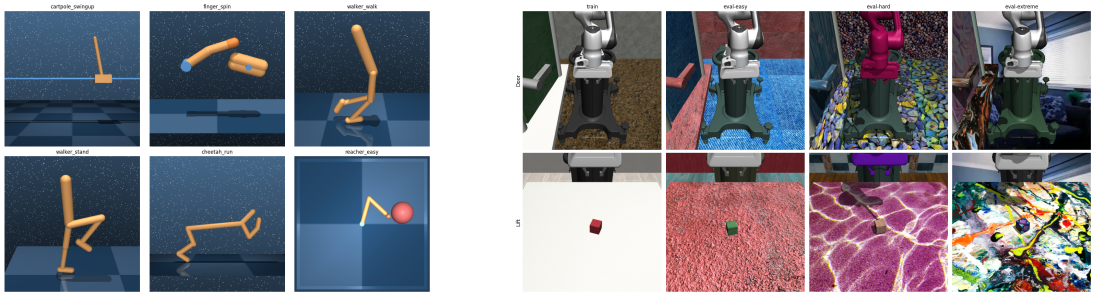
In this section, we provide empirical results of MBOG on diverse benchmarks with sophisticated experiment designs. We evaluate the generalization performance and sample efficiency with other baselines in relevant fields. We address the following questions via experiments: (i) how MBOG compares with other competitive baselines in observational generalization and sample efficiency, (ii) how our design choice affects the performance of MBOG, and (iii) how predicting consistent representation over the horizon impacts on observational generalization. We present our implementation details concerning the generalization benchmark and analyze the performance in the following.

5.1 Experimental Setup

In this section, we explain experimental configurations concerning the environments and baselines for performance comparison.

Environment. We evaluate MBOG over 6 tasks in the DeepMind control suite (DMC, [39]) and 2 tasks in robosuite [52]: *cartpole_swingup*, *finger_spin*, *walker_walk*, *walker_stand*, *cheetah_run*, and *reacher_easy* in DMC; *Door* and *Lift* in robosuite. We illustrate the tasks and environments in Figure 5.1. We train agents for each task with 1M gradient steps and evaluate the trained agents for 5 seeds. See further details regarding the environment and task setup in Appendix 7.1.

Baselines. We select state-of-the-art baselines in observational generalization problems to compare MBOG. Specifically, in model-free RL, SVEA [14] stabilizes off-policy Q-learning with data augmentation, SGQN [1] adapts self-supervised learning with attribution map and regularizes Q value learning, SRM [20] applies a spectrum augmentation to increase robustness toward spatial corruption, and PIEG [49] plugs large CNN pretrained with ImageNet for consistent representation. To compare the performance of MBOG with backbone model-based RL, we also evaluate the performance of TD-MPC. Regarding implementation details of baselines, see Appendix 7.1.



(a) DeepMind Control suite tasks.

(b) Robosuite tasks and evaluation types.

Figure 5.1: **Environments and tasks.** We consider locomotion and manipulation tasks for observational generalization. We address a set of diverse generalization tasks per environment.

5.2 Results

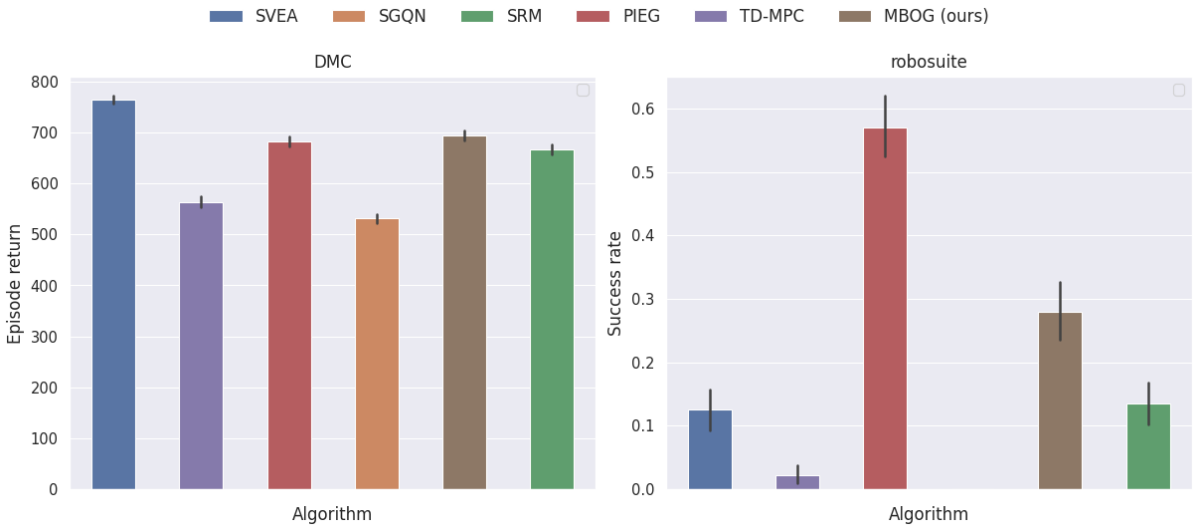
In this section, we provide rigorous explanations about each experiment we have designed in Section 5. We describe the background of each experimental setup and analyze empirical results.

5.2.1 Observational Generalization and Sample Efficiency

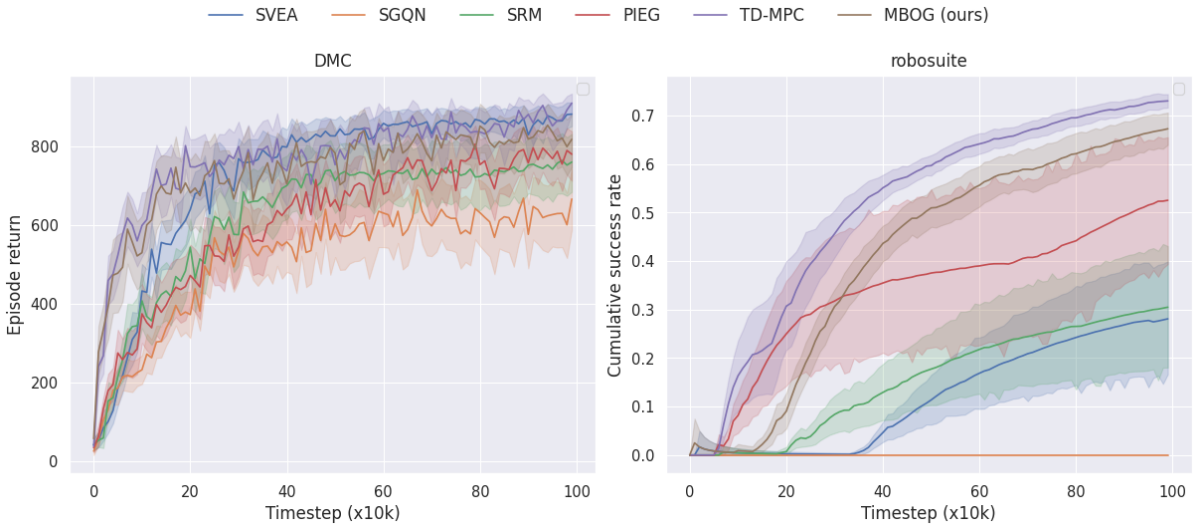
We provide aggregated results of performance comparison in Figure 5.2 and Table 5.1. MBOG proves superior sample efficiency over model-free RL in most cases and preserves similar sample efficiency compared to the backbone model-based RL, TD-MPC. In addition, MBOG demonstrates remarkable generalization performance in experiments although it fails to outperform all other baselines. It is worth noting that MBOG outperforms its backbone model, TD-MPC, in generalization performance with a trivial sacrifice of sample efficiency. Considering MBOG does not enforce any algorithmic modifications in model learning and planning with the model, the significant margin of generalization performance supports the validity of the proposed method to alleviate the out-of-distribution shift problem in observational generalization. See full experimental results in Appendix 7.3.

Table 5.1: **Quantitative comparison of generalization performance.** We quantitatively compare the performance of each algorithm here. Episode return and success rate are reported over tasks in DMC and robosuite, respectively. We average the results over 5 seeds.

Environment	Task	SVEA	SGQN	SRM	PIEG	TD-MPC	MBOG (ours)
DMC	cartpole_swingup	819.67 ±163	635.06±123.50	816.51±182.86	655.53±197.25	678.66±300.24	766.99±206.51
	finger_spin	814.97±305	760.97±307.79	814.93±316.14	780.77±214.09	617.33±356.70	721.62±305.08
	walker_walk	767.19±156	471.38±112.66	886.38±155.28	880.76±165.36	578.05±367.33	814.50±203.11
	walker_stand	947.18±102	876.18±163.26	142.16±29.37	937.51±102.51	667.06±321.88	883.47±143.34
	cheetah_run	435.99±176	226.14±90.50	502.99±155.63	249.32±112.64	379.52±252.26	238.15±95.86
	reacher_easy	801.57±349	217.68±343.67	834.44±321.28	586.87±453.16	461.69±457.92	744.19±367.91
		764.43±275.44	531.24±329.88	666.23±343.59	681.79±329.60	563.72±365.09	694.82±318.48
robosuite	Door	0.0±0.0	0.0±0.0	0.01±0.07	0.92±0.28	0.02±0.14	0.38±0.49
	Lift	0.25±0.43	0.0±0.0	0.27±0.44	0.23±0.42	0.03±0.16	0.18±0.39



(a) Evaluation results. Episode returns are averaged over 14 evaluation tasks.



(b) Sample efficiency results. Episode return and cumulative success rate of evaluations during training are reported in DMC and robosuite, respectively. Evaluation during training is averaged over 10 episodes.

Figure 5.2: **Experimental results.** We compare the generalization performance and sample efficiency of baselines in diverse experiments. MBOG demonstrates strong sample efficiency compared to previous model-free RL methods and improved generalization ability. TD-MPC also exhibits strong sample efficiency but fails to generalize to unseen images.

5.2.2 Ablation of Design Choices

We investigate several possible design choices for improving the generalization performance of model-based RL. Toward this objective, we examine the performance of variants of MBOG: dynamic and consistent augmentation, different strong augmentation, and another auxiliary task for representation learning. In the following, we explain the past candidates of MBOG and provide a summarized result in Figure 5.3.

Dynamic and consistent augmentations. We contend that using both weak and strong augmentation contributes to the increased generalization performance of MBOG. Since typical model-based RL

predicts future transition samples over the horizon, we suggest a novel model learning scheme with those augmentations. However, it might be unclear whether applying **dynamic** augmentation over the horizon benefits the generalization performance. Considering the horizontal state $s_{t:t+H}$ with the horizon H during model learning, MBOG augments the states with **both** weak and strong augmentation over the horizon to constrain latent consistency (Section 4.3) and regularization (Section 4.4). The horizontal states from time-step t to $t+H$ are augmented as:

$$s^w_{t:t+H} = \{\tau^w_k(s_k, \upsilon^w_k) : \upsilon^w_k \sim \Upsilon^w, k \in \{t, t+1, \dots, t+H\}\}$$
$$s^s_{t:t+H} = \{\tau^s_k(s^w_k, \upsilon^s_k) : \upsilon^s_k \sim \Upsilon^s, k \in \{t, t+1, \dots, t+H\}\},$$

where the random augmentation function τ can depend on time-step or not. We refer to τ_t as consistent augmentation over the horizon if $\tau_t = \tau_{t+k} \forall k \in [0, H]$ is satisfied. If the augmentation function differs over the horizon, i.e., $\tau_t \neq \tau_{t+k}, \forall k \in [0, H]$, we denote it as a dynamic augmentation function. MBOG adopts the **dynamic augmentation** for both weak and strong augmentations, i.e., $\tau_t \neq \tau_{t+k}, \forall k \in [0, H]$. We denote MBOG with consistent augmentation as `MBOG_CONST_AUG` later in experiments.

Different strong augmentation. Motivated by prior results [13, 14, 49], we choose *random-overlay* as strong augmentation for observational generalization in model-based RL. However, as proposed in [14, 23, 25], other strong augmentation methods can become strong candidates for observational generalization in Deep RL. Among potential augmentations, we opt *random-conv* as another strong augmentation method, which has exhibited remarkable performance in previous works. Precisely, we refer *random-overlay* and *random-conv* augmentation as:

$$\tau^{s,\text{overlay}}(o, \tilde{o}) = (1-\delta)o + \delta\tilde{o}$$
$$\tau^{s,\text{conv}}(o, w) = \texttt{CONV}(o, w)$$

where $o \in \mathcal{O}^{B\times C\times H\times W}$ is the original images with the batch size B, channel C, height H, and width W, respectively. δ is a linear interpolation coefficient, $\tilde{o} \sim \mathcal{D}$ is an overlaying image sampled from a dataset unrelated to the task. We set the default value of δ as 0.5. `CONV` stands for 2-dimensional convolution operation over the batch dimension and $w \in \mathbb{R}^{N\times C_k\times H_k\times W_k} : w \sim \mathcal{N}(0, 1)$ is the convolution filter randomly initialized from the normal distribution and kernel number N, channel C_k, height H_k, and width W_k, respectively. We implement the same strong augmentation over the channel dimension to obtain the state $s_t = \{o_t, o_{t-1}, \cdots, o_{t-k+1}\}$. While MBOG chooses *random-overlay* as strong augmentation, we denote `MBOG_CONV` as MBOG replaced with *random-conv*.

Different strong augmentation. Learning proper representation for vision-based RL plays a critical role in sample efficiency and generalization performance. Prior literature [1, 17, 24, 30, 49] has explored the field by leveraging popular representation learning techniques in computer vision. As in Section 4.4, we intend the encoder to predict robust representation regardless of distracting components (e.g., background image). Hence, we consider two variants of auxiliary representation learning task: *SODA* [13] and *CURL* [24] where we use the same auxiliary task from SODA for representation learning in Section 4.4:

$$\mathcal{L}_{CURL}(\theta; q, k) = \left(\log \frac{\exp q^T W k_+}{\exp q^T W k_+ + \sum_{j=0}^{B} \exp q^T W k_j}\right),$$

where $q = h_\theta(\tau^w(s_t, \upsilon^q))$ is the anchor, $k = h_{\theta^-}(\tau^w(s_t, \upsilon^k))$ is the key, and W is the weight kernel for bilinear product. υ^q and υ^k are separately sampled parameters for generating anchor and key images from the same distribution Υ^w. We denote `MBOG_CURL` as MBOG replacing the regularization loss (i.e., $\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{SODA}}$) with $\mathcal{L}_{\text{CURL}}$.

Aggreagated results. We provide the aggregated experimental results in Figure 5.3. We compare MBOG with `MBOG_CONST_AUG`, `MBOG_CONV`, and `MBOG_CURL` in *finger_spin* and *walker_walk* tasks. MBOG proves superior sample efficiency and generalization performance compared to other options, validating that the proposed method is the most reasonable choice among available options. See further experimental details regarding this comparison in Appendix 7.3.

Figure 5.3: **Experiments comparing design choices.** We have considered several methodologies for building generalizable model-based RL. Among the potential choices, MBOG proves its superiority in experiments.

5.2.3 Visualized Consistent Representation

To reduce the prediction error of the world model for observational generalization, we propose a novel approach that constrains the world model learning with data augmentation and regularization. Through the approach, the world model becomes generalizable to unseen observations for evaluation and avoids forgoing the superior sample efficiency of model-based RL. The idea below the method is that it is important not only to constrain a consistency to the latent transition model that recursively predicts the latent transitions but also to regularize the encoder to predict robust representation regardless of distracting factors. Hence, we conduct experiments to validate the motivation that MBOG would demonstrate consistent prediction ability regardless of perturbations while TD-MPC fails. We set the conditions for consistent representation: (1) the latent representations predicted by the transition model should be aligned with the future latent representations from the encoder and (2) the encoder should predict consistent representations between the original and unseen images. We collect the horizontal replay transitions by rolling out the trained model and feed the same input (i.e., $(s_t, a_t, s_{t+1})_{t:t+H}$) to MBOG and TD-MPC to predict the latent representations. Since the representation often has more than two dimensions, we visualize the collected representations from the latent transition model d_θ and encoder h_θ with UMAP [28] in Figure 7.10, a popular manifold learning method to extract two-dimensional coordinates from high-dimensional representations.

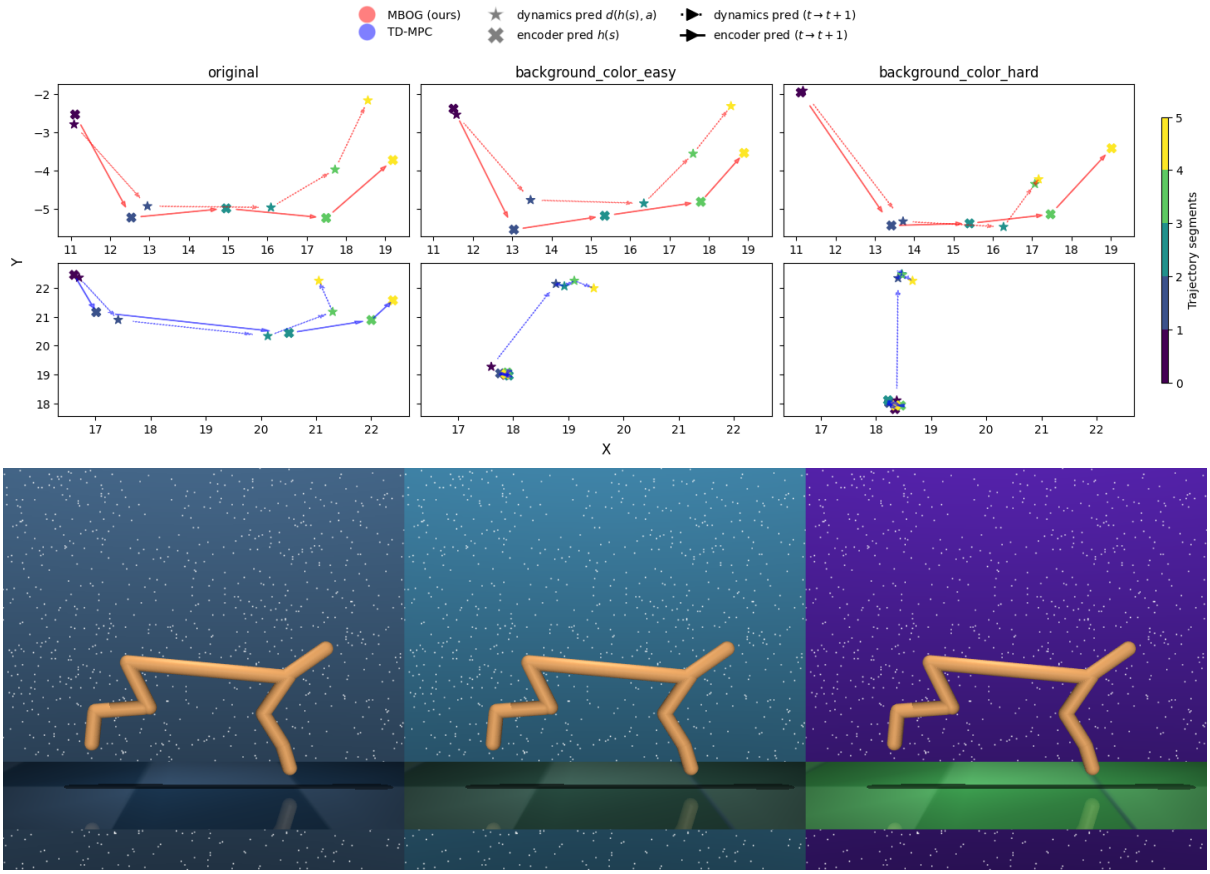


Figure 5.4: **Visualization of embeddings.** We visualize the embedding vectors using UMAP. (Top) MBOG shows consistent representation over the horizon and types of evaluation. (Bottom) Illustration of types used for extracting representation over different types of generalization (*original*, *background_color_easy*, and *background_color_hard* from left-side).

While TD-MPC struggles to output consistent representation over the horizon in evaluation tasks (i.e., *background_color*), MBOG demonstrates unvarying and aligned representation between the latent dynamics prediction $z^*_{t+1} = d_\theta(z_t, a_t)$ and the encoder prediction $z^\times_{t+1} = h_\theta(s_{t+1})$. Furthermore, MBOG manifests similar prediction ability regardless of generalization types (i.e., between *original*, *background_color_easy*, and *background_color_hard*) while TD-MPC predicts far-away representation from the original (trained) distribution, demolishing the accuracy of latent transition dynamics model. See additional discussions in Appendix 7.3.

5.3 Discussion and Future Work

MBOG achieves superior generalization ability to unseen image inputs with similar sample efficiency to TDMPC. By constraining the world model learning with data augmentation and regularization by representation learning, MBOG can achieve a remarkable generalization performance without losing the superior sample efficiency of model-based RL. However, there are a few setbacks for model-based RL to overcome in observational generalization. First, the performance gain of MBOG is still below the state-of-the-art model-free RL in observational generalization. While it is notable that MBOG improves the generalization ability over TD-MPC, previous works (e.g., SVEA [14]) achieve better generalization

performance than MBOG in experiments. One possible direction for future work is to adapt the learned world model during evaluation by few-shot learning (e.g. [45]). Second, the computing cost of planning is prohibitively high. Since the model is used not only for gathering samples during training but also for planning an optimal action during evaluation, reducing the computing cost raised by the model rollout should be the next step to adopting model-based RL into observational generalization in RL. Third, the empirical results are limited to a narrow distribution of tasks. We conduct extensive experiments in DMC and robosuite. While baselines manifest impressive generalization performance in DMC, all methods including MBOG and the state-of-the-art model-free method fail to achieve meaningful results for most tasks in robosuite. We suggest that incorporating further investigations on manipulation agents with visual input into observational generalization would be thrilling future work. We hope that future work will resolve the remaining problems and push the boundaries of model-based RL in observational generalization.

Chapter 6. Conclusion

We propose MBOG, a model-based RL that empirically demonstrates strong generalization ability over unseen image input without sacrificing sample efficiency by employing recipes from model-free RL in observational generalization. By constraining the world model to predict consistent representation with data augmentation and representation learning, MBOG successfully solves the observational generalization problem. We contemplate potential design choices and verify the chosen option outperforms other baselines. We provide extensive results for comparing the performance between model-free and model-based RL. However, there remains room for further improvements in model-based RL with observational generalization. We believe that further transitions from the model-based RL field into the observation generalization realm enable exciting collaborations.

Bibliography

[1] D. Bertoin, A. Zouitine, M. Zouitine, and E. Rachelson. Look where you look! saliency-guided q-networks for generalization in visual reinforcement learning. *Advances in Neural Information Processing Systems*, 35:30693–30706, 2022.

[2] M. Botvinick, S. Ritter, J. X. Wang, Z. Kurth-Nelson, C. Blundell, and D. Hassabis. Reinforcement learning, fast and slow. *Trends in cognitive sciences*, 23:408–422, 2019.

[3] J. Buckman, D. Hafner, G. Tucker, E. Brevdo, and H. Lee. Sample-efficient reinforcement learning with stochastic ensemble value expansion. *Advances in neural information processing systems*, 31, 2018.

[4] M. Deisenroth and C. E. Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472, 2011.

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

[6] I. Goodfellow. Deep learning, 2016.

[7] D. Ha and J. Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[8] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[9] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[10] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[11] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[12] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[13] N. Hansen and X. Wang. Generalization in reinforcement learning by soft data augmentation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13611–13617. IEEE, 2021.

[14] N. Hansen, H. Su, and X. Wang. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. *Advances in neural information processing systems*, 34:3680–3693, 2021.

[15] N. Hansen, X. Wang, and H. Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022.

[16] N. Hansen, H. Su, and X. Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

[17] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[18] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[19] S. Huang, Y. Sun, J. Hu, S. Guo, H. Chen, Y. Chang, L. Sun, and B. Yang. Learning generalizable agents via saliency-guided features decorrelation. *Advances in Neural Information Processing Systems*, 36, 2024.

[20] Y. Huang, P. Peng, Y. Zhao, G. Chen, and Y. Tian. Spectrum random masking for generalization in image-based reinforcement learning. *Advances in Neural Information Processing Systems*, 35: 20393–20406, 2022.

[21] M. Janner, J. Fu, M. Zhang, and S. Levine. When to trust your model: Model-based policy optimization. *Advances in neural information processing systems*, 32, 2019.

[22] T. Kurutach, I. Clavera, Y. Duan, A. Tamar, and P. Abbeel. Model-ensemble trust-region policy optimization. *arXiv preprint arXiv:1802.10592*, 2018.

[23] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas. Reinforcement learning with augmented data. *Advances in neural information processing systems*, 33:19884–19895, 2020.

[24] M. Laskin, A. Srinivas, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International conference on machine learning*, pages 5639–5650. PMLR, 2020.

[25] K. Lee, K. Lee, J. Shin, and H. Lee. Network randomization: A simple technique for generalization in deep reinforcement learning. *arXiv preprint arXiv:1910.05396*, 2019.

[26] T. Lillicrap. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[27] S. Liu, Z. Chen, Y. Liu, Y. Wang, D. Yang, Z. Zhao, Z. Zhou, X. Yi, W. Li, W. Zhang, et al. Improving generalization in visual reinforcement learning via conflict-aware gradient agreement augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23436–23446, 2023.

[28] L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[29] V. Mnih. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[30] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[31] J. Pari, N. M. Shafiullah, S. P. Arunachalam, and L. Pinto. The surprising effectiveness of representation learning for visual imitation. *arXiv preprint arXiv:2112.01511*, 2021.

[32] J. Schulman. Trust region policy optimization. *arXiv preprint arXiv:1502.05477*, 2015.

[33] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[34] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[35] A. Stone, O. Ramirez, K. Konolige, and R. Jonschkowski. The distracting control suite–a challenging benchmark for reinforcement learning from pixels. *arXiv preprint arXiv:2101.02722*, 2021.

[36] A. Stooke, K. Lee, P. Abbeel, and M. Laskin. Decoupling representation learning from reinforcement learning. In *International conference on machine learning*, pages 9870–9879. PMLR, 2021.

[37] R. S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

[38] R. S. Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

[39] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. d. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[40] C. R. Taylor. Dynamic programming and the curses of dimensionality. In *Applications of dynamic programming to agricultural decision problems*, pages 1–10. CRC Press, 2019.

[41] H. Van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[42] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pages 1995–2003. PMLR, 2016.

[43] Z. Wang, Y. Ze, Y. Sun, Z. Yuan, and H. Xu. Generalizable visual reinforcement learning with segment anything model. *arXiv preprint arXiv:2312.17116*, 2023.

[44] G. Williams, A. Aldrich, and E. Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015.

[45] S. Yang, Y. Ze, and H. Xu. Movie: Visual model-based policy adaptation for view generalization. *Advances in Neural Information Processing Systems*, 36, 2024.

[46] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

[47] D. Yarats, I. Kostrikov, and R. Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International conference on learning representations*, 2021.

[48] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *Proceedings of the aaai conference on artificial intelligence*, volume 35, pages 10674–10681, 2021.

[49] Z. Yuan, Z. Xue, B. Yuan, X. Wang, Y. Wu, Y. Gao, and H. Xu. Pre-trained image encoder for generalizable visual reinforcement learning. *Advances in Neural Information Processing Systems*, 35:13022–13037, 2022.

[50] Z. Yuan, S. Yang, P. Hua, C. Chang, K. Hu, and H. Xu. Rl-vigen: A reinforcement learning benchmark for visual generalization. *Advances in Neural Information Processing Systems*, 36, 2024.

[51] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.

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

Chapter 7. Appendix

7.1 Implementation Details

In this section, we describe the implementation details concerning the environments and baselines. We bring overall source code from RL-ViGen [50][1]. We thank the authors of RL-ViGen for providing comprehensive source code.

Environment. We consider 14 evaluation types in DMC and 3 evaluation types in robosuite. In DMC, we account for the *type* and *difficulty* of the tasks for observational generalization performance evaluation. By following [13, 50], we propose a unique set of observational generalization categories:

- *background_color*: Change the background color of the agent (e.g., terrain grid or background sky color).
  - Uniformly sample the parameters of the color, i.e., *(r,g,b)*, from the *easy* and *hard* distribution in Figure 7.1 for difficulties.
- *cam_pos*: Change the position of the tracking camera's focus by randomly adding noise offset.
  - Let the initial position of the tracking camera's focus be $X_{\text{cam}} = (x_i, y_i, z_i)$ in Euclid space.
  - Sample a random offset δ from the uniform distribution with different bounds: $\mathcal{U}(-0.08, 0.08)$ for *easy* and $\mathcal{U}(-0.15, 0.15)$ for *hard* difficulty.
  - Inject the offset to the initial position of the camera; $X_{\text{cam}} = (x_i + \delta, y_i + \delta, z_i + \delta)$.
- *background_video*: Overlay the background with the randomly sampled natural video.
  - Sample a random video with the same width and height as the original image from a set of natural videos [35].
  - Overlay the video only to the background sky for *easy* and to all backgrounds including the ground terrain other than the agent for *hard* difficulty.
- *light_position*: Change the position and orientation of the tracking light of the agent.
  - Following the approach used in [35], the tracking light's coordinate is parameterized as the spherical coordinate; (ϕ, θ, r) where ϕ is azimuth, θ is inclination, and r is the radius of the sphere.
  - Sample ϕ from the normal distribution $\mathcal{N}(\pi/6, 1)$ for *easy* and $\mathcal{N}(\pi/3, 1)$ for *hard* difficulty.
  - Sample $\theta \sim \mathcal{N}(2\pi, 1)$ and transform the initial pose of the tracking light X_{light} to (ϕ, θ, r) where $r = \sqrt{X_{\text{light}}}$.
- *light_color*: Change the color of the tracking light of the agent.
  - Uniformly sample the parameters of the color, i.e., *(r,g,b)*, from the *easy* and *hard* distribution in Figure 7.1 for difficulties.

[1] https://github.com/gemcollector/RL-ViGen

- *moving_light*: Rotate the tracking light of the agent around the agent.
  - Likewise in *light_position*, the spherical coordinate of the tracking light is randomly initialized as (ϕ, θ, r).
  - Let the speed of azimuth rotation as $\Delta_\phi = \pi/200$ for *easy* and $\Delta_\phi = \pi/100$ for *hard* difficulty.
  - Rotate the tracking light counterclockwise at azimuth axis; $(\phi, \theta, r) \leftarrow (\phi, \theta, r) + (\Delta_\phi, 0, 0)$.
- *object_color*: Change the color of the body color of the agent.
  - Uniformly sample the parameters of the color, i.e., *(r,g,b)*, from the *easy* and *hard* distribution in Figure 7.1 for difficulties.

where *easy* and *hard* difficulties exist for 7 types of evaluation. In robosuite, we consider 3 types of evaluation with the franka panda manipulator: *eval-easy*, *eval-hard*, and *eval-extreme*, which is predefined by [50]. We illustrate example images of DMC in Figure 7.2 and robosuite in Figure 7.3.

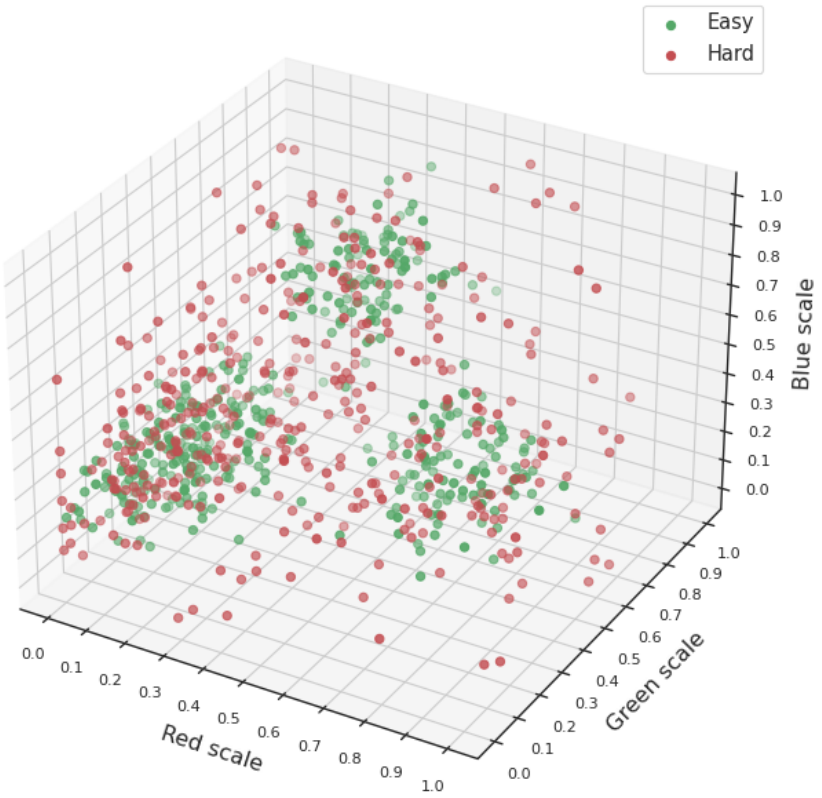


Figure 7.1: **Predefined distribution for evaluation.** We uniformly sample the color parameters from predefined space for *easy* and *hard* difficulties in DMC. Unnormalize the scaled color values for each color RGB space.

Figure 7.2: **Evaluation set in DMC.** We consider 7 types and 2 difficulties for generalization evaluation in DMC. We provide motivating example images in *walker_walk* task.

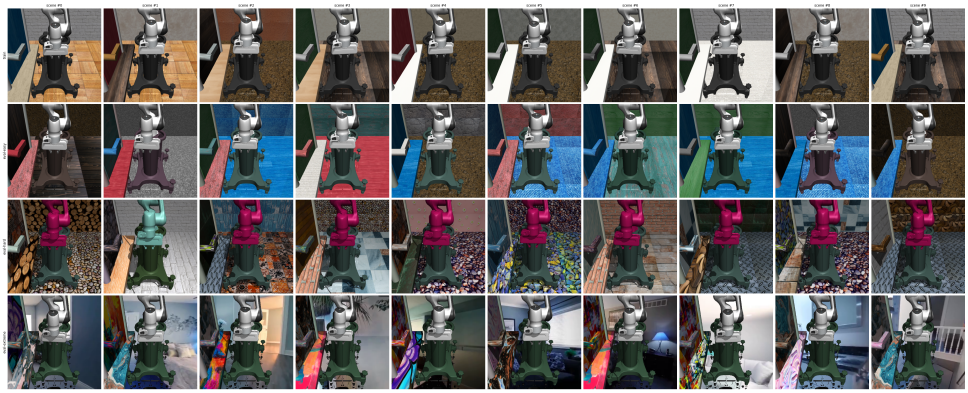


Figure 7.3: **Evaluation set in robosuite.** We consider 3 types of generalization evaluation in robosuite. We provide motivating example images in *Door* task: each row stands for *train*, *eval-easy*, *eval-hard*, and *eval-extreme*; each column corresponds to a different scene in the same type.

Baselines. We consider 4 competitive baselines for model-free (SVEA [14], SGQN [1], SRM [20], and PIEG [49]) and TD-MPC([15]) for model-based algorithms for comparison. We bring the source code of the model-free baselines from the RL-ViGen benchmark since the implementation is straightforward.

We implement MBOG upon the official repository of TD-MPC [15][2]. We intend to maximize the same hyperparameters and task-specific parameters (e.g., action repeat) across experiments. We list common parameters used across tasks in Table 7.1 and algorithm-specific hyperparameters in Table 7.2 and 7.3. We remark that we pursue unified task-specific parameters for reproducible results and concise comparison of baselines.

Table 7.1: **Common hyperparameters.** We list commonly used task-specific parameters for **all** baselines including MBOG across experiments (DMC and robosuite).

Parameter	Value
Discount factor	0.99
Replay buffer size	Unlimited (same with T_g)
Action repeats	2
Frame stack k	3
Pixel RGB image space	$o_t \in \mathcal{O}^{84\times84\times3}$
Maximum episode length	1,000 (DMC), 500 (robosuite)
Batch size	512 ($walker\_\{walk,stand\}$ tasks), 256 (otherwise)
Total gradient steps T_g	1,000,000
Total seeding steps	5,000 (model-based), 4,000 (model-free)
Periodic evaluation steps during training	10,000
Number of episodes per evaluation during training	10
N steps for TD target	1 (model-based), 3 (model-free)
MLP hidden layer dimension	512 (model-based), 1024 (model-free)
Number of CNN convolution filters	32
Latent dimension	50
Target network EMA weight	1e-2

For model-free baselines, we compute N-step TD target for value function learning; $(r_t + r_{t+1} + \cdots + r_{t+N}) + \gamma Q(s_{t+N}, \pi(s_{t+N}))$. The image dimension used for training and evaluation is $84 \times 84 \times 9$: $s_t = \{o_t, o_{t-1}, o_{t-2}\}$ where $o_t \in \mathcal{O}^{84\times84\times3}$. We stack the consequent images along the color channel axis and repeat a specific amount of the same action extracted from the policy or model planning by following prior works.

Table 7.2: **Baseline hyperparameters.** We list algorithm-specific hyperparameters for **model-free** baselines across experiments (DMC and robosuite). We denote that other hyperparameters not described here can be found in Table 7.1.

Hyperparameter	Value
Periodic critic target network (θ^-) update steps	1
Clip constant for the stochastic actor	3e-2
Learning rate for the auxiliary task	3e-4 (SGQN)
Attribution mask quantile	0.95 (SGQN)

[2] https://github.com/nicklashansen/tdmpc

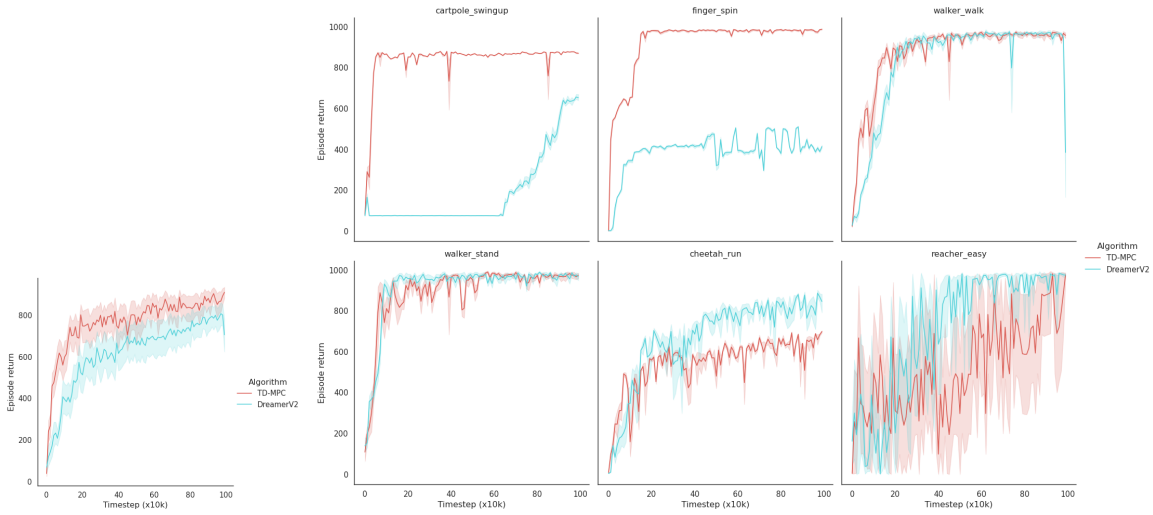
Table 7.3: **MBOG hyperparameters.** We list all hyperparameters of MBOG for completeness while highlighting a subset of hyperparameters only provided to MBOG. We borrow the same hyperparameters of TD-MPC [15].

Hyperparameter	Value
// Replay buffer	
Prioritized experience replay parameters	$\alpha = 0.6, \beta = 0.4$
// MPC planning	
Planning Horizon H	5
Initial parameters for CEM	$\mu_0 = 0, \sigma_0 = 1$
Population size	512
Elite fraction	64
Iterations	8
Policy fraction	5%
Particles	1
Momentum coefficient	0.1
Sampling temperature	0.5
// Model learning	
Temporal coefficient λ	0.5
Reward loss coefficient c_1	0.5
Q value loss coefficient c_2	0.1
Latent consistency loss coefficient c_3	2
// Optimization	
Learning rate η	3e-4
Periodic target network (θ^-) update steps δ	2
Optimizer	Adam($\beta_1 = 0.9$,$\beta_2 = 0.999$)
Exploration schedule (std)	Linear(0.5, 0.05, 25,000 steps)
Planning horizon schedule	Linear(1, 5, 25,000 steps)
// MBOG hyperparameters	
Weak random augmentation Υ^w	*random-shift*: `padding` $p = 4$
Strong random augmentation Υ^s	*random-overlay*: { `linear interpolation` $\delta = 0.5$; `Image dataset` $\mathcal{D} =$ `Places [51]`
Weak and strong augmentation ratio ζ	1.0
Regularization coefficient α	1

7.2 Discussions

Model-based RL. Our main contribution to this paper is to present a model-based RL method that empirically demonstrates strong generalization ability without sacrificing sample efficiency. We have compared two state-of-the-art model-based RL backbone algorithms, TD-MPC [15], and DreamerV2 [11], which exhibit strong sample efficiency over diverse continuous control problems. The reason for not choosing update-to-date versions of each model-based RL method is to reproduce the proper results

with the original paper (e.g., TD-MPC compared its performance with DreamerV2, which was the most recent version of the Dreamer series) and to benefit the architecture itself without further technical considerations since most recent versions of each algorithm, TD-MPC2 [16], and DreamerV3 [12], contain extensive techniques for improving performance. We provide reproduced empirical results to validate that TD-MPC would exhibit better sample efficiency than DreamerV2 in Figure 7.4. Since we aim to achieve sample-efficient RL for observational generalization, we choose TD-MPC as our backbone model-based RL algorithm. To provide a more rigorous comparison of the two models, those two model-based algorithms train and exploit the latent transition model that takes the low-dimensional latent state and action as inputs and outputs the next latent state to solve continuous control tasks with the high-dimensional image input. Dreamer learns the world model by predicting latent future states, actions, and **state-value** functions (V functions) and **reconstructing** the given image by contrastive learning. In contrast, TD-MPC trains the world model by predicting latent future states, actions, and **action-value** functions (Q functions) and **avoids reconstructing** the high-dimensional images. While Dreamer plans the optimal action with the trained world model and **actor policy**, TD-MPC derives the action by planning an optimal action with the **model predictive controller**. Since Dreamer employs representation learning with the reconstruction objective, adapting our method to Dreamer with the reconstructive techniques for observational generalization [1, 43] would be exciting future work.

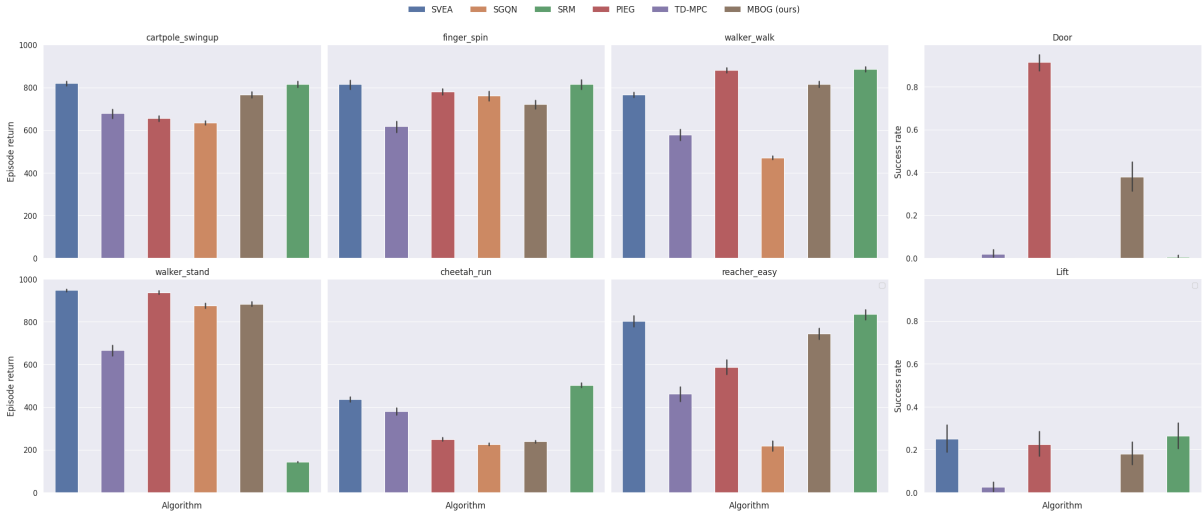


(a) Total comparison result. (b) Comparison over tasks in the DeepMind Control suite.

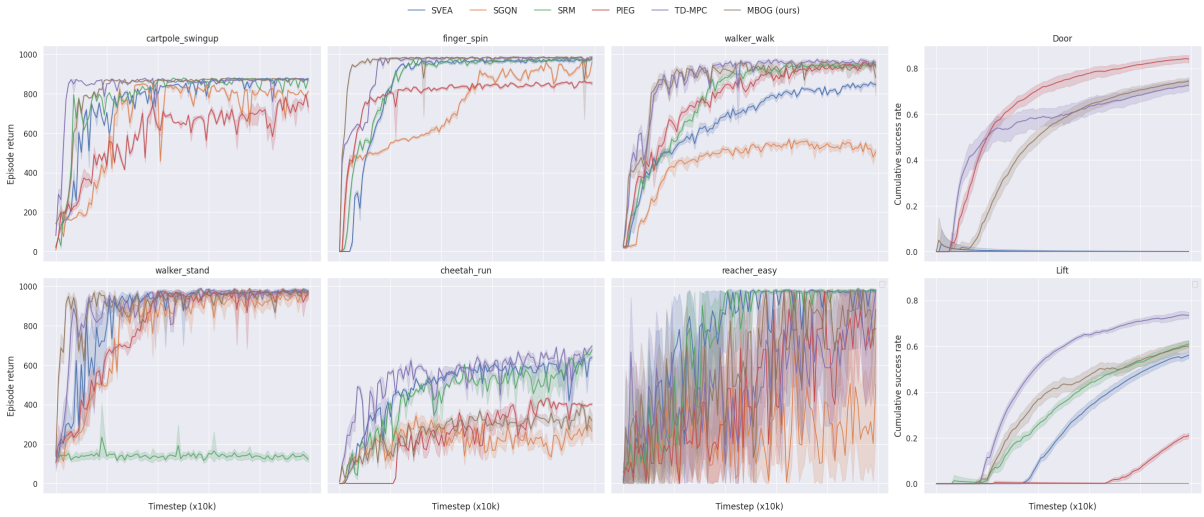
Figure 7.4: **Comparison of sample efficiency between model-based RL methods.** TD-MPC demonstrates superior sample efficiency over DreamerV2 in 6 tasks in the DeepMind Control suite benchmark.

7.3 Additional Results

We provide entire results for extensive experiments here. Concerning the main results, we additionally plot the generalization performance and sample efficiency across the tasks and evaluation types.



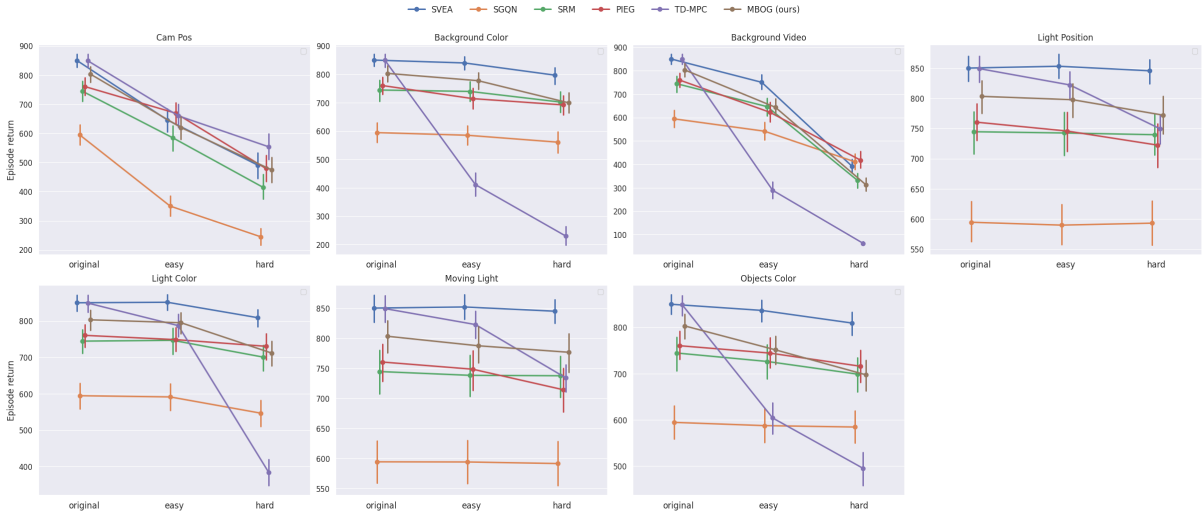
(a) Evaluation results over environments and tasks.



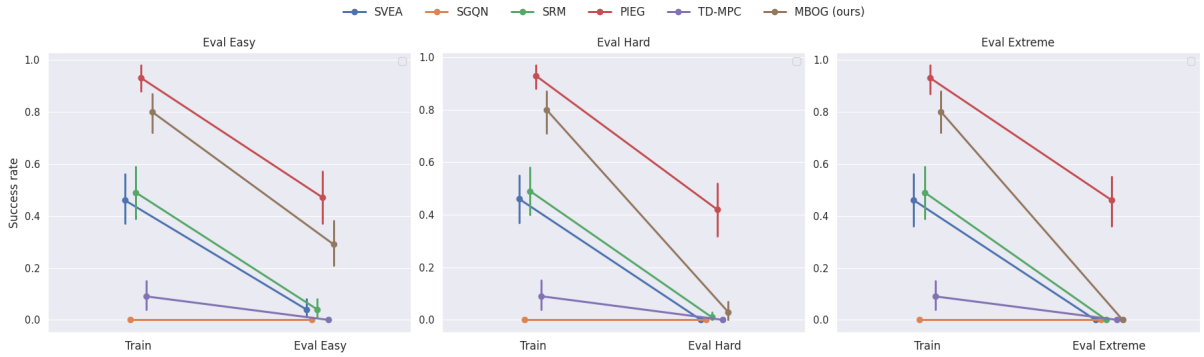
(b) Sample efficiency over environments and tasks.

Figure 7.5: **Additional experimental results over tasks.** Episode returns and cumulative success rates over tasks are reported in DMC and robosuite, respectively. All evaluation results are averaged over 5 seeds and sample efficiency results are averaged over 10 episodes during training.

Overall performance. One can find that PIEG outperforms other baselines at the *Door* task in the robosuite environment. We conjecture that PIEG demonstrates a strong generalization performance at the *Door* task since PIEG exploits the pretrained encoder from ImageNet. ImageNet is a huge dataset that contains diverse images from everyday lives, e.g. door images. Hence, we presume that the representation encoded from the pretrained encoder might distinguish the object well while other baselines should learn the effective representation first. This enables superior sample efficiency and generalization performance for PIEG.



(a) Episode return over evaluation types.



(b) Success rate over evaluation types.

Figure 7.6: **Evaluation results over evaluation types.** We compare the performance of baselines in a total of 14 types. Generalizable RL methods show a monotonic decrease between original and generalization types (e.g., original-easy-hard). All evaluation results are averaged over 5 seeds and sample efficiency results are averaged over 10 episodes during training.

In Figure 7.6, we provide the relative performance of baselines between the training environment and evaluation tasks. While all baselines indicate decreased performance depending on the severity of generalization, TD-MPC demonstrates the steepest decrease among the baselines. These results support the idea that conventional model-based RL suffers from the distribution shift when the trained model is given unseen input during evaluation. As one can see in Figure 7.5 and 7.6, almost every baseline shows moderate generalization performance in DMC while struggling in the robosuite environment. We suggest that current algorithms including state-of-the-art methods for observational generalization have demonstrated limited results to some extent and there is room for further improvements in problem formulation for better generalization.

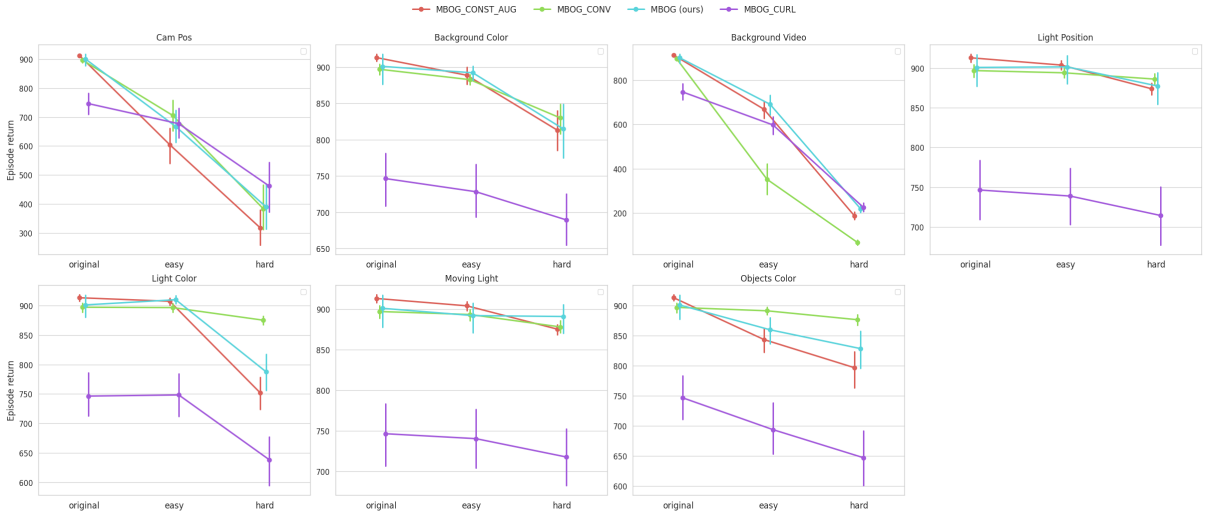


Figure 7.7: **Evaluation results over evaluation types for options.** We compare the performance of our design choices in a total of 14 types. MBOG shows a monotonic decrease between original and generalization types (e.g., original-easy-hard) compared to other options. All evaluation results are averaged over 5 seeds and sample efficiency results are averaged over 10 episodes during training.

Design choices. We provide additional experimental results comparing possible options over environments in Section 5.2. While other options prove competitive performance, `MBOG_CURL` struggles for every generalization category except *cam_pos* task. `MBOG_CURL` outperforms in *cam_pos_hard* task compared to other choices. We conjecture that *cam_pos* task does not disturb the original image with *color*, leading to a trivial decrease in performance of `MBOG_CURL` that does not employ strong augmentation. `MBOG_CONV` shows impressive performance in *light_color* and *object_color* tasks, where the image is mainly distracted with colors. Since `MBOG_CONV` augments the image with random convolution operation during training, `MBOG_CONV` may show relatively strong generalization performance compared to other baselines. Example images comparing the popular strong augmentation techniques can be found in Figure 7.8.

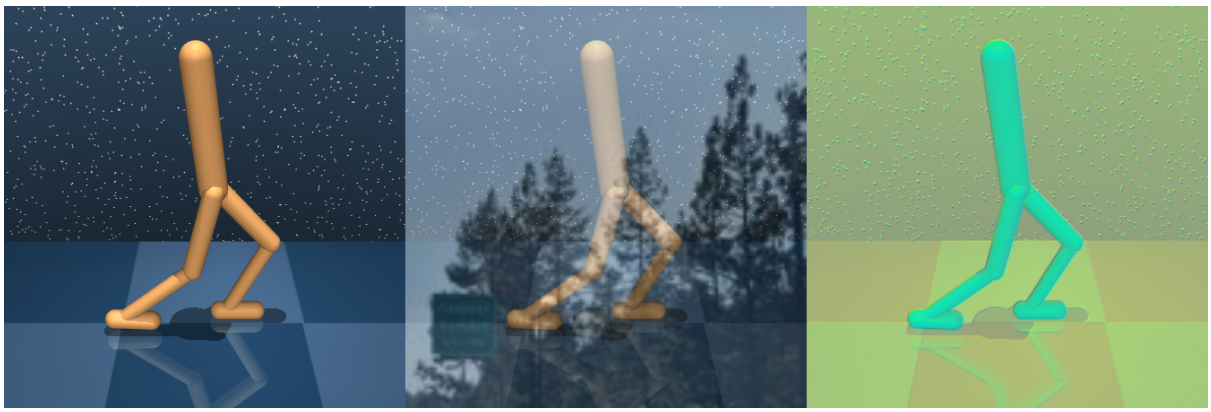


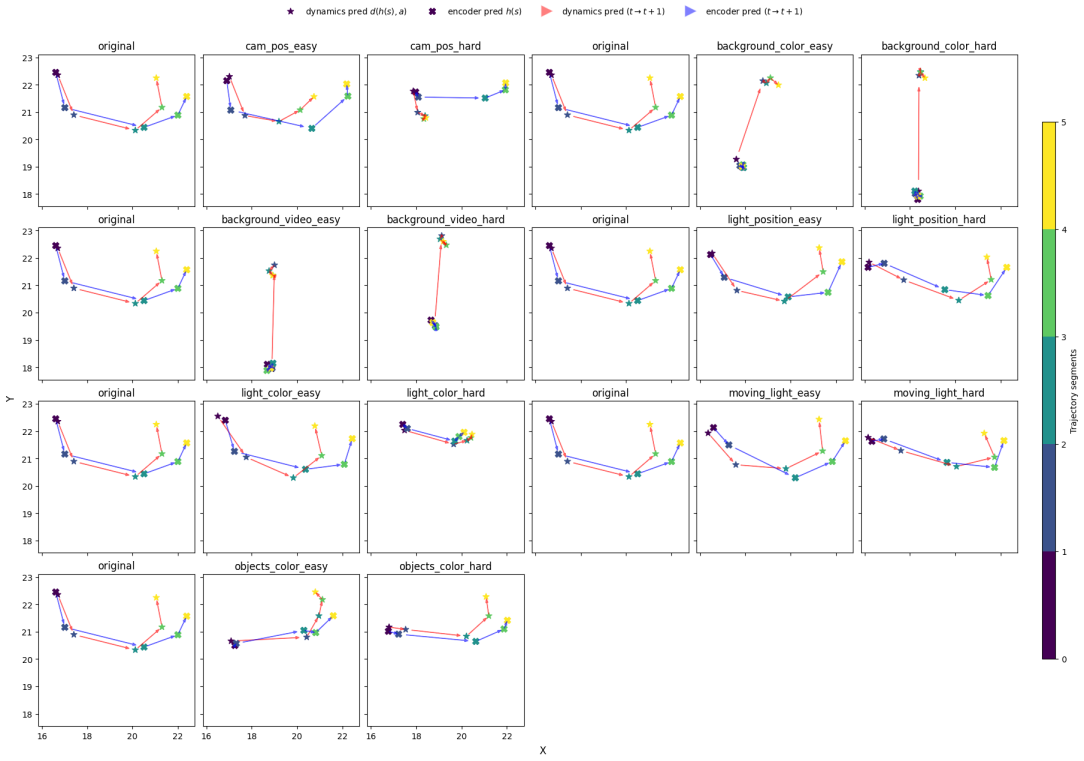
Figure 7.8: **Example images comparing strong augmentations.** Illustrations of how the image is augmented in *walker_walk* task. (Left) Original image, (Center) *random-overlay* augmentation, (Right) *random-conv* augmentation.

Embedding visualization. We provide full plots for embedding experiments. We choose *cheetah_run* for embedding experiments. We collect a set of horizontal images with the trained model across the

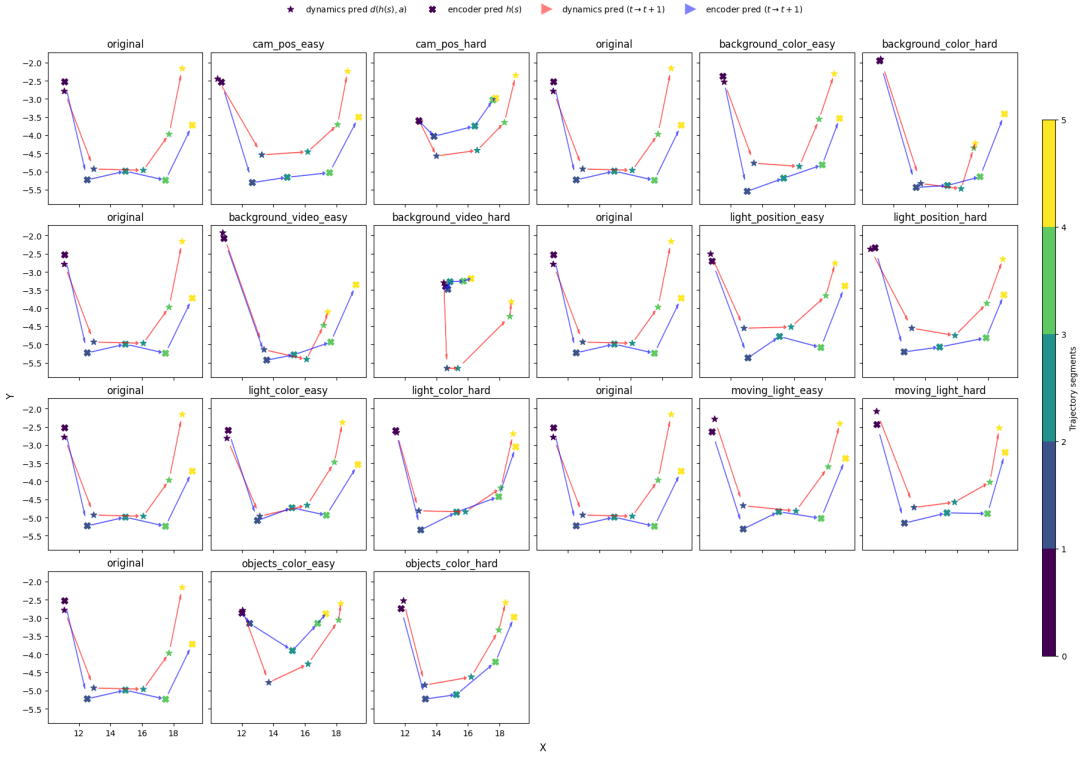
training and evaluation tasks and feed the images to the model to extract representations. For given states $s_{t:t+H}$, we aggregate the horizontal representations from the latent transition model and instantaneous representations from the encoder: horizontal representations $z^{\times}_{k+1} = d_\theta(s_t, a_t) \forall k \in [t, t+H-1]$; instantaneous representations $z^{*}_{k} = h_\theta(s_k) \forall k \in [t, t+H]$. After, we obtain 2D latent coordinates from the aggregated representations using U-MAP [28].

Figure 7.9: **Visualization of types for the embedding experiment.** We plot horizontal images used for extracting representations over the generalization types. Each row stands for a different task and each column stands for the horizontal time step.

In Figure 7.10, the embeddings from TD-MPC show inconsistent representations while MBOG indicates consistent and aligned representation over generalization tasks. Furthermore, MBOG demonstrates accurate predictions of latent transition dynamics models across tasks, supporting the assertion that learning consistent representation plays a significant role in enhancing the generalization performance.



(a) **Visualization of embeddings from TD-MPC.**



(b) **Visualization of embeddings from MBOG.**

Figure 7.10: **Visualization of embeddings.** Star and X markers correspond to z_t^* and $z_t^\times$, respectively. Arrows indicate sequential transitions from each representation. Trajectory segments are $s_{t:t+H}$.