Multi-View Masked World Models for Visual Robotic Manipulation

Younggyo Seo [*1] **Junsu Kim** [*1] **Stephen James** [2] **Kimin Lee** [3] **Jinwoo Shin** [1] **Pieter Abbeel** [4]

Abstract

Visual robotic manipulation research and applications often use multiple cameras, or views, to better perceive the world. How else can we utilize the richness of multi-view data? In this paper, we investigate how to learn good representations with multi-view data and utilize them for visual robotic manipulation. Specifically, we train a multi-view masked autoencoder which reconstructs pixels of randomly masked viewpoints and then learn a world model operating on the representations from the autoencoder. We demonstrate the effectiveness of our method in a range of scenarios, including multi-view control and single-view control with auxiliary cameras for representation learning. We also show that the multi-view masked autoencoder trained with multiple randomized viewpoints enables training a policy with strong viewpoint randomization and transferring the policy to solve real-robot tasks without camera calibration and an adaptation procedure. Video demonstrations are available at: https://sites.google.com/view/mv-mwm.

1. Introduction

The camera is a ubiquitous instrument for robot vision that provides rich information about a workspace from various viewpoints. Thus it has been a widely-used technique for roboticists to utilize multiple cameras for solving complex manipulation tasks (Akkaya et al., 2019; Akinola et al., 2020; James et al., 2022). However, prior work utilize multi-view data naively as inputs and has yet to investigate how to learn effective multi-view representations. Considering recent studies have shown the benefit of single-view representation learning for control (Nair et al., 2022; Radosavovic et al., 2022), it is desirable to explore the potential of multi-view representation learning for visual robotic manipulation.

[*]Equal contribution [1]KAIST [2]Dyson Robot Learning Lab [3]Google Research [4]UC Berkeley. Correspondence to: Younggyo Seo <younggyo.seo@kaist.ac.kr>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

A notable exception is the work of Sermanet et al. (2018), which learns view-invariant representations via contrastive learning. However, enforcing viewpoint invariance assumes that all viewpoints share similar information and thus requires careful curation of positive and negative pairs, similar to other contrastive approaches that often depend on complex design choices about sampling such pairs (Arora et al., 2019). This can make it challenging to learn representations from diverse multi-view data and limit its applicability to a narrow distribution of visual robotic manipulation setups. Instead, we aim to develop a simple multi-view representation learning method effective for more diverse setups.

In this paper, we present Multi-View Masked World Model (MV-MWM), a reinforcement learning framework that trains a multi-view masked autoencoder for representation learning and a world model to solve visual manipulation tasks. Our autoencoder consists of a synergistic combination of *view-masking*: which masks viewpoints at random, and *video autoencoding*: which reconstructs video frames of both masked and unmasked viewpoints. We find our autoencoder effectively learns representations that capture useful information of the current viewpoint but also the cross-view information from different viewpoints. For behavior learning, we learn a world model on frozen representations from either single-view or multi-view data, which is particularly feasible as the autoencoder consists of vision transformer (Dosovitskiy et al., 2021) layers that take inputs of variable sizes. We then train actor and critic with imaginary trajectories from the world model (Hafner et al., 2021).

We highlight the main contributions of this paper below:

- We present Multi-View Masked World Model, a reinforcement learning framework that trains a multi-view masked autoencoder with a view-masking and learns a world model upon autoencoder representations.
- We demonstrate the effectiveness of MV-MWM in various visual robotic manipulation setups. These setups include (i) a *multi-view* control where agents operate on multi-view data, (ii) a *single-view* control where agents operate on single-view data but use auxiliary cameras for representation learning, and (iii) a *viewpoint-robust* control where agents operate on single randomized viewpoint but use multiple randomized viewpoints for representation learning, as illustrated in Figure 1. Our

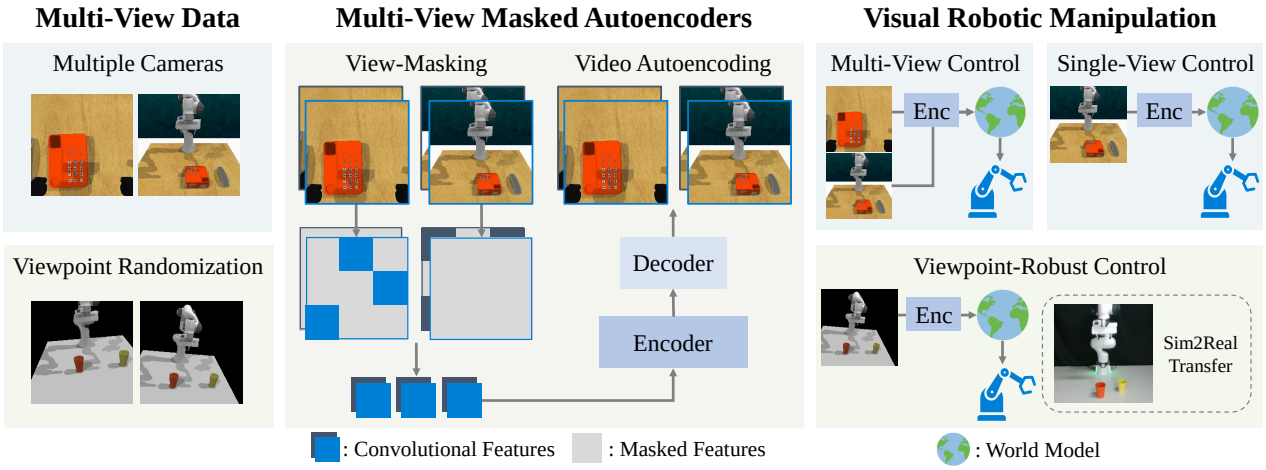


Figure 1. Illustration of our framework. Given multi-view data from multiple cameras or multiple randomized viewpoints, we mask viewpoints from video frames at random and train a multi-view masked autoencoder to reconstruct pixels of both masked and unmasked viewpoints. We then learn a world model upon frozen autoencoder representations to solve tasks from various robotic manipulation setups, including a multi-view control, a single-view control, and a viewpoint-robust control in both simulation and real-world.

experiments on RLBench (James et al., 2020) show that our method outperforms single-view representation learning baselines (Radford et al., 2021; He et al., 2021; Seo et al., 2022a) and a multi-view representation learning baseline (Sermanet et al., 2018).

- We show that MV-MWM can solve real-world robotic manipulation tasks by transferring a policy trained in simulation to a real-robot without camera calibration. We further show that MV-MWM works on a range of viewpoints and even with a hand-held camera subject to rotation or shaking while solving the tasks; showcasing impressive visual servoing robustness.

2. Related Work

Visual control with multiple cameras Leveraging multiple cameras has long been considered a practical and feasible technique in robotics, as the camera is usually an affordable and ubiquitous device (Sola et al., 2008; Carrera et al., 2011; Yang et al., 2021). Based on recent advances in computer vision and robot learning, there have been several approaches that utilize multi-view data from multiple cameras for visual control (Sermanet et al., 2018; Akinola et al., 2020; Zhan et al., 2020; Chen et al., 2021a; Hsu et al., 2022; Jangir et al., 2022; Shridhar et al., 2022; Guhur et al., 2022). While most approaches utilize multi-view data directly as inputs for robots, recent works have demonstrated that self-supervised learning that learns view-invariant representations (Sermanet et al., 2018) or 3D keypoints (Chen et al., 2021a) can be useful for downstream tasks. Yet these approaches assume viewpoints have similar characteristics or require multiple cameras for both representation learning and behavior learning phases, limiting their applicability to a narrow set of setups. Instead this work aims to develop a framework that can learn representations from diverse viewpoints and leverage them for various setups.

Unsupervised representation learning for visual control Most prior researches on representation learning for visual control have focused on solving control tasks using the representations learned with single-view data (Watter et al., 2015; Oord et al., 2018; Gelada et al., 2019; Hafner et al., 2019; Yarats et al., 2021; Seo et al., 2022b). We instead demonstrate that multi-view representations can be also useful for single-view control. Another line of works related to our work have demonstrated that pre-training with self-supervised learning enables agents to solve control tasks with frozen representations (Stooke et al., 2021; Schwarzer et al., 2021; Nair et al., 2022; Parisi et al., 2022; Xiao et al., 2022; Radosavovic et al., 2022). Our framework also learns to solve tasks with frozen representations but it differs in that representations are continually updated using the online samples throughout training. Incorporating pre-training into our framework would be an interesting future direction.

3. Multi-View Masked World Models for Visual Robotic Manipulation

We present Multi-View Masked World Models (MV-MWM), a reinforcement learning framework that learns multi-view representations and utilize them for visual robotic manipulation. Our method builds on top of the Masked World Models (MWM; Seo et al. 2022a) framework, which learns a world model on frozen masked autoencoder features. We first introduce how to learn multi-view representations in Section 3.1. We then describe in Section 3.2 how to utilize

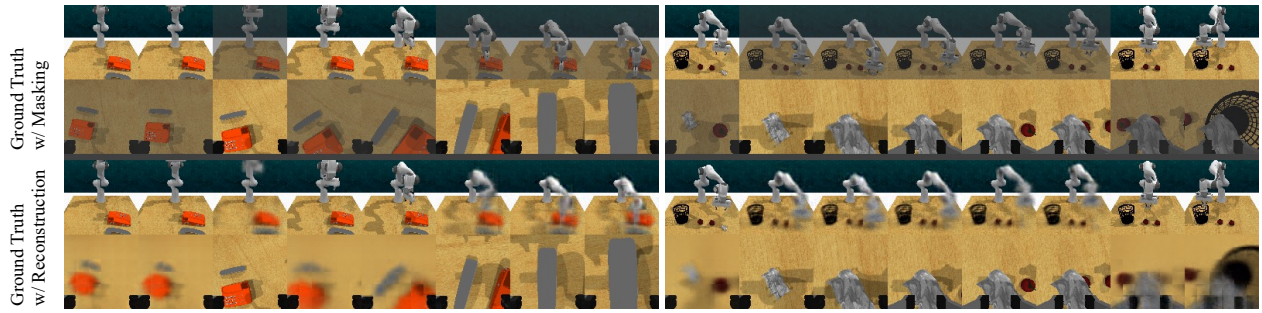


Figure 2. Masked view reconstruction on Phone On Base (left) and Put Rubbish in Bin (right) tasks from RLBench (James et al., 2020). We visualize ground-truth frames with masked viewpoints (upper two rows) and ground-truth frames with reconstructed frames (lower two rows). We find that the model can reconstruct masked viewpoints, successfully capturing the location of objects in the scene.

them for learning world models and behaviors to solve visual manipulation tasks. We provide the overview of our framework in Figure 1. The key difference to MWM is detailed in Appendix B and Algorithm 1.

3.1. Multi-View Representation Learning

For representation learning with multi-view data, we introduce a new model called Multi-View Masked Autoencoder (MV-MAE). Our main idea is to train a *video masked autoencoder* (Feichtenhofer et al., 2022; Tong et al., 2022) with *view-masking* to reconstruct missing pixels of randomly masked viewpoints. We also incorporate the idea of a prior work (Seo et al., 2022a) that masks convolutional features instead of pixel patches (He et al., 2021) and predicts rewards to learn representations capturing fine-grained details required for visual control. We first describe each component in detail and provide the formal objective.

Convolutional feature embedding Unlike prior work that masks random pixel patches (He et al., 2021), we embed camera observations into convolutional feature maps and mask these features following the design of Seo et al. (2022a). This is based on the observation where masked image modeling with pixel patch masking can make it difficult for the model to learn fine-grained details within patches. Specifically, we downsample $96 \times 96 \times 3$ inputs images to convolutional feature maps with the spatial size of 6×6 by introducing 4 convolutional layers. We separately process observations from each viewpoint with convolutional layers that share parameters. For each viewpoint, we add fixed 2D sin-cos position embeddings (Chen et al., 2021b) to the features. We also add learnable 1D parameters representing each viewpoint and timestep to features of each video frame from different viewpoints, following Geng et al. (2022) that introduces parameters for vision and language inputs.

View masking To learn cross-view information from multiple viewpoints, we introduce a novel view-masking strategy that masks all the features from a randomly selected viewpoint. Specifically, we mask randomly selected viewpoints from video frames by randomly sampling one viewpoint for each frame. We also mask randomly selected features from remaining viewpoints (see Figure 1) because we want the autoencoder to learn not only cross-view information but also the information within each viewpoint by reconstructing raw visual observations with masked features. Then we flatten the unmasked features and concatenate them into a single sequence. We empirically find that the proposed view-masking can be more effective than uniform-masking scheme by explicitly encouraging multi-view representation learning (see Figure 7(a) for supporting experiments).

Video autoencoding Despite its potential, the proposed masked view reconstruction objective might be too challenging for the autoencoder without any access to information from missing viewpoints. To address this issue, we consider a combination of video masked autoencoding (Feichtenhofer et al., 2022; Tong et al., 2022) and the proposed view-masking strategy. Because the autoencoder attends to unmasked neighbor frames from the same view, the model can focus on modeling important information such as the movement of robot arms, while ignoring redundant information such as background for reconstructing masked viewpoints (see Figure 2 for examples of masked view reconstruction). Specifically, our encoder processes a sequence of unmasked features from all viewpoints and video frames through a series of vision transformer (ViT; Dosovitskiy et al. 2021) layers. Then we concatenate a set of mask tokens with encoded features, and add learnable parameters for each viewpoint and frame to corresponding features and mask tokens. The decoder processes them through ViT layers and linearly projects them into pixel patch predictions. We also follow the idea of Seo et al. (2022a) that predicts a reward to encode task-relevant information.

Objective Let $o^v_{t,T} = \{o^v_t, ..., o^v_{t+T-1}\}$ be a video from a viewpoint $v \in \mathcal{V}$ where t is current timestep, T is window

size of the video autoencoder, and $\mathcal{V}$ is a set of available viewpoints. Given videos $\{o^v_{t,T}\}_{v\in\mathcal{V}}$ from multiple viewpoints, rewards $r_{t,T} = \{r_1, ..., r_{t+T-1}\}$, and a mask ratio of m, MV-MAE consists of following components:

$$\begin{aligned}
&\text{Convolution stem:} && h^v_{t,T} = f^{\texttt{conv}}_\phi(o^v_{t,T}) \\
&\text{View-masking:} && h^m_{t,T} \sim p^{\texttt{mask}}(h^m_{t,T} \,|\, \{h^v_{t,T}\}_{v\in\mathcal{V}}, m) \\
&\text{ViT encoder:} && z^m_{t,T} \sim p_\phi(z^m_{t,T} \,|\, h^m_{t,T}) \\
&\text{ViT decoder:} && \begin{cases} \{\hat{o}^v_{t,T}\}_{v\in\mathcal{V}} \sim p_\phi(\{\hat{o}^v_{t,T}\}_{v\in\mathcal{V}} \,|\, z^m_{t,T}) \\ \hat{r}_{t,T} \sim p_\phi(\hat{r}_{t,T} \,|\, z^m_{t,T}) \end{cases}
\end{aligned} \tag{1}$$

We train the model to reconstruct pixels and predict rewards, which corresponds to optimizing model parameters ϕ by minimizing the negative log-likelihood as below:

$$\mathcal{L}^{\texttt{mvmae}}(\phi) = -\ln p_\phi(\{o^v_{t,T}\}_{v\in\mathcal{V}} \,|\, z^m_{t,T}) - \ln p_\phi(r_{t,T} \,|\, z^m_{t,T})$$

3.2. World Model and Behavior Learning

Representations Once we learn multi-view representations, we leverage them for learning a world model and utilize the world model for visual control. A favorable property of MV-MAE is that the ViT encoder can extract representations from single-view images even if the encoder is trained with multi-view video data. Based on this property, we learn a world model for solving visual robotic manipulation tasks from two setups: *multi-view* and *single-view* control. These setups vary in the availability of viewpoints during the control phase, yet both adopt a multi-view data for representation learning. To provide input to the world model, we extract representations by encoding multi-view data $\{o^v_t\}_{v\in\mathcal{V}}$ or single-view data $o^{\tilde{v}}_t$ by using the MV-MAE encoder in Equation 1 with $m = 0$ and $T = 1$.[1] We exploit the notation by denoting both representations as z_t because the objective for learning the world model is same for both single-view and multi-view world models.

World model learning Following Seo et al. (2022a), we implement the world model as a variant of recurrent state-space model (RSSM; Hafner et al. 2019) that takes frozen autoencoder representations as inputs and reconstruction targets. The world model consists of following components:

$$\begin{aligned}
&\text{Encoder:} && s_t \sim q_\theta(s_t \,|\, s_{t-1}, a_{t-1}, z_t) \\
&\text{Decoder:} && \begin{cases} \hat{z}_t \sim p_\theta(\hat{z}_t \,|\, s_t) \\ \hat{r}_t \sim p_\theta(\hat{r}_t \,|\, s_t) \end{cases} \\
&\text{Dynamics model:} && \hat{s}_t \sim p_\theta(\hat{s}_t \,|\, s_{t-1}, a_{t-1})
\end{aligned} \tag{2}$$

The encoder extracts state s_t from previous state s_{t-1}, previous action a_{t-1}, and current autoencoder representations z_t. The dynamics model learns to predict s_t without an access to z_t, which enables the model to predict forward into the future. Following Hafner et al. (2021), we utilize discrete latents for s_t. The decoder reconstructs z_t to provide learning signal for model states and predicts r_t to allow for computing rewards from future states without decoding future autoencoder representations. All model parameters θ are jointly optimized by minimizing the negative variational lower bound (Kingma & Welling, 2014):

$$\begin{aligned}
\mathcal{L}^{\texttt{wm}}(\theta) = &-\ln p_\theta(z_t \,|\, s_t) - \ln p_\theta(r_t \,|\, s_t) \\
&+ \beta\, \mathrm{KL}\big[q_\theta(s_t|s_{t-1}, a_{t-1}, z_t) \,\|\, p_\theta(\hat{s}_t|s_{t-1}, a_{t-1})\big],
\end{aligned}$$

where β is a scale hyperparameter.

Behavior learning For behavior learning, we employ the actor-critic learning scheme of DreamerV2 (Hafner et al., 2021) where the objective is to train a policy that maximizes the predicted future values by gradients propagated through the world model. Specifically, a stochastic actor and a deterministic critic is defined as below:

$$\begin{aligned}
&\text{Actor:} && \hat{a}_t \sim p_\psi(\hat{a}_t \,|\, \hat{s}_t) \\
&\text{Critic:} && v_\xi(\hat{s}_t) \approx \mathbb{E}_{p_\theta}\left[\textstyle\sum_{i\le t} \gamma^{i-t} \hat{r}_i\right]
\end{aligned} \tag{3}$$

where $\{(\hat{s}_t, \hat{a}_t, \hat{r}_t)\}^H_{t=1}$ is predicted from initial state $\hat{s}_0$ using the stochastic actor and dynamics model in Equation 2. Given a λ-return (Schulman et al., 2015) defined as below:

$$V^\lambda_t \doteq \hat{r}_t + \gamma \begin{cases} (1-\lambda) v_\xi(\hat{s}_{t+1}) + \lambda V^\lambda_{t+1} & \text{if } t < H \\ v_\xi(\hat{s}_H) & \text{if } t = H \end{cases} \tag{4}$$

Algorithm 1 Multi-View Masked World Models.
Key differences to MWM (Seo et al., 2022a) in gray.

Initialize parameters ϕ, θ, ψ, ξ
Initialize a buffer $\mathcal{B}$ and a fixed expert buffer $\mathcal{B}^{\texttt{e}}$
for each timestep t **do**
  // COLLECT TRANSITIONS
  Update state $s_t \sim q_\theta(s_t|s_{t-1}, a_{t-1}, z_t)$
  Sample action $a_t \sim p_\psi(a_t|s_t)$
  Add transition to replay buffer $\mathcal{B}$
  // MULTI-VIEW REPRESENTATION LEARNING
  Sample $(\{o^v_{j,T}\}_{v\in\mathcal{V}}, r_{j,T}) \sim \mathcal{B}$
  Update ϕ by minimizing $\mathcal{L}^{\texttt{mvmae}}(\phi)$
  // WORLD MODEL LEARNING
  Sample $(\{o^v_j\}_{v\in\mathcal{V}}, a_{j-1}, r_j) \sim \mathcal{B}$
  Obtain z_j from either $\{o^v_j\}_{v\in\mathcal{V}}$ (multi-view control) or $o^{\tilde{v}}_j$ (single-view control with $\tilde{v}$)
  Update θ by minimizing $\mathcal{L}^{\texttt{wm}}(\theta)$
  // BEHAVIOR LEARNING
  Obtain initial state $\hat{s}_0$ and imagine $\{(\hat{s}_i, \hat{a}_i, \hat{r}_i)\}^H_{i=1}$
  Sample expert demonstration $(\{o^v_j\}_{v\in\mathcal{V}}, a^{\texttt{e}}_j) \sim \mathcal{B}^{\texttt{e}}$
  Update ξ by minimizing $\mathcal{L}^{\texttt{critic}}(\xi)$
  Update ψ by minimizing $\mathcal{L}^{\texttt{actor}}(\psi)$
end for

[1] We extract image representations as our world model operates upon observations from each timestep with a recurrent architecture.

the critic is trained to regress the λ-return and the actor is trained to maximize the λ-return with gradients backpropagated through the world model. Moreover, to better utilize expert demonstrations which we assume the access to by following the setup of James & Davison (2022), we introduce a behavior cloning loss that encourages the actor to imitate expert actions. Objectives are summarized as below:

$$\mathcal{L}^{\texttt{critic}}(\xi) \doteq \mathbb{E}_{p_\theta, p_\psi}\left[\sum_{t=1}^{H-1} \frac{1}{2}\left(v_\xi(\hat{s}_t) - \text{sg}(V_t^\lambda)\right)^2\right]$$

$$\mathcal{L}^{\texttt{actor}}(\psi) \doteq \mathbb{E}_{p_\theta, p_\psi}\left[-V_t^\lambda - \eta \,\text{H}\left[a_t|\hat{s}_t\right]\right] - \ln p_\psi(a_t^{\texttt{e}}|s_t)$$

where sg is a stop gradient operation, η is a scale hyperparameter for an entropy $\text{H}\left[a_t|\hat{s}_t\right]$, and $a_t^{\texttt{e}}$ is an expert action.

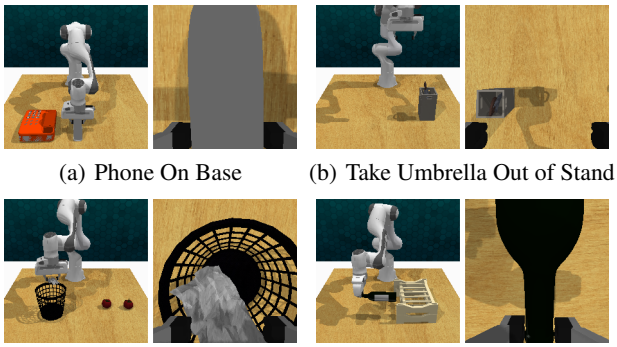
To summarize, we iterate the processes of (i) training the *video autoencoder* with *view-masking* for multi-view representation learning, (ii) learning the world model and behaviors for *multi-view* or *single-view* control, (iii) collecting samples with environment interaction. We highlight the key differences between MWM and MV-MWM in Algorithm 1.

4. Experiments

We evaluate MV-MWM on challenging visual robotic manipulation tasks from RLBench (James et al., 2020) – a standard benchmark for vision-based robotics which has been shown to serve as a proxy for real-robot experiments (James & Davison, 2022). Furthermore, we evaluate the zero-shot performance of our method on real-world by transferring the trained agent to control real-robots. We designed our experiments to explore the benefit of multi-view representation learning on practical and important robotics scenarios. Specifically, we aim to investigate the following questions:

- Can MV-MWM learn multi-view representations useful for various visual robotic manipulation setups?
- Can MV-MWM trained in simulation be transferred to solve real-world visual robotic manipulation tasks?
- How does MV-MWM compare to baselines in terms of sample-efficiency and asymptotic performance?
- What is the effect of each component in MV-MWM?

Implementation We build our implementation upon the official implementation of MWM (Seo et al., 2022a) and implementation details are same unless otherwise specified. We run 8 parallel simulators to accelerate training by avoiding the bottleneck from a slow simulator. Our autoencoder consists of the 8-layer ViT encoder and 6-layer ViT decoder, where the embedding dimension is set to 256. We use the same set of hyperparameters for all experiments. We provide more implementation details in Appendix A.



(a) Phone On Base (b) Take Umbrella Out of Stand

(c) Put Rubbish in Bin (d) Stack Wine

Figure 3. Examples of multi-view data consisting of front and wrist camera observations used in our experiments. Front camera observations provide a broad look at a robot workspace and wrist camera observations provide a closer look at target objects.

Environment For all experiments, we use only 96×96 RGB observations from each camera; proprioceptive state and depth are not used. While RLBench is originally designed to evaluate the performance in a sparse reward setup, we design dense rewards for our experiments. Moreover, to ease the difficulty of exploration in large action space, we enforce a robot gripper to be in an upright position except for a case where rotation is required for solving the task. Following James & Davison (2022), we fill a replay buffer with expert demonstrations. Unlike prior approaches that utilize path planner with the policy to output next best gripper pose (James & Davison, 2022; James et al., 2022; Shridhar et al., 2022; James & Abbeel, 2022a;b), our RL agent outputs relative change in gripper position. We provide further details in Appendix A.

Baselines We first compare MV-MWM with MWM to evaluate the benefit of multi-view representation learning. We note that both use the same amount of training data. We also consider baselines that utilize frozen pre-trained representations. Specifically, we consider CLIP (Radford et al., 2021) and MAE (He et al., 2021) representations as they have recently shown to be effective for robotic manipulation (Shridhar et al., 2021; Radosavovic et al., 2022). Moreover, to compare our method with other multi-view representation learning method designed for visual control, we consider time contrastive network (TCN; Sermanet et al. (2018)) that enforces view-invariance through contrastive learning as a baseline. We call these baselines as CLIP+WM, MAE+WM, and TCN+WM to denote that we learn world models upon these representations. All methods use frozen representations for world model and behavior learning. We note that MV-MWM, MWM, and TCN+WM learn representations from scratch throughout training, but we do not fine-tune the representations of CLIP+WM and MAE+WM for a comparison with frozen pre-trained representations. We provide

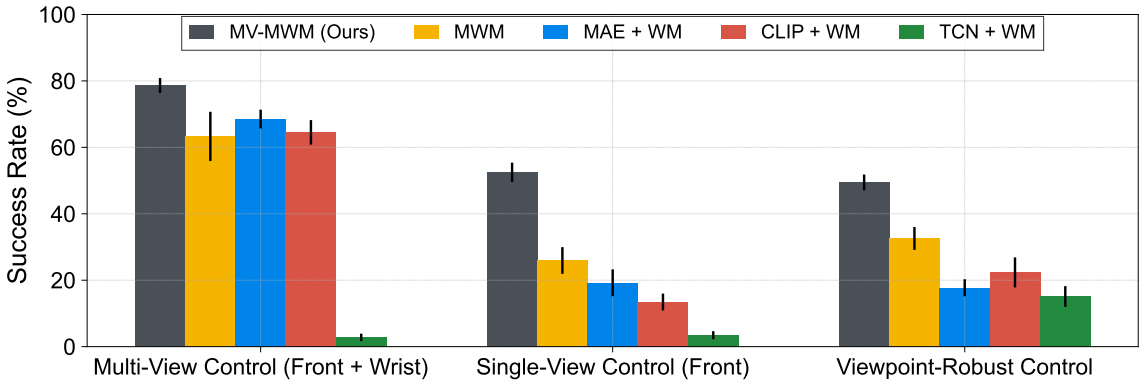


Figure 4. Aggregate success rate on five multi-view and single-view control tasks and two viewpoint-robust control tasks. We report the average success rate evaluated using the last five model checkpoints. The result shows the mean and stratified bootstrap confidence interval across 32 runs for viewpoint-robust control and 20 runs otherwise. We provide the learning curve for all tasks in Appendix C.

more details on baselines in Appendix B.

4.1. Multi-View and Single-View Control

Multi-view control setup We evaluate MV-MWM on a multi-view control setup where the agent operates on both front and wrist cameras, which is a widely-used setup for visual manipulation with multiple cameras (Zhan et al., 2020; James & Davison, 2022; Jangir et al., 2022). For all baselines, we extract representations from each viewpoint and use concatenated features as inputs to the agent. For our method, we use multi-view representations from our autoencoder as we previously mentioned in Section 3.2.

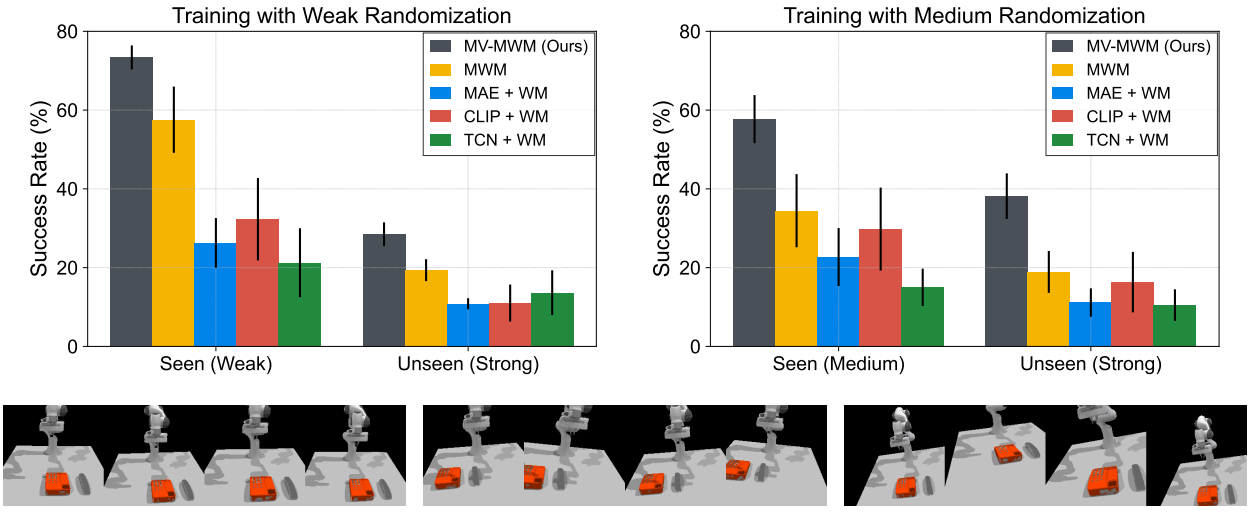
Single-view control setup We also consider a single-view control setup to investigate whether multi-view representation learning with auxiliary cameras can be helpful for training a single-view agent. This can be useful for a practical scenario where we can utilize additional cameras during training, but the robot should operate on a single camera at deployment time. We note that this setup has also been investigated in Sermanet et al. (2018), but we use more different types of cameras for representation learning. Specifically, we learn visual representations using multi-view data consisting of front and wrist camera observations and train the RL agent that operates on the front camera.

Results In Figure 4, we observe that MV-MWM outperforms MWM, which shows that multi-view representation learning can be helpful for both multi-view and single-view control. We find that TCN significantly fails to solve most of the tasks in both setups. This shows the critical drawback of TCN, which suffers from mode collapse when negative samples are too similar (*e.g.,* wrist camera observations look similar after grasping the objects in our case). Interestingly, we also find that MAE+WM and CLIP+WM achieve non-zero success rates, which shows that large pre-trained models can extract useful representations. However, MV-MWM largely outperforms both baselines, demonstrating that multi-view representation learning with in-domain data can be crucial for visual robotic manipulation. Given the results, it would be an interesting direction to integrate pre-training with our multi-view representation learning by employing recently developed efficient fine-tuning techniques (Gao et al., 2021; Zhang et al., 2021; Sharma et al., 2023).

4.2. Viewpoint-Robust Control

Problem setup We consider a viewpoint-robust control setup where we learn a policy robust to camera perturbations, which can enable us to deploy robots to real-world without a tedious camera calibration (see Section 4.3 for our real-robot experiments). However, we observe that learning to solve tasks under hard viewpoint randomization is a very challenging problem. To address this, we propose to utilize multi-view representation learning by generating multiple randomized viewpoints and learn representations with them (see Figure 1 for illustration). Then we train a single-view agent on randomized viewpoints upon these representations to solve tasks with randomized cameras.

Experimental setup We randomize the position and orientation of the front camera at every episode. We construct three randomization type, which represents how strongly viewpoints are randomized: *weak*, *medium*, and *strong* as exemplified in Figure 5. For faster experimentation, we modify the tasks to be more easier by making a target object be located upright (*i.e.,* not rotated related to the table). Because we aim to evaluate viewpoint-robust control agents to solve real-world robotic tasks, we also modify the colors of a simulated workspace to be similar to colors of a real-world setup and apply brightness and contrast augmentation to videos throughout training. For evaluation, we train all agents on weak and medium randomization setups



(a) Weak randomization (b) Medium randomization (c) Strong randomization

Figure 5. Aggregate success rate on two viewpoint-robust control tasks under both seen and unseen viewpoint randomization setups exemplified in the second row. We report the average success rate evaluated using the last five model checkpoints. The result shows the mean and stratified bootstrap confidence interval across 8 runs. We provide the learning curve for all tasks in Appendix C.

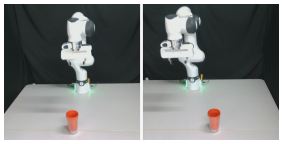
and report the performance under both seen (*i.e.,* weak and medium) and unseen (*i.e.,* strong) randomization setups.

Results Figure 5 shows the performance of visual control agents on both seen and unseen randomization setups. We first observe that our single-view baseline, MWM, can achieve competitive training performance on weak randomization setup. This aligns with the observation of Sadeghi et al. (2018) that shows a recurrent policy can be robust to randomized viewpoints. However, we observe that the performance of MWM significantly degrades as randomization gets stronger, implying that the recurrent architecture alone is not enough for viewpoint-robust control. We find that other baselines also struggle on medium randomization setup, failing to improve their generalization performance on unseen randomization setup. On the other hand, MV-MWM learns to solve the tasks under medium randomization and outperforms all baselines on both seen and unseen setups (see Figure 4 for aggregate performance). We also find that TCN + WM trained on randomized viewpoints achieves non-zero success rates, in contrast to results with the front and wrist cameras in Section 4.1. We hypothesize this is because contrastive learning becomes easier when using similar viewpoints. Nonetheless, MV-MWM largely outperforms TCN + WM, which highlights the benefit of our simple yet effective masked view reconstruction objective.

4.3. Real-Robot Manipulation via Sim-to-Real Transfer

Setup We also evaluate the zero-shot performance of agents trained in simulation for solving real-world manipulation tasks. We deploy the agents trained under medium randomization to solve the Pick Up Cup task without camera calibration and adaptation procedures. We conduct experiments with the Franka Research 3 and use MoveIt2 library based on ROS 2 framework for controlling the arm. We use RGB observations from RealSense D435f camera.

Table 1. Zero-shot sim-to-real transfer performance of viewpoint-robust control agents trained in simulation for solving a real-world visual robotic manipulation task.



Method	Success rate
MAE + WM	7.1%
MWM	11.3%
MV-MWM	**74.7%**

Results For evaluation, we measure the success rate across 3 viewpoints and 20 randomized cup positions for each viewpoint. Table 1 shows the real-world performance of MV-MWM and two baselines we selected for their overall similarity to our method. We find that MV-MWM can solve the task without camera calibration and largely outperforms all baselines. We further evaluate MV-MWM on an extreme setup where we use a hand-held camera which is subject to shake, rotation, and translation while solving the task. We note that viewpoint is not randomized within the episode for training our world model, which makes the setup more challenging. Surprisingly, Figure 6 shows that MV-MWM can solve the task on this challenging setup only using RGB observations, which showcases the effectiveness of our method for real-world visual robotic manipulation. Video demonstrations are available at our webpage: https://sites.google.com/view/mv-mwm.

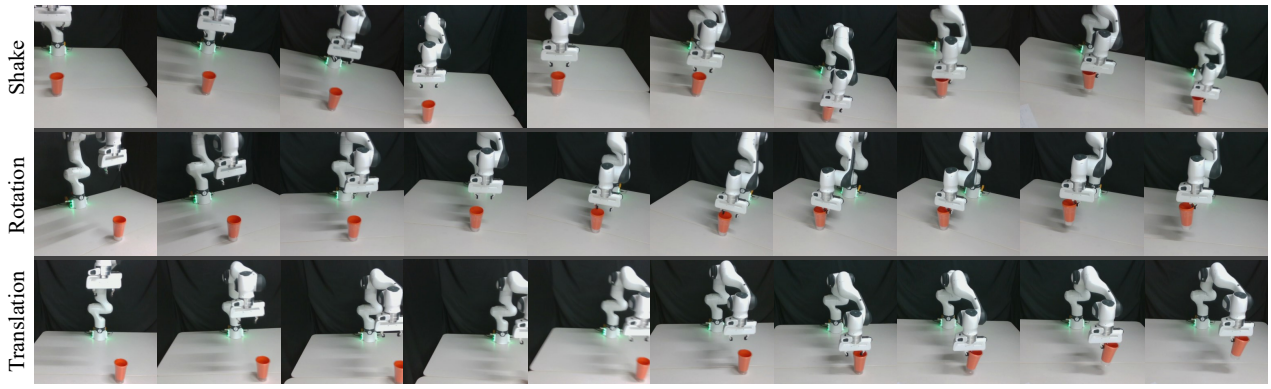
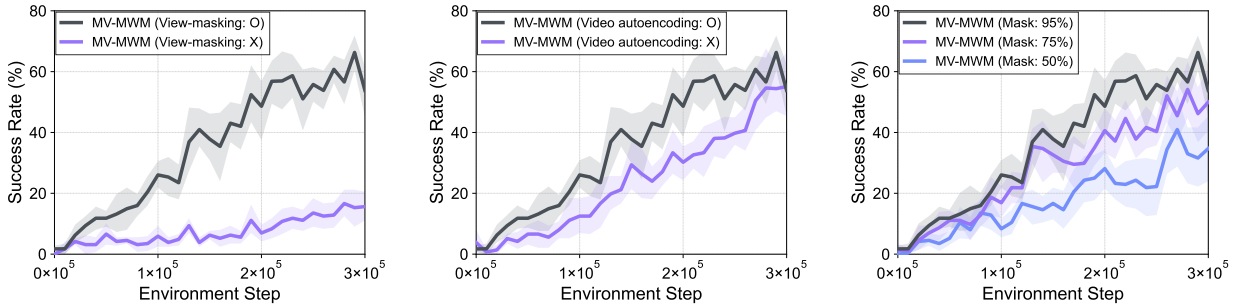


Figure 6. We study the robustness of MV-MWM to camera perturbations in a real-world by considering an extreme setup where the agent operates on a hand-held camera subject to shake, rotation, and translation. We observe that MV-MWM can solve the task using only RGB observations without proprioceptive states and any adaptation procedure. Best viewed as videos provided in the webpage: https://sites.google.com/view/mv-mwm.



(a) View-masking (b) Video autoencoding (c) Masking ratio

Figure 7. Learning curves of single-view visual control agents operating on the front camera for solving three manipulation tasks from RLBench (James et al., 2020), investigating the effect of (a) view masking, (b) video autoencoding and (c) masking ratio. The solid line and shaded regions represent the mean and stratified bootstrap confidence interval across 12 runs.

4.4. Ablation Study and Analysis

Effect of view-masking We evaluate the performance of MV-MWM with and without the proposed view-masking scheme in Figure 7(a). Specifically, we consider a baseline trained with the uniform-masking scheme that mask random features from multi-view inputs, *i.e.,* MV-MWM (View-masking: X). We observe that performance largely degrades without view-masking, which shows that view-masking is crucial for multi-view representation learning.

Effect of video autoencoding Figure 7(b) shows that sample-efficiency of our method significantly improves with video autoencoding. This shows that enabling the model to have access to unmasked frames of the same view ease the difficulty of masked view reconstruction, making the combination of view-masking and video autoencoding synergistic.

Masking ratio Figure 7(c) shows that MV-MWM performance keeps increasing with a higher masking ratio. We hypothesize this is because spatial information redundancy (He et al., 2021) is more significant in visual observations from manipulation tasks than natural images. This also aligns with the observation of prior work (Tong et al., 2022) where 90% masking ratio has shown to be effective for videos.

Scaling property In Figure 17(a) and Figure 17(b) available at Appendix D, we also investigate whether MV-MWM can be further scaled up for better performance by improving the number of gradient steps and increasing the model size. We find that training with more gradient steps and larger models can further improve the sample-efficiency.

Data augmentation for viewpoint-robust control To investigate the importance of using visual observations from randomized cameras, we consider a baseline that uses images perturbed with data augmentation (*i.e.,* rotation, translation, brightness, and contrast) for multi-view representation learning. As shown in Figure 18(a) available at Appendix D, we observe that using the images from physically perturbed images largely outperforms the baseline based on data augmentation. This is because such a randomized camera can provide images from different perspectives that contain additional information which is not available from the single,

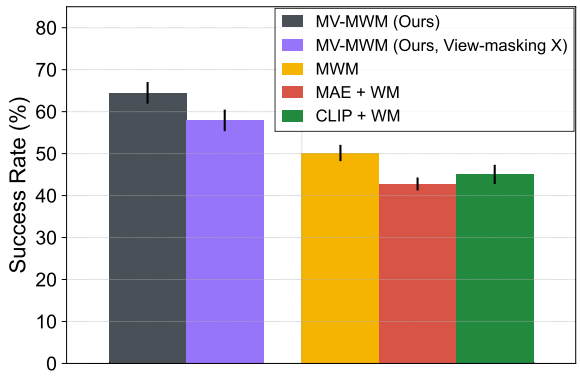


Figure 8. Aggregate success rate of imitation learning agents on five single-view control tasks. The result shows the mean and stratified bootstrap confidence interval across 20 runs.

fixed camera viewpoint. On the other hand, data augmentation fails to give such information useful for implicitly capturing 3D information of a robot workspace. Given this result, investigation into how to set up a real-world robot learning setup with randomized cameras would be an interesting and important future research direction.

We provide learning curves for ablation study in Appendix C and additional analysis in Appendix D.

4.5. Imitation Learning Experiments

Setup Finally, we investigate the effectiveness of our multi-view representation learning method on imitation learning (IL) setup, which is a widely-considered setup in the field of robotics. We consider a setup where we train visual control agents only with behavior cloning (BC) instead of RL to solve the tasks. Specifically, we consider the single-view control setup as in Section 4.1, using the same set of five manipulation tasks. For training, we use 100 expert demonstrations collected with the scripted policies available from the RLBench simulator. Unlike in the RL setup, we follow a setup of prior work that utilizes the output next best gripper pose. We use the same architecture as in RL experiments but find that using more stronger L2 weight decay largely improves the performance by preventing the overfitting. We also disable video autoencoding in IL experiments because the model is trained until convergence in this setup so that there is no problem from training difficulty from the view-masking as in RL setup. For baselines, we consider the same set of baselines as previous experiments but exclude TCN due to its overall low performance. For evaluation, we measure the average success rate over 500 episodes where the object position is randomized every episode.

Results In Figure 8, we observe that the trend in IL experiments is the same as in prior experiments, where our method, MV-MWM, outperforms all the baselines. In particular, MV-MWM outperforms MWM, which uses the same amount of training data, by a large margin (14.35%p). We also find that the proposed view-masking scheme significantly improves the performance, *e.g.,* the view-masking scheme improves the performance from 57.92% to 64.48%. This experimental result shows that the benefit of multi-view representation learning along with the proposed view-masking scheme is consistent across both RL and IL setups, highlighting the effectiveness of our method for diverse, practical robotic manipulation setups.

5. Discussion

Limitation and future directions One limitation of our work is that considered tasks are simple in that they do not require a long-horizon planning and involve a single object. Scaling up our framework to solve more challenging tasks is a direction we hope to investigate in future works. For instance, incorporating a more scalable architecture (Jaegle et al., 2021) along with large-scale pre-training on large datasets (Deng et al., 2009; Dasari et al., 2019) would be an interesting direction that can improve the generalization capability of visual manipulation system while having the benefit of multi-view representation learning. Another interesting direction would be to design a viewpoint randomization setup for real-world robot learning, where it is non-trivial to aggressively randomize viewpoints as we have done in sim-to-real transfer experiments.

Conclusion We present Multi-View Masked World Models, a reinforcement learning framework that learns multi-view representations and utilize them for diverse visual robotic manipulation setups. We conduct extensive experiments and find that our method consistently outperforms various baselines across a range of tasks in both simulation and real-world environments. We hope this work encourages future research to further explore the potential of multi-view representation learning for visual robotic manipulation.

Acknowledgements We would like to thank Danijar Hafner, Sangwoo Mo, Youngwoon Lee, Xingyu Lin, Hao Liu, Jongjin Park, Carlo Sferrazza, Sihyun Yu, and anonymous reviewers for helpful comments. This work was partially supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2022-0-00953, Self-directed AI Agents with Problem-solving Capability; No.2019-0-00075, Artificial Intelligence Graduate School Program (KAIST)) and KAIST-NAVER Hypercreative AI Center. This material is based upon work supported by the Google Cloud Research Credits program with the award (N6U8-0LLR-JDTW-JNWP). We also appreciate NVIDIA Corporation (https://www.nvidia.com/) and Cirrascale Cloud Services (https://cirrascale.com/) for providing compute resources.

References

Akinola, I., Varley, J., and Kalashnikov, D. Learning precise 3d manipulation from multiple uncalibrated cameras. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

Akkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., Petron, A., Paino, A., Plappert, M., Powell, G., Ribas, R., et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

Arora, S., Khandeparkar, H., Khodak, M., Plevrakis, O., and Saunshi, N. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.

Carrera, G., Angeli, A., and Davison, A. J. Slam-based automatic extrinsic calibration of a multi-camera rig. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

Chen, B., Abbeel, P., and Pathak, D. Unsupervised learning of visual 3d keypoints for control. In *International Conference on Machine Learning*, 2021a.

Chen, X., Xie, S., and He, K. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021b.

Dasari, S., Ebert, F., Tian, S., Nair, S., Bucher, B., Schmeckpeper, K., Singh, S., Levine, S., and Finn, C. Robonet: Large-scale multi-robot learning. In *Conference on Robot Learning*, 2019.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 2009.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Feichtenhofer, C., Fan, H., Li, Y., and He, K. Masked autoencoders as spatiotemporal learners. *arXiv preprint arXiv:2205.09113*, 2022.

Gao, P., Geng, S., Zhang, R., Ma, T., Fang, R., Zhang, Y., Li, H., and Qiao, Y. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021.

Gelada, C., Kumar, S., Buckman, J., Nachum, O., and Bellemare, M. G. Deepmdp: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning*, 2019.

Geng, X., Liu, H., Lee, L., Schuurams, D., Levine, S., and Abbeel, P. Multimodal masked autoencoders learn transferable representations. *arXiv preprint arXiv:2205.14204*, 2022.

Guhur, P.-L., Chen, S., Garcia, R., Tapaswi, M., Laptev, I., and Schmid, C. Instruction-driven history-aware policies for robotic manipulations. *arXiv preprint arXiv:2209.04899*, 2022.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, 2019.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.

Hsu, K., Kim, M. J., Rafailov, R., Wu, J., and Finn, C. Vision-based manipulators need to also see from their hands. *arXiv preprint arXiv:2203.12677*, 2022.

Jaegle, A., Gimeno, F., Brock, A., Vinyals, O., Zisserman, A., and Carreira, J. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, 2021.

James, S. and Abbeel, P. Coarse-to-Fine Q-attention with Learned Path Ranking. *arXiv preprint arXiv:2204.01571*, 2022a.

James, S. and Abbeel, P. Coarse-to-Fine Q-attention with Tree Expansion. *arXiv preprint arXiv:2204.12471*, 2022b.

James, S. and Davison, A. J. Q-attention: Enabling efficient learning for vision-based robotic manipulation. *IEEE Robotics and Automation Letters*, 2022.

James, S., Ma, Z., Arrojo, D. R., and Davison, A. J. RLBench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020.

James, S., Wada, K., Laidlow, T., and Davison, A. J. Coarse-to-Fine Q-attention: Efficient learning for visual robotic manipulation via discretisation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Jangir, R., Hansen, N., Ghosal, S., Jain, M., and Wang, X. Look closer: Bridging egocentric and third-person views with transformers for robotic manipulation. *IEEE Robotics and Automation Letters*, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

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

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. In *Advances in Neural Information Processing Systems*, 2018.

Parisi, S., Rajeswaran, A., Purushwalkam, S., and Gupta, A. The unsurprising effectiveness of pre-trained vision models for control. *arXiv preprint arXiv:2203.03580*, 2022.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

Radosavovic, I., Xiao, T., James, S., Abbeel, P., Malik, J., and Darrell, T. Real world robot learning with masked visual pre-training. In *Conference on Robot Learning*, 2022.

Sadeghi, F., Toshev, A., Jang, E., and Levine, S. Sim2real viewpoint invariant visual servoing by recurrent control. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Schroff, F., Kalenichenko, D., and Philbin, J. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

Schwarzer, M., Rajkumar, N., Noukhovitch, M., Anand, A., Charlin, L., Hjelm, D., Bachman, P., and Courville, A. Pretraining representations for data-efficient reinforcement learning. In *Advances in Neural Information Processing Systems*, 2021.

Seo, Y., Hafner, D., Liu, H., Liu, F., James, S., Lee, K., and Abbeel, P. Masked world models for visual control. In *Conference on Robot Learning*, 2022a.

Seo, Y., Lee, K., James, S., and Abbeel, P. Reinforcement learning with action-free pre-training from videos. In *International Conference on Machine Learning*, 2022b.

Sermanet, P., Lynch, C., Chebotar, Y., Hsu, J., Jang, E., Schaal, S., and Levine, S. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, 2018.

Sharma, M., Fantacci, C., Zhou, Y., Koppula, S., Heess, N., Scholz, J., and Aytar, Y. Lossless adaptation of pretrained vision models for robotic manipulation. In *International Conference on Learning Representations*, 2023.

Shridhar, M., Manuelli, L., and Fox, D. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, 2021.

Shridhar, M., Manuelli, L., and Fox, D. Perceiver-actor: A multi-task transformer for robotic manipulation. *arXiv preprint arXiv:2209.05451*, 2022.

Sola, J., Monin, A., Devy, M., and Vidal-Calleja, T. Fusing monocular information in multicamera slam. *IEEE transactions on robotics*, 2008.

Stooke, A., Lee, K., Abbeel, P., and Laskin, M. Decoupling representation learning from reinforcement learning. In *International Conference on Machine Learning*, 2021.

Tong, Z., Song, Y., Wang, J., and Wang, L. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Advances in Neural Information Processing Systems*, 2022.

Watter, M., Springenberg, J., Boedecker, J., and Riedmiller, M. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in neural information processing systems*, 2015.

Xiao, T., Radosavovic, I., Darrell, T., and Malik, J. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

Yang, A. J., Cui, C., Bârsan, I. A., Urtasun, R., and Wang, S. Asynchronous multi-view slam. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

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

Zhan, A., Zhao, P., Pinto, L., Abbeel, P., and Laskin, M. A framework for efficient robotic manipulation. *arXiv preprint arXiv:2012.07975*, 2020.

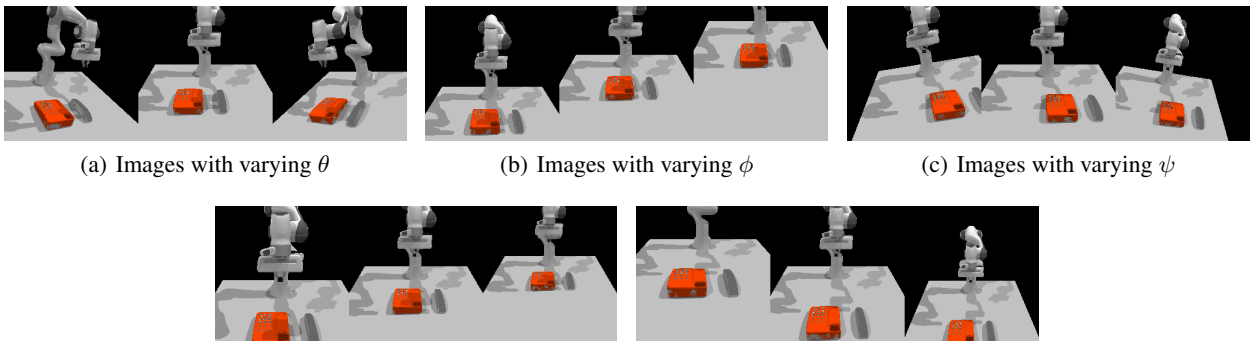
Zhang, R., Fang, R., Zhang, W., Gao, P., Li, K., Dai, J., Qiao, Y., and Li, H. Tip-adapter: Training-free clip-adapter for better vision-language modeling. *arXiv preprint arXiv:2111.03930*, 2021.

A. Implementation Details

RLBench details For RLBench experiments, we designed dense rewards for five manipulation tasks used in our experiments. We first construct waypoints where the robot should reach (*i.e.,* phone and base positions in Phone On Base). Then we define the reward to be the distance between the gripper and the next checkpoint. In tasks that does not need rotation in solving (*i.e.,* Phone On Base, Pick Up Cup, Take Umbrella Out of Umbrella Stand, Put Rubbish in Bin), we disable rotation to reduce redundancy in exploring rotating actions. For disabling rotation, we use a path planner with identity quaternion to force the robot to be in an upright position. In these tasks, we train the RL agent to output relative change in (x, y, z) position. For tasks that requires rotation (*i.e.,* Stack Wine), RL agent is trained to output relative quaternion changes as well as (x,y,z) position changes. For viewpoint-robust control, we ease the difficulty of tasks. For Phone On Base, we make phone and base be located upright (*i.e.,* not rotated related to the table). For Pick Up Cup, we make the distractor cup colors be fixed as yellow instead of changing the color for each episode. For such tasks, we append asterisk (*) after the (shortened) task name; *i.e.,* Phone* and Cup*. For more details, we refer readers to the source code we have attached.

Viewpoint randomization We randomize viewpoint by adjusting position and orientation of front camera. Specifically, we randomize five parameters of camera position and orientation: θ, ϕ, d, h, ψ, which represents as follows:

- θ: angle that determines how camera is moved clockwise (from the front camera) with respect to the origin.
- ϕ: angle that determines how camera is tilted downward.
- ψ: angle that determined how camera is rolled clockwise.
- d: distance from the origin of the simulator.
- h: height of camera from the floor.



(a) Images with varying θ (b) Images with varying ϕ (c) Images with varying ψ

(d) Images with varying d (e) Images with varying h

Figure 9. Rendered images with varying randomization parameters.

We exemplify how each parameter affects viewpoint in Figure 9. By randomizing aforementioned five parameters, we design three randomization types (weak, medium, and strong) as follows.

- weak:

$$\theta \sim [-5°, 5°], \quad \phi \sim [26°, 28°], \quad \psi \sim [-5°, 5°], \quad d \sim [1.25°, 1.45°], \quad h \sim [1.5, 1.7]$$

- medium:

$$\theta \sim [-7.5°, 7.5°], \quad \phi \sim [25.5°, 28.5°], \quad \psi \sim [-7.5°, 7.5°], \quad d \sim [1.2, 1.5], \quad h \sim [1.45, 1.75]$$

- strong:

$$\theta \sim [-10°, -7.5°] \cup [7.5°, 10°], \quad \phi \sim [25°, 25.5°] \cup [28.5°, 29°], \quad \psi \sim [-10°, -7.5°] \cup [7.5°, 10°],$$
$$d \sim [1.15, 1.2] \cup [1.5, 1.55], \quad h \sim [1.4, 1.45] \cup [1.75, 1.8]$$

Note that we design the strong randomization, whose distribution does not overlap with that of weak or medium randomization, in order to evaluate the performance under unseen viewpoint conditions. When we start episodes to collect transitions for training, we sample two random viewpoints under the randomization type. In evaluation phase, we sample one random viewpoint under the randomization type. The sampled viewpoints are maintained for an episode, and we re-sample viewpoints when new episode begins.

Architecture and optimization details Our architecture is based on the publicly available source code of Seo et al. (2022a), which is implemented with tfimm[2] library. We use 8-layer ViT encoder and 6-layer ViT decoder. For each view and time step, we introduce additional 1D learnable parameters that have the same embedding size as transformer blocks. We add these parameters to 2D fixed sin-cos embeddings and add them to features. We note that these parameters are shared across the same times and the same views. We do not introduce separate parameters for randomized viewpoints in our viewpoint-robust control experiments. For optimization, we use Adam optimizer (Kingma & Ba, 2015) with the learning rate of $3e-4$, the weight decay of $1e-6$, and the batch size of 1024. For training MV-MAE, we apply warm-up learning rate scheduling over initial 2500 gradient steps from learning rate of 0. We take 1 gradient step per every 16 environment steps. We follow the training schemes and details of Seo et al. (2022a) regarding the architecture, unless otherwise specified.

Computation We use 24 CPU cores (Intel Xeon CPU @ 2.2GHz) and 1 GPU (NVIDIA A100 40GB GPU) for our experiments. We find that there is no significant difference between all methods with regard to wall time because rendering speed of the RLBench simulator is a bottleneck rather than algorithmic difference. Running experiments over 300k environment steps for MV-MWM takes approximately 12 hours.

Hyperparameters We report the hyperparameters used in our experiments in Table 2.

Table 2. Hyperparameters used in our experiments. Unless otherwise specified, we use the same hyperparameters used in MWM (Seo et al., 2022a).

Hyperparameter	**Value**
Representation learning	
Image observation	$96 \times 96 \times 3$
Image normalization	Mean: $(0.485, 0.456, 0.406)$, Std: $(0.229, 0.224, 0.225)$
Autoencoder batch size	1024
Autoencoder initialization steps	10000
Autoencoder warm-up steps	2500
Autoencoder learning rate	$3 \cdot 10^{-4}$
Autoencoder masking ratio	0.95
Autoencoder ViT encoder size	8 layers, 4 heads, 256 units
Autoencoder ViT decoder size	6 layers, 4 heads, 256 units
Behavior learning	
Action repeat	1
Max episode length	150
Early episode termination	True (when path planner fails)
Reward normalization	True
Number of expert demonstrations	50 (single-view and multi-view control), 100 (viewpoint-robust control)
World model batch size	36
World model expert batch size	12
World model sequence length	50
World model tradeoff (β)	1.0
World model ViT encoder size	2 layers, 4 heads, 128 units
World model ViT decoder size	2 layers, 4 heads, 128 units

[2] https://github.com/martinsbruveris/tensorflow-image-models

B. Baselines

B.1. Masked world models

Masked world models (MWM; Seo et al. 2022a) is a visual model-based reinforcement learning framework which decouples visual representation learning and dynamicd model learning. Specifically, MWM trains a self-supervised vision transformer (ViT; Dosovitskiy et al. 2021) to reconstruct pixels given masked convolutional features for representation model. Then, a world model is trained on top of the frozen visual representations. The major methodological difference between MV-MWM and MWM is that MV-MWM learns cross-view information from multiple viewpoints by training a multi-view masked autoencoder but MWM only considers visual information within each viewpoint. Specifically, MWM does not employ pair information between multiple viewpoints in representation learning, but considers images from different viewpoints as independent instances in training. Whereas, MV-MWM considers cross-view information along with a synergistic combination of view-masking and video-autoencoding in contrast to MWM that trains an image autoencoder with uniform masking. With this methodological difference, MV-MWM allows for learning world models that can be useful for a range of important and practical visual robotic manipulation setups.

B.2. Time contrastive network

Time contrastive network (TCN; Sermanet et al. 2018) is a contrastive approach that learns view-invariant representations by attracting the representations of simultaneous viewpoints but making the representations from the same viewpoints be far located. For a given (anchor) frame in one viewpoint, we sample a positive and negative frame as follows. For the positive frame, we use the frame that has the same timestep as the given anchor frame but from another viewpoint. For the negative frame, we choose a frame that is a temporally faraway frame from the same viewpoint. Specifically, we sample a random frame among frames that are at least 30 timesteps away from the anchor frame. After building a triplet of anchor, positive, and negative frame, we train the encoder model with triplet loss (Schroff et al., 2015), which is formulated as follows:

$$\mathcal{L}_{\texttt{TCN}} = \texttt{max}(\|f(o^a) - f(o^p)\|_2^2 - \|f(o^a) - f(o^n)\|_2^2 + \alpha, 0), \tag{5}$$

where α is margin, $f(\cdot)$ refers embedding of a frame, and o^a, o^p, and o^n are anchor, positive, and negative frame, respectively. In our implementation of TCN, we use same encoder architecture with MV-MWM for a fair comparison; 8-layer ViT encoder. For embedding $f(\cdot)$, we use class embedding from the last layer of ViT encoder. In the control phase, we freeze the encoder and use average pooled token embeddings as an input for the RL agent. For RL agent, we employ the same world model and policy architecture with MV-MWM for a fair comparison.

B.3. Pretrained MAE and CLIP with world model (MAE+WM, CLIP+WM)

Masked autoencoder (MAE; He et al. 2021) learns visual representation in a self-supervised manner by training a vision Transformer (ViT; Dosovitskiy et al. 2021) to reconstruct masked patches. Contrastive language-image pre-training (CLIP; Radford et al. 2021) learns visual representation by aligning embedding of text and image with contrastive learning. Recently, it has been shown that pre-training MAE with (in-the wild) large-scale dataset, and then training a control module on top of frozen representation can solve real-world robotic manipulation tasks (Radosavovic et al., 2022; Shridhar et al., 2021). To compare such training scheme with ours, we design MAE+WM and CLIP+WM, which learn world model upon frozen representation of pretrained MAE or CLIP. For a fair comparison, we match the world model size with that used in MV-MWM: *i.e.,* 2 layers and 4 heads for the world model ViT encoder and decoder. For the pretrained MAE and CLIP, we employ open-sourced pretrained model from huggingface transformers library.[3][4] For the input of pretrained MAE and CLIP, we use 224 $\times$ 224 RGB observation from each camera. Then, we feed 7 $\times$ 7 patches into the world model.

[3] https://huggingface.co/facebook/vit-mae-base
[4] https://huggingface.co/openai/clip-vit-base-patch32

C. Full Experimental Results

C.1. Multi-view Control with Front and Wrist Camera

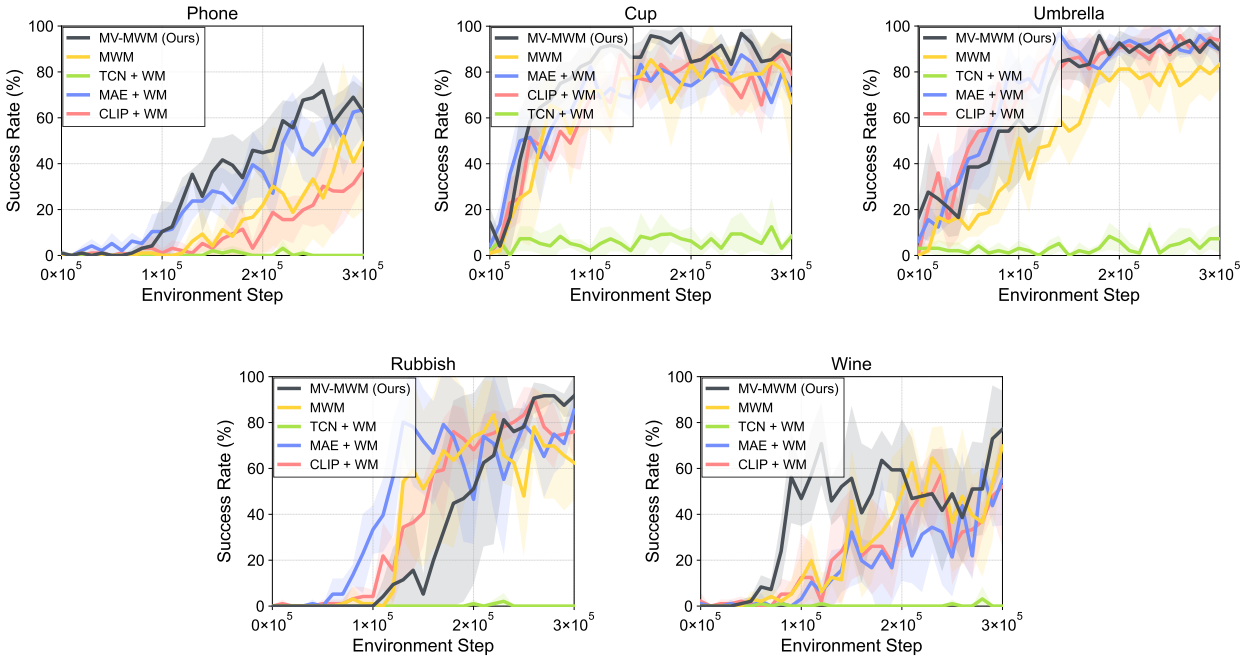


Figure 10. Learning curves of RL agents that operate on front and wrist camera observation for solving five tasks from RLBench as measured on the success rate. The solid line and shaded regions represent the mean and standard deviations, respectively, across 4 runs.

C.2. Single-view Control with Front Camera

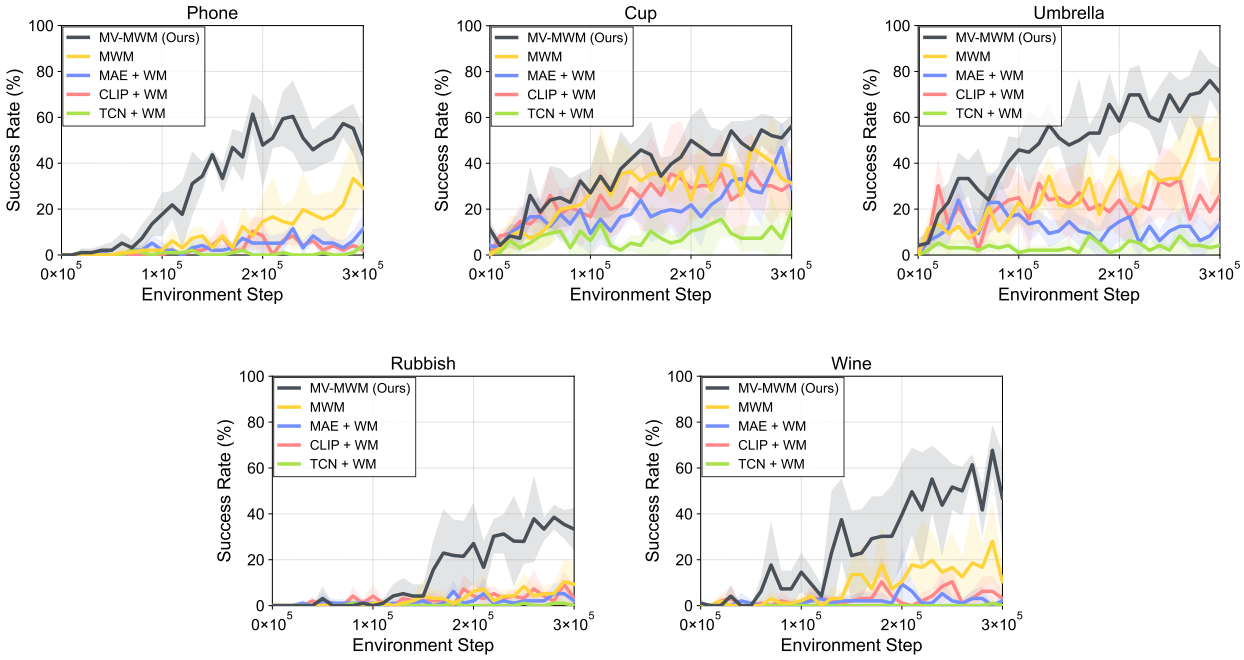


Figure 11. Learning curves of RL agents that operate on front camera observation for solving five tasks from RLBench as measured on the success rate. The solid line and shaded regions represent the mean and standard deviations, respectively, across 4 runs.

C.3. Viewpoint-Robust Control

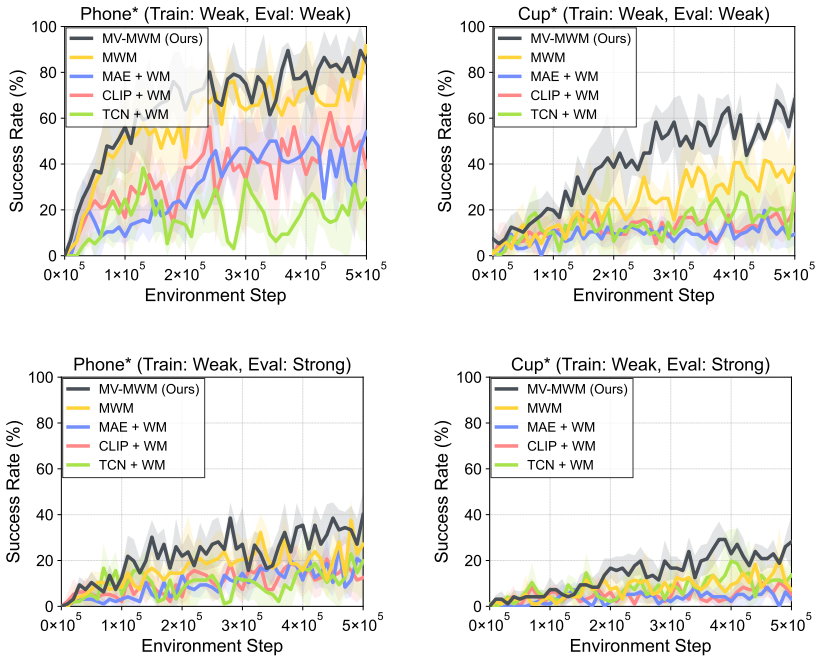


Figure 12. Success rate on seen (weak) and unseen (strong) viewpoints by RL agents trained on weak randomization across two tasks from RLBench. The solid line and shaded regions represent the mean and standard deviations, respectively, across 4 runs.

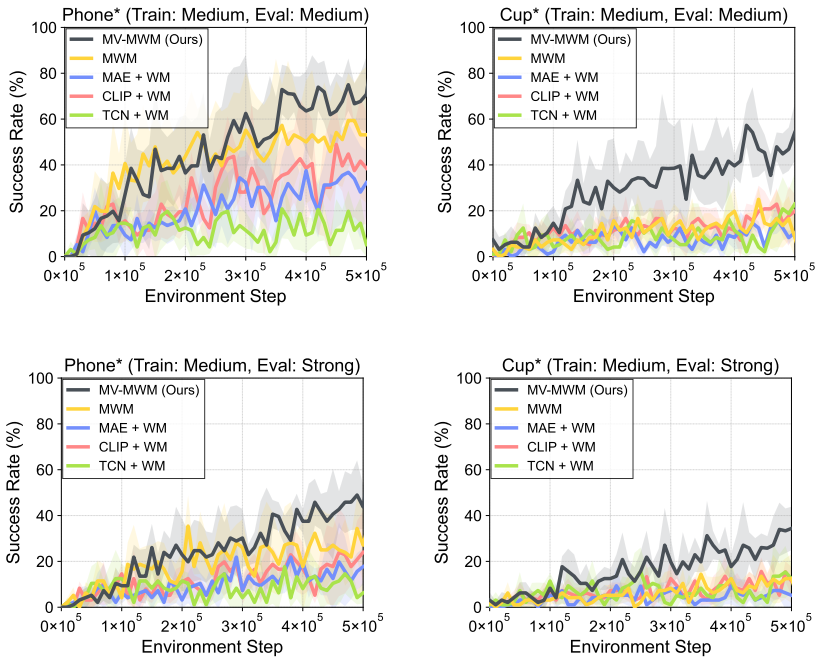


Figure 13. Success rate on seen (medium) and unseen (strong) viewpoints by agents trained on medium randomization across two tasks from RLBench. The solid line and shaded regions represent the mean and standard deviations, respectively, across 4 runs.

C.4. Ablation Study and Analysis

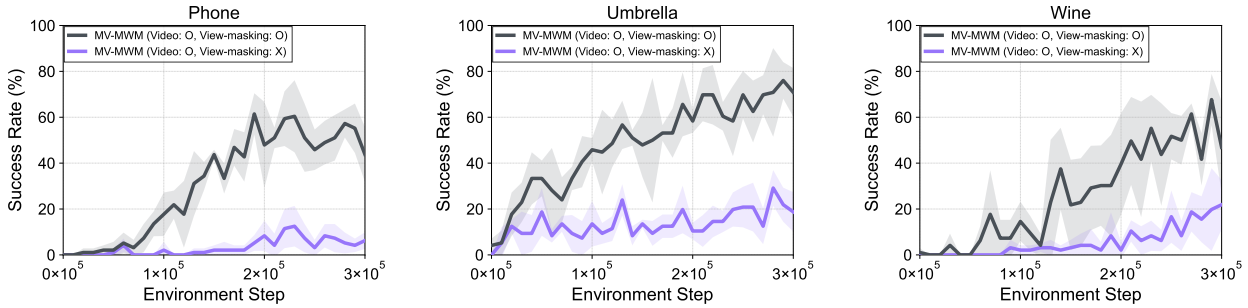


Figure 14. Effect of view masking. The solid line and shaded regions represent the mean and standard deviation across 4 runs.

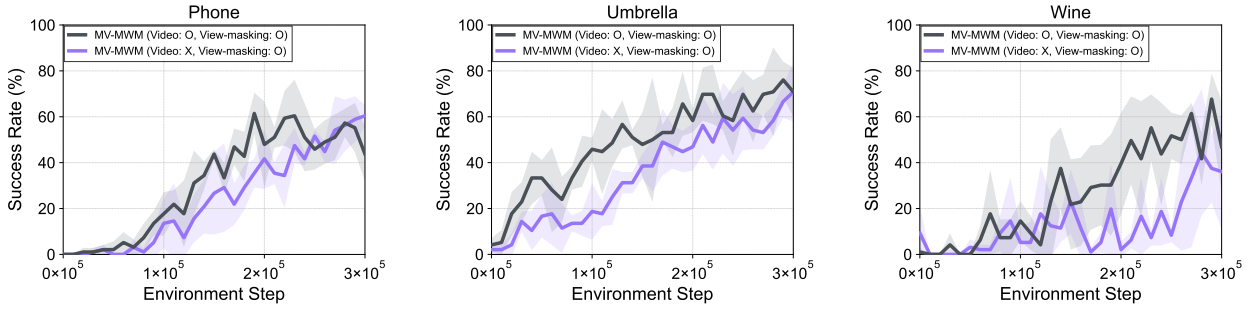


Figure 15. Effect of video autoencoding. The solid line and shaded regions represent the mean and standard deviation across 4 runs.

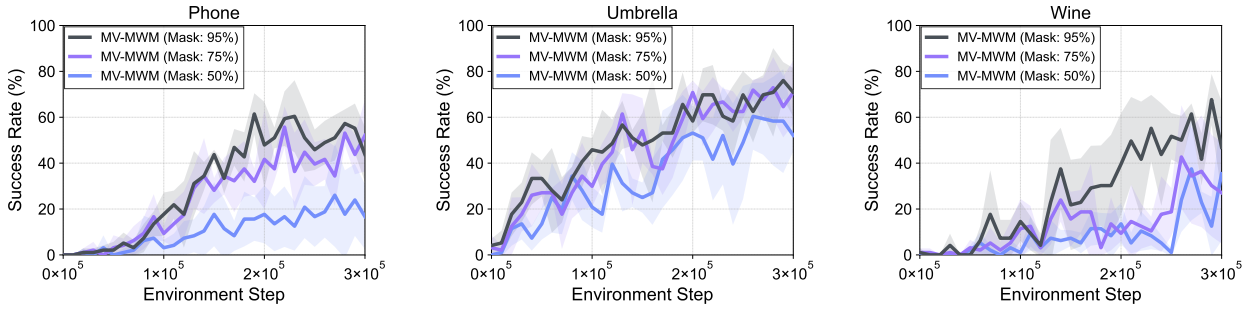
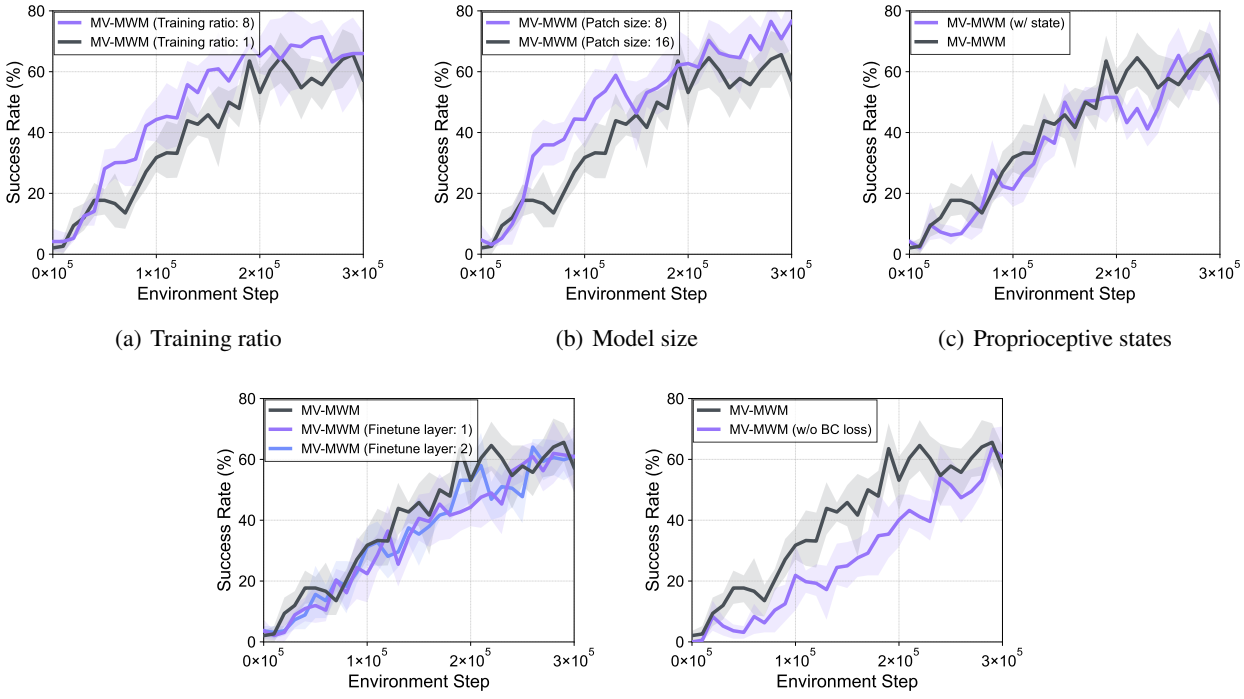


Figure 16. Effect of mask ratio. The solid line and shaded regions represent the mean and standard deviation across 4 runs.

D. Additional Experiments



(a) Training ratio (b) Model size (c) Proprioceptive states

(d) Finetuning MV-MAE encoder (e) Behavior cloning loss

Figure 17. Aggregate learning curves on Phone On Base and Take Umbrella Out of Stand in single-view control to investigate the effect of (a) training ratio, (b) model size, (c) proprioceptive states, (d) fine-tuning MV-MAE encoder, and (e) auxiliary behavior cloning loss with expert demonstration. The solid line and shaded regions represent the mean and standard deviation across 4 runs.

Training ratio In Figure 17(a), we investigate whether MV-MWM can be scaled up for better performance by adjusting the training ratio, which is the number of gradient steps per every 16 environment steps. We find that sample-efficiency can be further improved with training ratio higher than 1. This insight would bring practical guidance for future researchers when applying MV-MWM to new tasks.

Model size We further investigate the scaling property of MV-MWM by scaling up the model size in Figure 17(b). We increase the number of patches in ViTs by reducing the patch size of MV-MAE from 16 to 8. We find that larger model achieve higher asymptotic performance as well as higher sample-efficiency.

Proprioceptive states In Figure 17(c), we analyze whether feeding auxiliary proprioceptive states into the world model could improve performance. We observe that additional proprioceptive states does not make performance boost. We hypothesize that this is because our method implicitly learns 3D information by learning cross-view information from multi-view data, thus including proprioceptive state does not make large information gain.

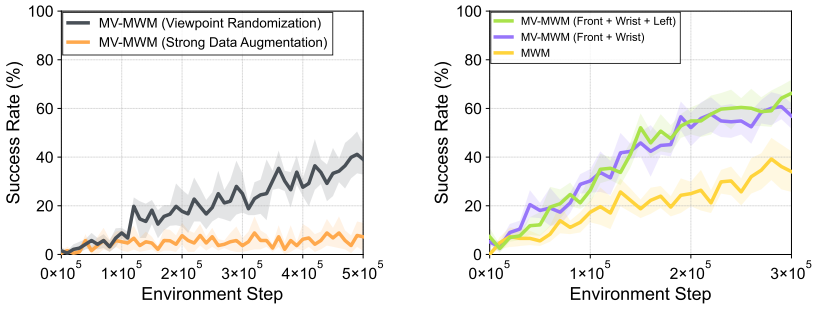
Fine-tuning MV-MAE encoder We investigate whether fine-tuning MV-MAE encoder when training world model could improve performance; we fine-tune the last one or two layers of the MV-MAE encoder and freeze all the other layers of the MV-MAE encoder. We observe that fine-tuning does not make gains in Figure 17(d), which shows that our visual representation learning scheme enables the autoencoder to effectively capture information required for solving the task.

Behavior cloning loss with expert demonstration Figure 17(e) shows that how behavior cloning loss using expert demonstrations affect performance. We find that using behavior loss improves sample-efficiency, yet the asymptotic performance converges to that without the behavior cloning loss. The behavior cloning loss could implicitly reduce an exploration space into area nearby expert demonstrations, which accelerates training especially in the early phase.

Data augmentation for viewpoint-robust control To investigate the importance of using visual observations from randomized cameras, we consider a baseline that uses images perturbed with data augmentation (*i.e.,* rotation, translation, brightness, and contrast) for multi-view representation learning. As shown in Figure 18(a), we observe that using the images from physically perturbed images largely outperforms the baseline based on data augmentation. This is because such a randomized camera can provide images from different perspectives that contain additional information which is not available from the single, fixed camera viewpoint. On the other hand, data augmentation fails to give such information useful for implicitly capturing 3D information of a robot workspace. Given this result, investigation into how to set up a real-world robot learning setup with a randomized camera would be an interesting and important future research direction.

Three cameras experiments To assess how the number of viewpoints used for representation learning affects the performance, we train MV-MWM with more than two cameras in multi-view representation learning: three cameras of {Front, Wrist, and Left Shoulder}. As shown in Figure 18(b), we find that including additional Left Shoulder camera does not largely affect the performance compared to using only Front and Wrist cameras. We hypothesize this is because a widely-used camera configuration with Front and Wrist camera is already sufficient for capturing the information required for completing the considered tasks. It would be an interesting future direction to investigate the importance of specific camera viewpoints for solving a variety of tasks.

Longer training step In Figure 19, we report the experimental results with larger training steps to provide asymptotic results of our analysis experiments. We observe that the performance gain from view-masking is significant over longer training horizon, which shows the effectiveness of the proposed masking scheme. On the other hand, because the video autoencoding is introduced to improve the sample-efficiency at the initial phase of training by addressing the difficulty of training with the view-masking, we find that the benefit of employing the video autoencoding becomes less significant in a more long-term manner. Moreover, we observe that high masking ratio is crucial for sample-efficiency but asymptotic performance with different masking ratios is similar. We hypothesize this is because our proposed view-masking scheme can asymptotically encourage the model to learn useful representations even with the low masking ratio as 50%.



(a) Data augmentation (b) Three cameras

Figure 18. (a) Aggregate learning curves on Phone On Base* and Pick Up Cup* in viewpoint-robust control to investigate the effect of using randomized cameras. (b) Aggregate learning curves of multi-view visual control agents for solving three manipulation tasks.

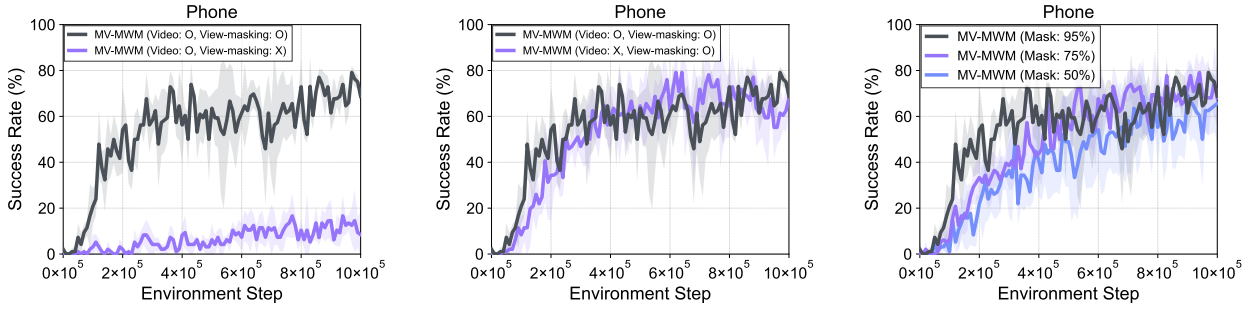


Figure 19. Learning curves of single-view visual control agents operating on the front camera for solving Phone On Base task from RLBench (James et al., 2020), investigating the effect of (a) view masking, (b) video autoencoding and (c) masking ratio. The solid line and shaded regions represent the mean and standard deviation across 4 runs.