Perception Stitching: Zero-Shot Perception Encoder Transfer for Visuomotor Robot Policies

Anonymous authors
Paper under double-blind review

Abstract

Vision-based imitation learning has shown promising capabilities of endowing robots with various motion skills given visual observation. However, current visuomotor policies fail to adapt to drastic changes in their visual observations. We present Perception Stitching that enables strong zero-shot adaptation to large visual changes by directly stitching novel combinations of visual encoders. Our key idea is to enforce modularity of visual encoders by aligning the latent visual features among different visuomotor policies. Our method disentangles the perceptual knowledge with the downstream motion skills and allows the reuse of the visual encoders by directly stitching them to a policy network trained with partially different visual conditions. We evaluate our method in various simulated and real-world manipulation tasks. While baseline methods failed at all attempts, our method could achieve zero-shot success in real-world visuomotor tasks. Our quantitative and qualitative analysis of the learned features of the policy network provides more insights into the high performance of our proposed method.

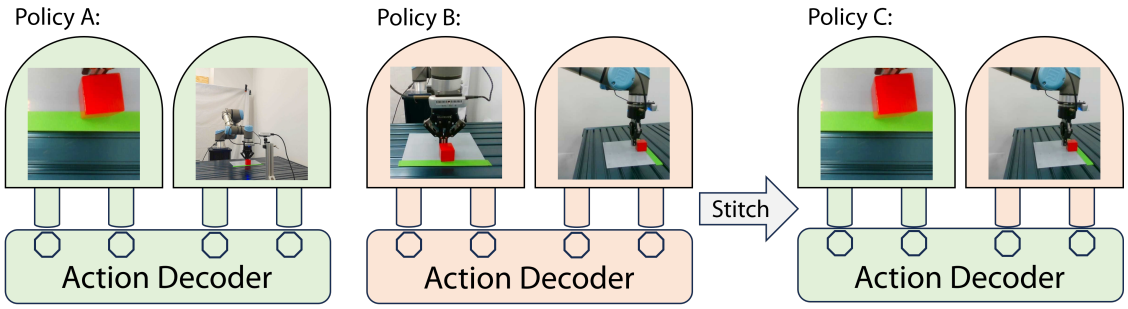


Figure 1: **Perception Stitching**: "Policy A" was trained with an in-hand camera and a front-view camera. "Policy B" was trained with a close-up camera and a side-view camera. Perception Stitching enables zero-shot stitching of the original Policy A and B by reusing their relevant components for each sensing configuration to form a "Policy C". "Policy C" can maintain strong zero-shot transfer performance with an in-hand camera and a side-view camera.

1 Introduction

Despite recent advances in vision-based imitation learning for acquiring diverse motor skills (Chi et al., 2023; Fu et al., 2024), a significant challenge in deploying visuomotor policies in real-world settings is ensuring the perceptual configurations are identical during training and policy execution. With the growth of hybrid robot datasets where robots are trained to perform similar tasks across the world, it remains difficult to share their learning experiences, even under the same task but with different visual observations. Such a challenge often stems from the unique configurations and perspectives each camera setup brings, which, while enriching the dataset, complicates the sharing of learned experiences across different systems. Different institutions worldwide usually place different sensors at different perspectives when collecting their datasets. In these cases, previously collected datasets or trained policies are not interchangeable, and new training data must be collected in each environment.

Instead, what if we could directly stitch the perception encoder trained in one visuomotor policy to the rest of the components of another visuomotor policy? This approach would enable zero-shot transfer of the

trained visuomotor policies to a novel combination of perceptual configurations. To address the challenges associated with zero-shot transfer across different settings, previous efforts have aimed to learn an invariant feature space or universal representation for quick adaptation to new environments, with approaches ranging from extensive pre-training with video data (Nair et al., 2022), employing contrastive learning between two policies to find a common feature space (Gupta et al., 2017), and concentrating on low-dimensional data over vision-based observations (Jian et al., 2023). In particular, recent studies have proposed to achieve fast policy transfer (Jian et al., 2023) by aligning the latent representations of different perception encoders with relative representation (Moschella et al., 2022). However, it remains unclear how to scale similar approaches from few-shot transfer to zero-shot transfer and high-dimensional observations.

We present **Perception Stitching (PeS)** (Fig. 1) to enable zero-shot perception encoder transfer for visuomotor robot policies. Our approach advances the previous studies through a novel training scheme under various camera configurations and effectively processing high-dimensional image data. Our approach can train modular perception encoders for specific visual configurations (e.g. camera parameters and positions) and reuse the trained perception encoders in a novel environment in a plug-and-go manner. In the simulation, we evaluate PeS in five different robotic manipulation tasks, each with six unique visual configurations. It constantly shows significant performance improvement compared to three baseline methods and two ablation studies. We also evaluate PeS in four real-world manipulation tasks. While the baseline struggles to get any successful attempts, PeS achieves pronounced success rates, indicating that directly stitching modular perception encoders in the real world has been turned from impossible to possible by our work. Additionally, we contribute quantitative and qualitative analysis to provide more insights on the high performance of our proposed method.

2 Related Work

Learning Visuomotor Policy for Robotic Manipulation A wide range of previous work has focused on learning visuomotor policy for robotic manipulation (Levine et al., 2016; Finn et al., 2016; 2017b; Kalashnikov et al., 2018; Srinivas et al., 2018; Ebert et al., 2018; Zhu et al., 2018; Rafailov et al., 2021; Jain et al., 2019; Hämäläinen et al., 2019; Florence et al., 2022; Brohan et al., 2022; 2023; Padalkar et al., 2023; Sermanet et al., 2018). Certain works have investigated the impact of camera placements (Zaky et al., 2020; Hsu et al., 2022), design of the hardware and software (Zhao et al., 2023; Fu et al., 2024; Kim et al., 2023), novel network architectures and optimization techniques (Dasari & Gupta, 2021; Kim et al., 2021; Zhu et al., 2023; Abolghasemi et al., 2019; Ramachandruni et al., 2020; Brohan et al., 2023; 2022; Padalkar et al., 2023; Chi et al., 2023; Li et al., 2023). In this work, we show that perception encoders trained with Behavior Cloning (BC) (Pomerleau, 1988) often lack the flexibility for module reuse. Our Perception Stitching (PeS) method enables perceptual knowledge reuse and facilitates zero-shot transfer between diverse visual configurations, advancing existing research on fast adaptable visuomotor policy design.

Robot Transfer Learning Transfer learning has long been considered as a primary challenge in robotics (Tan et al., 2018; Taylor & Stone, 2009). In the context of reinforcement learning, many previous work transfer different components such as policies (Devin et al., 2017; Konidaris & Barto, 2007; Fernández & Veloso, 2006), parameters (Finn et al., 2017a; Killian et al., 2017; Doshi-Velez & Konidaris, 2016), features (Barreto et al., 2017; Gupta et al., 2017), experience samples (Lazaric et al., 2008), value functions (Liu et al., 2021; Zhang & Zavlanos, 2020; Tirinzoni et al., 2018), and reward functions (Konidaris & Barto, 2006). In imitation learning, additional studies have made progress via domain adaptation (Kim et al., 2020; Yu et al., 2018), querying unlabeled datasets (Du et al., 2023), abstracting and transferring concepts (Lázaro-Gredilla et al., 2019; Shao et al., 2021), or conditioning on other information such as language instructions (Stepputtis et al., 2020; Lynch & Sermanet, 2020; Jang et al., 2022) and goal images (Pathak et al., 2018). Our work focuses on neural network policy sub-module reuse through direct stitching to achieve zero-shot transfer.

Sim-to-real transfer is another important topic within transfer learning (Sadeghi et al., 2018; James et al., 2019; Zhang et al., 2019; Tobin et al., 2018; Mehta et al., 2020; James et al., 2017; Nguyen et al., 2018; Rusu et al., 2017). To merge the sim2real gap, previous methods include domain adaptation (Zhang et al., 2019; Tobin et al., 2018; Mehta et al., 2020; James et al., 2017), adopting a progressive network (Rusu et al., 2017; 2016), fine-tuning the visual layers (Sadeghi et al., 2018), training a generator that translates real-world images to canonical simulation images (James et al., 2019), or training a CycleGAN (Zhu et al., 2017) to

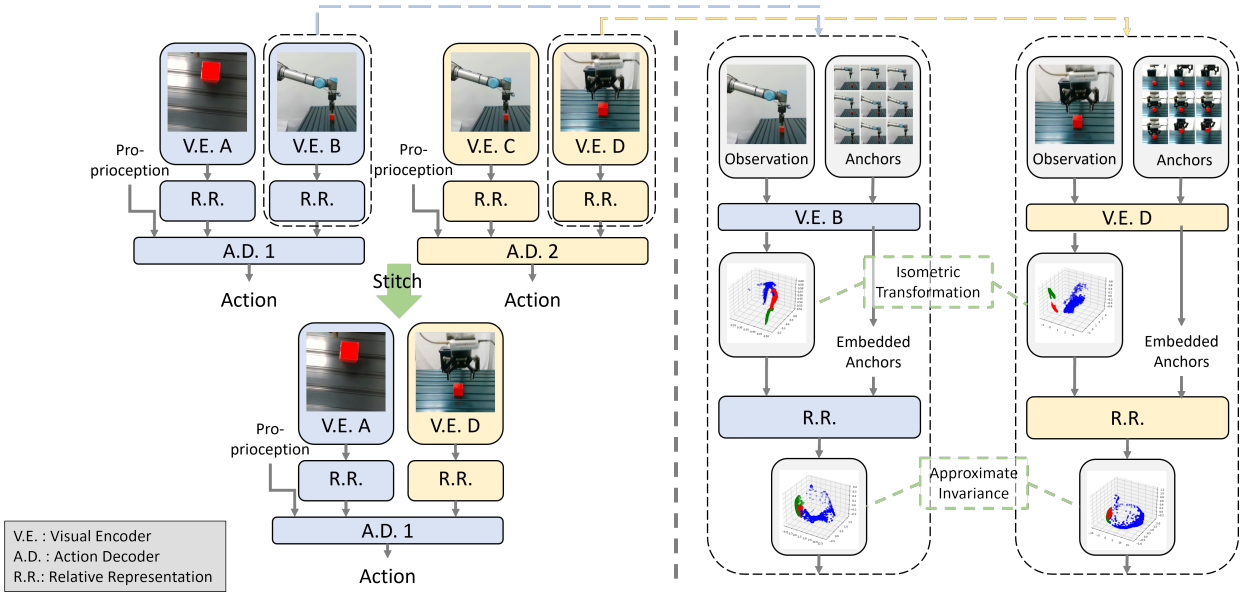


Figure 2: **Method Overview.** Two visual encoders process the RGB images from two cameras separately, and the latent representations are concatenated with the proprioception of the robot end effector state. The original latent representations of the images are observed to have an approximate isometric transformation relationship. Relative representations with disentanglement regularization can maintain an approximate invariance and, therefore, help achieve high zero-shot transfer performance.

synthesize real images from simulation images(Nguyen et al., 2018). Most recent research has proposed to learn invariant or universal feature representations for transfer learning (Gupta et al., 2017; Jian et al., 2023; Nair et al., 2022). Our work fits in this category. Unlike previous studies, our method does not require a large amount of online data for pre-training (Nair et al., 2022) or training two policies simultaneously(Gupta et al., 2017). Compared with a similar previous work by Jain et al. (2019), our method is not limited to low dimensional observations, does not require few-shot fine-tuning, and can solve much more difficult tasks.

Compositional Robot Learning Compositional robot learning reuses the learned knowledge saved in some portion of the policy network in new tasks instead of training from scratch (Pfeiffer et al., 2023; Devin, 2020; Alet et al., 2018; Chen et al., 2020). It has achieved promising results in multi-task learning (Yang et al., 2020; Hussing et al., 2023; Chen et al., 2022; Kwiatkowski et al., 2022; Hu et al., 2022), transfer learning (Devin et al., 2017; Jian et al., 2023; Gupta et al., 2017), and lifelong learning (Mendez et al., 2022; Mendez & Eaton, 2022; Méndez, 2022). Some previous works train a graph of network modules and generate different paths to connect specific modules for different tasks (Yang et al., 2020; Mendez et al., 2022), but the network modules cannot be reused out of the graph it embeds in. Devin et al. (2017) reuses the network module without aligning latent space representations and thus fails to achieve satisfying performance in more complex tasks. Jian et al. (2023) aligns the latent spaces by selecting anchor states and calculating relative representations (Moschella et al., 2022) at the module interface, which no longer requires simultaneously training multiple policies. However, the results are limited to low-dimensional observations and few-shot fine-tuning in simple tasks such as pushing. In this work, we extend its success to high-dimensional image observations and more difficult tasks such as cube stacking and door opening with zero-shot transfer.

3 Perception Stitching: Learning Reusable Perception Network Module

Perception Stitching (PeS) (Fig. 2) is a compositional vision-based robot learning framework that allows zero-shot transfer of perceptual knowledge between different camera configurations. It is designed to be compatible with conventional behavior cloning algorithms (Bain & Sammut, 1995; Ross et al., 2011; Torabi et al., 2018; Chi et al., 2023), various visual encoder structures such as CNN (LeCun et al., 1989) and ResNet (He et al., 2016), and action decoder structures such as MLP (Hinton et al., 2012; Mandlekar et al., 2021) and LSTM (Hochreiter & Schmidhuber, 1997). In this section, we will demonstrate the two main components of PeS: the modular visuomotor policy design and the latent space alignment for transferable representation.

3.1 Modular Visuomotor Policy Design

Consider an environment E with observation o^E. We denote the observations from two camera views as o_1^E and o_2^E, and the robot proprioception of the end effector position, orientation, and gripper open width as o_p^E. For an MLP-based policy, we denote it as $\pi_E(a^E \mid o^E)$ parameterized by a function $\phi_E(o^E)$. For an RNN-based policy, we denote it as $\pi_E(a^E, h_{t+1}^E, c_{t+1}^E \mid o^E, h_t^E, c_t^E)$ parameterized by a function $\phi_E(o^E, h_t^E, c_t^E)$, where h_t^E is the hidden state of the RNN network at time step t and c_t^E is the cell state. The visuomotor policy can be decomposed into the visual encoder g_1^E for camera 1, the visual encoder g_2^E for camera 2, and the action decoder f^E. We can represent the MLP-based policy as Eq. 1 and the RNN-based policy as Eq. 2.

$$\phi_E(o^E) = \phi_E(o_1^E, o_2^E, o_p^E) = f^E(g_1^E(o_1^E), g_2^E(o_2^E), o_p^E), \tag{1}$$

$$\phi_E(o^E, h_t^E, c_t^E) = \phi_E(o_1^E, o_2^E, o_p^E, h_t^E, c_t^E) = f^E(g_1^E(o_1^E), g_2^E(o_2^E), o_p^E, h_t^E, c_t^E), \tag{2}$$

Without loss of generality, we demonstrate the perception stitching process with the MLP-based policy. With two visuomotor policies $f^{E_1}(g_1^{E_1}(o_1^{E_1}), g_2^{E_1}(o_2^{E_1}), o_p^{E_1})$ and $f^{E_2}(g_1^{E_2}(o_1^{E_2}), g_2^{E_2}(o_2^{E_2}), o_p^{E_2})$ in two environments E_1 and E_2 with different cameras, we define perception stitching as constructing another visuomotor policy network $f^{E_1}(g_1^{E_1}(o_1^{E_3}), g_2^{E_2}(o_2^{E_3}), o_p^{E_3})$ by initializing the visual encoder 1 with parameters from $g_1^{E_1}$, visual encoder 2 with parameters from $g_2^{E_2}$, and action decoder with parameters from f^{E_1}. Then this stitched policy is zero-shot transferred to the new environment E_3 with the same perception configuration 1 as that of E_1 and perception configuration 2 as that of E_2. For example, as shown in Fig. 2 (left half), we train policy ϕ_{E_1} in E_1 with an in-hand camera and a side-view camera, and policy ϕ_{E_2} in E_2 with a side-view camera and a front view camera. Now, if we need a policy for E_3 with an in-hand camera and a front-view camera, we stitch the visual encoder $g_2^{E_2}$ to the $g_1^{E_1}$ and f^{E_1} to form a stitched policy that directly works in E_3. Note that though we formalize PeS set up with dual-camera settings, PeS is not constrained with only two cameras, as demonstrated in our experiments in the Section 4.2. In addition, although we choose the action decoder 1 for the experiments in section 4, the choice of which action decoder to be used doesn't affect the performance of PeS, and the experiment results of comparing the influence of the two action decoders are presented in Appendix D.

3.2 Latent Space Alignment for Transferable Representation

Relative Representation Simply stitching one portion of a neural network to another neural network usually cannot yield optimal performance in the target environment due to the misalignment of latent space (Jian et al., 2023; Devin et al., 2017). Previous works have observed an approximate isometric transformation relationship between the latent representations trained with different random seeds (Moschella et al., 2022) and different robot kinematics (Jian et al., 2023). These isometric transformations include rotation, reflecting, rescaling, and translation. In this work, we observe a similar phenomenon in the latent spaces of the visual encoders. Hence, we calculate a relative representation at the latent space (Jian et al., 2023; Moschella et al., 2022) to align the latent features from different policy modules.

As shown in Fig. 2, we first collect a set $\mathbb{A}$ of anchor images $\boldsymbol{a}^{(j)}$ from the dataset $\mathbb{D}$ for the behavior cloning: $\mathbb{A} = \{\boldsymbol{a}^{(j)}\} \subseteq \mathbb{D}$. By applying the visual encoder g to both the input image $\boldsymbol{s}^{(i)}$ ($\boldsymbol{s}^{(i)} \in \mathbb{S}$) and the anchor image $\boldsymbol{a}^{(j)}$, we obtain their embedded forms $\boldsymbol{e}_{\boldsymbol{s}^{(i)}} = g(\boldsymbol{s}^{(i)})$ and $\boldsymbol{e}_{\boldsymbol{a}^{(j)}} = g(\boldsymbol{a}^{(j)})$.

We want to project the embedded input images to a coordinate system consisting of the embedded anchor images, and if this coordinate system is invariant to isometric transformations, we can alleviate the latent space misalignment issue. Therefore, we calculate a similarity score $r = \text{sim}\left(\boldsymbol{e}_{\boldsymbol{s}^{(i)}}, \boldsymbol{e}_{\boldsymbol{a}^{(j)}}\right)$ between an embedded task state and an embedded anchor state where $sim : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Then the relative representation (Jian et al., 2023; Moschella et al., 2022) of the input image $\boldsymbol{s}^{(i)}$ with respect to the anchor set $\mathbb{A}$ is given by:

$$\boldsymbol{r}_{\boldsymbol{s}} = (\text{sim}(\boldsymbol{e}_{\boldsymbol{s}^{(i)}}, \boldsymbol{e}_{\boldsymbol{a}^{(1)}}), \text{sim}(\boldsymbol{e}_{\boldsymbol{s}^{(i)}}, \boldsymbol{e}_{\boldsymbol{a}^{(2)}}), \ldots, \text{sim}(\boldsymbol{e}_{\boldsymbol{s}^{(i)}}, \boldsymbol{e}_{\boldsymbol{a}^{(|A|)}})) \tag{3}$$

We choose the cosine similarity due to its invariance to reflection, rotation, and re-scaling. Additionally, we add a normalization layer before calculating cosine similarity to mitigate the translation transformation. Therefore, this relative representation is invariant to the isometric transformation, leading to better latent space alignment.

Compared with previous works (Jian et al., 2023; Moschella et al., 2022), we develop two novel techniques for the latent space alignment of visuomotor robot policies: (1) a novel anchors selection method designed for imitation learning and (2) the use of a disentanglement regularization for better latent space alignment.

Anchors Selection. When PeS is applied to visuomotor policies with two visual encoders, these two visual encoders encode images observed in two different cameras, and our proposed anchor selection method utilizes this correspondence between these two visual configurations. As shown in Fig. 3, after collecting a dataset $\mathbb{D}_1$ in an environment E_1, we perform k-means (Hartigan & Wong, 1979) on $\mathbb{D}_1$ and select the images closest to the cluster centers as the anchors in the anchor set $\mathbb{A}_1$. We then replay the trajectories of the dataset $\mathbb{D}_1$ in another environment E_2, which requires the robot to accomplish the same task as in E_1, but the cameras are different. There is no additional policy training required due to the replay. The dataset $\mathbb{D}_2$ collected by replaying these trajectories in E_2 has states corresponding to the states in $\mathbb{D}_1$. We then use the indices of anchors in $\mathbb{A}_1$ to select the anchor set $\mathbb{A}_2$.

Disentanglement Regularization. In addition to the negative log-likelihood loss (L_{BC}) used for the standard behavior cloning algorithm, we also apply a disentanglement regularization (Wang et al., 2022) L_{disent} to further refine the latent space alignment. We first calculate the covariance of the kth and l^{th} dimension of the batch of embedded representations with

$$\operatorname{cov}(z_k, z_l) = \frac{1}{N-1} \sum_{i=1}^{N} (z_{ik} - \bar{z}_k) \cdot (z_{il} - \bar{z}_l), \tag{4}$$

where z is a batch of latent representations embedded by the ResNet before going through the relative representation calculation process. z_{ik} and z_{il} are the values of the kth and l^{th} dimension of the ith embedded data point. $\bar{z}_k$ is the mean of the kth dimension across all N data points in the batch, calculated as $\bar{z}_k = \frac{1}{N}\sum_{i=1}^{N} z_{ik}$. $\bar{z}_l$ is the mean of the lth dimension, similarly calculated. Then the disentanglement loss is calculated by:

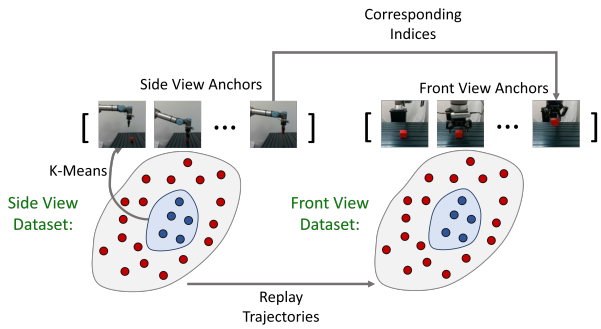


Figure 3: **Anchors Selection.** Select the anchor images in one dataset with the k-means algorithm (Hartigan & Wong, 1979). Replay the trajectories of the first dataset to collect another dataset with a different camera. We select the images with the corresponding indices of the anchors in the first dataset as the anchors in the second dataset.

$$L_{\text{disent}} = \frac{1}{Z(Z-1)} \sum_{k=1}^{Z} \sum_{l=1, l\neq k}^{Z} |\operatorname{cov}(z_k, z_l)|, \tag{5}$$

where Z is the dimensionality of the latent space. Overall, this disentanglement loss calculates the covariance of a latent representation feature z_k with all other features z_l ($l \neq k$), sums up these absolute values, normalizes it with $Z-1$, and calculates the mean over all the features z_k ($k = 1, 2, ..., Z$). By encouraging different features at the latent space to be independent with each other, we encourage them to capture different underlying factors hidden in the observation (e.g. object color, position etc.). The disentangled representation has been used in many applications in supervised learning (Tran et al., 2017; Kim & Mnih, 2018; Chen et al., 2018; Higgins et al., 2017; Quessard et al., 2020; Higgins et al., 2018; Zbontar et al., 2021; Ermolov et al., 2021; Bardes et al., 2021), and we empirically find it significantly improves the performance of PeS in difficult tasks.

The final PeS loss function is

$$L_{PeS} = L_{BC} + \lambda L_{disent}, \tag{6}$$

where we choose the weight $\lambda = 0.002$. For an ablation study, we also experiment without the disentanglement loss

$$L_{PeS(w/o\ disent.\ loss)} = L_{BC}. \tag{7}$$

We also experiment with replacing the disentanglement loss with $L1$ and $L2$ normalizations of the latent representations

$$L_{PeS(w.\ l1\&l2\ loss)} = L_{BC} + \lambda_1 \frac{1}{N} \sum_{i=1}^{N} \|z_i\|_1 + \lambda_2 \frac{1}{N} \sum_{i=1}^{N} \|z_i\|_2, \tag{8}$$

where the weights $\lambda_1 = 0.001$, $\lambda_2 = 0.001$, z_i is the ith latent representation, and N is the batch size. These L1 and L2 regularizations have been used in previous work (Nair et al., 2022) to avoid the state-distribution

shift failure in imitation learning (Ross et al., 2011) by limiting the latent space dimension. We find it improves the zero-shot transfer performance in some cases but generally doesn't perform as well as the disentanglement loss.

3.3 Implementation Details

Each input image is a $(84, 84, 3)$ RGB image. The visual encoder consists of a ResNet-18 network (He et al., 2016), followed by a spatial-softmax layer (Finn et al., 2016; Mandlekar et al., 2021), and then a 256-dimensional last layer at the module interface. We use 256 anchors to match the dimension of the latent representation. The eight-dimensional proprioception consists of the end effector position (3D), end effector quaternion (4D), and the gripper open width (1D). It is embedded by a 64-dimensional linear layer and then concatenated with two latent representations of the two images. We use the RNN-based action decoder for the Can and Door Open task and the MLP-based action decoder for other tasks. We list all the parameters of the neural network in the Appendix A with our code base. The actions are output by a Gaussian Mixture Model in the last layer of the policy network where the eight-dimensional action vector is the delta value of the 8D proprioception. Before the perception stitching and zero-shot transfer, all policies are trained to 100% success rates with BC (Pomerleau, 1988). The dataset of each task contains 200 trajectories of the expert demonstrations. The pseudo code of PeS is presented in Appendix B.

4 Experiments

Our experiments aim to (1) evaluate the effectiveness of PeS on enhancing zero-shot policy transfer on the stitched policies and (2) understand the mechanisms behind the performance advantage of PeS. Our experiments consist of five parts. First, we evaluate the zero-shot transfer performance of PeS for training double-camera-view visuomotor policies in five simulated manipulation tasks with six variations of camera configurations. Next, we apply PeS to single-camera-view and triple-camera-view policies to test its generalizability to arbitrary camera number settings. We then assess the performance of PeS in four real-world manipulation tasks with three different camera configurations. In addition, we analyze the latent representations of the visual encoders both quantitatively and qualitatively to understand the effectiveness of PeS in latent space alignment. Lastly, we adopt Gradient-weighted Class Activation Mapping (Grad-CAM) (Selvaraju et al., 2017) to visualize the regions that the visuomotor policies focus on, offering an intuitive insight into the success of PeS.

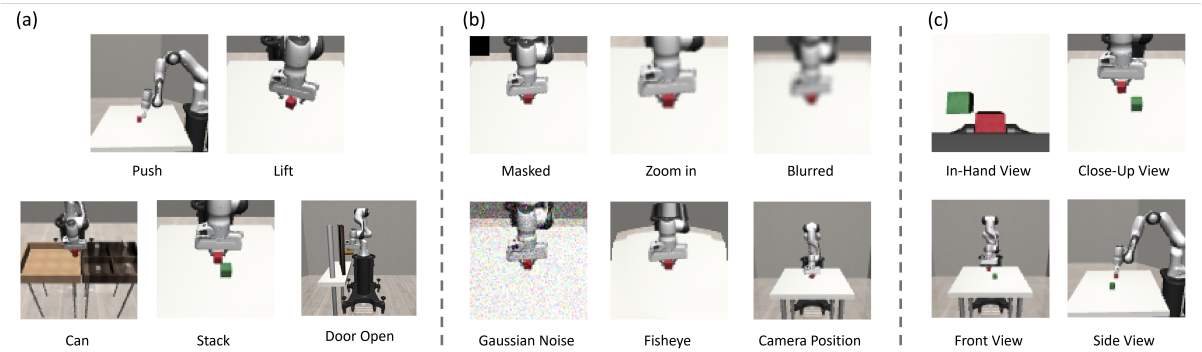


Figure 4: **Simulation Experiment Setup.** (a) Five simulation tasks from Robomimic (Mandlekar et al., 2021) benchmark. (b) Six camera configuration variations include. (c) Four camera mounting positions.

4.1 Simulation Experiments with Double Camera Views

Evaluation Tasks We evaluate PeS in five manipulation tasks from the Robomimic (Mandlekar et al., 2021) benchmark (Fig. 4(a)): (1) Push: a robot starts from a random initial position in the air and pushes the cube forward for a certain distance (30 cm in simulation, 20 cm in real world). (2) Lift: a robot lifts up a cube randomly placed on the table. (3) Can task: a robot picks up a can on its right and places it into the bins on its left. (4) Stack: a robot picks up the red cube on the table and stacks it on the green cube. (5) Door Open: a robot grabs the handle and pulls the door open.

Camera Configurations For each task, we test PeS with six camera configurations in Fig. 4(b). (1) Masked: A black square mask is put on the upper-left corner of the image to mimic a partially occluded camera lens. Although the mask makes little difference for humans, our experiments suggest a strong negative effect on current visuomotor policies. (2) Zoom in: zoom in the camera to enlarge the object in a smaller field of view. 3) Blurred: Gaussian blur effect is added to the image. (4) Gaussian Noise: Gaussian noise is added to the image to simulate low-light condition image capture. (5) Fisheye: a fisheye effect is simulated. (6) Camera Position: the camera is put in one out four possible positions as in Fig. 4(c).

The first five configurations used in-hand and close-up cameras. For example, in the Push-Blurred experiment, we first train policy 1 with a normal in-hand camera and a blurry close-up camera, and policy 2 with a blurry in-hand camera and a normal close-up camera. Then we stitch the close-up view visual encoder of the policy 2 to the in-hand view visual encoder and action decoder of the policy 1 and test this stitched policy in the Push task with a normal in-hand camera and a normal close-up camera without any fine-tuning. The last configuration is designed to test zero-shot transfer ability across different camera positions. Policy 1 is trained with an in-hand camera and a front-view camera. Policy 2 is trained with a close-up view camera and a side-view camera. Then the side-view encoder of policy 2 is stitched to the in-hand view encoder and action decoder of policy 1, and tested with the in-hand and side-view cameras.

Baselines We adopt three baseline methods and two ablation studies: (1) Devin et al. (2017) uses a task network module to embed the observation (task state) agnostic to the robot itself, concatenates it with the proprioceptive robot state, and then inputs it into the downstream robot network module. For a fair comparison, the task module and the robot module use the same structure as our PeS method. The key difference is that Devin et al. (2017) does not adopt the relative representation and does not apply our disentanglement regularization. (2) Cannistraci et al. (2023) (linear) uses four similarity functions, L_1, L_2, L_∞, *cos* to calculate four relative representations and then calculates the element-wise linear sum of the four relative representations to be the final latent state passed over to the action decoder. The weights of the L_1, L_2, L_∞, *cos* similarities are 0.05, 0.1, 0.05, and 0.8. In contrast, our PeS method only uses the cosine similarity. (3) Cannistraci et al. (2023) (nonlinear) also adopts these four similarity functions but passes them through a non-linear layer (consists of a LayerNorm, a linear layer, and a Tanh) followed by an element-wise summation. (4) The PeS (w/o disent. loss) ablation method does not have the disentanglement loss Eq. 7. (5) The PeS (w. l1 & l2 loss) replaces the disentanglement loss in PeS with L1 and L2 penalty, as shown in Eq. 8.

		Mask	Zoom in	Blurred	Noise	Fisheye	Camera Position	Average
Push	Devin et al. 2017	60.7±10.6	8.7±4.99	16.7±3.77	59.3±6.80	29.3±7.36	19.3±5.73	32.3
	Cannistraci et al. 2024 (linear)	89.3±4.11	94.0±2.83	64.7±1.89	74.7±6.18	74.0±2.83	78.7±2.49	79.2
	Cannistraci et al. 2024 (non-linear)	12.7±1.89	18.7±4.99	42.8±3.27	23.3±0.94	6.0±4.32	5.3±2.49	18.1
	PeS (w/o disent. loss)	**100.0±0.0**	86.0±2.83	80.7±9.84	**100.0±0.0**	**100.0±0.0**	**100.0±0.0**	94.5
	PeS (w. l1 & l2 loss)	88.7±4.99	95.3±1.89	90.0±5.66	**100.0±0.0**	93.3±0.94	80.7±4.99	91.3
	PeS	**100.0±0.00**	**100.0±0.00**	**95.3±0.94**	**100.0±0.0**	92.7±2.50	**100.0±0.00**	**98**
Lift	Devin et al. 2017	0.0±0.00	5.3±2.49	48.0±5.89	9.3±4.11	14.7±4.99	36.0±1.63	18.9
	Cannistraci et al. 2024 (linear)	72.7±3.77	64.0±2.83	86.0±4.32	68.7±1.88	88.7±1.88	57.3±2.49	72.9
	Cannistraci et al. 2024 (non-linear)	89.3±2.49	36.0±3.27	52.7±3.40	93.3±2.49	16.7±2.49	21.3±0.94	51.6
	PeS (w/o disent. loss)	83.3±6.60	80.7±5.73	**93.3±0.94**	91.3±5.73	79.3±2.49	**93.3±2.49**	86.9
	PeS (w. l1 & l2 loss)	**97.3±2.49**	85.3±0.94	90.7±0.94	86.0±4.32	88.0±1.63	84.7±3.77	88.7
	PeS	92.7±2.50	**94.7±1.89**	89.3±4.11	**96.0±1.63**	**88.7±0.94**	93.0±0.03	**92.4**

Table 1: **Zero-Shot Transfer Success Rates in basic Simulation tasks.** We test all methods across six different visual configurations and also calculate the average performance. In the two basic tasks, Push and Lift, PeS and its two ablation methods can all get about a 90% success rate on average. The Cannistraci et al. 2024 (linear) baseline has 70% - 80% of success rate, while the Cannistraci et al. 2024 (non-linear) and Devin et al. 2017 baselines perform poorly with about 20% - 50% of success rate.

Results Tab. 1 shows that PeS and its two ablation methods can all achieve satisfying performance with around 90% of success rates on Push and Lift. Cannistraci et al. 2024 (linear) has lower success rates around 70% - 80%, suggesting that introducing a linear combination of multiple similarity measurements for the relative representation hurts the model's performance. Since isometric transformation is the major transformation observed at the latent space, the single cosine similarity of PeS offers simple but effective features. Cannistraci et al. 2024 (non-linear) performs worse than the linear baseline and reaches about 20% to 50% success rates. We hypothesize that the nonlinear combination introduces learnable parameters for

		Mask	Zoom in	Blurred	Noise	Fisheye	Camera Position	Average
Can	Devin et al. 2017	19.3±5.25	24.7±1.89	2.67±1.89	6.0±4.32	29.3±3.40	1.3±1.89	13.9
	Cannistraci et al. 2024 (linear)	33.3±0.94	48.0±1.63	48.7±2.49	65.3±0.94	26.7±3.77	34.7±3.77	42.8
	Cannistraci et al. 2024 (non-linear)	72.7±0.94	24.7±2.49	37.3±4.99	42.7±3.40	8.7±1.89	39.3±1.89	37.6
	PeS (w/o disent. loss)	44.7±8.06	**89.3±4.11**	34.7±4.11	30.7±6.80	**92.7±2.50**	44.7±3.40	56.1
	PeS (w. l1 & l2 loss)	47.3±0.94	58.7±1.88	54.0±8.64	36.0±7.12	58.7±1.88	64.7±6.60	53.2
	PeS	**83.3±5.24**	**89.3±2.49**	**74.0±2.83**	**78.7±4.11**	56.0±2.83	**78.7±2.49**	**76.7**
Stack	Devin et al. 2017	0.7±0.94	8.0±1.63	0.7±0.94	24.0±2.83	0.0±0.00	14.0±3.27	7.9
	Cannistraci et al. 2024 (linear)	47.3±0.94	62.0±4.32	32.7±3.77	30.7±0.94	54.0±8.64	14.7±6.18	40.2
	Cannistraci et al. 2024 (non-linear)	10.0±1.63	12.0±0.00	0.0±0.00	3.3±0.94	0.0±0.00	0.7±0.94	4.3
	PeS (w/o disent. loss)	34.0±11.43	10.7±4.11	62.0±10.71	34.0±7.12	22.7±3.77	26.0±4.32	31.6
	PeS (w. l1 & l2 loss)	92.7±0.94	**98.0±0.00**	62.7±6.60	24.0±4.90	59.3±7.36	58.7±1.88	65.9
	PeS	**94.7±0.94**	96.7±0.94	**90.0±1.63**	**96.7±1.89**	**97.3±2.49**	**80.0±4.90**	**92.6**
Door Open	Devin et al. 2017	9.3±4.11	5.3±0.94	0.0±0.00	4.0±1.63	0.7±0.94	0.0±0.00	3.2
	Cannistraci et al. 2024 (linear)	0.0±0.00	1.3±0.94	10.7±2.49	10.7±4.99	2.0±1.63	47.3±9.29	12
	Cannistraci et al. 2024 (non-linear)	26.0±2.83	31.3±4.99	49.3±8.22	48.0±5.89	62.7±3.40	44.7±3.40	43.7
	PeS (w/o disent. loss)	24.7±7.71	44.0±2.83	34.7±3.77	0.7±0.94	36.7±0.94	23.3±3.40	27.4
	PeS (w. l1 & l2 loss)	4.0±1.63	**78.0±5.66**	3.3±0.94	2.0±1.63	42.7±4.99	6.0±3.26	22.7
	PeS	**58.7±4.11**	68.7±0.94	**70.7±0.94**	**52.7±3.40**	**64.7±4.99**	**48.7±3.40**	**60.7**

Table 2: **Zero-Shot Transfer Success Rates in difficult Simulation tasks.** The three difficult tasks include Can, Stack, and Door Open. We test all methods across six different visual configurations and also calculate the average performance. The three baselines achieve about 0% - 40% of success rates in these tasks. In the Can and Stack tasks, PeS achieves 75% - 95% of success rates, while the ablation methods achieves about 30% - 65% of success rates. In the Door task, PeS achieves 60.7% of success rates, while the two ablation methods only have 20% - 30% of success rates.

the non-linear summation process. However, these parameters are not optimized to align the latent spaces but are optimized for the accuracy of behavior cloning, leading to unaligned latent space and hurting the performance. Devin et al. 2017 baseline achieves very low success rates between 18% to 33%, indicating that the unaligned latent representations cause the failure of the stitched policy for zero-shot transfer.

Tab. 2 reports the performance in more difficult tasks: Can, Stack, and Door Open. PeS achieves 60% to 93% of success rates, while the success rates of all three baselines are lower than 43%. This result suggests that previous methods cannot solve the zero-shot transfer problem for difficult tasks with satisfactory performance, which usually involve complicated visual background (e.g., Can), long-horizon multi-stage motions (e.g., Stack), and articulated objects (e.g., Door Open). We also notice that the two ablation methods have significant performance drops (20% to 60%) in these difficult tasks, showing that the disentanglement regularization in PeS largely improves the zero-shot transfer performance. Although using L1 and L2 regularization also leads to good success rates in some cases, the disentanglement regularization has more consistent performance and higher success rates on average.

4.2 Simulation Experiments with Single Camera View and Triple Camera Views

Although most visuomotor policies adopt two cameras in their applications, we still carry out experiments for policies with only one camera and policies with three cameras to verify the generalizability of PeS to arbitrary camera number settings.

Single Camera View We train two policies with two different cameras separately. Each policy still has two visual encoders, but one visual encoder takes in a black image and the other encoder takes in the images from the single camera. For both encoders of the policy, they use

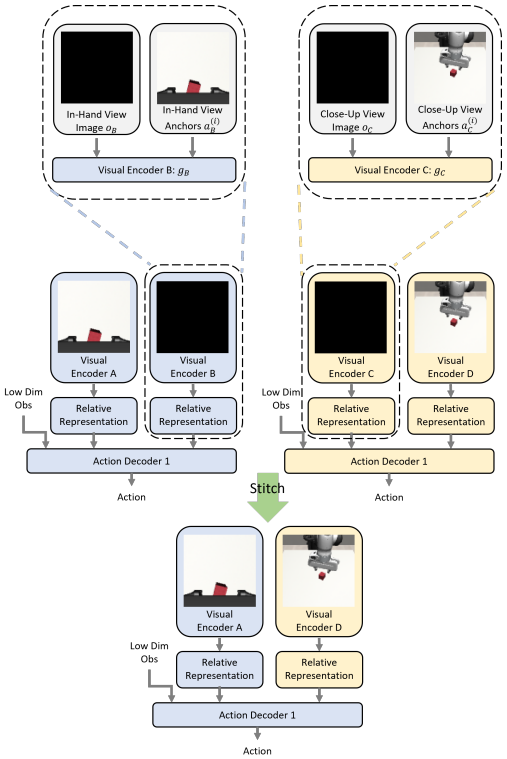


Figure 5: **Perception Stitching with Single Camera.** The original two policies are trained with only one camera. The other visual encoder takes in a black image. The corresponding anchors are selected with our proposed method 3. The visual encoder of the black image uses the same anchor images as the other encoder of this policy.

the same anchor images set collected by the only camera, because we assume that each policy is trained separately and only has access to the dataset of one camera.

In the perception stitching process, we stitch the visual encoder of policy 2 to the visual encoder and action decoder of policy 1, as shown in Figure 5, and assemble a stitched policy for double camera inputs. We suggest that this is the most reasonable setting for the perception stitching of single-camera policies. In contrast, if we remove the black image encoder and stitch the only vision encoder of the policy 1 to the action decoder of the policy 2, applying this reassembled policy in either environment 1 or 2 won't lead to better performance compared with the original policy trained to a maximum success rate in that environment.

	Push	Lift	Can	Stack	Door Open	Average
Devin et al. 2017	73.3±11.47	72.7±0.94	66.7±6.18	14.0±3.27	22.7±2.50	49.8
Cannistraci et al. 2024 (linear)	89.3±4.11	86.0±1.63	72.8±3.27	14.7±6.18	39.3±3.40	60.4
Cannistraci et al. 2024 (non-linear)	60.7±1.88	80.7±3.40	24.0±2.83	0.7±0.94	41.3±4.99	41.5
PeS (-w/o disent. loss)	88.7±4.11	**94.0±1.63**	91.0±3.40	26.0±4.32	32.7±3.77	66.5
PeS (w. l1 & l2 loss)	**94.7±2.49**	90.0±2.83	86.7±1.89	58.7±1.88	40.7±0.94	74.2
PeS	91.3±2.49	92.6±0.94	**94.6±0.94**	**80.0±4.90**	**72.7±3.77**	**86.2**

Table 3: **Zero-Shot Transfer Success Rates of single camera policies.** PeS achieves optimal performance in the three difficult tasks (Can, Stack, Door Open), and close-to optimal performance in the two basic tasks (Push, Lift). Its average success rate outperforms all the baselines and the ablation methods.

As shown in table 3, PeS outperforms the other methods in the three difficult tasks and achieves close-to-optimal performance in the two basic tasks. It has the highest average success rate over the five tasks.

	Mask	Zoom in	Blurred	Noise	Fisheye	Camera Position	Average
Devin et al. 2017	4.0±1.63	5.3±2.49	1.3±0.94	13.3±1.89	1.3±1.89	8.0±1.63	5.5
Cannistraci et al. 2024 (linear)	46.7±3.40	20.7±1.89	13.3±1.89	56.0±2.83	16.0±4.32	30.0±4.32	30.5
Cannistraci et al. 2024 (non-linear)	30.7±0.94	22.7±0.94	19.3±4.11	42.7±1.89	12.7±0.94	4.7±2.49	22.1
PeS (-w/o disent. loss)	21.3±2.49	20.7±1.89	72.7±3.77	56.7±4.11	37.3±4.99	9.3±1.89	36.3
PeS (w. l1 & l2 loss)	46.7±3.40	56.7±4.11	84.0±2.83	82.7±2.49	75.3±2.49	61.3±3.40	67.8
PeS	**97.3±2.49**	**92.7±0.94**	**93.3±0.94**	**90.0±1.63**	**94.6±0.94**	**86.7±2.49**	**92.4**

Table 4: **Zero-Shot Transfer Success Rates of triple camera policies.** All the experiments are carried out with the Stack task. PeS achieves optimal performance for all the six different visual configurations. Its average success rate has an around 25% advantage to the second best method.

Triple Camera Views We pick the Stack task and train policies with three visual encoders that embed three different visual configurations. For the first five visual configurations that add different effects to the observations, policies are all trained with the in-hand view, close-up view, and side view. Policy 1 has the effect (e.g. Gaussian noise) only in vision 3, and policy 2 has the same effect in vision 1 and vision 2. Then we stitch the encoder 3 of policy 2 to the rest parts of policy 1 and test the stitched policy with three normal cameras. For the last experiment that involves different camera positions, policy 1 is trained with the in-hand view, close-up view, and front view, and policy 2 is trained with close-up view, front view, and side view. Then we stitch the side view encoder of policy 2 to the in-hand view, close-up view encoders, and decoder of policy 1. The stitched policy is tested with in-hand view, close-up view, and side view. This triple camera perception stitching process is presented in Figure 6.

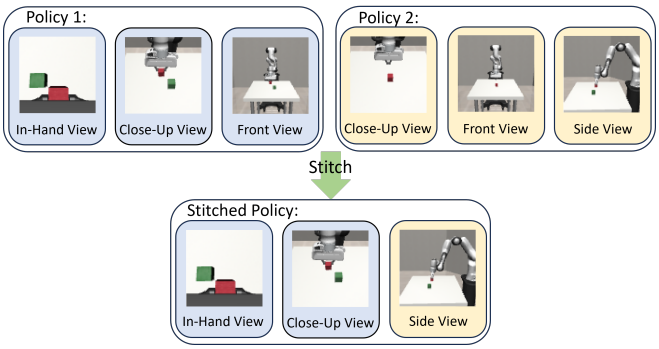


Figure 6: **Perception Stitching with Three Cameras.** The original two policies are trained with three cameras. In the Perception Stitching process, the visual encoder 3 of the policy 2 is stitched to the visual encoder 1 and 2 and the action decoder of the policy 1.

Table 4 reports the zero-shot transfer success rates for the Stack task with six different visual configurations. The PeS method consistently outperforms the other baseline and ablation methods for all six visual config-

urations. On average, it reaches a 92.4% success rate on the stacking task, which is about 25% higher than the second-best method.

4.3 Real World Experiment

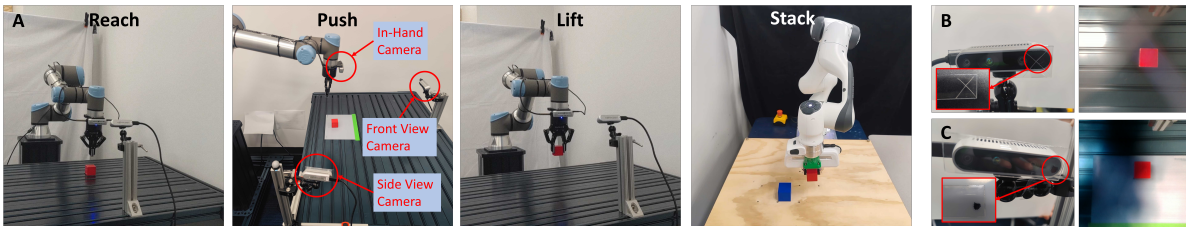


Figure 7: (A) Real-World Tasks: reaching, pushing, lifting, and stacking a cube. Three cameras are mounted at the front view, side view, and in-hand positions separately. (B) Broken lens. (C) Masked lens.

Camera Configurations We conducted Reach, Push, Lift, and Stack in our real-world experiments (Fig. 7). In Reach, the robot starts from some random position in the air and reaches the red cube. In this experiment, the policy 1 is trained with a front-view camera with a broken lens (Fig. 7) and a normal in-hand camera. The policy 2 is trained with a normal front-view camera and an in-hand camera with a broken lens. Then we stitch the front view encoder of the policy 2 to the in-hand view encoder and action decoder of the policy 1 and test the stitched policy with two normal cameras. In Push, the robot starts from some random position to reach the cube and then pushes the cube across the green line on the table. The training and testing process of Push is similar to that of the Reach, and the only difference is replacing the camera with a broken lens with a camera with a masked lens, as shown in Fig. 7. In Lift, the red cube is placed at a random position on the table with a random orientation. The robot needs to grab the cube and lift it up. In this experiment, the policy 1 is trained with in-hand and side-view cameras. The policy 2 is trained with side-view and front-view cameras. Then we stitch the front view encoder of the policy 2 to the in-hand view encoder and action decoder of the policy 1, and test the stitched policy with an in-hand camera and a front-view camera, as shown in the left half part of the Fig. 2. In Stack, the red cube is placed at a random position on the left side of the table, and the robot should lift it up and stack it on the blue cube on the right. Similar to Lift, it also involves the stitching between cameras at different positions (left, front, and right). Stack is the most difficult one in all the real-world experiments. It requires the most sophisticated manipulation skills and long-horizon multi-stage motions.

Results As shown in Tab. 5, PeS achieves 100% of success rate in Reach, 85% in Push, 80% in Lift, and 45% in Stack, while Devin et al. 2017 baseline gets 0% of success rates in all the experiments. In the experiments, we observe that PeS policies show more accurate motions compared with the baseline policies. In comparison, Devin et al. 2017 baseline policies cannot output actions accurately enough to accomplish the tasks, although they have some attempts to complete the tasks in some cases. Take the Lift task as an example, the robot trained with the baseline method reaches some positions close to the cube but always fails to grasp it. In the most difficult Stack task, PeS robot can still achieves many success cases, while the baseline robot cannot output meaningful motions. Please refer to the supplementary material for the experiment videos.

	Reach broken lens	Push masked lens	Lift different positions	Stack different positions
PeS	**100.0**	**85.0**	**80.0**	**45.0**
Devin et al. 2017	0.0	0.0	0.0	0.0

Table 5: **Zero-Shot Transfer Success Rates in Real World.** We report the success rates of three manipulation tasks in real world. Each success rate is calculated with 20 games. PeS achieves a 100% success rate in the easiest Reach task, 85% in the Push task, 80% in the Lift task, and 45% in the hardest Stack task. In comparison, the baseline method doesn't have any success case in all these tasks.

In summary, we find that the performance advantage of PeS is more pronounced in the real world than that in the simulation. We assume that this is because there are more noises and disturbances in the real world. To the best of our knowledge, PeS is the first method that enables vision-based zero-shot transfer in the real world via reassembling neural network components.

4.4 Analysis: Latent Space at Module Interface

To understand the mechanism behind the high success rate of PeS, we perform visualization and quantitative analysis of the latent representations of the visual encoders.

We choose the Push-Camera Position experiment in the simulation and visualize the corresponding latent representations to have an intuitive understanding of the mechanism of PeS. We first reduced the 256D representations to 3D with PCA(Hotelling, 1933) for visualization. Since the side view encoder of the policy 2 is stitched to the policy 1 at the position of its original front view encoder, we compare the latent representations of these two encoders. As shown in Fig. 8(a), the latent representations trained with PeS have similar shapes to each other. In comparison, the latent representations with the Devin et al. 2017 baseline are not similar but have an approximately isometric transformation (rotation in this case) relationship with each other.

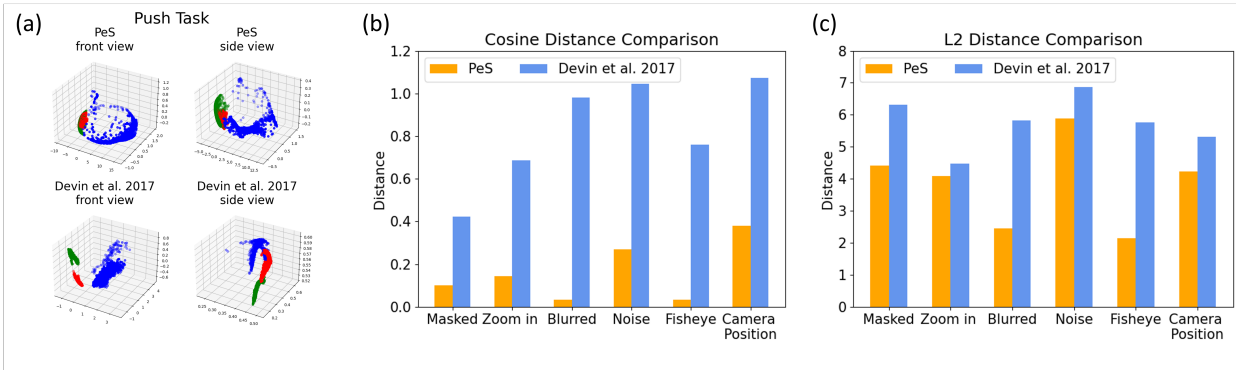
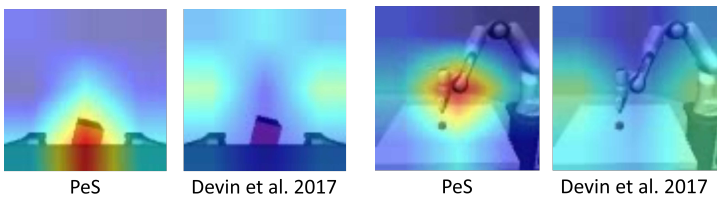


Figure 8: **Latent Space Analysis.** (a) We visualize the latent representations of the front-view encoder of policy 1 and the side-view encoder of policy 2 of the Push-Camera Position experiment. The 256D representations are reduced to 3D with PCA (Hotelling, 1933). The red dots represent samples where the robot's end effector is at higher positions, blue dots indicate medium heights and green dots correspond to lower positions near the cube. We compare our (PeS) method with the baseline (Devin et al. 2017) method. (b) & (c) We compare the distances of the latent representations in all the experiments in the Push task. One representation is from the second view encoder of policy 1 and the other is from the second view encoder of policy 2.)

For further quantitative analysis, we select the Push task in the simulation and calculate the pairwise distances of the latent representations in all Push experiments with different visual configurations. Fig. 8 (b) shows that PeS significantly reduces the cosine distances of the latent representations in all these experiments. Fig. 8 (c) shows that the L2 distances with PeS are generally smaller than that with the Devin et al. 2017 baseline, but the differences are not distinguishable in some cases (e.g. Push-Zoom in). More mathematical details of calculating the cosine and L2 distances can be found in Appendix C.

4.5 Analysis: Highlight Attention Regions with Grad-CAM

We visualize the attention map from the policy modules to further explain why certain policies work well while others fail. To this end, we modify the Gradient-weighted Class Activation Mapping (Grad-CAM) (Selvaraju et al., 2017) approach to highlight the regions that the policies pay attention to. Grad-CAM is widely applied to neural classification models with convolutional layers. To adapt the Grad-CAM from image classification to robot learning, we replace the before-softmax score y^c for class c of the image classification networks with the log-likelihood $l(a)$ of the robot action a in the training dataset. We denote the kth feature



(a) In-hand View (b) Side View

Figure 9: **Attention Heatmap.** (a) In the in-hand view, our policy pays attention to the cube on the table, while the baseline policy pays meaningless attention to the upper region of the image. (b) In the front view image, our policy pays attention to the robot end effector, while the baseline method has slightly more attention to the two sides of the image.

map activation output from the last convolutional layer as A^k. Then the backpropagated gradient of $l(a)$ with respect to A^k is computed as $\frac{\partial l(a)}{\partial A^k}$. We do global average pooling of these gradients over the width (indexed by i) and height (indexed by j) dimensions of the feature map to get the neuron importance weight :

$$\alpha_k^a = \overbrace{\frac{1}{Z}\sum_i\sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial l(a)}{\partial A_{ij}^k}}_{\text{gradients via backprop}} . \tag{9}$$

This weight α_k^a captures the "importance" of feature map k for robot action a. Then, the attention map $L^a_{\text{Grad-CAM}}$ is calculated as the weighted combination of forward activation maps followed by a ReLU (Selvaraju et al., 2017):

$$L^a_{\text{Grad-CAM}} = \text{ReLU} \underbrace{\left(\sum_k \alpha_k^a A^k\right)}_{\text{linear combination}} . \tag{10}$$

We apply ReLU because we are only interested in the features that have a positive influence on the actions. The intensity of these pixels should be increased in order to increase the log-likelihood $l(a)$ (Selvaraju et al., 2017). This $L^a_{\text{Grad-CAM}}$ is a heatmap of the same size as the convolutional feature maps A^k. We upsample it to the input image size with bilinear interpolation to get the final attention heatmap of the input image (Selvaraju et al., 2017). A larger value on this heatmap means this pixel contributes to a larger gradient of the log-likelihood of the robot action.

We choose the Lift-Camera Position experiment and visualize the attention heatmap of the stitched policies with PeS and the baseline separately. As shown in Fig. 9(a), from the in-hand camera, our policy is focusing on the cube between the two gripper fingers, which is intuitively what humans pay attention to in a Lift task. In comparison, the Devin et al. 2017 baseline policy has more attention to the upper middle part of the image, which is the desk surface and is not informative for this task. In Fig. 9(b), our policy is paying attention to the robot end effector from the side-view camera, while the baseline policy has more attention to the two sides of the image, while these regions have no crucial object. To sum up, the attention maps from the Grad-CAM suggest that the stitched policy with PeS performs better than the baseline because it can pay attention to the crucial regions for accomplishing the task, while the baseline policy cannot. This result provides an intuitive explanation of the good performance of PeS. Videos of the attention heatmaps during the manipulation process can be found in the supplementary material.

5 Discussion

Conclusion We present Perception Stitching (PeS), a method for zero-shot visuomotor policies transfer via latent spaces alignment. PeS aligns the latent spaces of different visual encoders by enforcing the relative representations invariant to isometric transformations and, therefore, allows the trained visual encoders to be reused in a plug-and-go manner. Our evaluation covers 30 simulation experiments and 4 real-world experiments with a variety of manipulation tasks and camera configurations. The results demonstrate significant performance improvement with PeS for zero-shot transfer, and its advantage is especially pronounced in real-world tasks. Moreover, we conduct quantitative and qualitative analysis to understand the mechanism of the superior performance of PeS. We hope that this work can inspire further exploration of the compositionality and modularity in robot learning.

Limitations and Future Work Though there is no additional training required during transfer, in order to obtain the anchor states, our framework still requires the robot to replay the trajectories in the previous dataset to collect a new dataset and then select anchors with the corresponding indices. Although this is feasible, as we have demonstrated in our real-world experiments, the trajectories replaying process in the real world usually takes longer time than collecting a new dataset by random sampling. Future work can develop an alternative algorithm for more efficient anchor selection. Another promising direction is to explore the combination of PeS with other robot transfer learning techniques that focus beyond vision encoder transfer, such as embodiment transfer and scale to large robot learning datasets.

References

Pooya Abolghasemi, Amir Mazaheri, Mubarak Shah, and Ladislau Boloni. Pay attention!-robustifying a deep visuomotor policy through task-focused visual attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4254–4262, 2019.

Ferran Alet, Tomás Lozano-Pérez, and Leslie P Kaelbling. Modular meta-learning. In *Conference on robot learning*, pp. 856–868. PMLR, 2018.

Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pp. 103–129, 1995.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Irene Cannistraci, Luca Moschella, Marco Fumero, Valentino Maiorca, and Emanuele Rodolà. From bricks to bridges: Product of invariances to enhance latent space communication. *arXiv preprint arXiv:2310.01211*, 2023.

Boyuan Chen, Robert Kwiatkowski, Carl Vondrick, and Hod Lipson. Fully body visual self-modeling of robot morphologies. *Science Robotics*, 7(68):eabn1944, 2022.

Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, 31, 2018.

Yutian Chen, Abram L Friesen, Feryal Behbahani, Arnaud Doucet, David Budden, Matthew Hoffman, and Nando de Freitas. Modular meta-learning with shrinkage. *Advances in Neural Information Processing Systems*, 33:2858–2869, 2020.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

Sudeep Dasari and Abhinav Gupta. Transformers for one-shot visual imitation. In *Conference on Robot Learning*, pp. 2071–2084. PMLR, 2021.

Coline Devin. *Compositionality and Modularity for Robot Learning*. PhD thesis, EECS Department, University of California, Berkeley, Dec 2020. URL `http://www2.eecs.berkeley.edu/Pubs/TechRpts/2020/EECS-2020-207.html`.

Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 2169–2176. IEEE, 2017.

Finale Doshi-Velez and George Konidaris. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *IJCAI: proceedings of the conference*, volume 2016, pp. 1432. NIH Public Access, 2016.

Maximilian Du, Suraj Nair, Dorsa Sadigh, and Chelsea Finn. Behavior retrieval: Few-shot imitation learning by querying unlabeled datasets. *arXiv preprint arXiv:2304.08742*, 2023.

Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *arXiv preprint arXiv:1812.00568*, 2018.

Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. In *International conference on machine learning*, pp. 3015–3024. PMLR, 2021.

Fernando Fernández and Manuela Veloso. Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pp. 720–727, 2006.

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 512–519. IEEE, 2016.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017a.

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *Conference on robot learning*, pp. 357–368. PMLR, 2017b.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on Robot Learning*, pp. 158–168. PMLR, 2022.

Zipeng Fu, Tony Z Zhao, and Chelsea Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. *arXiv preprint arXiv:2401.02117*, 2024.

Abhishek Gupta, Coline Devin, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. *arXiv preprint arXiv:1703.02949*, 2017.

Aleksi Hämäläinen, Karol Arndt, Ali Ghadirzadeh, and Ville Kyrki. Affordance learning for end-to-end visuomotor robot control. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1781–1788. IEEE, 2019.

John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR (Poster)*, 3, 2017.

Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230*, 2018.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

Kyle Hsu, Moo Jin Kim, Rafael Rafailov, Jiajun Wu, and Chelsea Finn. Vision-based manipulators need to also see from their hands. *arXiv preprint arXiv:2203.12677*, 2022.

Yuhang Hu, Boyuan Chen, and Hod Lipson. Egocentric visual self-modeling for legged robot locomotion. *arXiv preprint arXiv:2207.03386*, 2022.

Marcel Hussing, Jorge A Mendez, Anisha Singrodia, Cassandra Kent, and Eric Eaton. Robotic manipulation datasets for offline compositional reinforcement learning. *arXiv preprint arXiv:2307.07091*, 2023.

Divye Jain, Andrew Li, Shivam Singhal, Aravind Rajeswaran, Vikash Kumar, and Emanuel Todorov. Learning deep visuomotor policies for dexterous hand manipulation. In *2019 international conference on robotics and automation (ICRA)*, pp. 3636–3643. IEEE, 2019.

Stephen James, Andrew J Davison, and Edward Johns. Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task. In *Conference on Robot Learning*, pp. 334–343. PMLR, 2017.

Stephen James, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12627–12637, 2019.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pp. 991–1002. PMLR, 2022.

Pingcheng Jian, Easop Lee, Zachary Bell, Michael M Zavlanos, and Boyuan Chen. Policy stitching: Learning transferable robot policies. In *Conference on Robot Learning*, pp. 3789–3808. PMLR, 2023.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

Taylor W Killian, Samuel Daulton, George Konidaris, and Finale Doshi-Velez. Robust and efficient transfer learning with hidden parameter markov decision processes. *Advances in neural information processing systems*, 30, 2017.

Heecheol Kim, Yoshiyuki Ohmura, and Yasuo Kuniyoshi. Transformer-based deep imitation learning for dual-arm robot manipulation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8965–8972. IEEE, 2021.

Heecheol Kim, Yoshiyuki Ohmura, Akihiko Nagakubo, and Yasuo Kuniyoshi. Training robots without robots: deep imitation learning for master-to-robot policy transfer. *IEEE Robotics and Automation Letters*, 8(5): 2906–2913, 2023.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *International conference on machine learning*, pp. 2649–2658. PMLR, 2018.

Kuno Kim, Yihong Gu, Jiaming Song, Shengjia Zhao, and Stefano Ermon. Domain adaptive imitation learning. In *International Conference on Machine Learning*, pp. 5286–5295. PMLR, 2020.

George Konidaris and Andrew Barto. Autonomous shaping: Knowledge transfer in reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pp. 489–496, 2006.

George Dimitri Konidaris and Andrew G Barto. Building portable options: Skill transfer in reinforcement learning. In *Ijcai*, volume 7, pp. 895–900, 2007.

Robert Kwiatkowski, Yuhang Hu, Boyuan Chen, and Hod Lipson. On the origins of self-modeling. *arXiv preprint arXiv:2209.02010*, 2022.

Alessandro Lazaric, Marcello Restelli, and Andrea Bonarini. Transfer of samples in batch reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 544–551, 2008.

Miguel Lázaro-Gredilla, Dianhuan Lin, J Swaroop Guntupalli, and Dileep George. Beyond imitation: Zero-shot task transfer on robots by learning concepts as cognitive programs. *Science Robotics*, 4(26):eaav3150, 2019.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1 (4):541–551, 1989.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

Xiang Li, Varun Belagali, Jinghuan Shang, and Michael S Ryoo. Crossway diffusion: Improving diffusion-based visuomotor policy via self-supervised learning. *arXiv preprint arXiv:2307.01849*, 2023.

Chenyu Liu, Yan Zhang, Yi Shen, and Michael M Zavlanos. Learning without knowing: Unobserved context in continuous transfer reinforcement learning. In *Learning for Dynamics and Control*, pp. 791–802. PMLR, 2021.

Corey Lynch and Pierre Sermanet. Grounding language in play. *arXiv preprint arXiv:2005.07648*, 3, 2020.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J Pal, and Liam Paull. Active domain randomization. In *Conference on Robot Learning*, pp. 1162–1176. PMLR, 2020.

Jorge A Mendez and Eric Eaton. How to reuse and compose knowledge for a lifetime of tasks: A survey on continual learning and functional composition. *arXiv preprint arXiv:2207.07730*, 2022.

Jorge A Mendez, Harm van Seijen, and Eric Eaton. Modular lifelong reinforcement learning via neural composition. *arXiv preprint arXiv:2207.00429*, 2022.

Jorge Armando Méndez. *Lifelong Machine Learning of Functionally Compositional Structures*. PhD thesis, University of Pennsylvania, 2022.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodola. Relative representations enable zero-shot latent space communication. *arXiv preprint arXiv:2209.15430*, 2022.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

Phuong DH Nguyen, Tobias Fischer, Hyung Jin Chang, Ugo Pattacini, Giorgio Metta, and Yiannis Demiris. Transferring visuomotor learning from simulation to the real world for robotics manipulation tasks. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6667–6674. IEEE, 2018.

Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

Deepak Pathak, Parsa Mahmoudieh, Guanghao Luo, Pulkit Agrawal, Dian Chen, Yide Shentu, Evan Shelhamer, Jitendra Malik, Alexei A Efros, and Trevor Darrell. Zero-shot visual imitation. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 2050–2053, 2018.

Jonas Pfeiffer, Sebastian Ruder, Ivan Vulić, and Edoardo Ponti. Modular deep learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=z9EkXfvxta. Survey Certification.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

Robin Quessard, Thomas Barrett, and William Clements. Learning disentangled representations and group structure of dynamical environments. *Advances in Neural Information Processing Systems*, 33:19727–19737, 2020.

Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Offline reinforcement learning from images with latent space models. In *Learning for Dynamics and Control*, pp. 1154–1168. PMLR, 2021.

Kartik Ramachandruni, Madhu Babu, Anima Majumder, Samrat Dutta, and Swagat Kumar. Attentive task-net: Self supervised task-attention network for imitation learning using video demonstration. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4760–4766. IEEE, 2020.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Andrei A Rusu, Matej Večerík, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell. Sim-to-real robot learning from pixels with progressive nets. In *Conference on robot learning*, pp. 262–270. PMLR, 2017.

Fereshteh Sadeghi, Alexander Toshev, Eric Jang, and Sergey Levine. Sim2real viewpoint invariant visual servoing by recurrent control. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4691–4699, 2018.

Stefan Schaal. Learning from demonstration. *Advances in neural information processing systems*, 9, 1996.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 1134–1141. IEEE, 2018.

Lin Shao, Toki Migimatsu, Qiang Zhang, Karen Yang, and Jeannette Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434, 2021.

Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks: Learning generalizable representations for visuomotor control. In *International Conference on Machine Learning*, pp. 4732–4741. PMLR, 2018.

Simon Stepputtis, Joseph Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and Heni Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, 33:13139–13150, 2020.

Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27*, pp. 270–279. Springer, 2018.

Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7), 2009.

Andrea Tirinzoni, Rafael Rodriguez Sanchez, and Marcello Restelli. Transfer of value functions via variational methods. *Advances in Neural Information Processing Systems*, 31, 2018.

Josh Tobin, Lukas Biewald, Rocky Duan, Marcin Andrychowicz, Ankur Handa, Vikash Kumar, Bob McGrew, Alex Ray, Jonas Schneider, Peter Welinder, et al. Domain randomization and generative models for robotic grasping. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3482–3489. IEEE, 2018.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.

Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1415–1424, 2017.

Xin Wang, Hong Chen, Si'ao Tang, Zihao Wu, and Wenwu Zhu. Disentangled representation learning. *arXiv preprint arXiv:2211.11695*, 2022.

Ruihan Yang, Huazhe Xu, Yi Wu, and Xiaolong Wang. Multi-task reinforcement learning with soft modularization. *Advances in Neural Information Processing Systems*, 33:4767–4777, 2020.

Tianhe Yu, Chelsea Finn, Annie Xie, Sudeep Dasari, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. *arXiv preprint arXiv:1802.01557*, 2018.

Youssef Zaky, Gaurav Paruthi, Bryan Tripp, and James Bergstra. Active perception and representation for robotic manipulation. *arXiv preprint arXiv:2003.06734*, 2020.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International conference on machine learning*, pp. 12310–12320. PMLR, 2021.

Jingwei Zhang, Lei Tai, Peng Yun, Yufeng Xiong, Ming Liu, Joschka Boedecker, and Wolfram Burgard. Vr-goggles for robots: Real-to-sim domain adaptation for visual control. *IEEE Robotics and Automation Letters*, 4(2):1148–1155, 2019.

Yan Zhang and Michael M Zavlanos. Transfer reinforcement learning under unobserved contextual information. In *2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*, pp. 75–86. IEEE, 2020.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. Viola: Object-centric imitation learning for vision-based robot manipulation. In *Conference on Robot Learning*, pp. 1199–1210. PMLR, 2023.

Yuke Zhu, Ziyu Wang, Josh Merel, Andrei Rusu, Tom Erez, Serkan Cabi, Saran Tunyasuvunakool, János Kramár, Raia Hadsell, Nando de Freitas, et al. Reinforcement and imitation learning for diverse visuomotor skills. *arXiv preprint arXiv:1802.09564*, 2018.