Appendix

A Motion-Invariant Loss

The motion-invariant transform $\phi(\cdot)$, used to compute $\mathcal{L}_{\text{invar}}$ in Equation 5, follows the DHB motion-invariant framework [4]. Given trajectories $\{u_k\}_{k=t-T}^{t}$ with $T \geq 2$, we compute the relative position p_k and orientation r_k of the gripper with respect to the initial frame at $t-T$, where p_{t-T} and r_{t-T} are at the origin.

The differences $\Delta\mathbf{p}_k = \mathbf{p}_{k+1} - \mathbf{p}_k$ and $\Delta\mathbf{r}_k = \mathbf{r}_{k+1} - \mathbf{r}_k$ represent the linear and angular trajectory changes between $k+1$ and k. The initial linear frames are defined as:

$$\begin{aligned}
\hat{\mathbf{x}}_{p,k} &= \frac{\Delta\mathbf{p}_k}{\|\Delta\mathbf{p}_k\|}, \\
\hat{\mathbf{y}}_{p,k} &= \frac{\hat{\mathbf{x}}_{p,k} \times \hat{\mathbf{x}}_{p,k+1}}{\|\hat{\mathbf{x}}_{p,k} \times \hat{\mathbf{x}}_{p,k+1}\|}, \\
\hat{\mathbf{z}}_{p,k} &= \hat{\mathbf{x}}_{p,k} \times \hat{\mathbf{y}}_{p,k}.
\end{aligned}$$

Similarly, the initial angular frames are:

$$\begin{aligned}
\hat{\mathbf{x}}_{r,k} &= \frac{\Delta\mathbf{r}_k}{\|\Delta\mathbf{r}_k\|}, \\
\hat{\mathbf{y}}_{r,k} &= \frac{\hat{\mathbf{x}}_{r,k} \times \hat{\mathbf{x}}_{r,k+1}}{\|\hat{\mathbf{x}}_{r,k} \times \hat{\mathbf{x}}_{r,k+1}\|}, \\
\hat{\mathbf{z}}_{r,k} &= \hat{\mathbf{x}}_{r,k} \times \hat{\mathbf{y}}_{r,k}.
\end{aligned}$$

The directions of the axes in both frames are chosen to prevent discontinuities across time steps.

In the DHB transformation, the motion of a rigid body is separated into position and orientation using two frames. The two invariants are the norms of the relative positions and orientations between frames:

$$\begin{aligned}
m_{p,k} &= \|\Delta\mathbf{p}_k\|, \\
m_{r,k} &= \|\Delta\mathbf{r}_k\|.
\end{aligned}$$

These invariants, m_p and m_r, describe the translation of the linear and angular frames. Four additional values describe their rotation:

$$\begin{aligned}
\theta^1_{p,k} &= \arctan\left(\frac{\hat{\mathbf{x}}_{p,k} \times \hat{\mathbf{x}}_{p,k+1}}{\hat{\mathbf{x}}_{p,k} \cdot \hat{\mathbf{x}}_{p,k+1}} \cdot \hat{\mathbf{y}}_{p,k}\right), \\
\theta^2_{p,k} &= \arctan\left(\frac{\hat{\mathbf{y}}_{p,k} \times \hat{\mathbf{y}}_{p,k+1}}{\hat{\mathbf{y}}_{p,k} \cdot \hat{\mathbf{y}}_{p,k+1}} \cdot \hat{\mathbf{x}}_{p,k+1}\right), \\
\theta^1_{r,k} &= \arctan\left(\frac{\hat{\mathbf{x}}_{r,k} \times \hat{\mathbf{x}}_{r,k+1}}{\hat{\mathbf{x}}_{r,k} \cdot \hat{\mathbf{x}}_{r,k+1}} \cdot \hat{\mathbf{y}}_{r,k}\right), \\
\theta^2_{r,k} &= \arctan\left(\frac{\hat{\mathbf{y}}_{r,k} \times \hat{\mathbf{y}}_{r,k+1}}{\hat{\mathbf{y}}_{r,k} \cdot \hat{\mathbf{y}}_{r,k+1}} \cdot \hat{\mathbf{x}}_{r,k+1}\right).
\end{aligned}$$

This process produces the linear and angular invariant values $(m_{p,k}, \theta^1_{p,k}, \theta^2_{p,k})$ and $(m_{r,k}, \theta^1_{r,k}, \theta^2_{r,k})$, as established in the original work.

To ensure continuity, the computed frame rotations are transformed with $\sin(\cdot)$ and $\sin(2\cdot)$. The final transformation used in our regularization term is thus:

$$\phi\left(\{u_k\}_{k=t-T}^{t}\right) = \left\{ \begin{bmatrix} m_{p,k} \\ \sin(\theta_{p,k}^1) \\ \sin(2\theta_{p,k}^1) \\ \sin(\theta_{p,k}^2) \\ \sin(2\theta_{p,k}^2) \\ m_{r,k} \\ \sin(\theta_{r,k}^1) \\ \sin(2\theta_{r,k}^1) \\ \sin(\theta_{r,k}^2) \\ \sin(2\theta_{r,k}^2) \end{bmatrix} \right\}_{k=t-T}^{t-2},$$

yielding 10 variables with a length of $T-1$. When computing $\mathcal{L}_{\text{invar}}$, we use transformed values from two types of trajectories: 1) $\phi(\{\hat{u}_k\}_{k=t-T}^{t})$, the transformed values from the demonstration trajectories, and 2) $\phi(u_t, \{\hat{u}_k\}_{k=t-T}^{t-1})$, the transformed values from the given previous trajectories $\{\hat{u}_k\}_{k=t-T}^{t-1}$ and the predicted target u_t at time t. By calculating the L2 loss between these two transformed values and using it as a training loss, the predicted trajectories u_t are aligned with the demonstration trajectories in the motion-invariant space, given $\{\hat{u}_k\}_{k=t-T}^{t-1}$.

B Supplementary Evaluation in Simulation

We provide additional quantitative evaluations in simulation to further discuss cross-embodiment visuomotor policies. Specifically, we aim to address: 1) a comparison with existing cross-embodiment learning frameworks that utilize different action spaces, and 2) the applicability of the motion-invariant regularization term to varied neural network architectures and its impact on their performance.

Comparison with the Diffusion Policy Using Relative-Trajectory Actions To compare our method with existing works aimed at cross-embodiment learning of ego-centric visuomotor policies, we adapted the Diffusion Policy with the action space of relative end-effector trajectories, as used in the Universal Manipulation Interface [2], to our setting, where the visuomotor policy outputs gripper trajectories based on previous actions and visual observations. We refer to this baseline as Diffusion Policy (Relative Trajectory). This baseline serves as a reference for the Universal Manipulation Interface [2], but without the latency compensation process. As described in Section I, generalizing latency compensation across various robot embodiments is challenging because it requires fine-tuning for each target robot system. Therefore, we exclude the latency compensation process in our evaluation.

We used the same setup as the quantitative evaluation in Section IV-C, utilizing the same dataset of 150 task demonstrations with the *Abstract* embodiment. In Figure 9, we report the success rates for deploying Diffusion Policy (Relative Trajectory) with LEGATO and the Diffusion Policy baseline used in Section IV-C. For clarification, the Diffusion Policy baseline from Section IV-C is referred to as Diffusion Policy (Velocity). In our work, we considered only the action space of the differential pose of the gripper's task-space trajectories to minimize the impact of visual odometry errors during demonstration collection and to eliminate reliance on specified frames other than the gripper's pose. This ensures suitability across various robot platforms, including floating-base robot systems. Unlike our setting, training with the action space of relative trajectories requires additional geometric information, such as the initial gripper pose for a sequence of receding-horizon actions, to generate the training dataset. This information is not incorporated into other baselines or the LEGATO framework. Nevertheless, LEGATO achieved higher performance than Diffusion Policy (Relative Trajectory) in cross-embodiment settings for the *Closing the lid* and *Ladle reorganization* tasks, though not in deployments within the same domain. The performance gap was more significant in the *Ladle reorganization* task, which requires more complex manipulation skills. As noted by Chi et al. [2], Diffusion Policy (Relative Trajectory) outperformed Diffusion Policy (Velocity) in their evaluation, and this was also observed in our setting.

Impact of Training with the Motion-Invariant Regularization We provide an additional ablation study on the complementary use of the motion-invariant regularization term to enhance cross-embodiment transferability in policies, as shown in Figure 10. Specifically, we applied the motion-invariant regularization term $\mathcal{L}_{\text{invar}}$, as described in Section III-D and Appendix A, to BC-RNN [5], Diffusion Policy (Velocity) adapted from the Velocity Diffusion Policy introduced by Chi et al. [1], and Diffusion Policy (Relative Trajectory) adapted from the Diffusion Policy using

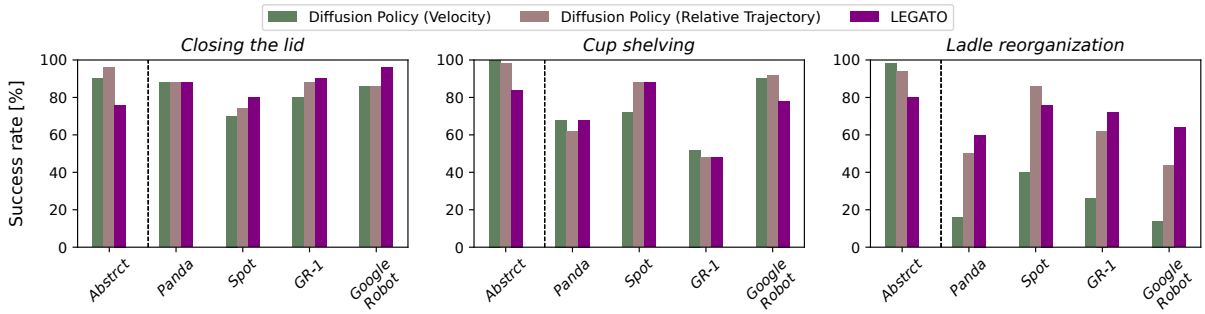

Figure 9: Comparison with different types of Diffusion Policy We report success rates from 50 trials of our LEGATO policies compared to Diffusion Policy baselines, using different action spaces: end-effector velocities [1] and relative end-effector trajectories [2]. Although the Diffusion Policy (Relative Trajectory) outperformed the Diffusion Policy (Velocity) in cross-embodiment settings, LEGATO policies achieved the highest success rates across most cross-embodiment settings, except for deployment on the *Goodle Robot* for *Cup shelving* and the *Spot* for *Ladle reorganization*.

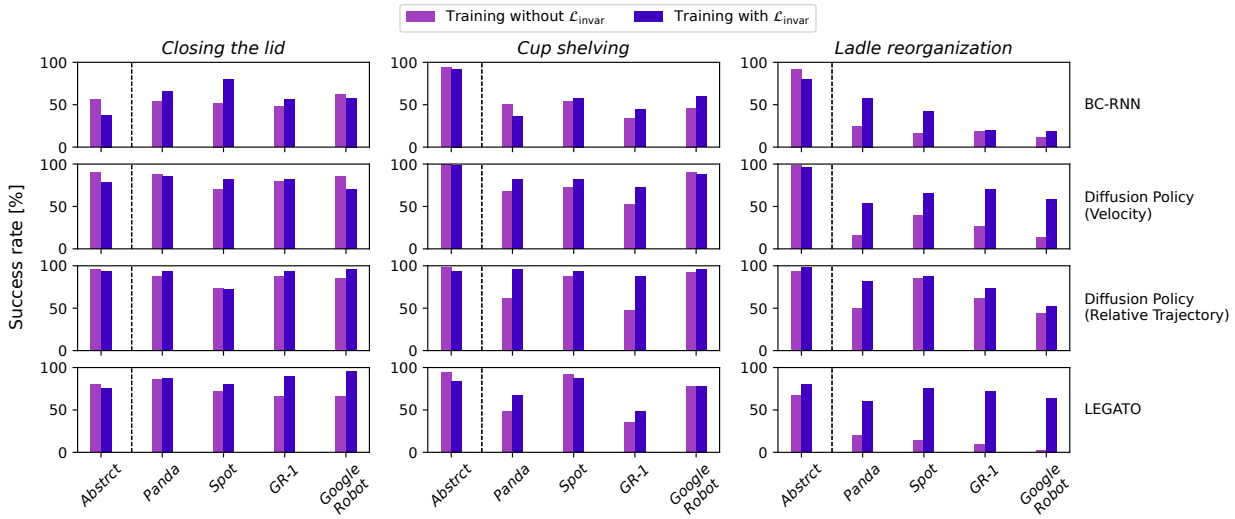

Figure 10: Impact of training with the motion-invariant regularization We evaluate the success rates of policies trained with and without the motion-invariant regularization, $\mathcal{L}_{\text{invar}}$, over 50 trials across varied architectures. Each policy is trained on demonstrations from the *Abstract* embodiment. While the success rates on the *Abstract* embodiment (used for demonstrations) decrease with the motion-invariant regularization, the success rates improve in cross-embodiment settings, where policies are deployed to other robots.

relative end-effector trajectories as the action space in the Universal Manipulation Interface [2]. Similar to the quantitative evaluation in Section IV-C, all baselines are trained on the same dataset of 150 task demonstrations using the *Abstract* embodiment. Additionally, the same IK optimizer with consistent parameters is used within a single robot.

As outlined in Equation 4 of Section III-D, the motion-invariant regularization terms are added to the original loss functions for each baseline. In BC-RNN, the motion-invariant loss is added to the negative log-likelihood between the predicted and demonstration actions. For Diffusion Policy, the motion-invariant loss is integrated with the L2-based DDPM loss during the denoising process. To adapt the formulation of the motion-invariant loss for the sequence of receding-horizon actions used in Diffusion Policy, the adapted loss is defined as follows:

$$\mathcal{L}_{\text{invar}} = \sum \|\phi(\{u_k\}_{k=t}^{t+P}, \{\hat{u}_k\}_{k=t-T}^{t-1}) - \phi(\{\hat{u}_k\}_{k=t-T}^{t+P})\|^2,$$

where P is the prediction horizon. The sequence of actions of length $P+1$ is transformed into the motion-invariant space for regularization.

Our findings indicate that incorporating motion-invariant regularization during training generally reduces success rates when deploying policies on the *Abstract* embodiment but enhances the performance of trained policies in cross-embodiment settings with different robot embodiments, regardless of the neural network architecture. This highlights the applicability of leveraging motion invariance across various neural network architectures and its effectiveness for cross-embodiment learning.

C Implementation Details

The visuomotor policy π_H predicts target poses for the handheld gripper at 10 Hz. The IK optimization π_L realizes these target poses by retargeting them into whole-body motions, updating target joint positions and body orientation at 100 Hz. In simulation, we applied low-level PD control for each joint and body at 500 Hz. For *Spot*, we additionally computed joint positions for the legs by solving IK analytically based on the target body pose. For *Google Robot*, body motion was controlled similarly to other arm joints with PD control, though using high gains. In real robot setups, we controlled the robots through APIs provided by the manufacturers. For quantitative evaluation on *Panda*, we used `JOINT_IMPEDANCE` mode via Deoxys [6] for joint position control. In the demonstration on *Spot*, we directly streamed one-point trajectories for arm joint positions and body pose through Boston Dynamics' Spot SDK.

D Demonstrations in Simulation

Task demonstrations in simulation use the same tracking camera setup as in real-robot evaluations—a Realsense T265 [3]. To replicate real-world human demonstration behaviors, visual odometry data from the tracking camera is mapped to simulated handheld gripper motions in the *Abstract* embodiment or to IK commands for teleoperated simulation robots. The button interface for triggering grasp actions and recording data is kept consistent with the real-world setup. However, unlike real-world demonstrations, simulation does not require physical interaction with the handheld gripper. Therefore, shared gripper components were removed, and a simplified handle was used to reduce the workload on the human demonstrators.

References

[1] C. Chi *et al.*, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, 2024.

[2] C. Chi *et al.*, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," in *Robotics: Science and Systems*, 2024.

[3] *Intel RealSense SDK*, `https://github.com/IntelRealSense/librealsense`.

[4] D. Lee, R. Soloperto, and M. Saveriano, "Bidirectional invariant representation of rigid body motions and its application to gesture recognition and reproduction," *Autonomous Robots*, 2018.

[5] A. Mandlekar *et al.*, "What matters in learning from offline human demonstrations for robot manipulation," in *Conference on Robot Learning*, 2022.

[6] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu, "Viola: Imitation learning for vision-based manipulation with object proposal priors," in *Conference on Robot Learning*, 2022.