Body Schema Policy: Learning a Body-Aware Vision–Action Interface for Efficient Humanoid Loco-Manipulation

Anonymous authors
Paper under double-blind review

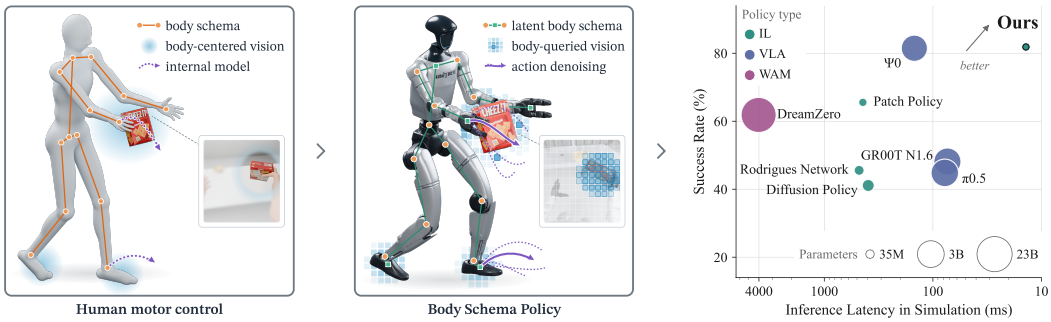

Figure 1: **From the human body schema to a latent body schema.** Left: humans act through a body schema, body-centered vision, and an internal model. Middle: Body Schema Policy gives the robot a latent body schema on its kinematic tree; body-queried vision reads image patches, and action denoising generates the motion. Right: success rate on the nine simulated G1 whole-body tasks against inference latency; marker size encodes parameter count on a log scale. BSP is served with ten DDPM iterations and plotted at its fast-path latency (Table 22).

Abstract

Inspired by the human body schema, we study robot morphology as a shared interface between visual perception and action generation. We introduce Body Schema Policy (BSP), a diffusion policy that organizes vision and action around the robot's body. Given parent–child connectivity, BSP learns body representations that jointly guide visual processing and action generation. Joint and link features retrieve spatial information from a pretrained visual encoder, connecting visual evidence to the movements being generated. On nine simulated humanoid whole-body loco-manipulation tasks, BSP with 35M parameters outperforms Diffusion Policy on every task and achieves comparable aggregate success to the Ψ_0 foundation model with 75× fewer parameters. These results support body schema as a practical architectural principle for learning compact and effective whole-body visuomotor policies. Videos and additional material: `https://anonymous.4open.science/w/Body-Schema-Policy/`.

1 Introduction

Human motor control integrates internal-model predictions with sensory feedback (Wolpert et al., 1995), organized by a body schema, an action-oriented representation of the body's structure, posture and movement (Maravita & Iriki, 2004; Hoffmann et al., 2010). The same knowledge shapes learning from others: observed actions map onto the observer's own motor representations, and watching a movement improves its later execution (Flanagan & Johansson, 2003; Mattar & Gribble, 2005), so demonstrations are read through an existing understanding of how the body moves.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Robot learning pursues generality by scaling pretrained models and demonstration data (Black et al., 2025b; Ye et al., 2026). Yet common policy architectures represent actions as flat coordinate sequences, leaving the body's structure implicit, so data and model capacity must recover how action coordinates, articulated motion, and visual evidence relate. We propose to organize both *motion generation and visual processing around a learned latent body schema*: for humanoid loco-manipulation, relating what the robot sees to coordinated motion of its base, torso, arms, and hands.

Such a representation must express how joint motions compose through the body. In our forward-kinematics study, MLPs and Transformers, even morphology-masked ones, struggle to extrapolate beyond the training range. Graph-based architectures supply connectivity (Sferrazza et al., 2025), but connectivity alone does not specify motion composition. The Rodrigues Network (Zhang et al., 2026) learns a rotational operator, but its periodic basis lacks the linear displacement of prismatic joints and mobile bases. We introduce the **Neural Screw Operator**, a learnable SE(3) operator that transports features along the kinematic tree through a unified, smooth family of revolute, prismatic, and helical motions.

On this operator we build **Body Schema Policy** (BSP), a diffusion policy with a shared body representation for perception and action (Figure 2). Given parent–child connectivity, BSP learns joint and link features conditioned on the state and candidate action chunk. The operator propagates these features through the body; they then query spatial patches from a pretrained visual encoder, and the retrieved evidence conditions a shared decoder that refines the action. As denoising changes the candidate movement, it also changes the features through which the policy reads the image.

Across ten robots, the operator extrapolates rotational and translational motion in forward kinematics. On nine simulated whole-body tasks from the SIMPLE Unitree G1 benchmark (Wei et al., 2026b), the 35M-parameter BSP learns from about 100 demonstrations per task and outperforms Diffusion Policy on every task, reaching 24 and 27 of 30 successes on two tasks where DP scores zero. It achieves comparable aggregate success to the Ψ_0 humanoid foundation model (Wei et al., 2026a) with approximately 75× fewer parameters, and exceeds larger vision–language–action and world-action baselines in aggregate, including GR00T N1.6 (NVIDIA, 2025) and DreamZero (Ye et al., 2026).

Our contributions are:

- **A body-aware vision–action interface** (Section 4): body representations jointly guide spatial visual retrieval and action generation.
- **A learnable rigid-motion operator** (Section 3): screw-motion transport makes kinematic structure a computational prior.
- **Compact humanoid control** (Section 5): a 35M-parameter policy comparable to Ψ_0 in aggregate on Humanoid whole-body tasks.

2 Related Work

Body representations and policy architectures Graph policies (Wang et al., 2018; Sanchez-Gonzalez et al., 2018), MetaMorph (Gupta et al., 2022), and Body Transformer (Sferrazza et al., 2025) use morphology to organize policy computation; implicit kinematic policies (Ganapathi et al., 2022) embed a differentiable kinematic chain in the controller. The Rodrigues Network (Zhang et al., 2026) learns rotation-based operators for forward kinematics and action denoising; its hand-reconstruction variant lets link features attend image tokens, but its published policy uses low-dimensional state. We extend it to full screw motion and investigate the visuomotor control upon it.

Geometric structure in robot learning Screw theory and the product-of-exponentials formulation describe articulated rigid motion (Murray et al., 1994; Lynch & Park, 2017). Differentiable kinematics and optimization embed geometry in learning (Sutanto et al., 2020; Pineda et al., 2022), and equivariant networks (Fuchs et al., 2020; Satorras et al., 2021) and policies (Wang et al., 2024; Yang et al., 2025) encode transformation symmetries. We instead model how joint motions compose along a kinematic tree: the Neural Screw Operator learns it through SE(3) exponentials with learned screw axes, including the non-periodic translation of prismatic joints and mobile bases.

Visual interfaces for generative policies Diffusion Policy (Chi et al., 2023), ACT (Zhao et al., 2023), and flow-based VLAs (Black et al., 2025b) learn action chunks from visual observations; world-action models jointly generate video and actions (Li et al., 2026; Ye et al., 2026). DINOv2 (Oquab et al., 2024) supplies spatial patch features, Patch Policy (Zhou et al., 2026) keeps dense visual tokens at the policy interface, and robot-centric pooling (Zhuang et al., 2024) uses proprioception to guide visual aggregation. SEM (Lin et al., 2026) conditions joint queries on analytic kinematics, CroSTAta (Minelli et al., 2025) modulates attention by learned state transitions over the execution history, and morphology-aware VLA experts (Suzuki et al., 2026) add per-joint kinematic tokens alongside temporal action tokens. BSP instead builds visual queries from action-conditioned joint and link representations updated by learned rigid-motion transport; the same representations condition action decoding.

3 Latent Body Schema

3.1 From human body schema to robot latent body schema

Body schema The brain's body schema is an action-oriented model of which parts the body has, how they connect, and how moving one part moves the rest (Head & Holmes, 1911; Maravita & Iriki, 2004; Hoffmann et al., 2010). We give a policy a latent analogue: features attached to the joints and links of its kinematic tree, updated by learned rigid motions, that read the image and condition action generation. This section builds the operator; Section 4 builds the policy on it.

3.2 Screw motions for articulated robots

A robot is a tree of $D{+}1$ links and D one-DoF joints, joint d taking parent link $p(d)$ to child link $c(d)$. By Chasles' theorem (Murray et al., 1994; Lynch & Park, 2017) every rigid displacement is a screw motion, and forward kinematics (FK) is a product of them: with S_d the twist of joint d in the root frame at home and M_ℓ the home pose of link ℓ,

$$P_\ell = P_0\Big(\prod_{d\in\pi(\ell)} \exp(\hat{S}_d\,\theta_d)\Big)M_\ell = \tilde{P}_\ell\,M_\ell, \tag{1}$$

$\pi(\ell)$ the root-to-ℓ path and P_0 the root pose. Since $P_\ell = \tilde{P}_\ell M_\ell$ is linear in $\tilde{P}_\ell$, the operator carries $\tilde{P}_\ell$ and a per-link linear head applies M_ℓ. A twist $\xi = (\omega, v) \in \mathfrak{se}(3)$ has matrix form $\hat{\xi} = \left(\begin{smallmatrix}\hat{\omega} & v\\ 0 & 0\end{smallmatrix}\right)$, with $\hat{\omega} \in \mathfrak{so}(3)$ the cross-product matrix, and generates the motion $\exp(\hat{\xi}\,\theta) \in \mathrm{SE}(3)$, in closed form

$$\exp(\hat{\xi}\,\theta) = \begin{pmatrix}\exp(\hat{\omega}\theta) & t_\xi(\theta)\\ 0 & 1\end{pmatrix}, \qquad \exp(\hat{\omega}\theta) = I + \frac{\sin\varphi}{\|\omega\|}\hat{\omega} + \frac{1-\cos\varphi}{\|\omega\|^2}\hat{\omega}^2, \tag{2}$$

$$t_\xi(\theta) = \Big(\theta I + \frac{1-\cos\varphi}{\|\omega\|^2}\hat{\omega} + \frac{\varphi-\sin\varphi}{\|\omega\|^3}\hat{\omega}^2\Big)v, \qquad \varphi = \|\omega\|\,\theta. \tag{3}$$

For $\omega = 0$ the motion is the pure translation $t_\xi(\theta) = \theta v$. The rotation block is Rodrigues' formula; the translation block is its integral, and it holds the one term rotation cannot: the advance along the axis.

For $\omega \neq 0$, only the axial advance $h\theta\omega$ of the screw motion, with pitch $h = \omega^\top v/\|\omega\|^2$, is non-periodic in θ (Proposition 1, Appendix B), so a kernel built on $\{1, \cos\theta, \sin\theta\}$, as in the Rodrigues Network (Zhang et al., 2026), contains the motion of every revolute joint and no screw with nonzero pitch: it can fit a slide on a bounded range but cannot extrapolate one (Section 5.1).

3.3 Neural Screw Operator

For a fixed twist, $G(\theta) = \exp(\hat{\xi}\theta)$ is a one-parameter subgroup of $\mathrm{SE}(3)$. Our single-joint operator learns its generator and transports a 4×4 link feature by right multiplication,

$$G(\theta) = \exp(\hat{\xi}\,\theta), \qquad \xi = (\omega, v) \text{ learned}, \qquad F^{\text{out}} = F^{\text{in}}\,G(\theta). \tag{4}$$

Two choices define it. *What is learned* is the generator: six numbers, the axis and the linear velocity of the screw. Each motion it produces is therefore rigid; revolute, prismatic, and helical motions

are all in reach; a physical twist recovers the joint's own motion, and $v = 0$ a pure rotation about a learned axis. The Rodrigues Network (Zhang et al., 2026) learns instead free coefficient matrices in front of 1, cos, and sin, which leave SE(3) and, by Proposition 1, exclude the advance. *What drives it* is a scalar θ: the physical joint coordinate when fitting FK (Section 5.1), or one latent angle per joint read from the joint's features in the policy (Section 4). The pair has a gauge, $(s\,\xi, \theta/s)$ gives the same motion, and the link-feature basis is itself learned, so a policy twist is identified only up to that basis and the angle's scale (Appendix G), not as calibrated geometry.

3.4 Multi-channel two-sided transport

Each link carries C_ℓ matrix-valued feature channels $L_\ell \in \mathbb{R}^{C_\ell \times 4 \times 4}$. Joint d owns $2R$ learnable twists $\xi_{d,r} \in \mathfrak{se}(3)$, right-action slots $1, \dots, R$ and left-action slots $R+1, \dots, 2R$, with motions $G_{d,r}(\theta) = \exp(\hat{\xi}_{d,r}\theta)$ and channel-mixing weights $\alpha^{\mathrm{R}}_d, \alpha^{\mathrm{L}}_d \in \mathbb{R}^{R \times C_\ell \times C_\ell}$. The right and left kernels are

$$U_d[i,o](\theta_d) = \sum_{r=1}^{R} \alpha^{\mathrm{R}}_{d,r,i,o}\, G_{d,r}(\theta_d), \qquad \bar{U}_d[i,o](\theta_d) = \sum_{r=1}^{R} \alpha^{\mathrm{L}}_{d,r,i,o}\, G_{d,R+r}(\theta_d), \tag{5}$$

and they transport the parent's features through the two-sided product

$$\widetilde{L}_{d,o} = \sum_{i=1}^{C_\ell} \big(L_{p(d),i}\, U_d[i,o](\theta_d) + \bar{U}_d[i,o](\theta_d)\, L_{p(d),i} \big), \tag{6}$$

abbreviated $\mathrm{Screw}(L_{p(d)}, \Xi_d, \theta_d)$ with kernel $\Xi_d = \{\xi_{d,r}, \alpha_d\}$. As in the Rodrigues Network, the kernel acts on both sides. Right multiplication $L\,G$ appends the joint's motion to the path, as Eq. 1 accumulates it, and commutes with any change of root pose; left multiplication $G\,L$ acts in the fixed frame and does not (Appendix B).

The class contains FK exactly, residual updates included (Proposition 2, Appendix B); Appendix G checks that training from zero twists on restricted-range data *recovers* it, to a median axis error of 0.085°.

4 Body Schema Policy

The Body Schema Policy is a diffusion-policy denoiser $\epsilon_\phi(x_t^{(k)}, k, s_t, V)$: given a candidate action chunk $x_t^{(k)} \in \mathbb{R}^{H \times A}$, the proprioceptive state s_t, the diffusion step k, and the image's spatial tokens V, it predicts the noise on the chunk. Its internal state is a latent body schema (Section 3), joint features J, link features L, and a global token g on the robot's tree, which N=8 neural screw blocks update: transport along the tree in every block, body self-attention and body-queried cross-attention into the image in blocks 4 and 8 (Figure 2). The schema is then read out to condition the denoising of every action coordinate.

4.1 Schema of an action hypothesis

The G1 upper body has 31 joints and 32 links. Each of the 31 on-tree action coordinates, the joint targets $\mathbf{q}_{\text{upper}}$, belongs to one joint; the other five, base height z_{b} and navigation command $\mathbf{v}$ (v_x, v_y, turning flag, target yaw), are *off-tree*, executed as leg targets $\mathbf{q}_{\text{lower}}$ by the lower-body controller of the benchmark's decoupled whole-body control. Three linear maps of $u = [\mathrm{vec}(x_t^{(k)}),\, e(k),\, s_t]$, with step embedding $e(k)$, initialize $J_0 \in \mathbb{R}^{D \times C_J}$, $L_0 \in \mathbb{R}^{(D+1) \times C_\ell \times 4 \times 4}$, and g_0, so every token starts from the whole chunk, state, and step: the schema is a schema *of the current action hypothesis*.

4.2 Neural screw block

Screw layer Each block owns joint kernels Ξ_d and one scalar angle head shared across joints. It reads each joint's latent angle, gathers the parent link, and applies Eq. 6:

$$\theta_d = W_\theta J^{\mathrm{in}}_d + b_\theta, \qquad \widetilde{L}_d = \mathrm{Screw}(L^{\mathrm{in}}_{p(d)}, \Xi_d, \theta_d). \tag{7}$$

The transported features update the child links residually,

$$L^{\mathrm{out}}_{c(d)} = \mathrm{LN}\big(L^{\mathrm{in}}_{c(d)} + \widetilde{L}_d\big), \tag{8}$$

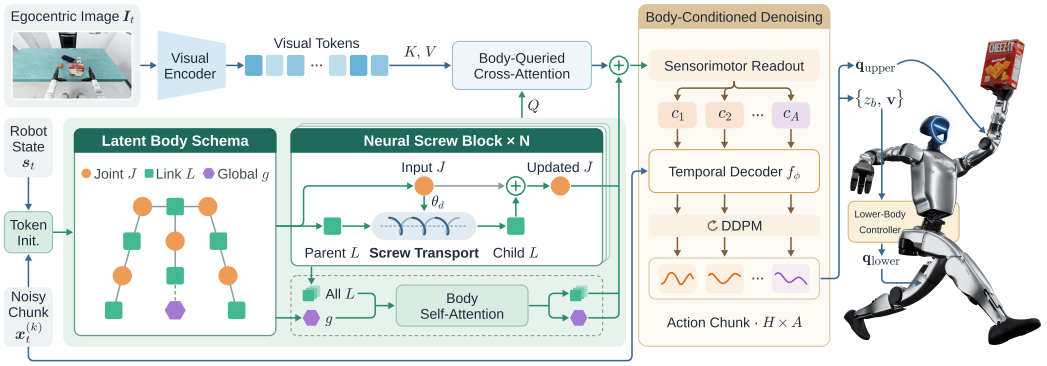

Figure 2: **Body Schema Policy architecture.** State s_t, step k, and the noisy chunk $x_t^{(k)}$ initialize joint (J), link (L), and global (g) features. Each of N neural screw blocks transports parent-link features to the child link through learned screw motions (Eq. 6); the child updates its joint. Blocks 4 and 8 add body self-attention (dashed) and body-queried cross-attention into the image tokens. A sensorimotor readout gives per-coordinate conditions $c_1, \ldots, c_A$ to a shared temporal decoder f_ϕ.

with LN over the whole $(D{+}1) \times C_\ell \times 4 \times 4$ link tensor (Appendix A); the root receives no message but shares the normalization, and joints are unchanged. This is latent FK propagation: connectivity and actuator identity are exact, while axes, pitches, and channel mixes are latent and learned, not copied from a robot description.

Joint layer Child links update their driving joints, closing the loop:

$$J_d^{\text{out}} = \text{Linear}_d\big(\text{vec}(L_{c(d)}^{\text{in}})\big) + J_d^{\text{in}}, \tag{9}$$

with a separate map $\mathbb{R}^{16C_\ell} \to \mathbb{R}^{C_j}$ per joint; links are unchanged.

Body self-attention In blocks 4 and 8, multi-head self-attention over the flattened link tokens $\text{vec}(L_\ell)$ and g mixes distant limbs (residual and LayerNorm on links and on g; joints unchanged). Token initialization, the block order, the two-sided kernel form, and the joint-plus-link readout follow the Rodrigues Network (Zhang et al., 2026), which attends in every block; we attend in blocks 4 and 8. The screw kernel, its angle head, and the two interfaces below are ours.

4.3 Reading the image: body-queried cross-attention

The image enters the policy here, in blocks 4 and 8, and nowhere else. A DINOv2-S/14 encoder (Oquab et al., 2024), fine-tuned end to end, maps the image I_t to 256 patch tokens, projected to $V \in \mathbb{R}^{256\times512}$ and mapped to a shared memory $\bar{V} = \text{LN}(W_{kv}(V + E))$, with learned positional embeddings E, that supplies the keys and values. The queries are body parts. Joint d asks with its own feature and its parent link's,

$$Q_d = W_q^J\,[J_d, \text{vec}(L_{p(d)})], \qquad Q_g = W_q^g\,g, \qquad Z = \text{MHA}([Q_1, \ldots, Q_D, Q_g], \bar{V}, \bar{V}), \tag{10}$$

and $W_o^J Z_d$ and $W_o^g Z_g$ are added to the joint and global features. Two properties follow. Each part attends over the full patch grid rather than one summary vector, and a joint's read is written to that joint alone, so it enters the tree through the joint's latent angle: visual evidence propagates down the tree along the same path as the joint's motion, and the last read conditions that joint's coordinate alone. Only the global token's read spreads body-wide, through the self-attention of block 8 and the off-tree conditions. And since Q_d is built from the current hypothesis, the read moves with denoising: image fixed, a different $x_t^{(k)}$ retrieves different evidence. The encoder is adapted by denoising gradients through this interface alone (Section 5.3).

4.4 Generating the action: body-conditioned denoising

The schema also conditions the output, coordinate by coordinate: a sensorimotor readout maps the final features to one condition per coordinate. An on-tree coordinate n, driven by joint d, gets $c_n = W_c[J_d, \mathrm{vec}(L_{p(d)})]$; an off-tree coordinate gets $c_n = W_g\, g + b_n$ with a learned coordinate-identity embedding b_n. One temporal decoder f_ϕ, a Conv1D stack of four FiLM-conditioned residual blocks shared by all A coordinates, then denoises each coordinate's own trajectory,

$$\hat{\epsilon}_{:,n} = f_\phi\big(x^{(k)}_{:,n};\ c_n,\ e(k)\big), \qquad n = 1, \ldots, A. \tag{11}$$

4.5 Training and inference

Training is standard DDPM (Ho et al., 2020; Chi et al., 2023): 100 steps, squared-cosine schedule, ϵ-prediction MSE on $x_t^{(k)} = \sqrt{\bar{\alpha}_k}\, a_t + \sqrt{1-\bar{\alpha}_k}\, \epsilon$, where a_t is the demonstrated chunk. Inference executes the predicted 16-step chunk before re-observing.

5 Experiments

Our experiments cover representation, performance and mechanism. Section 5.1 tests whether the Neural Screw Operator learns a latent body schema that extrapolates in forward kinematics. Section 5.2 measures how well the Body Schema Policy controls a humanoid's whole body in closed loop, in simulation and on a real G1. Section 5.3 traces where the gain comes from.

5.1 Learning a body schema that extrapolates

A capable latent body schema learns from limited experience and extrapolates to configurations it has never seen. Each model maps a root pose and joint configuration to every link's metric pose; it trains on the middle half of each joint's range and is tested on all of it (ten robots, five seeds, hyperparameters in Appendix G). The unstructured MLP and Transformer receive only the joint values; the structured Body Transformer, GCN, Rodrigues Network and our operator also receive the parent–child relation between links.

Generic networks fit well inside the data but fail beyond it. Trained on the full range, the generic networks reach $1.1{\times}10^{-4}$ at best and ours $4.3{\times}10^{-9}$ (Appendix G, Figure 8a). Trained on the middle half and tested on the full range, they err by at least $3{\times}10^{-2}$ on G1 (Figure 8b) and tear the body apart (Figure 8d, Appendix G); on every robot they average at least $11\times$ our error. The rotation-only Rodrigues operator (Zhang et al., 2026) suffices on most revolute robots but cannot slide: on Fetch, whose base translates, it is $1.7{\times}10^{7}\times$ worse, and $72\times$ even on the all-revolute G1.

5.2 Humanoid whole-body control in simulation and the real world

Setup We evaluate on nine G1 tasks from six categories of the SIMPLE benchmark (Wei et al., 2026b) (Figure 3, right): rigid-object grasping (BPMP, TTG), bimanual handover (HAND, BHT), mobile picking (XMP, XMBP), whole-body loco-manipulation (LPBT), non-prehensile pushing (POC) and articulated-object opening (OTC); Table 3 expands the names. Each task has about 100 demonstrations, motion-planned for BPMP and TTG and teleoperated elsewhere. BSP maps one egocentric RGB frame and the 36-D proprioceptive state to 16-step chunks of 36-D whole-body commands; in simulation these are executed by the decoupled whole-body controller shipped with GR00T N1.6 (NVIDIA, 2025). Every policy runs in closed loop in the benchmark's evaluator on 30 fixed episodes per task, 10 in each of its three evaluation levels: distractors (L0), materials and lighting (L1), and object and robot poses (L2). Success is the benchmark's own criterion for each task; where a released scene was defective we repaired it and redrew its episodes for every method.

On a real G1, we collect 15 demonstrations for each of three whole-body tasks (Figure 3, left) with SONIC's official teleoperation code and hardware (Luo et al., 2026): pick-and-place between two tables; picking up an item, opening a trash bin with a foot and throwing the item in; and pushing a trolley. DP, Ψ_0 and BSP learn from them and run 10 trials per task in the same physical setup (Appendix C.2).

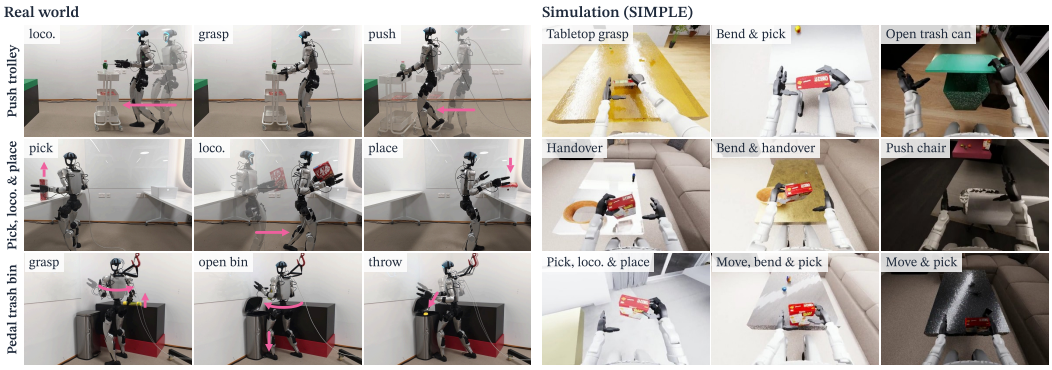

Figure 3: Evaluation tasks. Left: the three real-world G1 tasks, each in three phases. Pink arrows mark the motion; the arcs mark the torso turning toward the item and back to the bin, and a translucent overlay shows the robot earlier in the same phase. Right: the nine SIMPLE tasks in egocentric views, in reading order TTG, BPMP, OTC, HAND, BHT, POC, LPBT, XMBP and XMP (Table 3).

Table 1: Whole-body humanoid tasks on the SIMPLE G1 suite: successes on each of the benchmark's three evaluation levels (L0/L1/L2, 10 fixed episodes each); Total is out of 270. **Bold** = best per task by the 30-episode total, underline = second best; tied totals share a mark. †The published Rodrigues Network policy takes no image input; this row trains its denoiser, in its official design, with the same DINOv2-S encoder as BSP, fine-tuned end to end (Appendix D). BSP is served with ten DDPM iterations, its deployment setting (Appendix I). Table 3 expands the task names.

Method	XMP	BPMP	HAND	LPBT	TTG	XMBP	BHT	POC	OTC	Total
Ψ_0 (Wei et al., 2026a)	10/10/5	**10/9/9**	9/5/7	7/6/6	10/10/7	8/10/7	**9/9/9**	10/9/5	**8/8/8**	220
DP (Chi et al., 2023)	5/3/1	7/7/5	6/6/5	4/2/0	8/9/6	0/0/0	0/0/0	7/9/4	6/6/5	111
Rodrigues Network† (Zhang et al., 2026)	4/8/4	4/6/7	9/7/6	2/1/1	9/10/7	0/0/0	0/0/0	8/9/5	5/4/7	123
$\pi_{0.5}$ (Black et al., 2025a)	1/2/8	9/10/8	9/5/4	0/0/2	10/9/7	0/0/0	0/1/2	**10/10/5**	4/3/2	121
DreamZero (Ye et al., 2026)	**10/9/8**	6/7/5	**9/8/9**	5/1/0	9/10/6	0/0/0	7/9/6	10/9/5	8/5/6	167
GR00T N1.6 (NVIDIA, 2025)	5/4/3	8/8/6	7/6/8	0/0/0	9/10/7	5/6/6	0/0/0	7/5/1	7/4/8	130
Patch Policy (Zhou et al., 2026)	10/9/4	5/7/6	9/5/6	6/7/3	10/9/7	**9/9/9**	0/0/0	**10/10/5**	6/10/6	177
BSP (ours)	**10/10/7**	9/8/8	5/6/7	**9/7/8**	**10/10/8**	**9/9/9**	8/8/8	9/9/6	**5/9/10**	**221**

Baselines DP (Chi et al., 2023) and Ψ_0 (Wei et al., 2026a) are the benchmark's released checkpoints as shipped, one per task like every model we compare; where none is released, we train DP with the official recipe. On the same demonstrations we fine-tune three models of 3B to 22.9B parameters as served: $\pi_{0.5}$ (Black et al., 2025a), with a frozen vision–language model; GR00T N1.6 (NVIDIA, 2025); and the world-action model DreamZero (Ye et al., 2026). We retrain Patch Policy (Zhou et al., 2026) at BSP's parameter count. Appendices D and E give every recipe and statistical convention.

Simulation results BSP and Ψ_0 lead Table 1 with 221 and 220 of 270 successes; every other method totals 177 or fewer. Re-evaluating a checkpoint moves a cell by a few episodes, so we read totals and patterns rather than single cells (Appendix E.3). BSP matches Ψ_0's aggregate success with about 75× fewer parameters (35.04M served; sizes and latencies in Appendix I). The three long-horizon teleoperated tasks that add a bend or a base motion, LPBT, XMBP and BHT, separate the methods. Every method except BSP and Ψ_0 scores zero on at least one of them. BSP outperforms Diffusion Policy on every task, reaching 27 and 24 of 30 on XMBP and BHT, where DP scores zero.

Real-world results On the real robot we deploy a larger flow-matching variant of BSP (360.72M parameters) that predicts SONIC motion tokens instead of the joint-space actions of the simulator's decoupled whole-body controller (Appendix C.2). Both policies start from the same 10 initial placements of the robot and the objects. A trial succeeds when the object rests on the blanket on the second table (pick-and-place), when the item lands in the bin (trash bin), or when the trolley crosses a line 1 m ahead (trolley); a fall or a safety stop counts as a failure. Table 5 in Appendix C.2 lists the

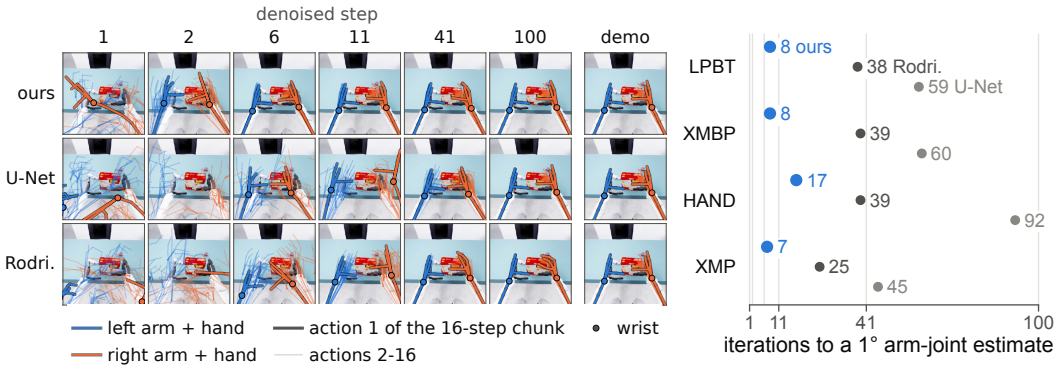

Figure 4: Denoising from noise. Left, LPBT: clean-action estimates $\hat{x}_0$ after k of 100 DDPM iterations, decoded through forward kinematics into the head-camera view for visualization; last column: the demonstration. Solid: the next action; light: the later actions of the 16-step chunk. Right: iterations until the estimated arm-joint commands are within 1° of the demonstration, per task. The U-Net is Diffusion Policy's design at BSP's size: a 13.47M trunk reading one pooled DINOv2 vector (not the 13.44M same-token trunk of Table 10); the Rodrigues Network is its published denoiser at the same size on the same vector, not the Table 1 row. All three share the DINOv2 encoder and start from the same noise.

successes. BSP succeeds in 5 of 30 trials and Ψ_0 in 23; DP completes no stage. In the simulated carry-box task the same variant matches Ψ_0 (Appendix C.2); on the robot it handles the upper-body part of each task well, while the lower-body motion it generates through SONIC's motion tokens is less reliable. Predicting locomotion tokens was not the design goal of the policy and remains to be improved. Appendix C.2 gives the data collection and training details, and the anonymous project page[1] shows videos.

5.3 Mechanism analysis: where the gain comes from

Ablations Table 10 changes one choice of BSP at a time on 90 episodes over three tasks, drawn on the released episode sets, so its counts compare within the table. Swapping in the Rodrigues operator gives 65 against 75, with a 3.6× larger trunk. Reading one image vector instead of body-queried tokens gives 60. A ResNet-18 trained from scratch in that path falls to 31. The Diffusion Policy U-Net on the same tokens scores 24 at 5.4× our trunk size and 0 at matched size; it also runs in bf16, ours in fp32 (Appendix E). On clean success, the encoder and the trunk are the load-bearing choices; the operator and the body-queried read cost 10 and 15 of 90 with one seed each. The next paragraphs measure instead how the policy reads its inputs.

Denoising with the latent body schema Where does the observation enter the computation? We score the clean-action estimate of every DDPM iteration against the demonstration, at one training frame per task, by normalized arm-joint MSE (Figure 4; Appendix H.1). After 2 of 100 iterations the input is still nearly pure noise, so the estimate is what the denoiser reads from the observation alone: BSP's error is already 0.05–0.08 on the four tasks, while the size-matched U-Net and Rodrigues Network are still at 0.86–1.02 and 0.73–0.99. After 11, BSP is at 0.001–0.004, the U-Net at 0.06–0.11 and the Rodrigues Network at 0.03–0.07. BSP reaches a 1° arm-joint estimate after 8, 8, 17 and 7 iterations on LPBT, XMBP, Handover and XMP; the Rodrigues Network after 25–39 and the U-Net after 45–92 (Appendix H.1). By 100 the Rodrigues Network has caught up, the U-Net's wrist ends 1.0 cm from the demonstration on Handover against 0.2 cm for BSP. The attention path is not the cause: a U-Net that cross-attends to the same patch tokens also starts from noise, its wrist 18.6 cm off after two iterations on Handover (Appendix H.1). Conditioned on the latent body schema, the observation determines the action from the first iterations; the chain refines it rather than builds it.

Body-queried vision Each joint's query reads the image from its own link, so we ask what each part reads. We map the sensitivity of a joint's command to every patch token the denoiser reads, the

[1] https://anonymous.4open.science/w/Body-Schema-Policy/

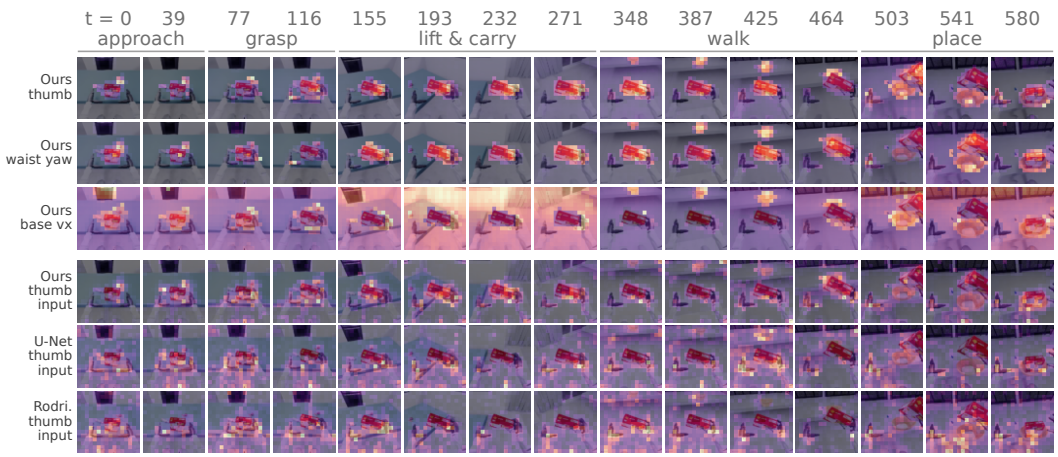

Figure 5: What three parts read over one LPBT episode. Top: sensitivity of BSP's right-thumb, waist-yaw and base-velocity commands to the patch tokens the denoiser reads, at 15 frames of a held-out episode; each heat map on its own maximum for better visualization. The thumb and waist stay on the box and the hand at it while the base reads the scene. Bottom: the thumb's sensitivity to the image patches at the encoder input for BSP, the Diffusion Policy U-Net and the Rodrigues Network, the level at which the three can be compared.

Jacobian of the action with respect to the tokens at the last denoising step, over 200 held-out LPBT frames, and score it against a permuted-joint null (Figure 5; Appendix H.2). BSP's read is compact: across the task's phases the manipulated object carries 2.4–3.3× its uniform share of the sensitivity, against 1.2–1.3× for a U-Net that cross-attends to the same tokens. The read is also part-specific, and nothing supervises it: the arm joints that grasp read the object more than a random other joint's read would in 90–100% of frames, while the base, which drives the lower body, reads it less in every frame, at 0.8–1.4× uniform, and spreads over the scene instead; its maps spread over the whole frame as the robot turns and sets off with the box (frames 155–271 in Figure 5), a scene-wide read rather than an object-centred one. Every joint of the same-token U-Net reads the same map, no part's excess over the null exceeding 0.003. The sensitivity is causal. Blanking a part's 26 most sensitive patches and re-running the chain moves that part's commands 2.1–5.2× more than blanking another part's equally salient patches on LPBT and XMBP, against 0.5–2.4× for the same-token U-Net; a randomly initialised denoiser gives 0.9–1.2×. The base is the exception at 0.9×, consistent with its diffuse read. Under a pixel-level selection that every arm can share, BSP's specificity exceeds theirs in 24 of 34 task–part cells and falls below in one (Appendix H.3). The latent body schema thus decides not only how each part moves but what it chooses to look at.

6 Conclusions and Discussion

We introduced the Body Schema Policy, a human-inspired diffusion policy that couples visual perception and action generation through a latent body schema. It is built on the Neural Screw Operator, a learnable, analytically smooth SE(3) exponential that generalizes kinematics-informed inductive bias to articulated robots. The policy outperforms Diffusion Policy on all nine tasks of a G1 whole-body suite and matches the Ψ_0 foundation model with 75× fewer parameters. Our prior concerns the functional form of rigid motion and how embodiment addresses visual evidence.

Limitations With limited compute, every arm we trained has one seed, and the two largest margins are against a baseline at zero. The interventions of Section 5.3 isolate the token interface, and only Section 5.1 isolates the operator. Our closed-loop evidence is on one embodiment, with ten real-world trials per task, and real-world deployment through the latent space lower-level controller still needs improvement. Morphology transfer (changed link lengths, tools, new robots) and WAM/VLA integrations remain open.

AI USE STATEMENT

In this work, we used generative AI tools (large language models) for editing and drafting manuscript text, for assistance with the training, evaluation, and analysis code, and for literature search. We have not used generative AI tools to generate research ideas or experimental results, and the remaining tasks with required disclosure are not applicable to this work. We have reviewed all AI-assisted work: LLM-assisted code was verified and tested by the authors, every number in the paper was traced to its evaluation record by the authors, and cited works were checked against their sources. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 27–30 Sep 2025a. URL https://proceedings.mlr.press/v305/black25a.html.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025b. doi: 10.15607/RSS.2025.XXI.010. URL https://www.roboticsproceedings.org/rss21/p010.html.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://www.roboticsproceedings.org/rss19/p026.html.

J. Randall Flanagan and Roland S. Johansson. Action plans used in action observation. *Nature*, 424(6950):769–771, 2003. doi: 10.1038/nature01861. URL https://www.nature.com/articles/nature01861.

Fabian Fuchs, Daniel Worrall, Volker Fischer, and Max Welling. SE(3)-transformers: 3d roto-translation equivariant attention networks. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://papers.nips.cc/paper/2020/hash/15231a7ce4ba789d13b722cc5c955834-Abstract.html.

Aditya Ganapathi, Pete Florence, Jake Varley, Kaylee Burns, Ken Goldberg, and Andy Zeng. Implicit kinematic policies: Unifying joint and cartesian action spaces in end-to-end robot learning. In *IEEE International Conference on Robotics and Automation*, 2022. URL https://arxiv.org/abs/2203.01983.

Agrim Gupta, Linxi Fan, Surya Ganguli, and Li Fei-Fei. Metamorph: Learning universal controllers with transformers. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2203.11931.

Henry Head and Gordon Holmes. Sensory disturbances from cerebral lesions. *Brain*, 34(2-3): 102–254, 1911.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Matej Hoffmann, Hugo Marques, Alejandro Hernandez Arieta, Hidenobu Sumioka, Max Lungarella, and Rolf Pfeifer. Body schema in robotics: a review. *IEEE Transactions on Autonomous Mental Development*, 2(4):304–324, 2010.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Luyao Zhang, Mingrui Yu, Zelin Gao, Nan Xue, Boyu Zhou, Xing Zhu, Mingyu Ding, Yujun Shen, and Yinghao Xu. Causal world modeling for robot control. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026. doi: 10.15607/RSS.2026.XXII.016.

Xuewu Lin, Tianwei Lin, Yun Du, Jitao Li, Hongyu Xie, Yiwei Jin, Lichao Huang, and Zhizhong Su. SEM: Enhancing spatial understanding for robust robot manipulation. In *IEEE International Conference on Robotics and Automation*, 2026. URL https://arxiv.org/abs/2505.16196.

Zhengyi Luo, Ye Yuan, Tingwu Wang, Chenran Li, Fernando Castañeda, Sirui Chen, Zi-Ang Cao, Jiefeng Li, David Minor, Qingwei Ben, Jinhyung Park, David Sami, Zi Wang, Xingye Da, Runyu Ding, Cyrus Hogg, Lina Song, Edy Lim, Eugene Jeong, Tairan He, Haoru Xue, Wenli Xiao, Simon Yuen, Jan Kautz, Yan Chang, Umar Iqbal, Linxi Fan, and Yuke Zhu. SONIC: Supersizing motion tracking for natural humanoid whole-body control. *Science Robotics*, 11(117):eaed4592, 2026. doi: 10.1126/scirobotics.aed4592.

Kevin M. Lynch and Frank C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017.

Angelo Maravita and Atsushi Iriki. Tools for the body (schema). *Trends in Cognitive Sciences*, 8(2): 79–86, 2004.

Andrew A. G. Mattar and Paul L. Gribble. Motor learning by observing. *Neuron*, 46(1):153–160, 2005. doi: 10.1016/j.neuron.2005.02.009. URL https://doi.org/10.1016/j.neuron.2005.02.009.

Giovanni Minelli, Giulio Turrisi, Victor Barasuol, and Claudio Semini. CroSTAta: Cross-state transition attention transformer for robotic manipulation. *arXiv preprint arXiv:2510.00726*, 2025.

Richard M. Murray, Zexiang Li, and S. Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

NVIDIA. GR00T N1.6: An open foundation model for generalist humanoid robots. https://research.nvidia.com/labs/gear/gr00t-n1_6/, 2025. Model release; code at https://github.com/NVIDIA/Isaac-GR00T.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2304.07193.

Luis Pineda, Taosha Fan, Maurizio Monge, Shobha Venkataraman, Paloma Sodhi, Ricky T. Q. Chen, Joseph Ortiz, Daniel DeTone, Austin Wang, Stuart Anderson, Jing Dong, Brandon Amos, and Mustafa Mukadam. Theseus: A library for differentiable nonlinear optimization. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://openreview.net/forum?id=K48UYo0glaJ.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4470–4479. PMLR, 2018. URL https://proceedings.mlr.press/v80/sanchez-gonzalez18a.html.

Víctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9323–9332. PMLR, 2021. URL https://proceedings.mlr.press/v139/satorras21a.html.

Carmelo Sferrazza, Dun-Ming Huang, Fangchen Liu, Jongmin Lee, and Pieter Abbeel. Body transformer: Leveraging robot embodiment for policy learning. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3407–3424. PMLR, 2025. URL https://proceedings.mlr.press/v270/sferrazza25a.html.

Giovanni Sutanto, Austin S. Wang, Yixin Lin, Mustafa Mukadam, Gaurav S. Sukhatme, Akshara Rai, and Franziska Meier. Encoding physical constraints in differentiable Newton-Euler algorithm. In *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, volume 120 of *Proceedings of Machine Learning Research*. PMLR, 2020. URL https://proceedings.mlr.press/v120/sutanto20a.html.

Kei Suzuki, Jing Liu, Ye Wang, Chiori Hori, Matthew Brand, Diego Romeres, and Toshiaki Koike-Akino. Embedding morphology into transformers for cross-robot policy learning. *arXiv preprint arXiv:2603.00182*, 2026.

Dian Wang, Stephen Hart, David Surovik, Tarik Kelestemur, Haojie Huang, Haibo Zhao, Mark Yeatman, Jiuguang Wang, Robin Walters, and Robert Platt. Equivariant diffusion policy. In *Conference on Robot Learning*, 2024. URL https://arxiv.org/abs/2407.01812.

Tingwu Wang, Renjie Liao, Jimmy Ba, and Sanja Fidler. Nervenet: Learning structured policy with graph neural networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=S1sqHMZCb.

Songlin Wei, Hongyi Jing, Boqian Li, Zhenyu Zhao, Jiageng Mao, Zhenhao Ni, Sicheng He, Jie Liu, Xiawei Liu, Kaidi Kang, et al. Ψ_0: An open foundation model towards universal humanoid loco-manipulation. In *Robotics: Science and Systems (RSS)*, 2026a. arXiv:2603.12263.

Songlin Wei, Zhenhao Ni, Jie Liu, Zhenyu Zhao, Junjie Ye, Hongyi Jing, Junkai Xia, Xiawei Liu, Michael Leong, Liang Heng, Di Huang, and Yue Wang. SIMPLE: Simulation-based policy learning and evaluation for humanoid loco-manipulation. *arXiv preprint arXiv:2606.08278*, 2026b.

Daniel M. Wolpert, Zoubin Ghahramani, and Michael I. Jordan. An internal model for sensorimotor integration. *Science*, 269(5232):1880–1882, 1995. doi: 10.1126/science.7569931. URL https://doi.org/10.1126/science.7569931.

Jingyun Yang, Zi-ang Cao, Congyue Deng, Rika Antonova, Shuran Song, and Jeannette Bohg. EquiBot: SIM(3)-equivariant diffusion policy for generalizable and data efficient learning. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 1048–1068. PMLR, 2025. URL https://proceedings.mlr.press/v270/yang25a.html.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Malik, Kyungmin Lee, William Liang, Nadun Ranawaka, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Ryan Julian, Danfei Xu, Yilun Du, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi Fan, and Joel Jang. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

Jialiang Zhang, Haoran Geng, Yang You, Congyue Deng, Pieter Abbeel, Jitendra Malik, and Leonidas Guibas. Rodrigues network for learning robot actions. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=IZHk6BXBST.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Proceedings of Robotics: Science and Systems*, 2023. URL https://arxiv.org/abs/2304.13705.

Gaoyue Zhou, Zichen Jeff Cui, Ada Langford, Bowen Tan, Yann LeCun, and Lerrel Pinto. Patch policy: Efficient embodied control via dense visual representations, 2026. URL https://arxiv.org/abs/2607.18236.

Zheyu Zhuang, Ville Kyrki, and Danica Kragic. Raising body ownership in end-to-end visuomotor policy learning via robot-centric pooling. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.

A Model details

A.1 G1 configuration

The policy acts on the G1 upper body, a tree of 31 joints and 32 links (Table 2). Three waist joints lead to the torso, where two arms of 7 joints each branch. Each hand has 7 joints in thumb, index, and middle chains from the wrist. The benchmark's lower-body controller executes the five off-tree coordinates, base height z_{b} and navigation command $\mathbf{v}$.

Table 2: Configuration of BSP on the G1 and of the Neural Screw Operator instantiation for FK fitting (Section 5.1). Rows below the middle rule describe the policy only; the FK instantiation has no joint or attention layers. Slot 1 is the first right-action slot.

	Policy (G1)	**FK fitting**
Kinematic tree	31 joints, 32 links	each robot's own tree
Blocks N	8, each with its own kernels Ξ_d and angle head	12
Transport width R	6: 6 right- and 6 left-action twists per joint	12
Link channels C_ℓ	16, each 4×4	4, each 4×4
Angle θ_d	latent, $W_\theta J_d + b_\theta$, shared by the 31 joints	raw joint coordinate
Link normalization	LayerNorm over the whole link tensor	none (identity)
Initial features	joint, link, and global tokens from three independent linear maps of u	link features from a linear map of the 12-D root pose
Initial twists	$0.01 \cdot \mathcal{N}(0, 1)$	zero
Initial channel mix α	identity in slot 1, $0.01 \cdot \mathcal{N}(0, 1)$ elsewhere	zero
Output	one condition c_n per action coordinate	per-link linear heads to 12-D poses
Joint channels C_j	8; the joint layer has one $16 \cdot 16 \to C_j$ map per joint	
Global token	256-D	
Attention blocks	4 and 8: link and global self-attention, then visual cross-attention	
Self-attention	256-D, 8 heads, dropout 0	
Cross-attention	8 heads into a 256-D shared visual memory $\bar{V}$	
Visual encoder	DINOv2-S/14, 22.06M parameters, 256 patch tokens	
Token projection	$384 \to 512$, 0.20M parameters, giving V	
Joint query	$[J_d, \mathrm{vec}(L_{p(d)})]$, $8 + 16 \cdot 16 = 264$ features; W_q^J: $264 \to 256$	
Other projections	W_{kv}: $512 \to 256$; W_q^g: $256 \to 256$; W_o^J: $256 \to C_j$; W_o^g: $256 \to 256$	
Context u	$[\mathrm{vec}(x_t^{(k)}), e(k), s_t]$ with H=16, A=36: $16 \cdot 36 + 256 + 36 = 868$ features	
Step embedding $e(k)$	sinusoidal(256)→Linear(256,1024)→Mish→Linear(1024,256)	
Condition c_n	128-D; on-tree W_c: $264 \to 128$; off-tree W_g: $256 \to 128$ plus b_n	
Temporal decoder	Conv1D($1 \to 128$, kernel 5), four residual blocks, output block; shared by all 36 coordinates	
Residual block	Conv1D–GroupNorm(8)–Mish; FiLM scale and bias from Mish→Linear(384→256) of $[c_n, e(k)]$; Conv1D–GroupNorm(8)–Mish; skip connection	
Output block	Conv1D–GroupNorm–Mish, then a 1×1 Conv1D to one channel; zero padding in every convolution	
Sampler	DDPM; the predicted clean sample is clipped to $[-1, 1]$	

A.2 Layer details

LayerNorm acts on the whole $(D+1) \times C_\ell \times 4 \times 4$ link tensor, with one mean and variance per sample and an elementwise affine map. It follows the screw layer and the self-attention. The global token has its own 256-D LayerNorm after self-attention; joint features and cross-attention residuals are not normalized. Each cross-attention site has its own 8-head attention module and shares W_{kv}, E, the memory LayerNorm, and W_q^J, W_q^g, W_o^J, W_o^g with the other site.

A.3 Twist initialization

The FK instantiation starts every twist and mix at exactly zero. Given the kinematic tree and the joint coordinates but not the URDF axes, it learns every twist from data (Appendix G). The R copies on each side start equal, receive identical gradients, and stay identical. In effect, the FK operator has one right- and one left-action twist per joint and block. The policy's $2R$ twist slots are latent, randomly initialized, and learned end to end (Table 2). The policy thus receives exact tree connectivity and actuator indexing, not calibrated geometry.

A.4 Input embeddings and heads

Table 2 lists the token initialization and the readout. Images are resized from 360×640 to 256×480 by nearest-neighbor sampling and center-cropped to 224×224, which keeps 46.7% of the width. They are then ImageNet-normalized, identically for DP and BSP. The denoiser ϵ_ϕ has 12.78M parameters, 7.12M of them in the $868 \to 32 \cdot 16 \cdot 16$ link-initialization map; Appendix I gives the totals.

B Theoretical details

B.1 Frame equivariance of two-sided transport

A change of root pose $T \in \mathrm{SE}(3)$ multiplies every link feature from the left, $L_{\ell,i} \mapsto T\, L_{\ell,i}$. For the two branches of Eq. 6,

$$\sum_i (T L_{p(d),i})\, U_d[i,o] = T \sum_i L_{p(d),i}\, U_d[i,o], \qquad \sum_i \bar{U}_d[i,o]\, T L_{p(d),i} \neq T \sum_i \bar{U}_d[i,o]\, L_{p(d),i}.$$

The right branch commutes with T, and so does the linear residual update. The left branch commutes only if T commutes with every $\bar{U}_d[i,o]$. In fp32, a numerical check finds the right branch commuting to within $7.6{\times}10^{-6}$ and the left branch off by $O(1)$. The right side is frame-equivariant transport and the left side frame-anchored mixing; LayerNorm lies outside this analysis.

B.2 Pitch decomposition

Proposition 1 (Pitch decomposition) *Let $h = \omega^\top v / \|\omega\|^2$ be the pitch of ξ and $v_\perp = v - h\,\omega$. Then*

$$t_\xi(\theta) = \frac{\sin\varphi}{\|\omega\|}\, v_\perp + \frac{1-\cos\varphi}{\|\omega\|^2}\, \hat{\omega} v_\perp + h\,\theta\,\omega, \tag{12}$$

so every entry of $\exp(\hat{\xi}\theta)$ lies in $\mathrm{span}\{1, \cos\varphi, \sin\varphi\}$ except the axial advance $h\theta\omega$, which is linear in θ.

Proof of Proposition 1 Write $v = v_\perp + h\omega$ in Eq. 3. The identities $\hat{\omega}\omega = 0$ and $\hat{\omega}^2 v_\perp = -\|\omega\|^2 v_\perp$ send the $h\omega$ part to $h\theta\omega$ and the $v_\perp$ part to the two periodic terms of Eq. 12. □

One twist therefore covers every one-DoF joint through its pitch: a revolute joint about an axis through the point q is the zero-pitch screw $(\omega, -\omega \times q)$; a prismatic joint is the limit $\omega \to 0$, in which the motion is the translation $v\theta$; nonzero pitch gives a helical joint. The coefficients are analytic in φ^2 and evaluated by Taylor branches near zero, so the operator and its gradients are smooth across the prismatic limit (Appendix I.2).

B.3 FK containment

Proposition 2 (FK containment) *Give every joint one right slot with $\xi_{d,1} = S_d$ and $\alpha^{\mathrm{R}}_{d,1} = I$, set every other twist and mix to zero, initialize channel 1 of the root link to P_0 and every other feature to 0, and stack N layers of Eq. 6 with residual child updates $L_{c(d)} \leftarrow L_{c(d)} + \widetilde{L}_d$ and no normalization. After N layers, channel 1 of a link ℓ at depth $\delta_\ell \le N$ equals $\binom{N}{\delta_\ell}\tilde{P}_\ell$ with $\tilde{P}_\ell$ the path product of Eq. 1, and the per-link linear head $X \mapsto X M_\ell / \binom{N}{\delta_\ell}$ returns the exact pose P_ℓ.*

Table 3: The nine SIMPLE tasks (Wei et al., 2026b) by category, with their demonstrations (LeRobot format, 50 fps), how the benchmark collected them, and its success criterion. A lift criterion is met at a single control step; every other condition must hold for the stated number of consecutive control steps.

			Demonstrations			
Task	**Abbr.**	**Category**	**Episodes**	**Frames**	**Collection**	**Success criterion**
BendPickMP	BPMP	Rigid-object grasping	100	15,509	Motion-planned	Lift of 0.090 m
TabletopGraspMP	TTG	Rigid-object grasping	100	12,226	Motion-planned	Lift of 0.018 m
Handover	HAND	Bimanual handover	100	45,530	Teleoperated	Containment for 5 steps
BendHandover	BHT	Bimanual handover	96	38,418	Teleoperated	Containment for 20 steps
XMovePick	XMP	Mobile picking	99	20,704	Teleoperated	Lift of 0.090 m
XMoveBendPick	XMBP	Mobile picking	100	23,664	Teleoperated	Lift of 0.080 m
LocomotionPick BetweenTables	LPBT	Whole-body loco-manipulation	99	62,764	Teleoperated	Object in the container, not hand-contacted, for about 10 steps
PushOfficeChair	POC	Non-prehensile pushing	99	31,679	Teleoperated	Chair moved more than 0.8 m along x for 15 steps
OpenTrashCan	OTC	Articulated-object opening	99	32,285	Teleoperated	Lid joint above 0.5 for 30 steps

Proof of Proposition 2 Write $L_\ell^{(n)}$ for channel 1 of link ℓ after n layers, and $\delta_\ell = |\pi(\ell)|$ for the depth of ℓ. With the stated kernel, Eq. 6 moves each channel through $\exp(\hat{S}_d\theta_d)$ separately, so channels 2 to C_ℓ stay zero. On channel 1, the residual update and the root read

$$L_{c(d)}^{(n)} = L_{c(d)}^{(n-1)} + L_{p(d)}^{(n-1)} \exp(\hat{S}_d\theta_d), \qquad L_0^{(n)} = P_0.$$

We claim $L_\ell^{(n)} = \binom{n}{\delta_\ell}\tilde{P}_\ell$, with $\binom{n}{k} = 0$ for $k > n$. It holds at $n = 0$. If it holds at $n-1$, then for $\ell = c(d)$, using $\tilde{P}_\ell = \tilde{P}_{p(d)} \exp(\hat{S}_d\theta_d)$ and $\delta_{p(d)} = \delta_\ell - 1$, Pascal's rule gives

$$L_\ell^{(n)} = \binom{n-1}{\delta_\ell}\tilde{P}_\ell + \binom{n-1}{\delta_\ell - 1}\tilde{P}_\ell = \binom{n}{\delta_\ell}\tilde{P}_\ell.$$

For $\delta_\ell \le N$ the coefficient is nonzero, and $X \mapsto XM_\ell/\binom{N}{\delta_\ell}$ maps $L_\ell^{(N)}$ to $\tilde{P}_\ell M_\ell = P_\ell$. □

C Tasks and data

C.1 Simulated tasks

Table 3 lists the nine SIMPLE tasks by the categories of Section 5.2. We call the seven tasks other than PushOfficeChair and OpenTrashCan the original tasks.

Success criteria Predicates come from each task's class in the benchmark code. Lift tasks score the instantaneous lift-height clip $\text{clip}((h - h_0)/L, 0, 1)$ against a criterion, listed as a height. The others accumulate reward while their condition holds and reset it otherwise. Heights and step counts imply no difficulty ranking. Every method on a task faces the same predicate.

Demonstrations Every model we train uses all demonstrations of its task, with no held-out split; only closed-loop success measures generalization. The benchmark authors corrected a time-axis defect in three state dimensions of the motion-planned demonstrations. The released DP and Ψ_0 checkpoints predate the fix; our BendPickMP and TabletopGraspMP models use the corrected data.

Observations and actions Each frame holds a 360×640 egocentric RGB image, the 36-D state and the 36-D whole-body command (Appendix A). BSP observes one frame and executes its full 16-step chunk before observing again. DP and BSP see a central 224-column crop of the frame resized to 480 columns. $\pi_{0.5}$ and Ψ_0 see the full resized frame; we leave this field-of-view difference uncorrected.

C.2 SONIC FOR REAL-ROBOT DEPLOYMENT

Data collection and robot control with SONIC whole-body control We collect the real-robot demonstrations and drive the G1 through SONIC's whole-body teleoperation stack (Luo et al., 2026) (Figure 6). The operator wears a PICO 4 Ultra headset and a motion tracker on each ankle and holds a PICO controller in each hand. SONIC's encoder maps the tracked motion to latent motion tokens, and its decoder turns each token into whole-body joint actions; the controller triggers and grips set the joint targets of the two Dex3 hands. We record the motion tokens and hand joint targets as the action, and the egocentric image and robot state as the observation. A policy trained on these records outputs rows of the same action space: the same frozen decoder turns its tokens into joint actions, and its hand targets go to the Dex3 hands. Data collection and deployment therefore share one action space and one low-level controller.

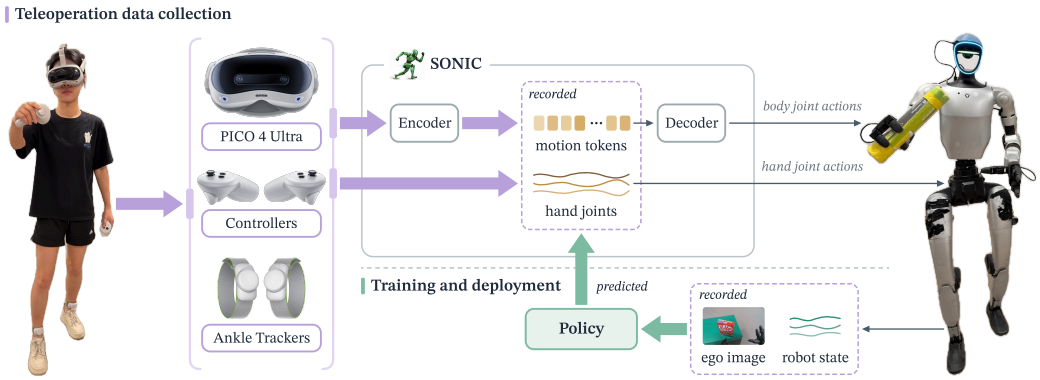

Figure 6: Data collection and robot control with SONIC whole-body control. Top: during teleoperation, the headset, controllers and ankle trackers drive SONIC, whose encoder yields motion tokens and whose decoder turns them into body joint actions; the controllers also give the hand joint actions. The motion tokens and hand joints are recorded as the action. Bottom: the recorded ego image and robot state form the observation, from which the policy predicts motion tokens and hand joints for the same decoder. Lavender marks teleoperation, green the policy, and grey the modules both share.

Action space The real robot is driven through SONIC's latent motion tokens (Luo et al., 2026) at 30 rows per second. Each row of an action chunk is a 64-D token on the tokenizer's FSQ grid, which the frozen SONIC decoder turns into whole-body joint targets, legs and locomotion included, plus 14 joint targets for the two Dex3 hands. Every policy we compare predicts rows of this space; none outputs a body joint angle.

BSP for SONIC token prediction No token coordinate belongs to a joint, so the body motion cannot be read out joint by joint as in Section 4.4. The variant we deploy keeps the body schema and reads the token stream from it through a transformer. The schema is a 31-joint graph of the G1 upper body (waist, two 7-DoF arms, two 7-joint hands; 32 links). Each joint carries a 16-channel latent, each link 32 channels of 4×4 frames, and a 512-D global token joins them. Twelve screw blocks propagate frames from parent to child, $F_{\text{child}} = F_{\text{parent}} \exp(\hat{\xi}\theta)$, with ξ drawn from a learned bank of screw axes per joint and θ a linear projection of the joint's feature rather than the measured joint position; nothing ties the schema to the physical configuration. One graph evaluation per flow step embeds the whole noisy 16×78 chunk, the flow time and the measured 43-D joint state into the joint, link and global features; three of the twelve blocks cross-attend to the 256 DINOv2 patch tokens with queries from the joint and link latents. Two heads read the schema. The token head is a 6-block, 1536-wide transformer over the 16 future rows. Each block runs temporal self-attention, cross-attention to the 31 joint features and the global token, a gated cross-attention to the patch tokens and a feed-forward layer, all under time modulation, and the head emits the 64-D token velocity of every row jointly. The fingers are read out per physical joint: a shared 1-D temporal convolution decoder (four FiLM blocks, kernel 5, width 256) denoises each finger joint's 16-step trajectory conditioned on that joint's own node feature. The encoder is DINOv2 ViT-S/14 on the centre 224×224 crop of the 256×480 frame with colour jitter, fine-tuned in simulation and frozen

on the real robot. Training is flow matching, $x_\sigma = (1-\sigma)a + \sigma\epsilon$ with $\sigma \sim U[0,1]$ and target $\epsilon - a$; sampling takes 10 Euler steps on Ψ_0's σ grid. The model has 360.72M parameters: 22.06M DINOv2 with its 0.20M projection, 46.25M graph, 4.14M finger decoder and 288.07M token head. On 640 training windows, its finger error is 0.0022 rad against 0.0073 and 0.0062 rad for a plain flow transformer and a widened DP U-Net of the same head size, while its token error is 0.0221 against 0.0222 and 0.0174: the schema helps the per-joint readout, not the token fit.

Simulation: carrying a box We first test the variant on the SONIC carry-box task of the SIMPLE benchmark, where the robot walks to a box, lifts it and places it on a table, under the benchmark's unmodified evaluator and the SONIC release controller. Every policy learns from the same 50 demonstrations and runs the 5 official scenes 6 times each, 30 episodes in all; the stage criteria were fixed before the runs and are scored from the logged simulator state for every policy alike. BSP trains with batch 32 for 40k steps and executes 6 of every 16 rows at 50 Hz. The plain head shares its data, recipe, flow objective and sampler, keeps Ψ_0's frozen vision–language model as encoder and replaces the schema by a plain transformer of the same head size; the released Ψ_0 is the post-trained carry-box checkpoint served with its own flags; DP is Ψ_0's baseline code trained on the same demonstrations. Table 4 gives the stage counts. Success is a tie at 8, 7 and 6 of 30 between BSP, the plain head and Ψ_0, reached with a smaller action head, a 22M image encoder in place of a 2.13B vision–language model and no robot pretraining. The profiles differ: Ψ_0 grasps in every episode and lifts in 26 but loses the box on the way to the table, while BSP starts walking in only 21 of 30 episodes, yet once it approaches it brings the box to table height in 17 of those 21, against 16 of 30 for Ψ_0. The counts cluster by scene, so we read the table as parity rather than a ranking.

Table 4: SONIC carry-box in simulation: episodes of 30 that reach each stage. Success is the placed column; the benchmark's own reward criterion, which needs the box to rest 45 further steps, gives BSP 7 of 30 and the others the same counts. Bold is best, underline second best.

Method	Approach	Grasp	Lift	Table height	Placed	Success
BSP (ours)	21	<u>21</u>	<u>17</u>	**17**	**8**	**8/30**
Plain head on Ψ_0's encoder	<u>26</u>	<u>21</u>	12	9	<u>7</u>	<u>7/30</u>
Ψ_0 (released)	**30**	**30**	**26**	<u>16</u>	6	6/30
DP (Ψ_0 baseline)	17	10	6	3	2	2/30

Real-world tasks We define three whole-body tasks on a real G1: pick-and-place between two tables, a trash-bin task and pushing a trolley. In the trash-bin task, the robot picks up an item, opens the bin with a foot and throws the item in. We collect 15 demonstrations per task with SONIC's official teleoperation code and hardware (Luo et al., 2026), and DP, Ψ_0 and BSP learn from them. Each policy runs 10 trials per task in the same physical setup, all three starting from the same 10 initial placements of the robot and the objects. A trial succeeds when the object rests on the blanket on the second table (pick-and-place), when the item lands in the bin (trash bin), or when the trolley crosses a line 1 m ahead (trolley). A fall or a safety stop counts as a failure. Table 5 lists the successes, and Figure 7 shows three failure cases of BSP.

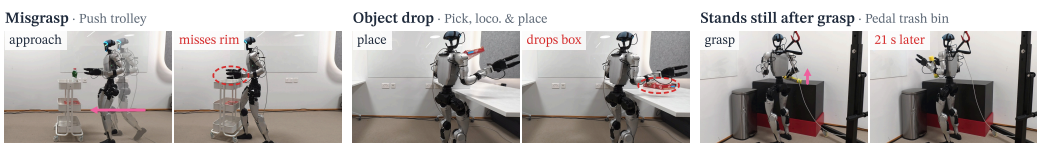

Figure 7: Three failure cases of BSP on the real robot. Left: on the trolley task the robot approaches but its hand misses the rim. Middle: on pick-and-place it reaches the second table but drops the box while placing it. Right: on the trash-bin task it grasps the item and then stands still, 21 s later, without stepping to the pedal.

Training and deployment On the robot, each policy trains per task on that task's 15 demonstrations (the trash-bin set has 6,503 frames at 30 fps, 6,248 16-row windows) with global batch 128, AdamW

Table 5: Real-world G1 tasks: trials of 10 that complete each stage, per task and policy; the last stage of each task is its success criterion. Every policy starts from the same 10 initial placements and learns from the same 15 demonstrations per task.

	Pick-and-place			**Trash bin (foot)**			**Trolley (hand)**			
Method	grasp	walk	place	grasp	open	throw	grasp	push	cross	Total
DP (Chi et al., 2023)	2	0	0	1	0	0	3	0	0	0/30
Ψ_0 (Wei et al., 2026a)	9	9	8	10	9	7	8	8	8	23/30
BSP (ours)	10	4	3	10	2	0	9	2	2	5/30

at 10^{-4}, 1000 warm-up steps and a 40k-step cosine schedule, stopped at 5k steps for both. BSP keeps its DINOv2 backbone frozen, so 338.66M of its 360.72M parameters train. Ψ_0 starts from its SONIC post-trained checkpoint and fine-tunes both the vision–language model and the action expert. DP keeps its official architecture and recipe, with the same encoder and training settings, its U-Net head widened to 338.35M parameters to match ours, and the same motion-token output. Ψ_0 and BSP are served on the robot's RTX 5090 desktop GPU, which they share with the SONIC controller. Table 6 lists how the two policies that succeed are served. Ψ_0 runs with real-time chunking, which freezes the 12 rows in flight, so a new camera frame affects the robot only after about 400 ms; BSP executes the new chunk from row 3–5 and reacts after about 130 ms, at a similar inference cost per executed row.

Table 6: Real-world serving of Ψ_0 and BSP, both after 5k training steps. Rows are motion-token steps; the lag is the time from a new camera frame to its first effect on the robot.

	Ψ_0	BSP (ours)
Flow steps	3	10
Chunk length	30 rows (1.0 s)	16 rows (0.53 s)
Rows executed per chunk	12 (0.4 s)	about 8 (0.27 s)
Inference per chunk	p50 136 ms, max 182 ms	p50 105 ms, p95 119 ms, max 141 ms
Inference per executed row	about 11 ms	about 13 ms
Lag before a new frame acts	about 400 ms	about 130 ms

D Baseline implementation details

Table 7 summarizes the baselines in the order of Table 1; Table 8 gives the settings of those we train. $\pi_{0.5}$ trains and serves through openpi, and GR00T N1.6 and DreamZero through their authors' code. Every other model we train shares one trainer, dataset reader and policy server.

D.1 Released checkpoints

We evaluate the benchmark's released Ψ_0 and DP checkpoints as shipped. For BendHandover, PushOfficeChair and OpenTrashCan, which have no released DP checkpoint, we train it from the benchmark's public trainer. We compare no reproduction with a released checkpoint and so claim no equivalence.

Ψ_0 The released checkpoints, one per task and fine-tuned on that task's demonstrations, pair a frozen Qwen3-VL-2B vision–language model with a 0.5B action head. Their configuration records bf16 mixed precision.

Diffusion Policy The released checkpoint is a ConditionalUnet1D trunk, FiLM-conditioned on the ResNet-18 feature vector. Table 20 pairs the same trunk with frozen DINOv2 as a matched-encoder control.

Table 7: The baselines at a glance, with BSP for reference. Weights are the benchmark's released checkpoint or a model we train or fine-tune; parameters count the model as served. *On BHT, POC and OTC, where no checkpoint is released, our run of the released recipe. [t] Trunk only. – = not reported.

Method	Weights	Visual input	Parameters	Chunk or sampler
Ψ_0 (Wei et al., 2026a)	Released	Full frame; frozen Qwen3-VL-2B	2.63B	–
DP (Chi et al., 2023)	Released*	224-column crop; pooled ResNet-18	73.13M[t]	–
Rodrigues Network (Zhang et al., 2026)	Our training	Fine-tuned DINOv2-S, pooled	61.47M	16-step chunk; 100 denoising steps
$\pi_{0.5}$ (Black et al., 2025a)	Our fine-tune	Full frame; frozen PaliGemma	3.62B	16-step chunk
DreamZero (Ye et al., 2026)	Our LoRA fine-tune, rank 4	320×176 input	22.9B	48-step chunk
GR00T N1.6 (NVIDIA, 2025)	Our fine-tune	–	3B	16-step chunk
Patch Policy (Zhou et al., 2026)	Our training	256 fine-tuned DINOv2-S/14 tokens	34.9–35.1M	16-step chunk
BSP (ours)	Our training	224-column crop; 256 fine-tuned DINOv2-S/14 tokens	35.04M	16-step chunk; 10-step DDPM (trailing)

Table 8: Training settings of the baselines we train or fine-tune. Our recipe is that of Appendix E; $\pi_{0.5}$, DreamZero and every model under our recipe train on four A100-40GB GPUs. *Only on BHT, POC and OTC. [‡]On POC and OTC. [§] DINOv2 runs under bf16 autocast. [¶] Outside the precision comparison of Appendix E. – = not reported.

Method	Steps	Global batch	Learning rate	Seed	Precision
DP*	released recipe			2026[‡]	bf16, trainer default
Rodrigues Network	our recipe			2027	fp32[§]
$\pi_{0.5}$	40k	128	10^{-4}, cosine, 1,000 warm-up	2027	set by openpi[¶]
DreamZero	20k	4 (authors: 8)	authors' default	2027	authors' default
GR00T N1.6	40k	24	10^{-4}, cosine, 1,000 warm-up	42	authors' default
Patch Policy	40k	64×2	our recipe	2027	–[§]

D.2 Models we train or fine-tune

Rodrigues Network The published Rodrigues Network policy does not take image observations. We therefore train its published denoiser with a DINOv2-S encoder like BSP's, fine-tuned end to end. We verified that the denoiser, the authors' `official_l6` preset with 39.22M parameters, is bit-identical to their code. The vector, the mean of the projected patch tokens, is concatenated with the state; the projection adds 0.20M parameters.

$\pi_{0.5}$ We fine-tune the openpi `pi05_droid` PyTorch base with the benchmark's own $\pi_{0.5}$ trainer. It freezes the PaliGemma vision–language model and trains the action expert and the action and time projections: 693M parameters, 19.2% of the model. We change only the seed and the action horizon, from 30 to 16 to match BSP.

DreamZero We fine-tune the released DreamZero-AgiBot base, a Wan2.1-I2V-14B video diffusion transformer with the authors' action head. The authors' trainer fits one LoRA adapter per task on their default target modules. The served model's 22.9B parameters comprise the 14B video backbone with the action modules and video encoders, and a frozen 5.7B text encoder.

Table 9: Shared training recipe and evaluation settings. Appendix D lists where a baseline departs from the training recipe; Appendix C gives the demonstrations and the success criterion of each task.

Setting	Value
Training	
Steps; batch size	40,000; 128
Optimizer	AdamW, weight decay 10^{-6}
Learning rate	10^{-4}; a fine-tuned DINOv2 encoder at $0.1\times$ that rate
Schedule	1,000-step warmup, then cosine decay
EMA	decay $\min(0.9999, (1+n)/(10+n))$ at optimizer step n; the official `power`= 0.75 is inert under its `use_ema_warmup=False`
Evaluated weights	raw weights at step 40,000
Observation horizon	1: one 360×640 egocentric RGB frame and the 36-D state
Prediction horizon	16 steps of the 36-D action
State normalization	official
Demonstrations	all episodes of the task; no held-out split
Precision	kinematic trunk fp32; U-Net trunk bf16; DINOv2 under bf16 autocast
Training seeds	one per arm
Hardware	four NVIDIA A100-40GB GPUs
Evaluation	
Simulator	Isaac Sim 4.5
Evaluator	SIMPLE release, revision `d010f0f`; `d46297e` on POC and OTC, and for the Rodrigues Network row on every task
Episodes	30 per task: 10 per evaluation level L0, L1, L2
Success	the task's native predicate (criterion v1)
Hardware	one NVIDIA RTX 6000 Ada GPU

GR00T N1.6 We post-train the released 3B base on each task with the authors' trainer and the settings of Table 8.

Patch Policy We retrain the reference diffusion head, a per-token conditioning MLP and an 8-layer, 4-head transformer decoder. The decoder cross-attends to the 256 DINOv2-ViT-S/14 patch tokens under the reference's patch-aware memory mask. We widen the head from 256 to 304 to match BSP's size.

E Training and evaluation protocol

E.1 Training recipe

Our models and ablation arms, in both trunk families, train with the recipe of Table 9. As in the official trainer, the EMA weights are copied into the live model at every 5,000-step checkpoint, so training continues from the average. We evaluate the raw weights saved at step 40,000, 5,000 optimizer steps after the last copy, not the EMA weights. Every model is trained or fine-tuned on NVIDIA A100-40GB GPUs; the number of GPUs differs by model (four under our recipe). Every closed-loop evaluation runs on a single NVIDIA RTX 6000 Ada GPU.

Precision The kinematic trunk runs in fp32, because the fused CUDA operators are fp32. This covers BSP, every kinematic ablation arm and the Rodrigues Network row, a kinematic arm in our trainer. Every U-Net trunk we train, in the DP reproduction and the U-Net ablation arms, runs in bf16, the trainer's default, except the pooled-vector U-Net of Figure 4, trained in fp32. Precision therefore co-varies with the trunk family; we did not retrain either family at the other precision. The operator, routing and encoder swaps, and the U-Net cells against each other, compare at one precision. The U-Net-against-kinematic contrasts do not, so we do not read those as purely architectural.

Table 10: Single-variable ablations of BSP on three tasks: successes out of 30, with the split over L0/L1/L2 in parentheses, and the three-task total Σ out of 90; one seed per arm. We took these counts on the released episode sets, so they compare within this table and not with Table 1. The operator swap puts the learned Rodrigues operator inside our exact interface and recipe; its trunk has 46.74M parameters against BSP's 12.98M, so the two are not matched in capacity. Each trunk swap replaces our trunk with the DP U-Net, which cross-attends to the same DINOv2 tokens from its time-step positions; the 13.44M one is matched to BSP's total size. Only the trunk swaps also change precision, bf16 against fp32 (Appendix E). DP (released) is the released checkpoint on the same sets.

Arm	trunk	operator	visual interface	encoder	**LPBT**	**XMBP**	**HAND**	Σ
BSP (flagship)	kinematic	screw	body-indexed tokens	DINOv2-S, fine-tuned	23 (8/8/7)	28 (10/8/10)	24 (8/8/8)	**75**
operator swap	kinematic	Rodrigues	body-indexed tokens	DINOv2-S, fine-tuned	19 (8/6/5)	25 (9/9/7)	21 (7/7/7)	65
routing swap (pooled)	kinematic	screw	pooled vector	DINOv2-S, fine-tuned	18 (8/6/4)	24 (7/9/8)	18 (6/8/4)	60
encoder swap (pooled)	kinematic	screw	pooled vector	ResNet-18, scratch	10 (5/3/2)	8 (3/3/2)	13 (6/3/4)	31
trunk swap	U-Net (69.67M)	—	time-step cross-attn.	DINOv2-S, fine-tuned	5 (3/2/0)	0 (0/0/0)	19 (6/7/6)	24
trunk swap, matched	U-Net (13.44M)	—	time-step cross-attn.	DINOv2-S, fine-tuned	0 (0/0/0)	0 (0/0/0)	0 (0/0/0)	0
trunk swap, frozen	U-Net (69.67M)	—	time-step cross-attn.	DINOv2-S, frozen	1 (0/1/0)	0 (0/0/0)	0 (0/0/0)	1
BSP-FROZEN	kinematic	screw	body-indexed tokens	DINOv2-S, frozen	4 (2/2/0)	20 (8/5/7)	11 (3/4/4)	35
DP (released)	U-Net	—	pooled vector	ResNet-18, scratch	6 (4/2/0)	0 (0/0/0)	14 (5/5/4)	20

E.2 Evaluation protocol

Every policy runs in closed loop in the SIMPLE evaluator on Isaac Sim 4.5 (Table 9). On the seven original tasks, a single-file patch accepts a legacy asset identifier on XMoveBendPick. The patch is inert on every other task, and we will release it with our code. Each task has 30 pre-generated episodes, 10 in each of three evaluation levels (Wei et al., 2026b). L0 places random distractor objects in the scene, L1 randomizes object materials and lighting, and L2 randomizes the initial poses of the target objects and the robot. Every method is scored by the task's native predicate: a lift height on four tasks, a run of consecutive control steps on five (Appendix C). Our models execute each 16-step chunk in full before observing again.

E.3 Statistical reporting

Each table cell counts successes in 30 fixed episodes. Its 95% Clopper–Pearson interval spans about ±0.19 in success rate near the middle of the range, $[0.31, 0.69]$ at 15/30, or about ±6 episodes. The simulator and policy server are not fully deterministic, so re-evaluating one checkpoint on the same episodes moves its count by a few episodes. A single-cell difference inside these bands does not separate two methods, so we read totals and patterns across tasks. Two methods that ran the same episodes are compared by an exact McNemar test on paired outcomes. Each method and ablation arm has one training seed, so no interval covers variation across training runs.

Zero counts No zero in Table 1 comes from a broken harness. Each 0/30 cell has a cell of another method on the same task, run on the same machine, that scored far above zero.

F Additional results on SIMPLE

F.1 Single-variable ablations

The Rodrigues Network row of Table 1 differs from BSP in two components, a rotation-only operator and a pooled visual path, over the same fine-tuned DINOv2-S encoder. Its 123 of 270 against 221 cannot separate the two, so Table 10 swaps one component at a time.

Table 11: Forward kinematics beyond the training range on ten robots (Section 5.1). Each model trains on the middle half of every joint's range; entries are the full-range 12-D link-pose MSE at metric scale of the final (60k) iterate, averaged over five seeds. Ratio: geometric mean over seeds of Rodrigues MSE over ours, with its [min–max] over seeds; ratios > 1 favor the Neural Screw Operator. Wins: seeds on which ours is better. DoF counts prismatic joints in parentheses. Both operators share one recipe, and ours is the smaller on every robot.

Robot	DoF (prism.)	**Screw (ours)**	**Rodrigues**	ratio [min–max]	wins
Fetch (mobile)	13 (5)	1.3×10^{-6}	2.0×10^{1}	1.7×10^{7} [7.2×10^{6}–3.0×10^{7}]	5/5
Franka (mobile)	12 (4)	5.6×10^{-6}	2.0×10^{-2}	7.5×10^{3} [1.6×10^{3}–6.4×10^{4}]	5/5
G1 (23-DoF)	23 (0)	7.0×10^{-7}	4.8×10^{-5}	72 [48–127]	5/5
Franka	9 (2)	4.7×10^{-6}	4.5×10^{-5}	43 [2.6–496]	5/5
Stretch	8 (5)	3.8×10^{-6}	7.4×10^{-6}	2.3 [0.95–6.1]	4/5
LEAP hand	16 (0)	4.8×10^{-5}	2.3×10^{-5}	0.58 [0.30–2.0]	1/5
PhantomX	18 (0)	8.5×10^{-6}	3.1×10^{-6}	0.41 [0.23–1.1]	1/5
Allegro hand	16 (0)	4.5×10^{-5}	1.3×10^{-5}	0.31 [0.20–0.49]	0/5
UR5	6 (0)	7.5×10^{-5}	2.1×10^{-5}	0.33 [0.15–0.84]	0/5
Go1	12 (0)	3.2×10^{-5}	5.8×10^{-6}	0.18 [0.16–0.23]	0/5

G Forward-kinematics experiments

G.1 Protocol

Each sample draws a joint configuration and a root pose, whose translation is uniform in $[-0.05, 0.05]^3$ m and whose orientation is a normalized 4-vector drawn uniformly from $[-1, 1]^4$, not uniform over rotations. Only the joint ranges differ: the middle half of each for training, the full range for testing. Supervision is every link's full 12-D pose at metric scale, without per-link normalization, which would hide metric-scale extrapolation error.

Both operators train with Adam for 60k iterations at batch 1024: learning rate 3×10^{-3}, 2k warmup iterations, cosine decay to 3×10^{-5}, gradient clipping at 1.0, fp32, seeds 0–4. The test error is the MSE over 200 fresh batches of 1024 full-range samples at the final (60k) iterate. Table 11 compares both operators; ours has fewer parameters on every robot, for example 0.10M against 0.14M on the mobile Franka.

G.2 Generic networks

Five generic networks without a motion operator run on all ten robots and five seeds with the same data and test protocol. The MLP uses the operators' recipe; the Transformer, Body Transformer, GCN and GAT use learning rate 3×10^{-4} with cosine floor 3×10^{-6}, and we have no record of them at 3×10^{-3}. Each is larger than the Neural Screw Operator (Table 12) and has 11.5× (Transformer, UR5) to $6.4\times10^{6}\times$ (GAT, Fetch) the Neural Screw Operator's error. On Fetch, all five stay at 0.24 to 8.6, below the Rodrigues operator's translation floor of 20. On the mobile Franka, all five are worse than Rodrigues, at 0.038 to 0.20 against 0.020. The kinematic-tree mask changes the Transformer's error by −12% on LEAP to +21% on UR5.

G.3 Inside versus beyond the training range

Table 13 lists the G1 numbers behind Figure 8a,b, plus GAT, which the figure omits because its G1 error exceeds GCN's. On fresh samples inside their training range, the middle-half-trained models err 50–820× less than on the full range.

For Figure 8a, we retrain each model on the full G1 range for five seeds, changing only the training distribution. The generic networks then reach 1.1×10^{-4} (Body Transformer) to 1.1×10^{-2} (MLP), at least 2.6×10^{4} times our 4.3×10^{-9}; Rodrigues reaches 6.3×10^{-8}.

Table 12: Generic networks under the protocol of Table 11: full-range test MSE of the final iterate, mean over five seeds; the last row gives the parameter range over the ten robots. The MLP has hidden widths 512–256–128–256–512. The Transformer and Body Transformer (Sferrazza et al., 2025) FK heads have 8 layers, width 256 and 4 heads; the Body Transformer adds its kinematic-tree attention mask. GCN and GAT are heads on the link graph. On every robot, each network averages at least 11× the error of the Neural Screw Operator.

Robot	**Screw (ours)**	MLP	Transformer	Body Tr.	GCN	GAT
Fetch (mobile)	1.3×10^{-6}	2.4×10^{-1}	6.1	6.4	5.5	8.6
Franka (mobile)	5.6×10^{-6}	8.7×10^{-2}	3.9×10^{-2}	3.8×10^{-2}	1.8×10^{-1}	2.0×10^{-1}
G1 (23-DoF)	7.0×10^{-7}	1.0×10^{-1}	3.4×10^{-2}	3.8×10^{-2}	3.6×10^{-2}	6.9×10^{-2}
Franka	4.7×10^{-6}	7.5×10^{-2}	3.2×10^{-2}	3.3×10^{-2}	5.6×10^{-2}	1.1×10^{-1}
Stretch	3.8×10^{-6}	1.2×10^{-2}	1.0×10^{-2}	1.0×10^{-2}	1.5×10^{-2}	2.2×10^{-2}
LEAP hand	4.8×10^{-5}	1.8×10^{-2}	6.5×10^{-3}	5.7×10^{-3}	8.4×10^{-3}	1.1×10^{-2}
PhantomX	8.5×10^{-6}	7.6×10^{-2}	3.2×10^{-2}	3.0×10^{-2}	2.1×10^{-2}	3.4×10^{-2}
Allegro hand	4.5×10^{-5}	8.7×10^{-3}	3.2×10^{-3}	3.1×10^{-3}	3.4×10^{-3}	5.2×10^{-3}
UR5	7.5×10^{-5}	1.7×10^{-3}	8.6×10^{-4}	1.0×10^{-3}	1.6×10^{-3}	3.5×10^{-3}
Go1	3.2×10^{-5}	2.2×10^{-2}	1.3×10^{-2}	1.2×10^{-2}	1.0×10^{-2}	1.5×10^{-2}
Parameters (M)	0.05–0.18	0.38–0.50	3.20–3.26	3.20–3.26	2.95–3.07	1.48–1.54

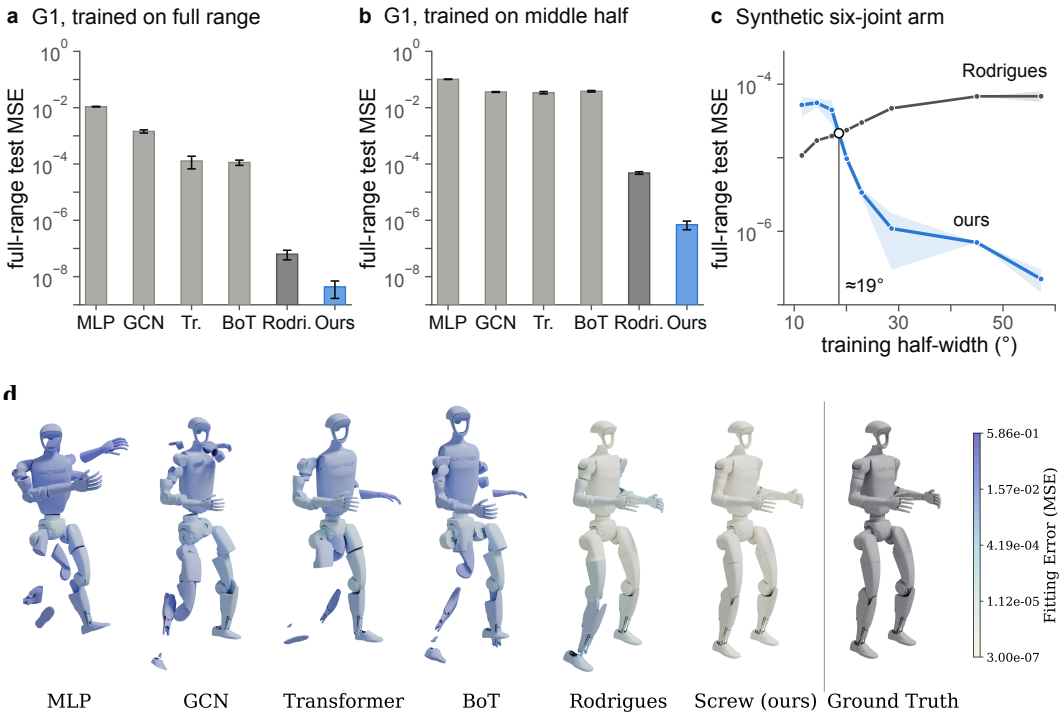

Figure 8: Full-range test error, log scale. **a**, **b** G1 models after 60k iterations, trained on the full joint range (a) or middle half (b); bars: five-seed mean; error bars: ±1 std (Table 13). Tr.: Transformer; BoT: Body Transformer (Sferrazza et al., 2025). **c** Synthetic six-joint arm against the training half-width (seed mean and range); the operators cross at ≈19°. **d** extrapolated forward-kinematics predictions on an example G1 configuration. Each link is colored by its pose error on a log scale shared across models; darker means larger error.

G.4 Renderings

Figure 8d renders the final (60k) middle-half-trained G1 checkpoints of Table 13 at seed 4, except GAT. Its configuration, from a batch of 4096 full-range samples, has the largest Rodrigues-to-ours

Table 13: G1 numbers behind Figure 8a,b, with GAT: parameters and test MSE of the final (60k) iterate, mean ± std over five seeds. Models trained on the middle half of each joint's range are tested on the full range and on fresh middle-half samples. Models trained on the full range are tested on the full range; they ran on an RTX 5090 with PyTorch 2.9.1 and the middle-half code unchanged, ours on the `screw_transform` kernel and Rodrigues on a kernel build with a later warp-shuffle fix (`__shfl_down_sync`) that its middle-half runs may have lacked. An MLP run with the same configuration on an Apple M4 Max reached 1.05×10^{-2}.

		Trained on middle half		**Trained on full range**
Model	Params (M)	full-range test	in-range test	full-range test
MLP	0.50	$(1.02 \pm 0.02)\times10^{-1}$	$(2.01 \pm 0.10)\times10^{-3}$	$(1.09 \pm 0.03)\times10^{-2}$
GCN	3.07	$(3.61 \pm 0.12)\times10^{-2}$	$(4.38 \pm 0.26)\times10^{-4}$	$(1.44 \pm 0.19)\times10^{-3}$
Transformer	3.26	$(3.42 \pm 0.32)\times10^{-2}$	$(4.56 \pm 0.72)\times10^{-5}$	$(1.28 \pm 0.61)\times10^{-4}$
Body Transformer	3.26	$(3.83 \pm 0.24)\times10^{-2}$	$(4.68 \pm 0.40)\times10^{-5}$	$(1.13 \pm 0.24)\times10^{-4}$
GAT	1.54	$(6.95 \pm 0.45)\times10^{-2}$	$(4.32 \pm 0.13)\times10^{-4}$	$(3.16 \pm 0.94)\times10^{-3}$
Rodrigues	0.27	$(4.81 \pm 0.50)\times10^{-5}$	$(1.78 \pm 0.12)\times10^{-7}$	$(6.30 \pm 2.36)\times10^{-8}$
Screw (ours)	0.18	$(6.99 \pm 2.38)\times10^{-7}$	$(1.39 \pm 0.24)\times10^{-8}$	$(4.34 \pm 2.65)\times10^{-9}$

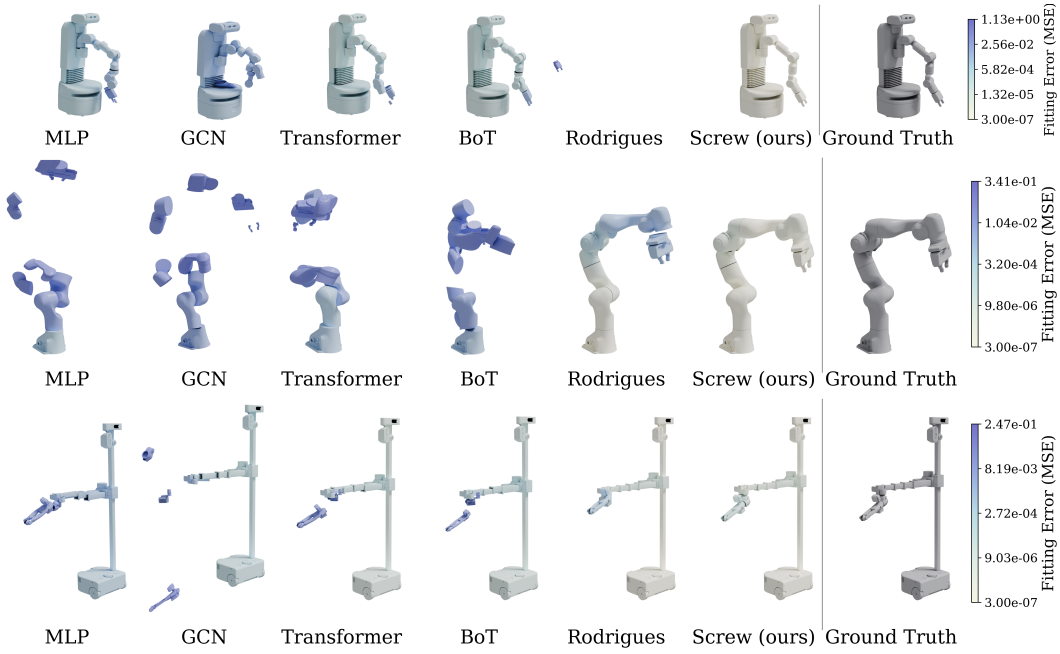

Figure 9: Per-link FK error on Fetch, the fixed-base Franka and Stretch, rendered as in Figure 8d with the same six models. Each robot is at Rodrigues' worst seed by logged full-range error at 60k: 1 for Fetch, 2 for Franka and 0 for Stretch. The same rule picks the configuration from the robot's own batch of 4096. There, Rodrigues and ours err by 0.84 and 2.4×10^{-8} on Fetch, 1.5×10^{-3} and 7.6×10^{-7} on Franka, and 9.7×10^{-5} and 7.0×10^{-6} on Stretch. Each strip has its own color scale, from 3×10^{-7} to the robot's largest per-link error: 1.1, 0.34 and 0.25. On Fetch, the rule skips configurations with Rodrigues MSE above 1; uncapped, it would pick one at 55. Even so, Rodrigues shifts every link by about 3.3 m through the mobile-base joints, so the robot leaves the frame. The panel still understates Rodrigues' typical Fetch error, a mean of 20.

error ratio among the ten with the largest Rodrigues error: 4.4×10^{-4} against 3.3×10^{-7}. Links are shaded by their 12-D pose MSE there, on one log scale from 3×10^{-7} to 0.586, the largest per-link error among the six models. Each link is drawn at $T_{\text{gt}}^{\text{disp}}\,(T_{\text{gt}}^{q})^{-1}\,T_{\text{pred}}^{q}$, which carries its pose error at the query configuration onto a common display pose. Figure 9 covers three more robots.

Table 14: Twists identified from the checkpoints behind Table 11 (final iterate, five seeds): median [max] over revolute (joint, seed) pairs. The reference is the URDF twist in the same frame, $(R_A a, -R_A a \times p_A)$ for a revolute and $(0, R_A a)$ for a prismatic joint, with rate 1; $A = (R_A, p_A)$ is the child-in-parent pose at the zero configuration and a the URDF axis. Axis error: angle between the learned and the URDF axis. Rate error: $\big|\,\|\omega\| - 1\big|$. Moment: the residual $\|v + \omega \times p\|$ in mm per rad at the URDF joint origin p, zero when the learned axis passes through the joint. Half-width: median training half-width of the revolute joints. Ours beats Rodrigues on the first five robots and loses on the last five.

Robot	joints (R/P)	half-width (°)	axis error (°)	rate error	moment (mm/rad)
Fetch (mobile)	8/5	90	0.029 [0.08]	$7.6{\times}10^{-5}$ [$1.4{\times}10^{-3}$]	1.1
Franka (mobile)	8/4	83	0.030 [0.29]	$3.7{\times}10^{-5}$ [$7.0{\times}10^{-3}$]	1.1
G1 (23-DoF)	23/0	55	0.021 [0.14]	$4.2{\times}10^{-5}$ [$1.9{\times}10^{-3}$]	0.7
Franka	7/2	83	0.019 [0.09]	$3.6{\times}10^{-5}$ [$2.3{\times}10^{-3}$]	0.7
Stretch	3/5	82	0.079 [0.36]	$2.8{\times}10^{-4}$ [$9.2{\times}10^{-3}$]	2.4
LEAP hand	16/0	34	0.259 [0.70]	$1.2{\times}10^{-3}$ [$6.9{\times}10^{-3}$]	4.5
PhantomX	18/0	75	0.065 [0.21]	$3.0{\times}10^{-4}$ [$2.2{\times}10^{-3}$]	1.1
Allegro hand	16/0	30	0.361 [1.36]	$3.4{\times}10^{-3}$ [$2.0{\times}10^{-2}$]	5.1
UR5	6/0	22	0.458 [1.88]	$7.5{\times}10^{-3}$ [$1.4{\times}10^{-2}$]	16.6
Go1	12/0	28	0.213 [0.54]	$1.8{\times}10^{-3}$ [$8.7{\times}10^{-3}$]	3.7
All ten	117/16	–	0.085 [1.88]	$4.4{\times}10^{-4}$ [$2.0{\times}10^{-2}$]	2.3

G.5 Twist identification

Each joint owns 12 blocks × 2 sides × R=12 raw twists. All start at zero (Appendix A), so the R copies on a side receive identical gradients and are bit-identical in all 50 checkpoints: 24 distinct raw twists per joint. A raw twist acts on learned 4×4 feature frames not constrained to SE(3), so only its rate $\|\omega\|$ is comparable with the URDF: $\hat{\xi}$ has eigenvalues 0 and $\pm i\|\omega\|$ in any feature basis.

We therefore read each joint's twist from the model's outputs. With $X(\theta) = T_p^{-1} T_c(\theta)$ the predicted child-link pose in the parent link's frame, rotations projected to SO(3), a product of exponentials gives $X(\theta)X(\theta_0)^{-1} = \exp\big(\hat{\xi}(\theta - \theta_0)\big)$ for a fixed ξ. Sweeping the joint over 49 points of its full test range at 16 test configurations of the other joints, we fit $\xi = (\omega, v)$ by least squares to the SE(3) logarithms of the chords from the range center θ_0, up to 3 rad for revolute joints. Applied to the URDF's own FK, it recovers every axis within 10^{-6} degrees and every rate within 10^{-14}.

Over the 585 revolute (joint, seed) pairs, the medians are 0.085° of axis error and $4.4{\times}10^{-4}$ of rate error (Table 14). Both are smaller on the robots where our operator beats Rodrigues, at 0.024° and $4.3{\times}10^{-5}$, than on those where it loses, at 0.23° and $1.3{\times}10^{-3}$. The losers include the four robots with the narrowest training windows: median half-widths of 22–34°, against 55–90° for the winners. PhantomX, trained on ±75°, loses although its joints are identified to 0.065°. Beyond the training window, on chords that end outside it, the medians are 0.091° and $4.8{\times}10^{-4}$. The six long-travel prismatic joints (Fetch base and torso, mobile-Franka base, Stretch lift) stay within 2.4° and a rate error of $8{\times}10^{-3}$ on every seed. The short-travel fingers and telescope segments, trained on half-widths of 1–3.25 cm, reach per-joint medians of 0.6–6°. In the raw parameters, twists within 5% of unit rate carry a median 99.3% of each revolute joint's motion effect, the RMS change of all predicted link poses when a twist's angle is frozen; UR5 is the exception, at 3.9%.

G.6 Synthetic arm

The synthetic arm of Figure 8c has six revolute joints and the same protocol and recipe. It covers eight joint ranges $\pm r$, with training half-widths of 11.5° to 57.3° and 1–3 seeds per range and model. Points are seed means of the best test MSE, which equals the final value at every width shown; bands span the seeds. The crossover at 18.6° interpolates log(ours/Rodrigues) linearly between the two bracketing widths.

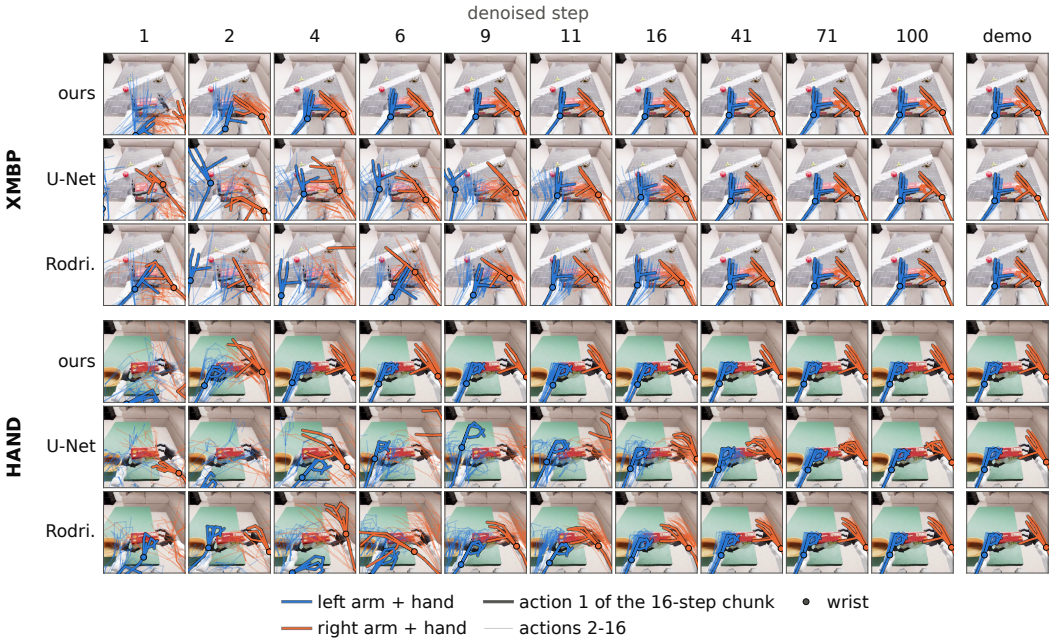

Figure 10: XMBP (episode 0, grasp frame) and Handover (episode 15, the handover frame), drawn as in Figure 4.

Where Rodrigues wins The Neural Screw Operator has one quantity more to learn than the Rodrigues operator: the pitch of each screw, which separates a revolute joint from a prismatic one (Proposition 1). Rodrigues hard-wires it to zero and pays nothing to identify it; ours needs enough rotation in the training data to pin it down. Rodrigues therefore wins on five of six all-revolute robots, by at most 5.5×, and on the synthetic arm whenever training covers less than ±19° of each joint; beyond that width the Neural Screw Operator overtakes it (Figure 8c). Only the Neural Screw Operator averages below 10^{-4} on all ten robots.

H Mechanism analyses

H.1 Denoising Efficiency

Figure 10 shows the XMBP and Handover strips behind the errors reported in Section 5.3, drawn like the LPBT strip of Figure 4. As there, the U-Net is Diffusion Policy's design at BSP's size, a 13.47M trunk reading one pooled DINOv2 vector (`unetpool_matched`: 35.72M parameters in total, fp32, seed 2027), and the Rodrigues Network is the size-matched variant on a pooled DINOv2 vector, not the row of Table 1. The strips follow the 100-step DDPM chain that the served policy runs; Table 1 runs every arm at its native sampler.

XMP, the fourth row of the right panel of Figure 4, has no strip here: at the grasp frame of episode 0, chosen by the same rule as on LPBT and XMBP, BSP's checkpoint of Table 1 and a pooled U-Net and Rodrigues Network trained on XMP with the same recipe as on the other tasks were captured on a second RTX 5090 machine. An earlier capture of BSP and the Rodrigues Network on the first machine, from the same checkpoints and initial noise, reaches 1° and 5° and crosses 5 and 2 cm of wrist error at the same iterations; its wrist error stays within 0.05 and 0.12 mm of the later capture's, and the largest difference in a joint coordinate, at the first iteration, is 3.1 and 13.2 mm.

After 11 of 100 iterations, the BSP wrist is 0.8, 0.9, 1.1 and 0.7 cm from the demonstration on LPBT, XMBP, Handover and XMP. The size-matched U-Net is at 5.2, 4.7, 5.9 and 4.1 cm, and the Rodrigues Network at 3.6, 3.4, 4.3 and 2.6 cm. The ordering also holds against the 69.67M U-Net of Table 10, which succeeds on Handover. On Handover, its wrist is 18.6, 2.9 and 0.8 cm from the

demonstration after 2, 11 and 100 iterations, against 6.5, 1.1 and 0.2 cm for BSP. After 100 iterations, the size-matched U-Net's wrist is 0.6, 0.4, 1.0 and 0.11 cm from the demonstration on LPBT, XMBP, Handover and XMP, against 0.5, 0.4, 0.2 and 0.06 cm for BSP.

For the right panel of Figure 4, the error of an iteration is the mean absolute difference between $\hat{x}_0$ and the demonstration chunk over the 14 arm-joint commands and the 16 steps of the chunk, in degrees, on the frames and captures of Figures 4 and 10 and on the XMP capture above; the panel marks the first iteration at which it falls below 1°; the 5° crossings are tabulated here, not drawn. Two crossings are not final (none on XMP): BSP on XMBP returns to at most 1.2° between iterations 11 and 23, and the U-Net on Handover hovers at 1° after iteration 92. To 1°, BSP, the Rodrigues Network and the U-Net take 8, 38 and 59 iterations on LPBT; 8, 39 and 60 on XMBP; 17, 39 and 92 on Handover; and 7, 25 and 45 on XMP. To 5°, they take 2, 9 and 15; 2, 8 and 11; 3, 10 and 19; and 2, 7 and 10. At 2, 11 and 100 iterations the normalized arm-joint MSE is $0.05/0.001/1.5\times10^{-4}$ (LPBT), $0.08/0.004/15\times10^{-4}$ (XMBP), $0.07/0.002/2.8\times10^{-4}$ (Handover) and $0.07/0.002/0.23\times10^{-4}$ (XMP) for BSP; for the U-Net it is $1.00/0.081/1.8\times10^{-4}$, $0.86/0.085/22\times10^{-4}$, $1.02/0.11/14\times10^{-4}$ and $0.97/0.064/0.65\times10^{-4}$, and for the Rodrigues Network $0.73/0.035/2.0\times10^{-4}$, $0.99/0.066/12\times10^{-4}$, $0.91/0.045/1.9\times10^{-4}$ and $0.78/0.027/0.33\times10^{-4}$.

H.2 Visual Sensitivity maps

Instrument For action row i, one of the 31 joints or five base commands, and image patch p, the sensitivity is $S_i(p) = \|\partial\hat{a}_i/\partial h_p\|_2$ at the last denoising step, where $\hat{a}_i$ is the row's command averaged over the executed 16-step horizon and the chain up to that step is held fixed. At level A, h_p is the DINOv2 output token of patch p as the denoiser reads it; at level B, it is the patch's embedding at the encoder input, so the gradient passes through the whole encoder. Level B is the only level defined for the pooled arms, whose level-A map is uniform by construction because they mean-pool the tokens. Level A is the level we report for the two token-reading arms, BSP and the same-token U-Net; level B is the level at which they can be compared with the pooled arms. Each row's map is normalized to unit mass over the 256 patches. The object region (on LPBT, the box) and the target region (the bowl) come from colour masks of the rendered frame, each dilated by one patch and checked by eye on 20 frames per task; the elsewhere region is the rest of the frame. The phases (approach, grasp, lift–transfer and carry–place) come from the demonstration's event anchors, and the 3 pre-approach frames are excluded. We use 200 held-out LPBT frames, 22, 20, 87 and 68 per phase. A region's enrichment is its mass divided by its share of the frame, so a uniform map scores 1. The permuted-joint null N1 is the mass that another row's map places on the region, averaged over 100 derangements of the row-to-map assignment within the kinematic tree, with five random tree joints as donors for the base rows. A group's excess is its own mass minus N1; we also report the fraction of frames in which its own mass is above N1. A random region of the same area gives a second null, N2, which is computed but not used here.

Arms BSP is the checkpoint of Table 1. The comparators are the size-matched pooled U-Net and pooled Rodrigues Network of Figure 4 (`unetpool_matched`: 35.72M parameters in total, fp32, seed 2027; `rodpool_matched`: 35.64M parameters in total, seed 2027). We also report the same-token U-Net of Table 16.

Results Table 15 gives the LPBT readings. At level A, across the four phases, the object carries 2.42 to 3.26 times its area share of BSP's sensitivity and 1.15 to 1.31 times the same-token U-Net's; BSP's right arm is above the null in 90 to 100% of frames and its base below it in every frame, while no group of the same-token U-Net departs from the null by more than 0.003. At level B, the object carries 1.79 to 2.45 times its area share of BSP's sensitivity, against 1.12 to 1.62 for the pooled U-Net, 1.19 to 1.48 for the pooled Rodrigues Network and 1.03 to 1.53 for the same-token U-Net. BSP's right arm places more mass on the object than the null in 91 to 96% of frames, while its base places less in every frame, by 0.078 to 0.099 of its sensitivity, and more on the rest of the frame. The pooled arms show the same signs, with the right arm above the null in 90 to 100% of frames and the base below it in 79 to 100%, but their base deficit is 0.010 to 0.027, and their right-arm excess, 0.026 to 0.042, is larger than BSP's 0.012 to 0.026.

Table 15: Sensitivity maps on LPBT at level A (the tokens the denoiser reads) and level B (the encoder input), by phase (22, 20, 87 and 68 frames); the pooled arms have no level-A read. Enrichment: the region's share of the sensitivity divided by its share of the frame (1 = uniform), over all 36 action rows. Excess: a group's mass on the region minus the permuted-joint null N1, as a fraction of the group's total sensitivity, with the fraction of frames above N1 in parentheses. R arm: the seven right-arm joints; waist: the three waist joints; base: the five base commands. Elsewhere: the frame outside the box and the bowl. Pooled U-Net and pooled Rodrigues: the size-matched comparators of Figure 4; same-token U-Net: the U-Net of Table 16.

Arm	Group	Approach	Grasp	Lift–transfer	Carry–place
Level A (tokens the denoiser reads), object region: enrichment, all 36 rows					
BSP	All	3.26	2.80	2.63	2.42
Same-token U-Net	All	1.31	1.26	1.20	1.15
Level A, object region: excess over N1 (fraction of frames above N1)					
BSP	R arm	+0.023 (0.95)	+0.025 (0.90)	+0.033 (0.99)	+0.044 (1.00)
	Waist	−0.041 (0.05)	−0.021 (0.35)	−0.003 (0.53)	−0.006 (0.44)
	Base	−0.228 (0.00)	−0.219 (0.00)	−0.302 (0.00)	−0.204 (0.00)
Same-token U-Net	R arm	+0.002 (1.00)	+0.002 (1.00)	+0.002 (1.00)	+0.002 (1.00)
	Waist	+0.000 (0.82)	+0.001 (0.85)	+0.001 (0.93)	+0.001 (0.94)
	Base	−0.001 (0.00)	−0.000 (0.30)	+0.000 (0.41)	−0.000 (0.43)
Level B (encoder input), object region: enrichment, all 36 rows					
BSP	All	2.45	2.16	2.10	1.79
Pooled U-Net	All	1.62	1.61	1.33	1.12
Pooled Rodrigues	All	1.48	1.45	1.19	1.20
Same-token U-Net	All	1.53	1.39	1.19	1.03
Level B, object region: excess over N1 (fraction of frames above N1)					
BSP	R arm	+0.012 (0.91)	+0.024 (0.95)	+0.026 (0.95)	+0.025 (0.96)
	Waist	−0.023 (0.09)	−0.022 (0.15)	−0.014 (0.25)	−0.016 (0.24)
	Base	−0.078 (0.00)	−0.094 (0.00)	−0.099 (0.00)	−0.089 (0.00)
Pooled U-Net	R arm	+0.026 (1.00)	+0.029 (1.00)	+0.038 (1.00)	+0.032 (1.00)
	Waist	−0.012 (0.09)	−0.021 (0.05)	−0.014 (0.14)	−0.011 (0.07)
	Base	−0.019 (0.00)	−0.023 (0.05)	−0.010 (0.21)	−0.010 (0.18)
Pooled Rodrigues	R arm	+0.032 (1.00)	+0.042 (1.00)	+0.041 (1.00)	+0.030 (0.90)
	Waist	−0.020 (0.05)	−0.022 (0.00)	−0.020 (0.02)	−0.027 (0.01)
	Base	−0.027 (0.00)	−0.023 (0.00)	−0.015 (0.10)	−0.020 (0.00)
Level B, elsewhere region: excess over N1 (fraction of frames above N1)					
BSP	Base	+0.078 (1.00)	+0.094 (1.00)	+0.099 (1.00)	+0.097 (1.00)
Pooled U-Net	Base	+0.019 (1.00)	+0.021 (0.95)	+0.010 (0.79)	+0.006 (0.71)
Pooled Rodrigues	Base	+0.027 (1.00)	+0.020 (0.95)	+0.014 (0.87)	+0.018 (0.90)

Other tasks At level B, the compact read is an LPBT result. On Handover, where the object is large and central, all three arms read it alike and close to uniform: over approach, grasp, lift–transfer and carry–place (53, 49, 78 and 17 frames), the object's enrichment is 1.45, 1.58, 1.11 and 0.93 for BSP, 1.36, 1.55, 1.06 and 0.81 for the pooled U-Net, and 1.42, 1.61, 1.18 and 0.81 for the pooled Rodrigues Network. On XMBP, the approach holds 53 frames with a defined object region and the grasp only 2; at the approach, BSP's enrichment is 1.50, against 0.96 and 1.00 for the pooled U-Net and Rodrigues Network.

H.3 Visual Intervention study

Intervention We ask whether each body-part group reads its own part of the image. The six groups are the left and right fingers (LF, RF), arms with wrists (LA, RA), waist and torso (W), and base and legs (B). For each of 256 training frames per task and each group, the token Jacobian at the last denoising step ranks the 256 image patches by how strongly they move the group's actions. The own swap replaces the top 26 patches in the image, before the encoder, with the mean frame's content. We then rerun the full 100-step chain from the same initial noise and record the change of the group's own actions (Figure 11, Table 16). The other swap replaces 26 equally salient patches chosen by another group, and the random swap 26 random patches.

Measures On each frame, the ratio divides the own change by the other change; we report its median with a 95% episode-cluster percentile interval from 10,000 resamples. P is the fraction of frames with own > other, where ties count as not greater. The specificity $s = (\text{own} - \text{other})/(\text{own} + \text{other})$ is averaged over the frames where it is defined. The counts $n_{\text{eff}}/n_0/n_\infty$ are the frames with a defined

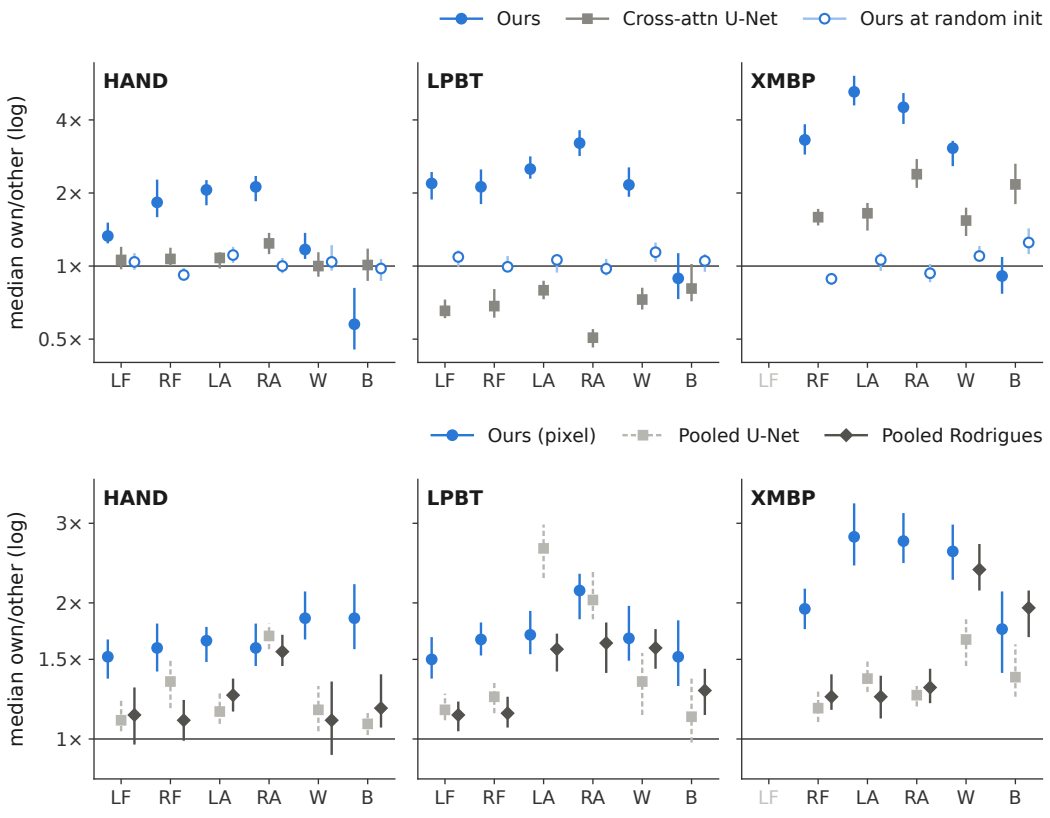

Figure 11: Part-specific visual dependence. Median own/other ratio per part with 95% episode-cluster intervals; 1× means that blanking the part's own patches and blanking another part's equally salient patches move the part's commands equally. Top: token-selected patches, BSP against the U-Net that reads the same tokens and against BSP at random initialization. Bottom: pixel-selected patches, BSP against the Diffusion Policy U-Net and the Rodrigues Network. LF, RF: left and right fingers; LA, RA: left and right arm with wrist; W: waist; B: base. XMBP never moves the left hand.

ratio, with own = other = 0, and with an infinite ratio. BSP and the same-token U-Net are the flagship and 69.67M trunk-swap checkpoints of Table 10. XMBP never moves the left hand.

Results On Handover, BSP's five upper-body ratios lie between 1.17 and 2.12; every U-Net ratio lies between 1.00 and 1.24. On LPBT, BSP's upper-body ratios lie between 2.12 and 3.21, while every U-Net ratio falls below 1, from 0.51 to 0.81. On XMBP, BSP reaches 3.06 to 5.22 on the four upper-body groups that move, the U-Net 1.54 to 2.39. On all three tasks, BSP's own swap displaces the wrist by 0.45 to 1.09 cm, the other swap by 0.14 to 0.45 cm. The base and legs are the exception: BSP's ratio stays at or below 0.91 on every task.

Pixel-selected patches The parameter-matched Rodrigues Network variant, a 13.4M denoiser on a pooled ResNet-18 trained from scratch, has no tokens to swap. For it and for BSP, we select the 26 patches on Handover from a pixel-level sensitivity map instead (Table 17). This variant is not the Table 1 checkpoint and succeeds in 1 of 90 closed-loop episodes on the three tasks. Its ratios lie between 1.29 and 1.60, against 1.52 to 1.85 for BSP. We do not compare pixel-selected with token-selected ratios. For BSP, the token-level and pixel-level maps have a cosine of 0.61 to 0.78 per group, and their top-26 patch sets share 4 to 8 patches.

Size-matched pooled comparators The size-matched pooled U-Net and pooled Rodrigues Network of Figure 4, which read a pooled DINOv2 vector rather than the ResNet-18 of the variant above, have no tokens either. We therefore repeat the study for them on all three tasks with pixel-selected

Table 16: Intervention study with token-selected patches, 256 training frames per task. Own, Other, Random: mean change of the group's own actions under each 26-patch swap, in raw action units ($\times 10^{-3}$). Wrist: isolated wrist displacement under the three swaps, for the arm groups. Ratio, P, s and the frame counts are defined in the text. LF/RF: left/right fingers; LA/RA: left/right arm and wrist; W: waist and torso; B: base and legs.

Task	Group	Arm	Own	Other	Random	Ratio (95% CI)	P	s	$n_{\text{eff}}/n_0/n_\infty$	Wrist (cm)
HAND	LF	BSP	9.6	4.6	3.3	1.33 (1.24–1.51)	0.69	0.17	256/0/0	N/A
HAND	RF	BSP	36.5	8.9	5.0	1.83 (1.59–2.27)	0.82	0.36	256/0/0	N/A
HAND	LA	BSP	11.7	6.0	4.7	2.06 (1.78–2.26)	0.84	0.30	256/0/0	0.51/0.27/0.21
HAND	RA	BSP	9.5	4.7	3.9	2.12 (1.85–2.35)	0.84	0.32	256/0/0	0.45/0.21/0.19
HAND	W	BSP	4.0	3.5	2.0	1.17 (1.07–1.37)	0.60	0.09	256/0/0	N/A
HAND	B	BSP	2.4	4.9	1.8	0.58 (0.45–0.81)	0.37	-0.18	256/0/0	N/A
HAND	LF	U-Net	18.8	17.6	10.6	1.06 (0.97–1.20)	0.54	0.06	256/0/0	N/A
HAND	RF	U-Net	8.3	4.8	3.4	1.07 (1.00–1.19)	0.57	0.05	256/0/0	N/A
HAND	LA	U-Net	11.7	11.1	5.8	1.08 (0.98–1.14)	0.56	0.03	256/0/0	0.55/0.54/0.26
HAND	RA	U-Net	11.3	8.8	5.0	1.24 (1.12–1.37)	0.67	0.12	256/0/0	0.58/0.37/0.21
HAND	W	U-Net	3.6	3.6	2.1	1.00 (0.90–1.14)	0.50	-0.00	256/0/0	N/A
HAND	B	U-Net	4.9	5.2	2.4	1.01 (0.87–1.18)	0.51	0.01	256/0/0	N/A
LPBT	LF	BSP	0.11	0.04	0.02	2.19 (1.88–2.44)	0.87	0.35	256/0/0	N/A
LPBT	RF	BSP	20.1	9.8	2.6	2.12 (1.80–2.50)	0.82	0.33	256/0/0	N/A
LPBT	LA	BSP	11.3	4.4	2.1	2.51 (2.29–2.83)	0.91	0.41	256/0/0	0.59/0.21/0.11
LPBT	RA	BSP	25.6	9.1	4.2	3.21 (2.84–3.63)	0.94	0.49	256/0/0	1.09/0.45/0.20
LPBT	W	BSP	10.8	5.0	2.1	2.16 (1.93–2.55)	0.85	0.35	256/0/0	N/A
LPBT	B	BSP	24.0	56.2	6.3	0.89 (0.73–1.13)	0.47	-0.04	256/0/0	N/A
LPBT	LF	U-Net	0.20	0.27	0.14	0.65 (0.61–0.73)	0.24	-0.18	256/0/0	N/A
LPBT	RF	U-Net	7.2	11.7	3.4	0.68 (0.61–0.80)	0.27	-0.17	256/0/0	N/A
LPBT	LA	U-Net	3.9	4.8	2.7	0.80 (0.73–0.87)	0.34	-0.10	256/0/0	0.19/0.23/0.14
LPBT	RA	U-Net	7.5	13.7	4.8	0.51 (0.46–0.55)	0.18	-0.29	256/0/0	0.36/0.73/0.24
LPBT	W	U-Net	2.6	3.3	2.1	0.73 (0.66–0.81)	0.35	-0.13	256/0/0	N/A
LPBT	B	U-Net	17.2	24.9	10.6	0.81 (0.72–1.02)	0.44	-0.05	256/0/0	N/A
XMBP	LF	BSP	0.00	0.00	0.00	N/A	N/A	N/A	0/256/0	N/A
XMBP	RF	BSP	19.9	7.3	5.2	3.31 (2.88–3.84)	0.95	0.51	256/0/0	N/A
XMBP	LA	BSP	14.2	2.8	2.3	5.22 (4.59–6.08)	0.99	0.64	256/0/0	0.68/0.14/0.11
XMBP	RA	BSP	17.3	4.3	3.0	4.51 (3.85–5.16)	0.98	0.60	256/0/0	0.70/0.19/0.14
XMBP	W	BSP	5.9	2.1	1.8	3.06 (2.58–3.28)	0.93	0.44	256/0/0	N/A
XMBP	B	BSP	4.6	6.2	2.0	0.91 (0.77–1.09)	0.45	-0.04	256/0/0	N/A
XMBP	LF	U-Net	0.00	0.00	0.00	N/A	N/A	N/A	0/256/0	N/A
XMBP	RF	U-Net	4.0	9.0	3.3	1.59 (1.47–1.72)	0.81	0.21	256/0/0	N/A
XMBP	LA	U-Net	5.1	3.5	2.7	1.65 (1.40–1.82)	0.83	0.21	256/0/0	0.23/0.16/0.12
XMBP	RA	U-Net	8.9	4.5	3.6	2.39 (2.10–2.76)	0.91	0.37	256/0/0	0.34/0.21/0.16
XMBP	W	U-Net	3.1	2.1	1.6	1.54 (1.33–1.74)	0.77	0.20	256/0/0	N/A
XMBP	B	U-Net	6.6	3.2	2.6	2.17 (1.80–2.64)	0.80	0.31	256/0/0	N/A

Table 17: Intervention study on Handover with pixel-selected patches, for BSP and the parameter-matched Rodrigues Network variant on a pooled ResNet-18. This variant is not the Table 1 checkpoint and fails Handover in closed loop. Columns as in Table 16.

Task	Group	Arm	Own	Other	Random	Ratio (95% CI)	P	s	$n_{\text{eff}}/n_0/n_\infty$	Wrist (cm)
HAND	LF	Rodrigues-matched (pixel)	14.4	7.3	6.3	1.31 (1.21–1.41)	0.74	0.14	256/0/0	N/A
HAND	RF	Rodrigues-matched (pixel)	8.7	4.9	3.3	1.29 (1.20–1.41)	0.75	0.17	256/0/0	N/A
HAND	LA	Rodrigues-matched (pixel)	16.0	10.3	7.2	1.50 (1.36–1.68)	0.76	0.20	256/0/0	0.70/0.47/0.33
HAND	RA	Rodrigues-matched (pixel)	14.6	9.5	6.7	1.49 (1.31–1.68)	0.78	0.20	256/0/0	0.62/0.40/0.29
HAND	W	Rodrigues-matched (pixel)	4.0	2.8	1.8	1.57 (1.32–1.80)	0.71	0.17	256/0/0	N/A
HAND	B	Rodrigues-matched (pixel)	6.0	5.2	3.1	1.60 (1.39–1.75)	0.72	0.18	256/0/0	N/A
HAND	LF	BSP (pixel)	9.5	8.7	3.3	1.52 (1.36–1.66)	0.75	0.18	256/0/0	N/A
HAND	RF	BSP (pixel)	17.2	7.8	5.0	1.59 (1.41–1.80)	0.74	0.27	256/0/0	N/A
HAND	LA	BSP (pixel)	11.7	6.9	4.7	1.65 (1.48–1.77)	0.81	0.22	256/0/0	0.52/0.31/0.21
HAND	RA	BSP (pixel)	9.5	6.0	3.9	1.59 (1.45–1.80)	0.81	0.22	256/0/0	0.44/0.27/0.19
HAND	W	BSP (pixel)	5.7	3.4	2.0	1.85 (1.66–2.12)	0.85	0.28	256/0/0	N/A
HAND	B	BSP (pixel)	6.2	3.6	1.8	1.85 (1.58–2.20)	0.75	0.27	256/0/0	N/A

patches, and run BSP under the same rule; its Handover cells are those of Table 17. The protocol is otherwise that of Table 16: the same 256 training tuples per task and six groups, 26 of the 256 patches per group, now ranked by the pixel-level map, own, other and random swaps with the mean frame's content before the encoder, and the full 100-step chain from the same initial noise; the row statistics come from the same code. Each difference, BSP minus the comparator, applies one

Table 18: Intervention study with pixel-selected patches: BSP and the size-matched pooled comparators of Figure 4, 256 training frames per task. Median own/other ratio, with a 95% episode-cluster interval for BSP. Δ: BSP's median minus the comparator's, with a paired 95% episode-cluster interval. XMBP never moves the left hand, so its LF row is undefined and omitted.

Task	Group	BSP (pixel)	Pooled U-Net	Pooled Rodrigues	Δ vs U-Net	Δ vs Rodrigues
HAND	LF	1.52 [1.36, 1.66]	1.10	1.13	0.41 [0.24, 0.57]	0.39 [0.17, 0.63]
HAND	RF	1.59 [1.41, 1.80]	1.34	1.10	0.25 [0.014, 0.51]	0.49 [0.23, 0.73]
HAND	LA	1.65 [1.48, 1.77]	1.15	1.25	0.50 [0.27, 0.62]	0.40 [0.21, 0.52]
HAND	RA	1.59 [1.45, 1.80]	1.69	1.56	−0.095 [−0.26, 0.12]	0.034 [−0.17, 0.24]
HAND	W	1.85 [1.66, 2.12]	1.16	1.10	0.69 [0.49, 0.96]	0.74 [0.48, 1.01]
HAND	B	1.85 [1.58, 2.20]	1.08	1.17	0.77 [0.47, 1.14]	0.68 [0.27, 1.05]
LPBT	LF	1.50 [1.36, 1.68]	1.16	1.13	0.34 [0.17, 0.53]	0.37 [0.20, 0.57]
LPBT	RF	1.66 [1.53, 1.81]	1.24	1.14	0.42 [0.24, 0.58]	0.52 [0.33, 0.67]
LPBT	LA	1.70 [1.54, 1.92]	2.64	1.58	−0.94 [−1.34, −0.50]	0.12 [−0.085, 0.39]
LPBT	RA	2.13 [1.84, 2.32]	2.03	1.63	0.094 [−0.29, 0.33]	0.49 [0.17, 0.81]
LPBT	W	1.67 [1.49, 1.97]	1.34	1.59	0.33 [0.043, 0.68]	0.076 [−0.19, 0.39]
LPBT	B	1.52 [1.31, 1.83]	1.12	1.28	0.41 [0.11, 0.72]	0.24 [−0.023, 0.58]
XMBP	RF	1.94 [1.75, 2.15]	1.17	1.24	0.77 [0.50, 1.00]	0.70 [0.44, 0.93]
XMBP	LA	2.80 [2.42, 3.32]	1.36	1.24	1.44 [1.08, 1.95]	1.56 [1.11, 2.11]
XMBP	RA	2.74 [2.45, 3.16]	1.25	1.30	1.50 [1.19, 1.93]	1.44 [1.10, 1.88]
XMBP	W	2.60 [2.25, 2.98]	1.66	2.37	0.94 [0.52, 1.36]	0.22 [−0.22, 0.63]
XMBP	B	1.75 [1.40, 2.12]	1.37	1.95	0.38 [−0.11, 0.79]	−0.20 [−0.65, 0.27]

episode-cluster resample stream per task, group and comparator (10,000 resamples) to both arms. As a known-answer check, the flagship LPBT cell run through the comparators' code path reproduces its earlier pixel-selected cell bitwise.

Of the 34 defined differences, 24 have a 95% interval above zero and one below: the left arm on LPBT against the pooled U-Net, whose ratio is 2.64 against 1.70 for BSP. The other nine span zero. Under the pixel rule, BSP's ratios on LPBT and XMBP lie between 1.50 and 2.80.

Diagnostics Table 19 repeats the study in two settings. With its denoiser at random initialization behind the pretrained encoder, BSP's ratios lie between 0.89 and 1.25 on every task and moving group; this model has no policy training and uses the flagship checkpoint's normalizer. On XMBP, we also train BSP and the U-Net on 50 demonstrations and test them on frames from held-out episodes. There, BSP's specificity exceeds the U-Net's by 0.17 on RF, over 243 paired frames, and by 0.08, 0.15 and 0.08 on LA, RA and W; each difference has a 95% interval above zero. On the base and legs, it is 0.42 lower. The held-out U-Net has 13 right-finger frames with own = other = 0, where s is undefined; they count under n_0.

I Model size, speed and the fused kernel

I.1 Size and speed

Table 20 lists sizes and speeds. The structural prior replaces generic capacity rather than adding to it. The 35.04M served parameters are 22.06M of DINOv2 and a 12.98M kinematic trunk, the 12.78M denoiser plus the 0.20M token projection, under half the size of the official DP denoising trunk alone. A 100-step DDPM action chunk takes 308.6 ms, against 324.5 ms for the U-Net with the same DINOv2 encoder; both are 100-step chunks. Table 1 serves BSP with ten DDPM iterations and trailing timestep spacing. On LPBT and XMBP the 100-step draws of the same checkpoints score 23 and 28 against the table's 24 and 27, while three iterations drop them to 12 and 20 (one draw each). BSP trains slower per step: its kinematic trunk runs in fp32 rather than bf16 and adds the operator's cost. The prior buys trainable parameters, memory, and serve latency, not training throughput.

I.2 Fused kernel

One fused CUDA kernel evaluates Eq. 6 end to end in fp32: the closed-form exponential, both action sides, and the channel mixing. Its gradients are analytic for twists, mixing weights, angles, and link features. The coefficients of Eqs. 2–3 become $0/0$ as $\varphi \to 0$ but are analytic in $x = \varphi^2$. For $x < 10^{-4}$

Table 19: Intervention diagnostics: BSP with its denoiser at random initialization behind the pretrained encoder, and the 50-demonstration XMBP checkpoints of BSP and the U-Net on frames from held-out episodes. Columns as in Table 16.

Task	Group	Arm	Own	Other	Random	Ratio (95% CI)	P	s	$n_{\text{eff}}/n_0/n_\infty$	Wrist (cm)
HAND	LF	BSP, random init	1.1	1.0	0.88	1.04 (0.96–1.13)	0.53	0.01	256/0/0	N/A
HAND	RF	BSP, random init	0.79	0.86	0.72	0.92 (0.88–0.98)	0.43	-0.03	256/0/0	N/A
HAND	LA	BSP, random init	1.1	0.99	0.91	1.11 (1.03–1.20)	0.59	0.05	256/0/0	0.06/0.06/0.05
HAND	RA	BSP, random init	0.89	0.86	0.75	1.00 (0.93–1.08)	0.50	0.01	256/0/0	0.05/0.05/0.04
HAND	W	BSP, random init	0.60	0.57	0.52	1.04 (0.96–1.22)	0.53	0.03	256/0/0	N/A
HAND	B	BSP, random init	3.2	3.3	2.7	0.98 (0.87–1.07)	0.49	-0.02	256/0/0	N/A
LPBT	LF	BSP, random init	1.1	0.98	0.97	1.09 (0.99–1.16)	0.55	0.04	256/0/0	N/A
LPBT	RF	BSP, random init	1.1	1.1	1.0	0.99 (0.95–1.10)	0.49	0.02	256/0/0	N/A
LPBT	LA	BSP, random init	0.72	0.70	0.69	1.06 (0.94–1.10)	0.54	0.01	256/0/0	0.05/0.05/0.05
LPBT	RA	BSP, random init	1.3	1.3	1.3	0.97 (0.92–1.07)	0.49	0.00	256/0/0	0.08/0.07/0.07
LPBT	W	BSP, random init	0.64	0.56	0.58	1.14 (1.04–1.25)	0.57	0.06	256/0/0	N/A
LPBT	B	BSP, random init	12.2	11.5	11.9	1.05 (0.95–1.12)	0.54	0.03	256/0/0	N/A
XMBP	LF	BSP, random init	0.00	0.00	0.00	N/A	N/A	N/A	0/256/0	N/A
XMBP	RF	BSP, random init	0.69	0.79	0.70	0.89 (0.83–0.94)	0.41	-0.06	256/0/0	N/A
XMBP	LA	BSP, random init	0.51	0.48	0.45	1.06 (0.95–1.14)	0.55	0.03	256/0/0	0.04/0.03/0.03
XMBP	RA	BSP, random init	0.68	0.72	0.62	0.93 (0.86–1.02)	0.45	-0.02	256/0/0	0.04/0.05/0.04
XMBP	W	BSP, random init	0.36	0.32	0.32	1.10 (1.05–1.21)	0.58	0.05	256/0/0	N/A
XMBP	B	BSP, random init	2.6	2.1	1.9	1.25 (1.12–1.43)	0.62	0.09	256/0/0	N/A
XMBP	LF	BSP, held-out	0.00	0.00	0.00	N/A	N/A	N/A	0/256/0	N/A
XMBP	RF	BSP, held-out	11.2	3.5	6.2	3.61 (2.86–4.36)	0.93	0.53	256/0/0	N/A
XMBP	LA	BSP, held-out	12.4	3.9	2.7	3.51 (3.00–4.03)	0.96	0.51	256/0/0	0.61/0.18/0.12
XMBP	RA	BSP, held-out	17.2	5.1	3.8	3.56 (2.90–4.60)	0.95	0.52	256/0/0	0.76/0.23/0.16
XMBP	W	BSP, held-out	7.3	2.5	1.8	2.92 (2.56–3.52)	0.94	0.46	256/0/0	N/A
XMBP	B	BSP, held-out	4.4	4.6	1.9	0.89 (0.82–1.13)	0.47	-0.03	256/0/0	N/A
XMBP	LF	U-Net, held-out	0.00	0.00	0.00	N/A	N/A	N/A	0/256/0	N/A
XMBP	RF	U-Net, held-out	15.5	3.2	1.4	2.19 (1.86–2.55)	0.81	0.34	243/13/5	N/A
XMBP	LA	U-Net, held-out	6.4	2.4	1.9	2.61 (2.36–2.87)	0.99	0.43	256/0/0	0.27/0.10/0.09
XMBP	RA	U-Net, held-out	7.0	3.1	2.7	2.30 (2.01–2.64)	0.92	0.37	256/0/0	0.35/0.13/0.12
XMBP	W	U-Net, held-out	3.7	1.6	1.2	2.42 (2.15–2.72)	0.91	0.38	256/0/0	N/A
XMBP	B	U-Net, held-out	6.1	2.8	2.0	2.59 (2.28–2.92)	0.91	0.39	256/0/0	N/A

Table 20: Size and speed of BSP, the official DP trunk, and the same U-Net trunk on pooled frozen DINOv2 (U-Net + DINO; Appendix D), measured on an RTX 5090 Laptop GPU with torch 2.9.1 and synthetic batches. Training is ms per step at batch 128; min/100k is minutes per 100k steps. Inference is one 100-step DDPM action chunk at batch 1. The U-Net trunk runs in bf16 and the kinematic trunk in fp32; DINOv2 runs under bf16 autocast in both. [t] Trunk only.

Model	total	trainable	ms/step	min/100k	chunk (ms)	fwd (ms)
Official DP (ResNet-18)	–	73.13M[t]	–	–	–	–
U-Net + DINO	95.38M	73.32M	94.0 ± 15.8	157	324.5 ± 5.2	3.17
BSP (ours)	35.04M	35.04M	140.7 ± 15.4	235	308.6 ± 0.5	2.86

the kernel switches to three-term Taylor branches and their derivatives,

$$\frac{\sin\varphi}{\varphi} = 1 - \tfrac{x}{6} + \tfrac{x^2}{120}, \qquad \frac{1-\cos\varphi}{\varphi^2} = \tfrac{1}{2} - \tfrac{x}{24} + \tfrac{x^2}{720}, \qquad \frac{\varphi-\sin\varphi}{\varphi^3} = \tfrac{1}{6} - \tfrac{x}{120} + \tfrac{x^2}{5040},$$

so the operator and its gradients stay smooth across joint types. The truncation error, about $x^3/5040 \approx 2\times10^{-16}$, lies far below fp32 precision. The kernel matches the eager PyTorch reference to 9.5×10^{-6} in the forward pass and to a relative error of 2.9×10^{-4} against central finite differences in the backward pass; with the linear part v of every twist set to zero it reproduces the Rodrigues rotation to 3.4×10^{-7}, within 2.9 units of fp32 machine epsilon, and repeated backward passes are bit-identical. Table 21 gives its timings: 8.5–12× faster than eager PyTorch. Over 8 screw layers and 100 DDPM steps, a served chunk spends about 29 ms in the kernel, against about 350 ms eager. The kernel is what lets the 308.6 ms chunk fit the 320 ms that a 16-step action chunk lasts at 50 Hz.

Table 21: Run time of the fused kernel at the policy's shapes (D=31, R=6, C_ℓ=16) against the eager implementation, batched `torch.matrix_exp` over the 4×4 twist generators plus an einsum contraction. RTX 5090 Laptop GPU, torch 2.9.1, CUDA-synchronized timing over 50 iterations after 10 warm-up iterations. The kernel is built with fast-math intrinsics by default; `SCREW_FAST_MATH=0` selects the IEEE build.

	batch 128 (train)		batch 1 (serve)	
	forward	fwd+bwd	forward	fwd+bwd
CUDA kernel (ms)	0.123	0.646	0.036	0.226
eager PyTorch (ms)	1.380	6.059	0.442	1.931
speedup	11×	9.4×	12×	≈8.5×

I.3 CUDA Graph Fast Path

On the evaluation seat, an RTX 6000 Ada at batch 1 through each policy's own serve handler (observation decode, transforms, sampling, de-normalisation and encoding; 20 warm-up and 100 timed calls; median per 16-step chunk), the 100-step DDPM chain of BSP takes 451 ms, against 394, 441 and 478 ms for Diffusion Policy, Patch Policy and the Rodrigues Network with the same sampler. A denoising iteration costs 4–5 ms at batch 1 for every diffusion arm whatever its size, so the step count sets the latency: the ten trailing iterations of Table 1 take 55.8 ms on the same seat, checkpoint and harness. These served numbers include the observation pipeline, unlike the synthetic-batch timing of Table 20. The fast path is a serving mode of the same network behind one flag, off in every evaluated cell. It computes the keys and values of the 256 visual tokens once per observation instead of once per iteration, runs the screw kernel in a batch-one tiling with static topology indices on the device, and replays the denoiser's forward pass as a captured CUDA graph; the encoder, the noise draw and the scheduler stay eager. On seven of the nine tasks, 110 held-out frames each, it returns action chunks bit-identical to the reference path, so it changes latency and not behaviour. Timed around `Policy.act` in the production environment and paired against the reference path on the same seat, the 100-step chain falls from 484.7 to 97.6 ms (4.97×) and the 10-step chain from 54.2 to 13.9 ms (3.89×); an RTX 5090 through the same entry point gives 101.4 and 14.1 ms. A 16-step chunk lasts 320 ms at 50 Hz; the served ten-step chain uses 13.9 ms of it.

Table 22 lists the served latency of every Table 1 arm on the same seat.

Table 22: Served latency of every Table 1 arm on one RTX 6000 Ada at batch 1 with no concurrent process, through each arm's own serve handler (observation decode, transforms, sampling, denormalisation and encoding): 20 warm-up and 100 timed calls, median and interquartile range in ms per returned action chunk; the last column divides by the steps executed per call. Latency is set by the sampler and the chunk length as much as by the network: the VLAs run their flow-matching steps after one backbone pass, and the diffusion policies run DDPM iterations that cost about 4 ms each at batch 1 whatever the network size. Parameters are the whole served model. Table 20 uses a different protocol (RTX 5090 Laptop, synthetic batches) and is not comparable in absolute terms. The Rodrigues Network row is the published ResNet-18 checkpoint, not the DINOv2-S variant of Table 1, whose latency we did not measure. † DINOv2 under bf16 autocast. ‡ 5.7B T5 text encoder in fp32 on the CPU, run once per episode. § Measured on a second RTX 6000 Ada with the same harness, not in the same job as the other rows. ¶ The fast path of Section I.3, timed around `Policy.act`; it returns the same actions as the reference path, and Figure 1 plots this row.

Arm	params	precision	sampler	steps/call	ms per chunk (IQR)	ms per step
Ψ_0	2.63B	bf16	flow, 10 steps	24	148.2 (146.4–151.3)	6.17
$\pi_{0.5}$ (VLM frozen)	3.62B	bf16	flow, 10 steps	16	77.7 (76.8–78.5)	4.86
GR00T N1.6	3.29B	bf16	flow, 4 steps	16	73.5 (70.9–77.0)§	4.60
DreamZero	22.92B	bf16‡	UniPC, 16 steps	48	4026.5 (3961.7–4078.9)	83.89
Diffusion Policy	84.3M	fp32	DDPM, 100 steps	16	394.0 (387.5–425.8)	24.62
Patch Policy	34.9M	fp32†	DDPM, 100 steps	16	441.4 (438.4–446.0)	27.59
Rodrigues Network	50.4M	fp32	DDPM, 100 steps	16	477.8 (472.1–486.9)	29.86
BSP (ours), reference path	35.0M	fp32†	DDPM, 10 steps	16	55.8 (54.3–57.1)	3.49
BSP (ours), fast path¶	35.0M	fp32†	DDPM, 10 steps	16	13.9 (13.6–14.3)	0.87