PAIRFORMER IS AN EFFECTIVE RELATIONAL EMBODIED LEARNER

Anonymous authors
Paper under double-blind review

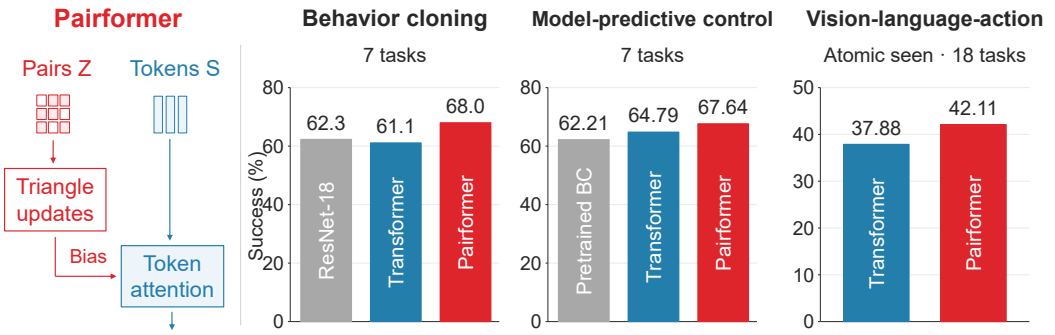


Figure 1: **Pairformer improves embodied learning.** Left: pair updates bias token attention; transitions and residual paths are omitted. Right: success rates in BC, MPC, and VLA Atomic seen (18 tasks), from Tables 3–5, using each setting's evaluation protocol. All axes start at zero; the VLA panel uses a separate scale.

ABSTRACT

Transformers thrive on large-scale data, yet robot experience remains costly and sparsely covers physical interactions. We investigate a **relational inductive bias** for embodied learning: maintaining and composing vector-valued representations of token pairs. We study **Pairformer** from AlphaFold 3, whose persistent pair states and triangle updates guide token attention while retaining a Transformer-compatible token interface. Diagnostic experiments show that Pairformer retains graph relations for subsequent queries and predicts contact-driven motion more accurately, with attention more closely aligned to collisions in a visual probe. We then test this computation in three embodied components: (i) observation encoders for diffusion policies, (ii) latent dynamics models for model-predictive control, and (iii) pretrained action predictors within $\pi_{0.5}$. On seven simulated manipulation tasks with 200 demonstrations per task, the encoder and dynamics adaptations improve mean success by 6.9 and 2.85 percentage points over their Transformer baselines. The action-predictor adaptation improves atomic-seen success by 4.23 points, with smaller gains on composite tasks. Together, these results support explicit relational computation as an effective architectural complement to data scaling and pretraining for embodied learning and control.

1 INTRODUCTION

Transformers underpin large-scale learning, including pretrained vision–language–action (VLA) models (Vaswani et al., 2017; Brohan et al., 2023; Kim et al., 2024; Physical Intelligence, 2025). Robot experience, however, remains costly and sparsely covers the interactions encountered during execution. Manipulation depends on relative configurations, sequences of contacts, and coordinated actions. These conditions motivate a complementary question to data scaling: *can an explicit relational inductive bias improve embodied learning with the data available?*

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.

Persistent pair representations give relational information its own evolving state (Battaglia et al., 2018; Bergen et al., 2021; Diao & Loynd, 2023). Standard self-attention aggregates tokens using weights computed from token features. Pair computation additionally maintains a vector for each ordered token pair, composes relations sharing a third token, and uses the refined features to guide token aggregation. Our hypothesis is that this organization helps retain and combine dependencies needed for subsequent decisions.

Pairformer offers a practical way to study this hypothesis within existing embodied models. Introduced in AlphaFold 3 (Abramson et al., 2024), it couples token and pair streams through triangle updates and pair-biased attention. Task-specific adapters initialize pairs while retaining the surrounding input and output interfaces. This compatibility permits backbone replacement in observation encoders and dynamics models, and pair-bias augmentation inside a pretrained action predictor.

Two diagnostic experiments probe whether pair computation retains relations for later queries and predicts their physical consequences. In directed reachability, all tested models solve queries known during encoding; only Pairformer retains discrimination when queries arrive afterward. In billiards, Pairformer shows closer attention–contact alignment in a collision probe, lower rollout error, and better strike selection. The shared finding is that persistent pair computation supports representations that remain useful for downstream decisions.

Robot experiments carry this question into **observation encoders**, **latent dynamics models**, and **pretrained action predictors**. The first two improve mean success over Transformer by 6.9 and 2.85 points on seven tasks with 200 demonstrations each. The $\pi_{0.5}$ adaptation gains 4.23 points on atomic-seen tasks, extending the study to pretrained policies.

Our contributions are threefold:

- We adapt established Pairformer computation to observation encoders, latent dynamics models, and pretrained action predictors while preserving their surrounding interfaces.
- We provide diagnostic evidence of reusable graph representations and contact-aligned attention, together with improved physical prediction and candidate-strike selection.
- We demonstrate improved embodied task success and characterize the adaptations through task-level comparisons, a triangle-removal dynamics ablation, and computational measurements.

2 Related Work

Relational inductive biases. Relational architectures organize computation around entities and their interactions (Battaglia et al., 2018; Bronstein et al., 2021). Interaction Networks and graph-network simulators learn object–relation dynamics, while neural relational inference estimates interaction graphs (Battaglia et al., 2016; Sanchez-Gonzalez et al., 2020; Kipf et al., 2018). SE(3)-Transformers and EGNN additionally impose geometric equivariance (Fuchs et al., 2020; Satorras et al., 2021). In robotics, RodriNet incorporates forward kinematics through a learnable Neural Rodrigues Operator (Zhang et al., 2025). Our focus is learned pair refinement at the existing observation, latent-state, and action-token interfaces of embodied models.

Pair representations and composition. Persistent pair features and third-token interactions appear in molecular and graph architectures. AlphaFold 2/3 use pair representations for structure prediction (Jumper et al., 2021; Abramson et al., 2024); Edge Transformer uses ordered edge states and triangular attention (Bergen et al., 2021). EGT, Relational Transformer, and TGT develop node/edge, relational, and triplet computation (Hussain et al., 2022; Diao & Loynd, 2023; Hussain et al., 2024). Graphormer and GraphGPS combine attention with structural information, while TokenGT tokenizes nodes and edges (Ying et al., 2021; Rampášek et al., 2022; Kim et al., 2022). An AF2-derived Pairformer also supports extremal graph search (Mehrabian et al., 2024). Our study brings AF3's established pair/single organization to embodied components with distinct token interfaces and learning objectives. Universal Transformers motivate our weight-tied recurrence control (Dehghani et al., 2019).

Observation encoding and action generation. Visuomotor policies provide distinct encoding and action-generation modules for adaptation. Diffusion Policy learns conditional action denoising

(Chi et al., 2023); RT-2, OpenVLA, and Octo broaden robot learning through pretraining and shared interfaces (Brohan et al., 2023; Kim et al., 2024; Octo Model Team et al., 2024). The π_0 family combines vision–language representations with flow-based action generation, and $\pi_{0.5}$ extends generalization through heterogeneous training (Black et al., 2024; Physical Intelligence, 2025). We retain the diffusion U-Net architecture when comparing observation encoders and preserve the pretrained single-stream architecture when augmenting the $\pi_{0.5}$ action predictor.

Predictive representations and planning. PETS uses dynamics ensembles for planning, Dreamer learns through latent imagination, and TD-MPC/TD-MPC2 combine latent prediction with terminal values (Chua et al., 2018; Hafner et al., 2019; Hansen et al., 2022; 2024). I-JEPA learns predictive embeddings; LeJEPA combines embedding invariance and SIGReg (Assran et al., 2023; Balestriero & LeCun, 2025). DINO-WM plans through pretrained patch features, while S4D couples structured scene prediction to inverse dynamics (Zhou et al., 2024; Li et al., 2026). Our comparison uses a shared BC-supervised embedding and environment-reward targets. The frozen behavior prior, critics, and controller are common to all dynamics variants, testing Transformer and Pairformer within the same planning system.

3 Preliminaries: Pairformer

Pairformer is the representation backbone introduced in AlphaFold 3 (Abramson et al., 2024). For N input tokens, it maintains single features $S \in \mathbb{R}^{N \times C_s}$ and pair features $Z \in \mathbb{R}^{N \times N \times C_z}$: s_i describes token i, while z_{ij} describes the ordered pair (i, j). Here C_s and C_z are channel widths. Task-specific adapters initialize these features from token inputs and available pair information. Each z_{ij} is a vector-valued relation retained and updated across blocks.

The relational bias lies in maintaining pair features and allowing relations sharing a token to interact. Training determines their content through the task objective. The following operators are inherited from AF3; Section 5 describes their task-specific interfaces.

Pair update. Each block first applies outgoing and incoming triangle multiplication, then starting- and ending-node triangle attention, followed by a pair transition. Multiplication combines two projected pair features sharing a third token k and sums over k; triangle attention selects pair messages using the third edge as an attention bias. The transition is a gated feed-forward network applied to each pair. Residual updates carry the refined pair state across blocks.

Single update. The updated pair features produce a bias for single-token attention. For head h,

$$A_{ij}^h = \mathrm{softmax}_j \left(\frac{(q_i^h)^\top k_j^h}{\sqrt{d_h}} + [W_b \, \mathrm{LN}(Z)_{ij}]_h \right), \tag{1}$$

where q, k, v are projections of normalized singles and d_h is the head width. Gated aggregation of values v_j^h updates S, followed by a single transition. Repeating these blocks yields (S, Z) for the task readout. Algorithm 1 gives the complete residual sequence; Appendices A–B provide operator details and analysis.

4 Probing Relational Computation

The usefulness of relational computation depends on what its representations retain and predict. Reachability probes whether one graph encoding can answer subsequent queries. Billiards tests prediction of contact-mediated motion, with attention diagnostics exposing information routing around collisions. Together, these probes motivate evaluating pair computation in the embodied components of Section 5.

4.1 Reachability: reusing relations across queries

The reachability test isolates representation reuse on randomly relabeled 24-node directed trees. **Single** exposes the query before encoding; **Multiple** supplies it only at readout, requiring one encoding to support different queries. A common graph embedding and readout surround a four-block

Transformer, a four-block Pairformer, or a 71-step weight-tied Looped Transformer. Models train on depths 1–8; independent tests include unseen depth 12 and average ten seeds. Appendix C gives query construction and optimization details.

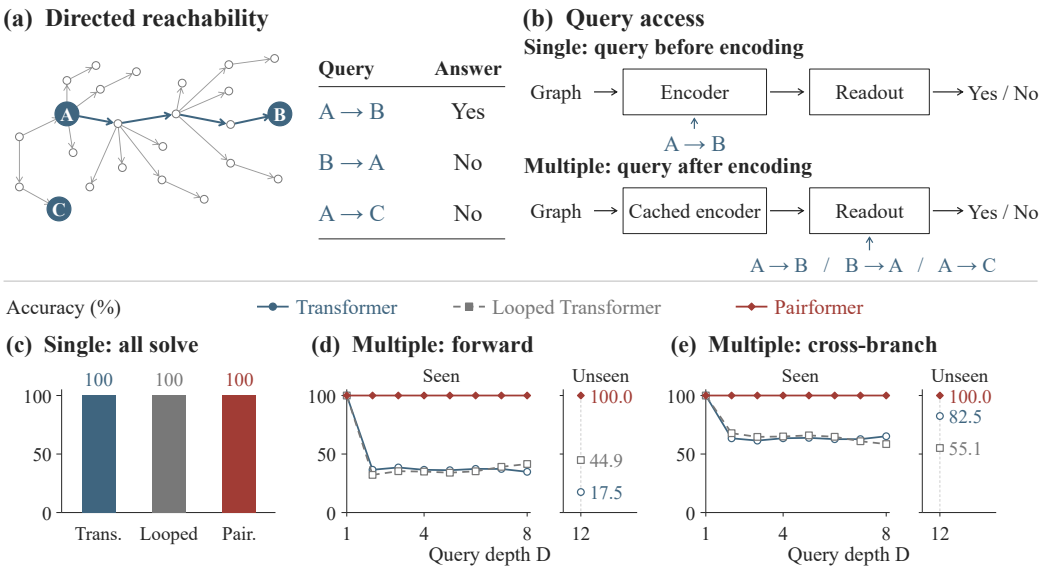


Figure 2: **Pairformer's advantage emerges when an encoding must be reused.** (a) Directed graph and query examples. (b) Single provides the query before encoding; Multiple supplies it after encoding. (c) All models solve Single. (d,e) Pairformer preserves both reachable and cross-branch discrimination in Multiple, including unseen depth 12. Curves average ten seeds; reverse-query accuracy is 100% throughout.

Table 1: Directed reachability in Multiple (%): mean accuracy across all ten training seeds. Forward measures correct acceptance of reachable queries; cross-branch measures correct rejection of unreachable queries. Reverse-query accuracy is 100% for every model and seed and is reported separately. All models also score 100% on every query type in Single, both IID and at $D12$.

	IID $D2$–$D8$		Unseen $D12$	
Model	Forward ↑	Cross-branch ↑	Forward ↑	Cross-branch ↑
Transformer	36.74	63.26	17.53	82.47
Looped Transformer	36.06	63.94	44.85	55.15
Pairformer	**100.00**	**100.00**	**100.00**	**100.00**

Query reuse separates the models (Table 1). All solve Single, but only Pairformer retains 100% forward and cross-branch accuracy in Multiple, including depth 12. The two component scores average to 50% for each token baseline, revealing lost discrimination under reuse; 71 recurrent steps do not restore it. The advantage thus concerns reusable relational encoding under this protocol. Appendix B.2 gives an idealized path-composition mechanism consistent with the result.

4.2 Billiards: Predicting Contact Dynamics

BilliardsRelay tests whether relational computation predicts physical consequences. An impulse drives a relay among six balls; a world model predicts 64 steps to select a strike toward a goal. Both models use the same state inputs, decoder, and schedule with 300,000 trajectories at contact depths $D1$–$D3$. Independent testing uses 2,000 layouts, including unseen $D4$, with 16 candidates each. Transformer and Pairformer have 3.79M and 4.32M parameters (Appendix D).

Pairformer's attention tracks contact more closely in the collision diagnostic (Figure 3). Across all 16 candidates of the displayed layout, attention correlates with next-step contact onset at $r = 0.747$,

versus $r = 0.531$ for Transformer. Aligned snapshots and temporal curves expose the response around contact. This one-layout measurement illustrates physically relevant information routing; aggregate prediction performance is evaluated separately below.

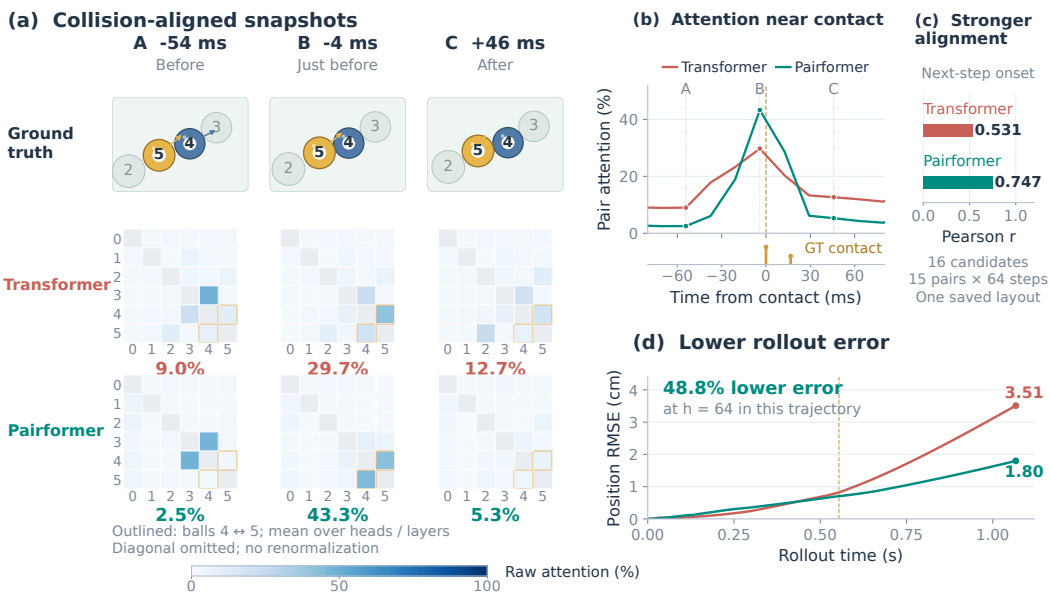


Figure 3: **Closer contact alignment accompanies lower prediction error.** (a,b) States, attention, and contact onsets at matched times A–C. (c) Attention–contact Pearson correlation over all 16 candidates of one layout. (d) Open-loop prediction error. Attention uses observed states; the seed-2 trajectory is selected post hoc. Appendix D.5 specifies measurement and selection.

Table 2: BilliardsRelay primary $D2$–$D3$ evaluation with 300,000 training trajectories. Position RMSE is in centimeters; selection success is a percentage. Both models predict 64 steps open loop. Best results are bolded.

	Position RMSE ↓			Selection success ↑
Model	$h = 16$	$h = 32$	$h = 64$	16 candidates
Transformer	0.270	0.890	2.960	79.27
Pairformer	**0.180**	**0.510**	**1.380**	**85.45**

Pairformer improves both rollout accuracy and strike selection (Table 2). On primary $D2$–$D3$ tests, 64-step RMSE falls from 2.960 to 1.380 cm (53.4%), while selection success rises from 79.27% to 85.45% (+6.18 points). Figure 4 illustrates a collision relay. Reachability demonstrates reuse of encoded relations; billiards shows that improved predictions support better action selection. We next examine these capabilities within robot policies and planners. Appendix D.4 reports the 30,000-trajectory comparison.

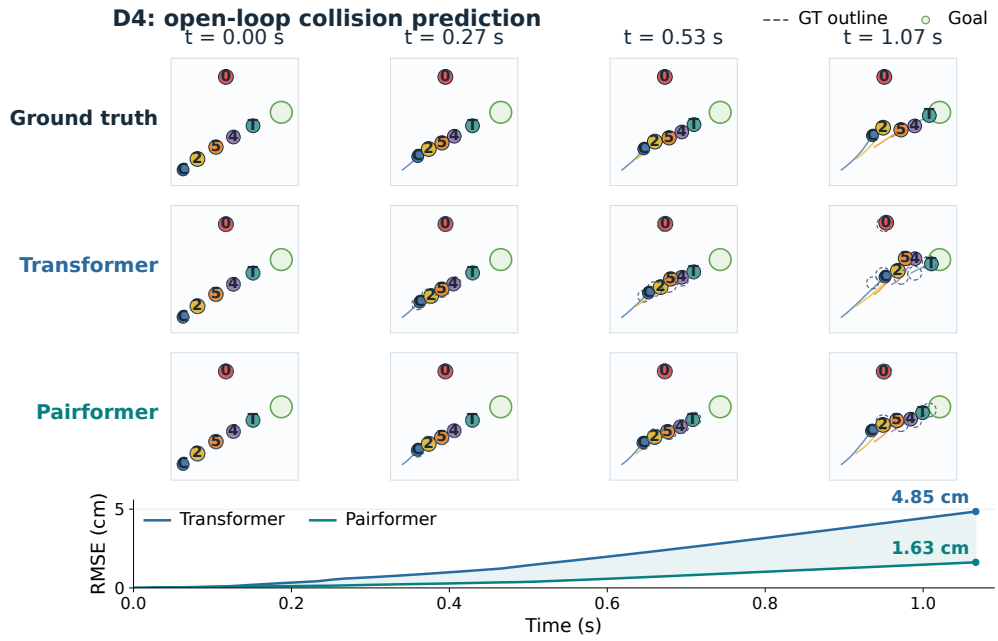


Figure 4: **Lower open-loop error through a collision relay.** Ground truth and original model rollouts for seed 2, layout D4-0000018, candidate 1. Shading marks the error difference. This example has the largest relative improvement among 32 archived visualizations and was selected post hoc.

5 Adapting Pairformer to Embodied Models

We insert pair computation where embodied models encode observations, predict transitions, or generate actions. Tokens represent visual patches and robot links, latent states, or action positions. Observation encoders and action predictors learn within policies; dynamics models predict consequences for planning with a fixed behavior policy. Each adaptation preserves the surrounding interfaces.

5.1 Observation encoders for diffusion policies

The observation encoder constructs the condition for a diffusion policy. Four RGB views and proprioception over two frames produce 152 visual-patch and robot-link tokens of width 256. Four Transformer or Pairformer blocks process them; Pairformer initializes 32-channel ordered pairs from single features and 14-dimensional pair geometry. Pooling visual and robot tokens within each frame and concatenating the projected frame representations yields a 512-dimensional condition. The policies use the same tokenizer, readout, and diffusion U-Net architectures (Chi et al., 2023), with each policy trained separately.

Action-denoising supervision jointly trains the encoder and policy. Given observation o, noisy action chunk a_t, diffusion step t, and sampled noise ϵ, both variants minimize

$$\mathcal{L}_{\mathrm{BC}} = \mathbb{E}\big[\|\epsilon - \epsilon_\psi(a_t, t; E_\theta(o))\|_2^2\big]. \tag{2}$$

Here E_θ is the encoder and ϵ_ψ the conditional U-Net. Pair features learn through the action objective. At inference, 100 DDPM denoising steps generate 32 eight-dimensional actions, of which the first eight are executed. Appendix E specifies encoder differences and training.

5.2 Latent dynamics models for MPC

The dynamics model predicts consequences of actions proposed by a common pretrained Transformer RGB-DP. We first adapt its visual stem using BC denoising, LeJEPA invariance/SIGReg reg-

ularization (Balestriero & LeCun, 2025), and preservation of the original policy's denoising outputs; all other BC weights remain frozen. This couples representation learning to demonstrated actions while maintaining compatibility with the behavior prior. The resulting representation supports both action proposals and predictive learning (Figure 5).

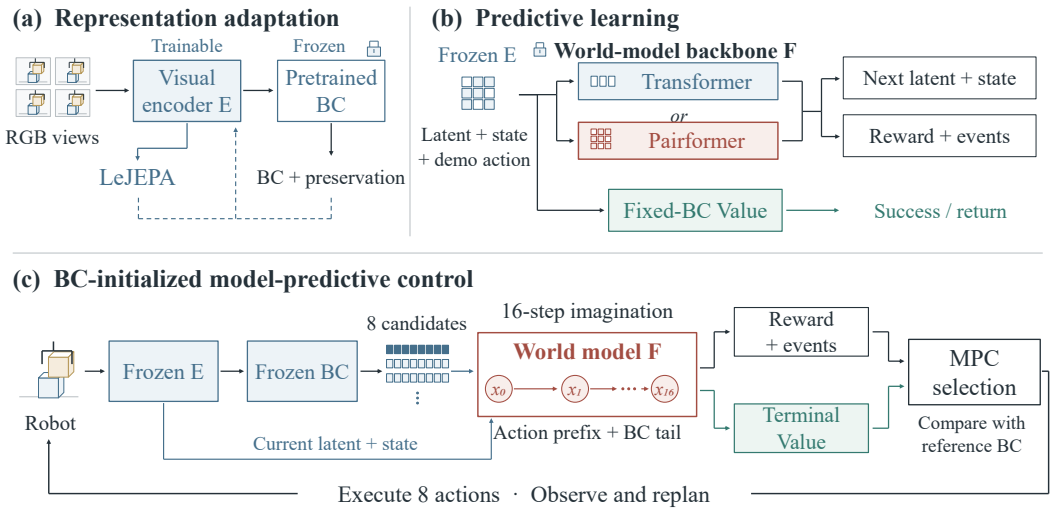


Figure 5: **BC-supervised representation learning and relational dynamics for MPC.** (a) BC denoising, preservation, and LeJEPA losses adapt one visual encoder through frozen BC weights; dashed arrows indicate supervision. (b) Frozen representations support dynamics learning and shared fixed-BC Value estimation. Transformer and Pairformer use matched next-state, reward, and event head architectures, fitted separately. (c) BC proposes eight action prefixes. Ensembles predict 16 steps with BC continuations, combining predicted environment rewards and events with terminal success/return Value. MPC compares candidates with the reference BC proposal, executes eight actions, and replans. Each x_h contains latent and proprioceptive history.

With the stem and BC frozen, two-frame histories of 64 visual latent positions and proprioception form 65 tokens. An action embedding conditions four Transformer or Pairformer blocks; the latter maintains a $65 \times 65 \times 32$ pair tensor. The same head architectures predict the next latent grid, proprioception, environment reward, and continuation/success/failure probabilities. Training combines 16-step autoregressive prediction with BC-output consistency, encouraging predicted features to remain usable by the frozen policy. Three independently fitted models form each dynamics ensemble.

Terminal values extend evaluation beyond the imagined horizon. Three shared critics estimate future success probability and environment return under frozen BC, conditioned on the remaining task budget. Action-chunk heads fit Bellman targets from recorded transitions; state-value heads learn average target estimates over BC-sampled actions. The same critics serve all dynamics variants. Reward labels come from synchronized demonstration replay; learned heads supply rewards during imagination (Appendix F).

MPC selects among eight BC-sampled action prefixes while retaining the reference BC sample as candidate zero. Each prefix contains eight actions and is scored over 16 imagined steps, with two common-noise BC continuations and three model/Value members. Let $\widehat{S}_c^m$ combine predicted success events with survival-weighted terminal success Value for candidate c and member m. The controller computes

$$\Delta_c^m = \widehat{S}_c^m - \widehat{S}_0^m, \qquad G_c = \mathrm{mean}_m\, \Delta_c^m - \mathrm{std}_m\, \Delta_c^m. \tag{3}$$

The controller selects the largest positive finite G_c, breaking ties by uncertainty-penalized predicted return, or retains candidate zero if none qualifies. The reference estimates improvement over BC; the ensemble penalty discounts inconsistent predictions. All variants execute eight actions and replan with this procedure.

5.3 PRETRAINED ACTION PREDICTORS

The pretrained action predictor is the flow-based action expert within $\pi_{0.5}$ (Physical Intelligence, 2025). We initialize a $50 \times 50 \times 64$ ordered pair tensor from noisy action tokens and relative positions. Each of the 18 expert layers applies the AF3 pair core and projects its output into an eight-head bias on attention between action positions. The vision–language prefix, joint attention normalization, pretrained Gemma single blocks, and action decoder retain their existing interfaces.

Zero initialization of the pair-bias output preserves the pretrained policy at the start of adaptation. Subsequent flow-matching training learns how pair refinement routes information among action positions, while the original single stream supplies vision–language conditioning. The adapter adds 2.681M parameters, or 0.080% of the full model, and uses the baseline's flow objective and sampling procedure. Appendix G specifies initialization, conditioning, and inference settings.

6 EMBODIED EXPERIMENTS

Embodied evaluations measure task success for the three component adaptations. Each comparison fixes its data and evaluation protocol.

6.1 OBSERVATION ENCODERS WITH LIMITED DEMONSTRATIONS

We compare ResNet-18, Transformer, and Pairformer encoders on seven ManiSkill tasks (Tao et al., 2024), using 200 demonstrations and 50 validation trajectories per task. ResNet-18 trains from scratch (He et al., 2016); Transformer and Pairformer use the same tokenizer and denoiser architectures. Token policies train for 300,000 updates with batch 64 and validation-loss selection of EMA checkpoints (Appendix E).

Table 3: BC success (%) with 200 training demonstrations and 50 shared validation trajectories. All models use a diffusion U-Net; Transformer and Pairformer use the same tokenizer architecture. Average weights tasks equally; row-best results are bolded.

	200 training demos		
Task	ResNet-18	Transformer	Pairformer
PushCube	99.0	98.5	**100.0**
PickCube	72.0	92.5	**98.5**
StackCube	45.5	40.5	**52.0**
PullCubeTool	**88.0**	77.5	86.5
StackPyramid	15.0	9.5	**15.5**
PegInsertionSide	35.5	29.0	**39.0**
PokeCube	81.0	80.0	**84.5**
Average	62.3	61.1	**68.0**

Pairformer achieves 68.0% mean success, versus 61.1% for Transformer and 62.3% for ResNet-18 (Table 3). Gains over Transformer span all seven tasks, largest on StackCube (+11.5 points), PegInsertionSide (+10.0), and PullCubeTool (+9.0). Pairformer leads six tasks; ResNet-18 remains stronger on PullCubeTool. StackPyramid improves by 6.0 points but remains difficult at 15.5%. The comparison evaluates the complete encoder, including geometric inputs and gated updates.

6.2 LATENT DYNAMICS MODELS FOR MPC

The dynamics comparison uses 200 demonstrations and 50 development trajectories on the same seven tasks. World models train with 16-step prediction, batch 16, and development-based plateau stopping. Selected checkpoints receive 200 independent test episodes per seed; Table 4 reports three-seed means. All variants share the Transformer BC prior, adapted embedding, Value ensemble, candidates, and controller (Appendix F).

Table 4: BC-to-MPC success (%, three-seed mean): 200 training demos and 200 independent test episodes per seed. All MPC variants share Transformer BC, the embedding, Value, and controller. Average weights tasks equally; row-best results are bolded, including ties.

	Direct policy	Transformer BC + LeJEPA + WM + MPC		
Task	Pretrained Transformer BC	Transformer WM	Pairformer WM w/o triangle	Pairformer WM
PushCube	99.5	99.5	**100.0**	**100.0**
PickCube	98.5	98.5	**99.5**	99.0
StackCube	37.0	40.5	**45.5**	44.5
PullCubeTool	82.5	83.5	81.0	**88.0**
StackPyramid	14.5	17.0	14.5	**18.0**
PegInsertionSide	25.5	32.0	35.0	**38.0**
PokeCube	78.0	82.5	81.5	**86.0**
Average	62.21	64.79	65.29	**67.64**

Both controllers improve over direct BC (62.21%): Transformer MPC reaches 64.79% and Pairformer MPC 67.64%, a 5.43-point gain over BC and 2.85 over Transformer dynamics. Backbone gains span all seven tasks, largest on PegInsertionSide (+6.0), PullCubeTool (+4.5), and StackCube (+4.0).

Triangle refinement improves aggregate performance, with task-dependent benefits. Removing it lowers mean success by 2.35 points to 65.29%, losing 7.0 points on PullCubeTool but leading on PickCube and StackCube. The ablation retains pair storage, transitions, and pair-biased attention while reducing capacity (Appendix F.5). Costs appear in Appendix F.6.

6.3 Pretrained action predictors

Both VLA variants train on RoboCasa365 Human300: 30,000 demonstrations across 300 tasks (Nasiriany et al., 2026), using three seeds, batch 64, and matched initialization and optimization. Evaluation covers 50 tasks with 50 episodes per task and seed, sharing scene splits, horizons, and per-task random streams (Appendix G).

Table 5: RoboCasa365 success (%). Column-best results are bolded.

Action expert Task count	Composite seen 16	Composite unseen 16	Atomic seen 18
Transformer ($\pi_{0.5}$)	10.00	1.00	37.88
Action Pairformer ($\pi_{0.5}$)	**10.25**	**1.25**	**42.11**

Action-pair computation improves all three groups (Table 5), most clearly on atomic-seen tasks: 37.88% to 42.11% (+4.23 points). Each composite group gains 0.25 points and remains difficult. Benefits to pretrained action generation thus vary across task groups.

7 Conclusion

Pairformer connects reusable relational representations to embodied task success. Diagnostic probes demonstrate graph-query reuse and improved physical prediction; robot experiments show gains in observation encoders, dynamics models, and action predictors. These findings support relational computation as a complement to data scaling and pretraining for embodied learning and control.

Limitations and future work. Simulation, coupled capacity/operator effects, and expensive pair computation limit this study. MPC cycles take tens of seconds (Appendix F.6); sparse updates and computational reuse could accelerate control. Broader contact diagnostics, parameter-matched controls, and physical-robot evaluation are important next steps.

AI USE STATEMENT

Generative AI coding assistants, including OpenAI Codex, assisted with research-idea development, experimental and architectural design, mathematical derivations, implementation and debugging, experiment orchestration, data-processing and evaluation code, interpretation of experimental records, manuscript writing, and plotting and diagram-generation code. Figures use vector or presentation primitives and plotted simulation data. Computational checks cover finite directed-graph identities, permutation behavior, attention identities, and prediction-to-return bounds; these checks accompany the written proofs. AI assistance is a tool in the research workflow, not a source of measured experimental outcomes. Responsibility for the scientific claims, data provenance, and submitted content rests with the human authors.

ETHICS STATEMENT

Robot experiments use simulation. Transfer to physical systems entails separate safety validation, including robustness to perception error, contact uncertainty, and out-of-distribution states. Human-demonstration datasets remain subject to their original licenses and usage conditions. The evaluation protocol separates development and independent test data, preserves seed-level comparisons, and identifies the selection procedure for qualitative examples. Private credentials and identifying infrastructure information are excluded from the manuscript.

REPRODUCIBILITY STATEMENT

Algorithm 1 and Appendix A specify the core operators and update order. Appendix B gives the mathematical proofs. Appendices C–G document input and output shapes, data-generation procedures, split rules, random seeds, optimizer schedules, checkpoint selection, and evaluation. The robot controller is specified in Algorithm 2, including candidate generation, shared-noise continuation, terminal handling, and replanning. These details distinguish task adapters and evaluation protocols throughout the study.

REFERENCES

Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, et al. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630:493–500, 2024. doi: 10.1038/s41586-024-07487-w. URL https://www.nature.com/articles/s41586-024-07487-w.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. *arXiv preprint arXiv:2301.08243*, 2023. URL https://arxiv.org/abs/2301.08243.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025. URL https://arxiv.org/abs/2511.08544.

Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*, 2016. URL https://arxiv.org/abs/1612.00222.

Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018. URL https://arxiv.org/abs/1806.01261.

Leon Bergen, Timothy J. O'Donnell, and Dzmitry Bahdanau. Systematic generalization with edge transformers. In *Advances in Neural Information Processing Systems*, 2021. URL https://arxiv.org/abs/2112.00578.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024. URL https://arxiv.org/abs/2410.24164.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023. URL https://arxiv.org/abs/2307.15818.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021. URL https://arxiv.org/abs/2104.13478.

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

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *arXiv preprint arXiv:1805.12114*, 2018. URL https://arxiv.org/abs/1805.12114.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1807.03819.

Cameron Diao and Ricky Loynd. Relational attention: Generalizing transformers for graph-structured tasks. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.05062.

Fabian B. Fuchs, Daniel E. Worrall, Volker Fischer, and Max Welling. SE(3)-transformers: 3D roto-translation equivariant attention networks. *arXiv preprint arXiv:2006.10503*, 2020. URL https://arxiv.org/abs/2006.10503.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019. URL https://arxiv.org/abs/1912.01603.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022. URL https://arxiv.org/abs/2203.04955.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.16828.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016. URL https://arxiv.org/abs/1512.03385.

Md Shamim Hussain, Mohammed J. Zaki, and Dharmashankar Subramanian. Global self-attention as a replacement for graph convolution. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022. URL https://arxiv.org/abs/2108.03348.

Md Shamim Hussain, Mohammed J. Zaki, and Dharmashankar Subramanian. Triplet interaction improves graph transformers: Accurate molecular graph learning with triplet graph transformers. In *Proceedings of the International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20768–20792, 2024. URL https://proceedings.mlr.press/v235/hussain24a.html.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596:583–589, 2021. doi: 10.1038/s41586-021-03819-2. URL https://www.nature.com/articles/s41586-021-03819-2.

Jinwoo Kim, Tien Dat Nguyen, Seonwoo Min, Sungjun Cho, Moontae Lee, Honglak Lee, and Seunghoon Hong. Pure transformers are powerful graph learners. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2207.02505.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. URL https://arxiv.org/abs/2406.09246.

Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, 2018. URL https://arxiv.org/abs/1802.04687.

Zhiyi Li, Peilin Wu, Xiaoshen Han, Ruojin Cai, and Yilun Du. Structured 4D latent predictive model for robot planning. *arXiv preprint arXiv:2607.01166*, 2026. URL https://arxiv.org/abs/2607.01166.

Abbas Mehrabian, Ankit Anand, Hyunjik Kim, Nicolas Sonnerat, Matej Balog, Gheorghe Comanici, Tudor Berariu, Andrew Lee, Anian Ruoss, et al. Finding increasingly large extremal graphs with AlphaZero and tabu search. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2024. URL https://arxiv.org/abs/2311.03583.

Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2603.04356.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024. URL https://arxiv.org/abs/2405.12213.

Physical Intelligence. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. URL https://arxiv.org/abs/2504.16054.

Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *arXiv preprint arXiv:2205.12454*, 2022. URL https://arxiv.org/abs/2205.12454.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. *arXiv preprint arXiv:2002.09405*, 2020. URL https://arxiv.org/abs/2002.09405.

Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. *arXiv preprint arXiv:2102.09844*, 2021. URL https://arxiv.org/abs/2102.09844.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, et al. ManiSkill3: GPU parallelized robotics simulation and rendering for generalizable embodied AI. *arXiv preprint arXiv:2410.00425*, 2024. URL https://arxiv.org/abs/2410.00425.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1706.03762.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform bad for graph representation? *arXiv preprint arXiv:2106.05234*, 2021. URL https://arxiv.org/abs/2106.05234.

Jialiang Zhang, Haoran Geng, Yang You, Congyue Deng, Pieter Abbeel, Jitendra Malik, and Leonidas Guibas. Rodrigues network for learning robot actions. *arXiv preprint arXiv:2506.02618*, 2025. URL https://arxiv.org/abs/2506.02618v1.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024. URL https://arxiv.org/abs/2411.04983.

A PAIRFORMER OPERATORS AND IMPLEMENTATION

The implementations use the established AF3 pair/single computation with task-specific adapters (Abramson et al., 2024). Given singles $S^{(0)} \in \mathbb{R}^{N\times C_s}$, pair inputs E_{ij}, and token validity m_i, initialize

$$z_{ij}^{(0)} = M_{ij}\big(W_{\text{left}} s_i^{(0)} + W_{\text{right}} s_j^{(0)} + E_{ij}\big), \quad M_{ij} = m_i m_j. \tag{4}$$

Endpoint order is preserved. The padding mask M controls valid tokens, and observed adjacency or geometry enters through E. Pair state persists through the blocks of one invocation; each new observation, dynamics step, or flow invocation initializes its own state.

Triangle multiplication. Each outgoing/incoming module forms independently parameterized gated projections of the current normalized pair state:

$$\bar{z}_{ij} = \mathrm{LN}(z_{ij}), \tag{5}$$

$$a_{ij} = M_{ij}(W_a \bar{z}_{ij}) \odot \mathrm{sigmoid}(W_{ga}\bar{z}_{ij}), \qquad b_{ij} = M_{ij}(W_c \bar{z}_{ij}) \odot \mathrm{sigmoid}(W_{gc}\bar{z}_{ij}), \tag{6}$$

$$t_{ij}^{\text{out}} = \sum_k a_{ik} \odot b_{jk}, \qquad t_{ij}^{\text{in}} = \sum_k a_{kj} \odot b_{ki}, \tag{7}$$

$$\Delta z_{ij} = \mathrm{sigmoid}(W_{go}\bar{z}_{ij}) \odot W_o \,\mathrm{LN}_{\text{mid}}(t_{ij}). \tag{8}$$

The incoming contraction follows `kjc,kic` $\rightarrow$ `ijc`. Residual addition updates the pair state before the next operator recomputes its projections.

Triangle attention. Starting and ending attention operate on $X_{ij} = \bar{z}_{ij}$ and $X_{ij} = \bar{z}_{ji}$, respectively. The head bias $\beta_{jk}^h = [W_\beta \bar{z}_{jk}]_h$ is constructed before this orientation change. For each anchored row,

$$\alpha_{i,j,k}^h = \mathrm{softmax}_k\left(\frac{(q_{ij}^h)^\top k_{ik}^h}{\sqrt{d_h}} + \beta_{jk}^h + \mu_{ik}\right), \tag{9}$$

$$u_{ij} = W_o\left[\mathrm{sigmoid}(W_g X_{ij}) \odot \mathrm{Concat}_h \sum_k \alpha_{i,j,k}^h v_{ik}^h\right]. \tag{10}$$

The validity mask follows the row orientation. Ending attention transposes its output back to the original pair order.

Single attention and transitions. Updated pairs condition single-token aggregation through a per-head bias:

$$B_{ij}^h = [W_b \,\mathrm{LN}(z_{ij})]_h, \qquad A_{ij}^h = \mathrm{softmax}_j\left(\frac{(q_i^h)^\top k_j^h}{\sqrt{d_h}} + B_{ij}^h + \mu_j\right). \tag{11}$$

Gated aggregation yields $\Delta s_i = W_o[\mathrm{sigmoid}(W_g \,\mathrm{LN}(s_i)) \odot \mathrm{Concat}_h \sum_j A_{ij}^h v_j^h]$. The query projection includes the AF3 query bias. Single and pair transitions use independent pre-normalized SwiGLU networks with hidden widths $4C_s$ and $4C_z$.

Task-specific alignment. Reachability, billiards, BC, and world-model backbones use the complete residual order in Algorithm 1. The action-predictor adapter combines its five pair operators with the retained pretrained Gemma single block. Dynamics implementations use interleaved projection channels; graph/BC implementations use a contiguous split. The corresponding weight permutation maps both layouts to the same projected tensors. Tokenization, pair initialization, and output heads are adapted to each task.

B PROPERTIES OF RELATIONAL COMPUTATION

The following properties connect persistent pair storage, third-index composition, and pair-biased attention. They characterize computational operations underlying the diagnostic probes; empirical performance is measured in Section 4.

Algorithm 1 Persistent directed-pair computation (AF3-inspired core)

Require: $S^{(0)} \in \mathbb{R}^{N\times C_s}$, $E \in \mathbb{R}^{N\times N\times C_z}$, validity m, depth L
$S \leftarrow S^{(0)}$; $M_{ij} \leftarrow m_i m_j$; initialize Z by Eq. (4)
for $\ell = 1, \dots, L$ **do** ▷ Independent parameters unless explicitly tied
  $Z \leftarrow Z + \mathrm{TriangleMultiplyOut}_\ell(Z, M)$
  $Z \leftarrow Z + \mathrm{TriangleMultiplyIn}_\ell(Z, M)$
  $Z \leftarrow Z + \mathrm{TriangleAttentionStart}_\ell(Z, M)$
  $Z \leftarrow Z + \mathrm{TriangleAttentionEnd}_\ell(Z, M)$
  $Z \leftarrow Z + \mathrm{PairTransition}_\ell(Z)$
  $B \leftarrow W_b^{(\ell)} \mathrm{LN}_b^{(\ell)}(Z)$
  $S \leftarrow S + \mathrm{GatedSelfAttention}_\ell(S, m, B)$
  $S \leftarrow S + \mathrm{SingleTransition}_\ell(S)$
end for
return (S, Z)

B.1 Pair storage and bilinear composition

Pairformer carries an evolving tensor through its blocks:

$$Z^{(\ell+1)} = G_\ell(Z^{(\ell)}; M), \qquad S^{(\ell+1)} = F_\ell(S^{(\ell)}, B_\ell(Z^{(\ell+1)}); m). \tag{12}$$

Each ordered relation retains C_z channels for subsequent updates and query readout. The product primitive composes two projected relations before summing over a third index.

Proposition 1 (Factorized bilinear composition)**.** *For $x, y \in \mathbb{R}^d$ and coefficients $\mathcal{T}_{opq} = \sum_{r=1}^{\chi} W_{or} A_{rp} B_{rq}$,*

$$\phi_o(x, y) = \sum_{p,q} \mathcal{T}_{opq} x_p y_q, \qquad \sum_k \phi(x_{ik}, y_{jk}) = W \sum_k (Ax_{ik}) \odot (By_{jk}). \tag{13}$$

Every bilinear map admits this factorization with $\chi \le d^2$.

Proof. Expanding the channel product gives the coefficient expression. Conversely, index intermediate channels by (p, q), select x_p and y_q with rows of A, B, and set $W_{o,(p,q)} = \mathcal{T}_{opq}$. Smaller widths give a factorized parameterization. □

The implemented triangle module further applies learned gates and normalization to this primitive.

B.2 Directed path composition

A Boolean abstraction explains how third-index composition can answer different reachability queries simultaneously. If channels expose both R_{ij} and R_{ji}, choosing $a_{ik} = R_{ik}$ and $b_{jk} = R_{kj}$ makes the raw outgoing contraction equal $(R^2)_{ij}$. For adjacency A, consider

$$R^{(0)} = A \vee I, \qquad R^{(\ell+1)} = R^{(\ell)} \vee \mathbf{1}[R^{(\ell)} R^{(\ell)} > 0]. \tag{14}$$

After ℓ updates, this recurrence represents paths of length at most 2^ℓ. The proof is inductive: multiplying two entries concatenates paths of length at most 2^ℓ; every path of length at most $2^{\ell+1}$ can be split into two such paths. Four idealized updates cover depth 12. This abstraction assumes Boolean thresholds and both relation orientations; learned AF3 blocks use continuous channels, gates, and normalization.

B.3 Pair bias and attention routing

For valid keys j, k, cancellation of the softmax denominator in Eq. (11) gives

$$\log \frac{A_{ij}}{A_{ik}} = \frac{q_i^\top (k_j - k_k)}{\sqrt{d_h}} + B_{ij} - B_{ik}. \tag{15}$$

Holding the token logits fixed, a change δ in the pair-bias difference multiplies these attention odds by e^δ. Pair refinement therefore has an explicit route to influence information aggregation, which motivates the attention–contact diagnostic.

B.4 PERMUTATION EQUIVARIANCE

Shared projections and jointly permuted inputs make the core equivariant to token reindexing. For permutation π, outgoing multiplication obeys

$$\sum_k a_{\pi(i),\pi(k)} \odot b_{\pi(j),\pi(k)} = \sum_r a_{\pi(i),r} \odot b_{\pi(j),r}. \tag{16}$$

The same change of variables applies to incoming multiplication. Softmax commutes with permutation of the key axis when biases and masks follow it; normalization, gates, and residual addition act identically at each index. Induction gives outputs $(PS, PZP^\top)$ for jointly permuted tokens, pair inputs, and masks. Position and geometry metadata move with their tokens.

C DIRECTED REACHABILITY PROTOCOL

C.1 DATA GENERATION AND SHORTCUTS

The directional v3.1 protocol uses directed out-trees with random recursive side branches. Each graph has 24 vertices. A forward source–descendant query is paired with its exact reversal and an unreachable query to a different branch. The graph remains in one weakly connected component. The cross-branch target and its predecessor are matched to the positive construction in local in/out degree, reducing endpoint-degree shortcuts. Random renumbering removes index-order cues. Labels are checked independently using directed breadth-first search and Boolean transitive closure.

Each family contributes three query types with labels $(1, 0, 0)$. Equal type counts therefore do *not* mean equal positive and negative label counts. An always-unreachable classifier achieves $2/3$ type-macro accuracy, so raw accuracy without direction-specific metrics is insufficient. The primary metric first averages sensitivity/specificity across the three types within a depth and then averages the specified depths.

Table 1 reports forward sensitivity and cross-branch specificity separately for IID depths 2–8 and unseen depth 12. Within each training seed, IID values average the seven depths; the table then gives the mean across all ten seeds. Reverse specificity and all Single query types are 100% for every model and seed. This reporting decomposition leaves the original primary endpoint, training, and validation checkpoint selection unchanged. The baseline component scores complement one another, yielding 50% when averaged over forward and cross-branch queries, or 66.67% when reverse queries are included, at depths 2–8 and 12. Hence component variation does not imply variation in the original aggregate metric.

Table 6: Fixed reachability data scale. Each family contributes three queries.

Split	Graph families	Queries	Depths
Train	4,096	12,288	1–8
Validation	1,024	3,072	1–8
Independent IID test	2,048	6,144	1–8
Independent depth test	2,048	6,144	12

Topology-hash buckets use the ratio 50/15/20/15 with split salt 27183; generator seeds are 96101–96104. Topology identity checks exclude graphs appearing in earlier development and confirmation banks. Training, validation, and independent evaluation use their respective fixed banks.

C.2 INPUT EMBEDDING, BACKBONES, AND OUTPUT

The common GraphEmbed-1 stem performs one-hop graph embedding before the tested backbone. The adjacency enters as an observed directed relation, not as an attention-validity mask. Single adds source/target roles before graph encoding; Multiple encodes the graph without a marked query and supplies its ordered endpoints only to the readout. Single and Multiple use the same number of training families and queries.

All backbones use $C_s = 64$, $C_z = 32$, and four single-attention heads. Transformer has four untied blocks with a static pair bias. Pairformer has four complete AF3-inspired blocks. Looped Transformer repeats one tied static-pair-bias block 71 times. The common query-readout architecture concatenates $[s_{\text{source}}, s_{\text{target}}, \text{mean}_i\, s_i, z_{\text{source,target}}]$, yielding 224 channels, followed by the archived normalization/MLP with hidden width 64 and a scalar logit. Ordered concatenation preserves the query direction. Exact parameter counts are 326,625, 509,665, and 116,193 for Transformer, Pairformer, and Looped Transformer.

Each run has 4,000 optimizer updates with batch 48, stratified as eight depths by three query types by two examples. AdamW uses learning rate 10^{-3} and weight decay 10^{-4}; gradient clipping is 1. Training uses FP32 with TF32 disabled. Checkpoints maximize validation directional macro accuracy with BCE as the tie-breaker. The ten seeds are 151, 157, 163, 167, 173, 179, 181, 191, 193, and 197.

C.3 Statistical comparisons and directional controls

The primary contrasts compare Pairformer against the two token baselines on Multiple IID $D2$–$D8$. Single measures query-conditioned solvability, while reverse and branch specificity diagnose directional and topological shortcuts. Seed-level contrasts are paired. The statistical protocol specifies an exact paired sign-flip test with Holm adjustment for the two primary comparisons and 10,000-resample seed/family hierarchical bootstrap intervals. The directional decomposition distinguishes learning forward composition from simply predicting every query as unreachable.

D BilliardsRelay Protocol and Visual Evidence

D.1 Simulator, state, and candidate construction

Each ball token is $[x, y, v_x, v_y, \omega_z, r, \log m, \mu, e] \in \mathbb{R}^9$, where r, m, μ, e denote radius, mass, friction, and restitution. Motion is planar: height and roll/pitch are constrained, gravity is zero, and there is no ground drag. There are no walls; the nominal one-meter table bounds reject out-of-bounds cases. Penetration above 5 mm is rejected. The simulator uses four 240-Hz substeps per 60-Hz observation, bounce threshold 0.01 m/s, contact offset 0.001 m, rest offset zero, and solver iterations 8/4.

Actions are initial cue-ball changes in planar velocity, with speeds 0.6–1.4 m/s; the equivalent impulse is $m\Delta v$. For each retained layout, the 16-candidate bank contains 1–8 successful candidates and at least eight failures, including at least four near-angle hard negatives within eight degrees. Relay depth refers to the temporal contact sequence, not static connected components. Goal radius ranges from 0.045 to 0.065 m. A model receives physical states and candidate actions; candidate scoring uses the externally specified goal.

D.2 Model and optimization

Training has 50,000 layout families at each of $D1, D2, D3$, with a fixed successful/failed pair per layout. Validation has 500 layouts each at $D2, D3$; main test has 500 each at $D1, D2, D3, D4$. The full 16-candidate bank is retained for validation/test selection. Every rollout has 65 frames spanning 64 transitions.

Both models use six blocks, $C_s = 192$, $C_z = 48$, six single heads, and four pair-attention heads in the Pairformer. Both receive the same 12-dimensional ordered pair inputs and use matching encoder/decoder architectures, trained separately. The decoder predicts five dynamic-state increments while retaining static physical attributes. Pair initialization is recomputed from the current state at each physical rollout step and persists through the six blocks within that step. Transformer has 3,792,797 parameters and Pairformer has 4,315,229.

Training seeds 0/1/2 each run 100,000 updates with global batch 256, AdamW learning rate 3×10^{-4}, weight decay 10^{-4}, gradient clipping 1, and Huber threshold 0.1. Computation is FP32 without TF32. The main configuration optimizes one-step prediction; multistep training loss is disabled. Checkpoint selection maximizes validation $D2$–$D3$ candidate success, with horizon-16 RMSE on

a fixed 5,000-trajectory validation subset as the tie-breaker. Test trajectories are never used for checkpoint selection.

D.3 METRICS AND VISUALIZATION

For a candidate trajectory c, horizon-h position error is

$$e_h(c) = 100\sqrt{\frac{1}{12}\sum_{i=1}^{6}\left\|\widehat{p}_{h,i}^{(c)} - p_{h,i}^{(c)}\right\|_2^2} \quad \text{cm}, \tag{17}$$

where coordinates are in meters. The reported RMSE is the average of these per-trajectory errors, not a square root taken after pooling all trajectories. The selected candidate minimizes $\min_{1\le h\le 64}\|\widehat{p}_{h,\text{target}}^{(c)} - g\|_2$. Selection success is the simulator's ground-truth outcome for that candidate. Thus improved selection does not by itself prove more accurate trajectories.

D.4 ADDITIONAL TRAINING-DATA BUDGET

The additional data-budget comparison keeps the model, 100,000-update optimization schedule, global batch 256, and evaluation layouts fixed while reducing training trajectories from 300,000 to 30,000. Table 7 retains both budgets for comparison. The 30k values are means over three training seeds. Pairformer improves its 64-step RMSE from 1.627 to 1.380 cm and selection success from 83.10% to 85.45% at the larger budget. Transformer has a different response, with higher RMSE and slightly lower selection success. The evidence therefore describes a model-dependent data-budget effect.

Table 7: BilliardsRelay primary $D2$–$D3$ results at two data budgets. The 30k entries are three-seed means. Optimization budgets and evaluation layouts are shared across data budgets. RMSE is in centimeters; success is a percentage.

		Position RMSE ↓			Selection success ↑
Train trajectories	Model	$h = 16$	$h = 32$	$h = 64$	16 candidates
30k	Transformer	0.235	0.699	2.051	79.50
	Pairformer	**0.179**	**0.540**	**1.627**	**83.10**
300k	Transformer	0.270	0.890	2.960	79.27
	Pairformer	**0.180**	**0.510**	**1.380**	**85.45**

Figure 4 shows the archived open-loop prediction example in the main text. The figure caption records its selection and output provenance. Uncertainty is estimated by a paired hierarchical bootstrap over training seeds and layouts, preserving all candidate trajectories within each sampled layout. The main depth comparison uses shared physical-parameter ranges, while $D4$ changes the contact-depth distribution.

D.5 CONTACT-ALIGNED ATTENTION DIAGNOSTIC

Figure 3 uses the archived seed-2 models and layout D4-0000018. Panels (a), (b), and (d) show candidate 2, selected post hoc; the separate trajectory visualization in Figure 4 uses candidate 1. The highlighted collision is between balls 4 and 5. The snapshots at frames 30, 33, and 36 align with labels A–C in the temporal panel, and arrows show ground-truth velocity.

We extract single-token attention, including the learned pair bias in Pairformer, and average over all six blocks and six heads. Both models receive the same observed states for attention extraction; their prediction-error curves instead use open-loop rollouts without ground-truth corrections. Heatmaps share a common scale, with no renormalization after diagonal masking. For each unordered ball pair, the temporal statistic averages its two directed attention weights.

The correlation panel pools all 16 saved candidates, 64 prediction steps, and 15 unordered ball pairs: 15,360 observations within one fixed layout. Attention at state x_t is compared with binary contact onset during $(t/60, (t+1)/60]$, aggregated from four simulator substeps. Pearson correlations are

0.531 for Transformer and 0.747 for Pairformer. The observations share a layout and have temporal dependence. The reported correlation characterizes this within-layout diagnostic; broader sampling is left to future work.

E DIFFUSION-POLICY PROTOCOL

E.1 COMPLETE REFERENCE ARCHITECTURE

The reference RGB input is $[B, 2, 4, 3, 128, 128]$: batch, observation time, camera, color channel, height, and width. Proprioception is $[B, 2, 25]$, consisting of nine joint positions, nine joint velocities, and a seven-dimensional TCP pose. Both token models receive the same proprioception and camera/kinematic information.

Both token policies use a five-convolution patch-stem architecture producing a 4×4 grid per camera: 16 patches per view and 64 visual tokens per frame. The robot MLP and fixed forward kinematics produce 12 link tokens per frame. Across two frames there are $2(64 + 12) = 152$ single tokens, each of width 256. The Transformer backbone has four untied pre-LayerNorm blocks, eight-head attention without a pair bias, and a $4\times$ GELU MLP. The Pairformer replacement initializes 32-channel pairs using single projections and encoded 14-dimensional pair geometry. Pair geometry uses RGB camera-ray proxies.

Both readouts independently max-pool visual and robot tokens per frame, concatenate the two 256-dimensional vectors, apply normalization and a projection to 256, and concatenate both frames to produce a 512-dimensional global condition. The conditional 1-D U-Net has channels 256/512/1024 and diffusion-timestep embedding width 128. It predicts noise for an action tensor $[B, 32, 8]$. Sampling uses 100 DDPM denoising steps and executes the first eight actions. Total parameter counts are 74,138,022 for Transformer RGB-DP and 75,642,754 for Pairformer RGB-DP. RGB Diffusion Policy (R3) uses a ResNet-18 initialized from scratch and has 116,279,398 parameters in its archived full policy configuration.

E.2 TRAINING AND EVALUATION CONTROLS

Both policies train their encoder and diffusion denoiser with the action-denoising objective in Eq. (2). The BC comparison uses 200 training demonstrations per task and 50 separate held-out validation trajectories. The reference implementation uses the following training and evaluation settings. The formal token-policy schedule uses 300,000 updates, FP32, global batch 64, three training seeds, and EMA checkpoint selection by validation diffusion loss. Evaluation uses 100 episodes per training seed with reset seeds 40000–40099 and success-once accounting. Cameras, action interpretation, observation channels, and horizon must match within each task and comparison.

The seven-task comparison includes PushCube-v1, PickCube-v1, StackCube-v1, PullCubeTool-v1, clean-data StackPyramid-v1, PegInsertionSide-v1, and PokeCube-v1. PickCube's goal-conditioned adapter adds the goal coordinates to the reference proprioceptive/context input. Other task differences, including horizons and dataset versions, are fixed within each model comparison. The 300-step and 512-step tool-use evaluations are distinct horizon settings; they are not pooled. The reference 100-episode evaluation and the hard-task 200-episode evaluation retain their own task-specific reset banks.

Table 3 reports aggregate success rates for the seven listed tasks at the 200-training-demo condition. The Average row is the unweighted mean of the seven displayed task rates, rounded to one decimal place. These aggregates are reported without a per-seed standard deviation or a significance claim. In the reference protocol, training trajectory IDs are fixed across architectures; demonstrations outside this subset are excluded from optimization and normalization fitting. Checkpoint selection uses only the same 50 held-out validation trajectories for all models. Any historically substituted training seed is recorded explicitly rather than treated as a paired seed.

Transformer and Pairformer use the same tokenizer, readout, and diffusion-decoder architectures, with weights trained separately for each policy. The ResNet-18 policy provides the broader system reference. Parameter counts specify the capacity differences among these configurations.

F BC-SUPERVISED LEJEPA AND DUAL-VALUE MPC

F.1 STAGE A: ONE SHARED EMBEDDING

Let E_θ be the shared visual patch stem and π_ψ the pretrained Transformer scene encoder and diffusion action decoder. Freeze ψ and adapt θ using two same-frame photometric views $\tau_1(o), \tau_2(o)$:

$$z^{(v)} = E_\theta(\tau_v(o)), \tag{18}$$

$$\mathcal{L}_A = \mathcal{L}_{\mathrm{BC}}\big(\pi_\psi(E_\theta(o), p), a^{\mathrm{demo}}\big) + \lambda_J \mathcal{L}_{\mathrm{LeJEPA}} + \lambda_P \mathcal{L}_{\mathrm{preserve}}, \tag{19}$$

$$\mathcal{L}_{\mathrm{LeJEPA}} = (1-\lambda_S)\mathcal{L}_{\mathrm{inv}}(z^{(1)}, z^{(2)}) + \lambda_S \mathcal{L}_{\mathrm{SIGReg}}, \qquad \lambda_S = 0.02. \tag{20}$$

$\mathcal{L}_{\mathrm{BC}}$ is the original diffusion-noise supervision, with gradient passing through the frozen decoder to the stem. The preservation loss matches the frozen original policy's denoising output at the same noisy actions, timestep, and augmented image; the original visual stem is a training-only teacher. The coefficients are $\lambda_J = 0.01$ and $\lambda_P = 1$. SIGReg is the LeJEPA distributional regularizer (Balestriero & LeCun, 2025); its empirical samples are trajectories in the batch at a fixed spatial token. Spatial positions and history frames are not flattened into additional independent samples. The regularizer operates on the current-frame view.

After Stage A, freeze both the shared stem and BC. An adapted-BC development control isolates representation changes. The direct-BC comparison in Table 4 measures the full representation-and-planning pipeline.

F.2 STAGE B: PREDICTIVE WORLD MODEL

The world model receives latent history $[B, 2, 64, 256]$, proprioceptive history $[B, 2, d_p]$, and an action $[B, 8]$. Here $d_p = 25$ in the reference configuration; the recovered PickCube policy additionally includes its three goal coordinates, giving $d_p = 28$. The two frames are concatenated per spatial position and projected to 256 channels. A projected proprioceptive-history token yields 65 tokens total. Position/type and action embeddings condition four Transformer or Pairformer blocks. Pairformer uses $65 \times 65 \times 32$ pairs. Four output heads predict the residual next-frame latent grid, next proprioception, scalar transition reward, and three logits for continuation, success, and failure. Three independently trained world models form each predictive ensemble.

At rollout time h, denote the valid-transition mask by v_h. The training objective is

$$\ell_{\mathrm{pred}}(h) = \ell_z(\hat{z}_{h+1}, z_{h+1}) + \ell_p(\hat{p}_{h+1}, p_{h+1}) + \ell_r(\hat{r}_h, r_h) + \ell_d(\hat{d}_h, d_h), \tag{21}$$

$$\mathcal{L}_B = \frac{\sum_{h=0}^{15} v_h\big(\ell_{\mathrm{pred}}(h) + 0.1\ell_{\text{BC-cons}}(h)\big)}{\sum_h v_h}, \tag{22}$$

$$\ell_{\text{BC-cons}}(h) = \big\|\epsilon_\psi(x_\tau, \tau; \hat{z}_{h+1}, \hat{p}_{h+1}) - \mathrm{sg}[\epsilon_\psi(x_\tau, \tau; z_{h+1}, p_{h+1})]\big\|_2^2. \tag{23}$$

The two BC-consistency terms share the same expert noisy action chunk x_τ and diffusion time τ. Although BC weights are frozen, gradients through its predicted-input evaluation supervise the world model. The latent, proprioceptive, and reward terms use mean squared error; the event term ℓ_d uses three-class cross-entropy. Valid-transition loss is averaged within each trajectory window and then across windows in the batch.

Original-environment reward means the actual `env.step` transition label: `normalized_dense` for the supported tasks and official `sparse` reward for StackPyramid, whose environment does not implement that dense mode. A learned head predicts the reward during imagination; MPC never queries future simulator states. The terminal transition and its reward are valid targets; post-terminal padding is masked. Time-limit truncation is handled through the remaining task budget rather than relabeled as physical failure.

The dataset uses 200 training and 50 development trajectories, with RGB, proprioception, reward, and event labels generated by synchronized replay. Stage B uses 16-step autoregressive prediction, global batch 16, FP32 without TF32, AdamW learning rate 10^{-4} and weight decay 10^{-6}, gradient clipping 10, and EMA. The development-based stopping rule requires at least 100,000 updates, learning rate 10^{-6}, and less than 0.5% improvement over 50,000 updates; the optimization budget is capped at 600,000 updates. PegInsertionSide uses a rebuilt content-disjoint split and retrained BC, rather than the earlier split containing duplicate trajectories across training and validation.

F.3 FIXED-BC DUAL VALUE

Three shared critics estimate future success probability $V_s^\pi(x, b)$ and environment return $V_r^\pi(x, b)$ under the frozen, action-clipped BC, with remaining task budget b. Each critic also has action-chunk heads Q_s^π and Q_r^π. A real chunk of $k \leq 8$ transitions supplies Bellman targets: success already reached gives target one; failure or exhausted budget gives zero continuation; otherwise the success target bootstraps V_s^π at the next history. The reward target sums observed environment rewards and bootstraps $\gamma^k V_r^\pi$ only for a continuing, non-exhausted chunk. We use $\gamma = 1$.

The Q heads fit these real-transition targets. The V heads distill target-Q values averaged over four freshly sampled BC chunks, not a maximization over actions or an expert Monte Carlo return interpreted as BC value. Success losses are binary cross-entropy and return losses are mean squared error after scaling by the task horizon. The two world-model variants share the same three critics and adapted BC. No new training interactions are collected.

F.4 CLOSED-LOOP BC-TAIL CANDIDATE EVALUATION

Algorithm 2 BC-initialized MPC with event prediction and dual Value

Require: Frozen E, π; three model/Value pairs (F_m, V_m); $H = 16$, $K = 8$, remaining budget b

for each real control cycle **do**
  Encode the latest two observations and proprioceptive states as history x_0
  Sample eight BC candidate prefixes; preserve the reference sample as candidate $c = 0$
  for each candidate c, member m, and common-noise continuation draw $u \in \{1, 2\}$ **do**
    $x \leftarrow x_0$; $S \leftarrow 0$; $R \leftarrow 0$; $q \leftarrow 1$; $L \leftarrow \min(H, b)$
    for $h = 0, \ldots, L - 1$ **do**
      if $h = 0$ **then**
        Use candidate prefix c
      else if $h \bmod K = 0$ **then**
        Query frozen BC on predicted history using shared tail noise
      end if
      $(\hat{x}', \hat{r}, \hat{\boldsymbol{e}}) \leftarrow F_m(x, a_h)$; $\boldsymbol{p} \leftarrow \mathrm{softmax}(\hat{\boldsymbol{e}})$
      $S \leftarrow S + q\, p_{\mathrm{success}}$; $R \leftarrow R + \gamma^h q \hat{r}$; $q \leftarrow q\, p_{\mathrm{continue}}$; $x \leftarrow \hat{x}'$
    end for
    if $L < b$ **then**
      $S \leftarrow S + q V_{s,m}^\pi(x, b - L)$; $R \leftarrow R + \gamma^L q V_{r,m}^\pi(x, b - L)$
    end if
    Store $S_{c,m,u} = S$ and $R_{c,m,u} = R$
  end for
  Average continuation draws; compute success gain G_c using Eq. (3)
  Select the finite alternative with largest positive G_c; break ties by conservative return
  If no alternative qualifies, choose $c = 0$
  Execute the chosen first K actions in the real environment; observe and replan
end for

The discount is one. Candidate alternatives use random streams separate from the reference BC; continuation draws share noise across candidates and ensemble members. Survival weight q softly marginalizes continuation events, rather than thresholding one binary termination logit. Terminal-transition reward is included before updating q. At the known task budget, no terminal Value is added. Each success score in Eq. (3) first averages the two draws. Both the paired-success-gain penalty and reward tie-break penalty use the population standard deviation across three members, with coefficient one. The gain threshold and tie tolerance are zero. Nonfinite alternatives are excluded; an invalid reference or absence of an eligible alternative returns the reference BC proposal. The ensemble penalties serve as uncertainty-aware ranking heuristics.

The current development evaluation uses the same reset seeds 10000–10049 across four arms: original Transformer BC, adapted BC without planning, Transformer WM + MPC, and Pairformer WM + MPC. Each training seed has 50 episodes per arm, with the task-specific episode limit (200 for StackCube), official success-once, first-success stopping, and action-bound clipping. All planning arms, including the no-triangle control in the final comparison, retain the Transformer BC behav-

ior prior. The direct pretrained Transformer BC reference compares complete systems, while the adapted-BC development control separates representation change from adding planning. Pairformer RGB-DP is evaluated as a policy backbone in Table 3, not as the initialization for these MPC arms.

For the independent fixed-protocol evaluation, we lock the validation-selected checkpoints after training plateau and retain the planner parameters above. Each training seed is evaluated on reset seeds 120000–120199, disjoint from the development bank; the environment budget, observation processing, success criterion, and clipping rule are unchanged. This fixed-protocol evaluation proceeds without a completed event/Value calibration gate. All training seeds and outcomes are retained, irrespective of improvement over BC. Table 4 reports this independent protocol, not the development scores; entries require complete three-seed coverage.

F.5 Removing triangle updates

The ablation removes the outgoing and incoming triangle multiplications and the starting- and ending-node triangle attentions from every world-model block. With PT denoting the pair transition, its pair recursion is

$$Z^{(\ell+1)} = Z^{(\ell)} + \mathrm{PT}_\ell(Z^{(\ell)}). \tag{24}$$

Pair initialization, padding masks, pair-to-single attention bias, single attention and transition, input/action adapters, and all four prediction heads are retained. Thus the control still stores ordered pairs and processes each pair across layers, but removes pair-to-pair composition through a third token.

The ablated world model is trained from initialization using the same demonstrations, training seeds, optimization protocol, loss weights, EMA, and development-based selection rule as the complete model. Each task/seed uses the same frozen Transformer BC, Stage-A embedding, and Value ensemble for all world-model variants. Candidate prefixes, common-noise continuations, horizon, execution chunk, reward definition, and reset bank are fixed. Removed modules reduce parameter count and computation; widths are not increased to compensate. Accuracy and cost are reported together rather than interpreting the ablation as parameter-matched.

F.6 Computational cost protocol

Table 8 separates training cost from prediction and decision latency. World-model training GPU-hours sum accelerator wall time across the three fitted WM members, including scheduled validation and checkpointing, but excluding queueing, replay/cache construction, BC pretraining, shared LeJEPA adaptation, and shared critic fitting. Shared critic fitting is accounted for separately. Resource records retain accelerator model, device count, updates, stopping point, and all continuation segments across migrations; lower wall time need not mean fewer GPU-hours. Standalone Transformer BC and Pairformer BC have no world-model fitting or prediction, hence the dashes in those columns. Pairformer BC is included only as a policy-computation reference in this cost table; all three MPC variants retain the same Transformer behavior prior.

Inference uses one NVIDIA RTX 4090, FP32 with TF32 disabled, and one environment. A fixed bank uses reset seeds 80000–80199, outside the development evaluation: each history repeats the initial observation twice, as at the first real control cycle. Saved inputs and hashes enforce matching across methods and training seeds; no policy actions or training labels are collected. Fifty warm-up calls precede 200 measured calls. Inputs are GPU-resident before timing and explicit CUDA synchronization brackets each call. Loading, rendering, simulation, transfer, compilation, and file I/O are excluded. Cycle timing includes encoding, diffusion sampling, model rollout, terminal Value, and selection as applicable. A timing process yields if another compute process starts on its GPU.

One world-model step measures one ensemble member's action-conditioned transition, including adapters and all four heads, without BC, Value, or planning; per-member means are averaged within a task. A full MPC cycle uses all three model/Value members, eight candidate prefixes, two common-noise continuation draws, horizon 16, and execution chunk eight. Direct BC times its actual single proposal. All BC calls retain 100 DDPM steps. Mean and P95 cycle times are computed within each task and then macro-averaged over the seven tasks. P95 is the macro-average of task-level percentiles. The table reports aggregate costs. Partial task coverage is archived separately

and never substituted for this seven-task aggregate. Timing runs independently of success evaluation on otherwise idle GPUs, using the same sealed checkpoints and planning settings.

Table 8: Control cost on one RTX 4090 with FP32 and one environment. Transformer BC and Pairformer BC are timed as standalone policies; only Transformer BC supplies the behavior prior for the MPC variants. Training cost covers world-model fitting; cycle latency covers encoding through action selection, including terminal Value for MPC and excluding simulation. MPC uses eight candidates, two continuation draws, a 16-step horizon, and three model/Value members. Entries are seven-task macro averages. Appendix F.6 defines timing and aggregation.

Method	WM training (GPU h)↓	WM step (ms)↓	Mean cycle (ms)↓	P95 cycle (ms)↓
Transformer BC	—	—	561.90	564.00
Pairformer BC	—	—	572.16	573.52
Transformer WM + MPC	122.0	0.870	32266.77	32376.65
Pairformer WM + MPC	371.6	4.907	35443.79	35609.17
w/o triangle	141.3	1.840	33026.40	33166.36

Pairformer increases mean cycle latency from 32.27 s to 35.44 s (9.8%) and world-model training cost from 122.0 to 371.6 GPU-hours. The measured cycles include shared BC sampling and terminal Value evaluation as well as the dynamics computations.

G Action-Pairformer Adaptation of $\pi_{0.5}$

The formal baseline is the original $\pi_{0.5}$ action expert. Its SigLIP/PaliGemma visual-language prefix and discretized-state conditioning remain unchanged. Three RGB views are resized to 224×224. The action expert has 18 layers and a 1,024-channel single representation, retaining its original adaptive normalization, rotary position encoding, grouped-query attention, MLP, and output projection. Its internal action tensor is $[B, 50, 32]$; the environment action uses the applicable 12 dimensions after the original adapter, not all padded channels.

The new pair tensor is $[B, 50, 50, 64]$. Initialization combines left/right action-token projections and relative action-time features once per flow-network evaluation. This state persists across all 18 layers, and is freshly initialized at the next flow evaluation. At each layer, the pair core supplies eight query-head biases. With a joint attention logit matrix L, the modification is

$$L' = L + \begin{bmatrix} 0 & 0 \\ 0 & B^{\text{action}} \end{bmatrix}, \tag{25}$$

where the block partition is prefix/action queries and keys. Original masking prevents prefix queries from seeing the action suffix. Action queries retain their original joint normalization across prefix and action keys, and the prefix KV cache is independent of pair-state evolution.

The pair-bias output projection is initialized at zero while the original pretrained parameters are loaded unchanged. With matched precision, masking, and random inputs, the initial policy output is preserved. Pair refinement affects the policy as its output projection learns. Parameter counts are 3,353,433,872 for the baseline and 3,356,114,896 for Action Pairformer, an addition of 2,681,024 parameters, approximately 0.080%.

Training uses the fixed Human300 selection with filter seed zero, 30,000 demonstrations, three training seeds 42/43/44, batch 64, BF16 computation with FP32 master weights, and the same base initialization. The archived schedule contains 75,001 optimizer updates with the final checkpoint directory indexed 75,000. Inference uses ten flow steps and executes five actions before the next prediction. Both variants use the same task labels and flow objective.

The official 50-task benchmark contains 18 atomic-seen, 16 composite-seen, and 16 composite-unseen tasks and uses the `pretrain` scene split. Each task has 50 episodes per training seed and uses the official registry's horizon. NumPy seed 7, Python seed 7, and policy JAX seed 0 are reset once per task and run continuously through its episodes. This supports independent task sharding

with matched random streams across models. This evaluation is separate from the earlier serial cross-task-RNG protocol.

The per-task evaluation record associates the task and scene split with the training seed, checkpoint, evaluation protocol, episode count, horizon, and episode outcomes. Overall success is a task macro-average, weighting the three groups by 18/50, 16/50, and 16/50.

G.1 What conditions the action-pair branch?

Let X_ν be the noisy action chunk at a flow invocation, C the cached visual–language/state prefix, and τ_ν the flow time. In the implemented $\pi_{0.5}$ adapter, initial suffix singles are the projected noisy actions. The pair initializer receives these singles and relative action indices Δ_{ij}, not C. Flow time enters the retained single-stream adaptive normalization, not the pair initializer or pair core. With masks fixed,

$$Z_\nu^{(0)} = I(W_a X_\nu, \Delta), \qquad Z_\nu^{(\ell+1)} = G_\ell(Z_\nu^{(\ell)}), \tag{26}$$

$$B_\nu^{(\ell)} = b_\ell(Z_\nu^{(\ell+1)}), \qquad S_\nu^{(\ell+1)} = F_\ell(S_\nu^{(\ell)}, C, B_\nu^{(\ell)}, \tau_\nu). \tag{27}$$

The action-pair branch is conditioned on the current noisy action tokens and relative action indices. The single stream receives the visual–language/state prefix and flow time, so the policy remains observation-conditioned throughout sampling. Later flow invocations receive action chunks updated by that conditioned policy.

H Model Sizes and Comparison Scope

The model counts below specify the capacity of each comparison. Pair state requires $O(N^2 C_z)$ storage, with triangle computation additionally depending on a third token index. The action-predictor adapter adds a small pair branch to a large pretrained expert; the other variants replace their token-processing backbone.

Table 9: Parameter counts of the reference configurations. World-model counts exclude the frozen BC and shared critics.

Comparison	Transformer	Pairformer
Reachability	326,625	509,665
BilliardsRelay	3,792,797	4,315,229
RGB-DP	74,138,022	75,642,754
MPC world model	3,594,013	5,098,439
$\pi_{0.5}$ full policy	3,353,433,872	3,356,114,896

The experiments evaluate complete, setting-specific architectural adaptations. Graph and billiards Transformer controls receive static pair information. The BC replacement additionally changes single-stream gating, transition activation, and geometric pair inputs. The pretrained action predictor retains its single stream and adds an action-pair branch. Removing triangle modules reduces world-model capacity. Parameter-matched and pair-local controls would further separate these factors; the reported robot comparisons concern the implemented configurations.