GROUPACT: FUNCTIONAL-GROUP COORDINATION FOR ROBOT ACTION GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large-scale robot models learn diverse motion patterns from data collected across embodiments. However, robot embodiments, particularly mobile manipulation and multi-arm platforms, comprise heterogeneous joints with distinct functional roles and motion characteristics. These differences shape the temporal and spatial structure of joint trajectories. Incorporating this structure as a physical prior can guide the learning of coordinated actions across heterogeneous control variables. In this work, we assign each joint a group label based on its functional role and motion characteristics and organize joints into action groups. Using the articulated structure of each embodiment, we further construct a joint-group graph that explicitly represents the relationships between joints and their functional groups. We then introduce **GroupACT**, a flow-matching policy that incorporates these structural priors into its action expert through group-mediated block and group-query tokens. The group-mediated block is designed for integration into Transformer-based action experts. Experiments on mobile manipulation in simulation and bimanual manipulation on a physical robot show that GroupACT outperforms the evaluated baselines on multiple tasks. These results support the effectiveness of functional-group priors for learning heterogeneous actions across the evaluated embodiments and tasks.

1 INTRODUCTION

Robots with mobile bases or multiple arms comprise subsystems with distinct functional roles and motion characteristics. For example, a mobile base moves the robot through the environment, while the two arms may perform complementary actions to complete a manipulation task. Differences in the physical properties and functions of these subsystems give rise to heterogeneous control signals and motion patterns.

Large robot datasets and pretrained vision language models (VLM) support policy learning for such heterogeneous actions (Open X-Embodiment Collaboration, 2023; Ghosh et al., 2024; Kim et al., 2025; Black et al., 2025b). Mobile ALOHA learns complex mobile manipulation behaviors by jointly predicting bimanual joint targets and base velocities from demonstrations (Fu et al., 2025). BiGym provides mobile bimanual tasks with action spaces that combine arm, gripper, and base or lower-body control (Chernyadev et al., 2025). HoMMI addresses whole-body manipulation involving the arms, torso, head, and mobile base, using a whole-body controller to translate hand-eye trajectories into motions under embodiment-specific constraints (Xu et al., 2026). Many vision language action (VLA) models similarly accommodate heterogeneous actions through unified action representations and shared action-generation architectures (Su et al., 2026; Ng et al., 2025). However, learning coordinated behaviors within these architectures often relies on extensive robot demonstrations and large-scale pretraining, both of which require substantial resources. This motivates incorporating structural priors that explicitly reflect the physical prior of robots.

Prior work incorporates robot structure into policy learning as an inductive prior for handling embodiment heterogeneity. Graph-based approaches organize information exchange according to physical connectivity: NerveNet propagates features along the robot graph, while Body Transformer uses sensor-actuator topology to constrain attention between body components (Wang et al., 2018; Sferrazza et al., 2025). Morphology-conditioned approaches adapt shared controllers to different embodiments. MetaMorph incorporates morphology as an input modality to a Transformer, while Uni-

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.

versal Morphology Control uses morphology-dependent modulation to adapt control parameters and interactions between body parts (Gupta et al., 2022; Xiong et al., 2023). Recent methods integrate embodiment information into action representations and attention mechanisms. SAT represents actions as joint-wise trajectories with functional and kinematic embeddings, while morphology-aware Transformers encode kinematic topology and joint attributes through attention and token representations (Lei et al., 2026; Suzuki et al., 2026). ACE-Ego-0 derives morphology tokens from robot URDF graphs to condition the action expert, and DexGrasp-Zero incorporates embodiment-specific physical constraints into a morphology-aligned graph representation (Li et al., 2026a; Wu et al., 2026). PhysGraph introduces attention biases based on kinematic connectivity, geometric proximity, contact relations, and anatomical structure for bimanual manipulation (Li et al., 2026b). However, these representations primarily emphasize joint and link attributes, physical relations, or morphology-conditioned control. In complex robots, joints form subsystems such as the base, arms, wrists, and grippers, each with distinct motion characteristics and coordination requirements. This motivates an explicit functional-group representation that structures both interactions within subsystems and coordination across them.

We propose **GroupACT**, a VLA model that introduces functional-group coordination through group-mediated blocks. The model combines a pretrained VLM backbone with an action expert and inserts group-mediated layers at selected depths of the action expert. Each group-mediated block organizes joint-token interactions according to an annotated joint-group graph. Joint tokens first interact within their assigned groups after VL-prefixed attention. Learned group queries, shared across group layers, then aggregate joint-group features through joint-group masked cross-attention. The resulting group tokens exchange information through inter-group self-attention. Each updated group representation is broadcast to its member joints and fused with their individual features through an MLP. Then updated joint tokens are passed to the next layer. Dense joint-attention layers remain between the group layers, allowing the model to alternate between direct joint-level interaction and subsystem-level coordination. Each block also attends to the projected final hidden state of the pretrained VLM, provided as a vision-language prefix. Action grouping therefore retains access to the visual and linguistic context.

The paper makes three contributions.

- We construct a joint-group graph based on the articulated structure and functional organization of each robot. The graph encodes joints to action groups according to their functional roles and motion types. For a bimanual mobile manipulator, these groups can represent the mobile base, each arm, and each end-effector, providing an explicit representation of functional subsystems.
- We introduce a group-mediated block that incorporates the physical structure encoded by the joint-group graph for Transformer-based VLA action experts. The block models both intra-group interactions among joints and inter-group coordination among functionally distinct subsystems.
- We evaluate GroupACT on mobile manipulation tasks in MS-HAB benchmark and bimanual coordination tasks on a physical robot. Both settings require coordinated control across heterogeneous joints. GroupACT outperforms the existing baselines on multiple tasks, supporting the effectiveness of functional-group priors for learning coordinated behaviors on the evaluated platforms.

2 Related Work

2.1 Vision-Language-Action Models

Diverse robot datasets support generalist policy learning and adaptation across observation and action spaces (Brohan et al., 2023; Open X-Embodiment Collaboration, 2023; Ghosh et al., 2024), while multimodal pretraining connects visual and linguistic knowledge to embodied reasoning and control (Driess et al., 2023; Zitkovich et al., 2023; Kim et al., 2025). Diffusion Policy, RDT-1B, and π_0 generate continuous action sequences through diffusion or flow matching, with RDT-1B additionally introducing a physically interpretable action space for heterogeneous robot data (Chi et al., 2023; Liu et al., 2025; Black et al., 2025b). RT-H and $\pi_{0.5}$ incorporate intermediate language descriptions or high-level semantic predictions to support task execution (Belkhale et al., 2024; Black

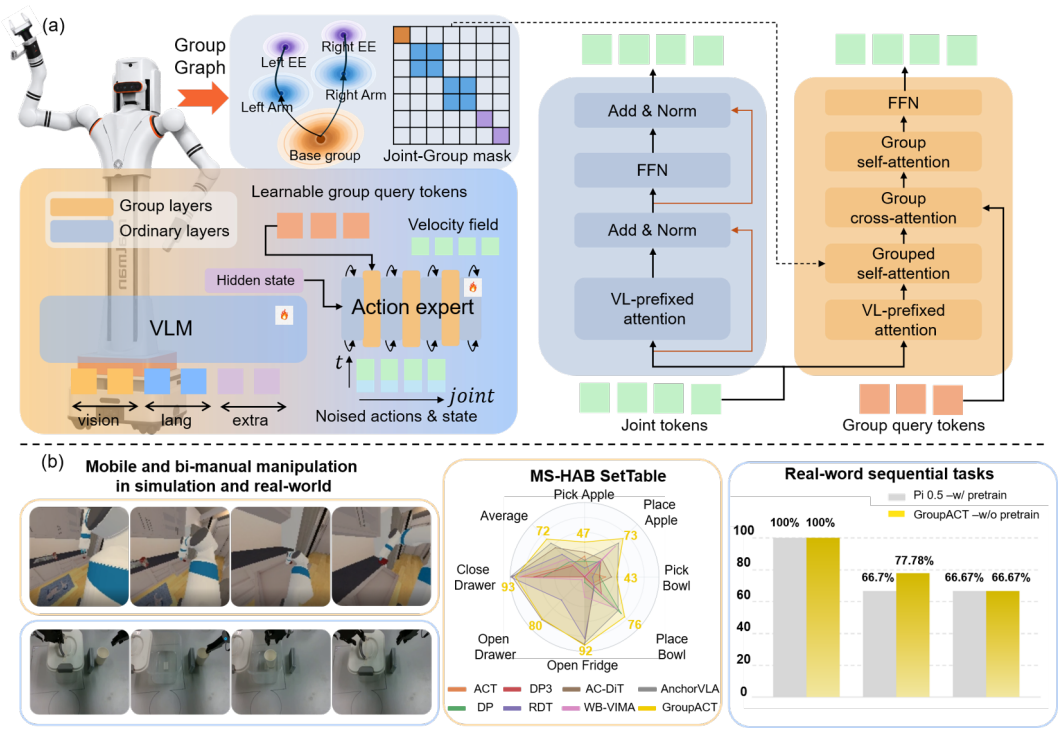

Figure 1: (a) **GroupACT** architecture. VLM hidden states serve as a prefix for action generation by the action expert. Group-mediated layers use joint-group masks derived from robot functional-group structure to combine masked intra-group self-attention with joint-to-group aggregation and inter-group coordination. Residual modules are not illustrated in group layer but applied at each attention sublayer. (b) Simulation and real-world evaluations. GroupACT achieves a mean success rate of 72% on MS-HAB SetTable, outperforming all evaluated baselines. On real-world tasks, it matches or exceeds the success rates of pretrained $\pi_{0.5}$.

et al., 2025a). Although these methods differ in their action representations and prediction objectives, they generally leave the distinction between intra-group joint interactions and inter-group coordination implicit. GroupACT makes this functional organization explicit by interleaving standard Transformer layers with group-mediated layers. This design provides a structural prior for learning heterogeneous action trajectories, while the intervening dense layers preserve direct joint-level interactions across groups.

2.2 Embodiment-Aware Policy Learning

Embodiment-aware policies incorporate robot structure into the representations used for control. NerveNet applies graph message passing over body structure, while shared modular policies reuse local controllers across actuators and coordinate them through messages (Wang et al., 2018; Huang et al., 2020). CASNET builds subsystem and whole-agent representations through cascaded recurrent encoders and decoders (Malik, 2019). Transformer-based approaches offer additional mechanisms: MetaMorph conditions control on modular morphology descriptions, and ModuMorph uses morphology-conditioned parameter generation and attention (Gupta et al., 2022; Xiong et al., 2023). Amorpheus also shows that restricting communication to physical connectivity is not uniformly beneficial across control settings (Kurin et al., 2021). Body Transformer, GET-Zero, and SEM encode embodiment through structured attention or graph-based state representations (Sferrazza et al., 2025; Patel & Song, 2025; Lin et al., 2025). Body Transformer also studies progressive pooling and interleaved masked and dense layers.

Recent work brings structural representations closer to VLA action generation. SAT represents actions as joint trajectories with functional and kinematic annotations, while morphology-embedded

Transformers introduce kinematic tokens, topology-dependent attention, and joint-attribute modulation (Lei et al., 2026; Suzuki et al., 2026). ACE-Ego-0 conditions action generation on URDF-derived robot morphology tokens and learned human surrogate representations (Li et al., 2026a). CrossDex and DexGrasp-Zero align embodiments through shared grasp spaces with retargeting and anatomical motion representations, respectively (Yuan et al., 2025; Wu et al., 2026). PhysGraph encodes physical interaction biases, while Instant Policy organizes demonstrations, observations, and actions as graphs (Li et al., 2026b; Vosylius & Johns, 2025).

However, these embodiment priors primarily encode physical robot structure, such as joint connectivity (Sferrazza et al., 2025) or morphological context (Li et al., 2026a). Given the captured embodiment information, these methods leave coordination among functional subsystems implicit. Restricting information flow to local kinematic connections may be suboptimal, as the joints most relevant to the motion of a given joint are not necessarily its immediate kinematic neighbors, particularly in mobile manipulators or bimanual robots (Kurin et al., 2021). Hence, we construct a joint-group graph based on joint functions and motion properties to make this functional organization explicit. The graph uses nodes and edges to represent the articulation of robots, while providing a higher-level structural prior over action groups.

3 METHOD

Problem Formulation. Given visual observations, a language instruction, and proprioceptive states, the policy generates an action chunk $A \in \mathbb{R}^{N\times H}$ for an embodiment with N action degrees of freedom (DoFs) and K functional groups, where H denotes the prediction horizon. Following SAT (Lei et al., 2026), we organize action chunks along the joint dimension: each row represents the trajectory of one action coordinate over the full prediction horizon. This representation supports grouping across action DoFs.

For each embodiment, the action DoFs are partitioned into K nonempty, pairwise disjoint functional groups $\mathcal{G}_1, \ldots, \mathcal{G}_K$. The joint-group mask $B \in \{0,1\}^{K\times N}$ is defined as

$$B_{gj} = \begin{cases} 1, & j \in \mathcal{G}_g, \\ 0, & \text{otherwise}, \end{cases} \qquad 1 \le g \le K, \quad 1 \le j \le N. \tag{1}$$

Each valid action DoF therefore belongs to exactly one functional group.

To construct joint tokens, separate linear projections encode the noised action trajectory A_j^τ at flow time τ and the aligned scalar state q_j, when available. A MLP fuses the resulting features into z_j^{AS}. We then add shared embeddings for the within-group index p_j and motion category r_j:

$$z_j = \left[z_j^{AS} + E_{\text{pos}}(p_j) + E_{\text{motion}}(r_j) \right]. \tag{2}$$

The exact definition of p_j and r_j for each embodiment can be found in the Appendix B. This construction combines joint-level action and state information with functional attributes for the subsequent group-mediated blocks.

For the attention operations below, Q, K, and V denote the query, key, and value matrices, respectively, and d is the query and key dimension within each head.

$$\text{Attention}(Q,K,V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V. \tag{3}$$

Given a binary attention mask M, its masked variant is

$$\begin{aligned} \text{Attention}(Q,K,V;M) &= \text{Softmax}\left(\frac{QK^T}{\sqrt{d}} + \mathcal{M}(M)\right)V, \\ \mathcal{M}(M)_{ij} &= \begin{cases} 0, & M_{ij} = 1, \\ -\infty, & M_{ij} = 0. \end{cases} \end{aligned} \tag{4}$$

Multi-head attention applies these operations to separately projected heads, concatenates their outputs, and applies an output projection.

3.1 VISION-LANGUAGE MEMORY AND FLOW MATCHING

As shown in Figure 1, our model uses a VLM to encode visual observations and language instructions. Joint tokens attend to the prefixed VL hidden states to predict the velocity field in the action expert. We use the official Qwen3-VL-4B-Instruct model and its associated processor as the VLM (Yang et al., 2025). The current camera observations and task instruction are formatted using the corresponding chat template and processed by the model. We apply shared RMS normalization and a MLP projection to the full sequence of final hidden states to obtain the observation memory $VL \in \mathbb{R}^{L\times D}$, where L is the sequence length.

The vision encoder and the pretrained base weights of the language model remain frozen. Low-rank adaptation (LoRA) (Hu et al., 2021) is applied exclusively to the query, key, value, and output projections in the language decoder.

The training targets are normalized joint action chunks A^1. For each sample, we draw $A^0 \sim \mathcal{N}(0, I)$ over the robot's actions and sample $\tau \sim \mathcal{U}[0, 1]$. We then construct the linear flow path and its target velocity:

$$A^\tau = (1-\tau)A^0 + \tau A^1, \qquad U = A^1 - A^0. \tag{5}$$

The predicted velocity $v_\theta \in \mathbb{R}^{B\times N_{\max}\times H}$ is trained using:

$$\mathcal{L} = \mathbb{E}\left[\frac{1}{NH}\left\|v_\theta - U\right\|_F^2\right], \tag{6}$$

which is the only training loss in our method.

For each sample, the squared error is averaged over all valid actions and the H prediction steps. Each action trajectory is normalized independently using an epsilon-stabilized affine transformation (Appendix B.5).

3.2 GROUP-MEDIATED BLOCKS

The main contribution is the group-mediated block introduced into the action expert, which augments joint-token interactions with explicit intra-group communication and inter-group coordination. The action expert, built upon the transformer structure, consists of ordinary and group-mediated layers. Ordinary layers combine VL-prefixed attention, and a joint-wise feed-forward network. A group-mediated layer first applies the same VL-prefixed attention, followed by intra-group self-attention with joint-group graph mask. Shared group queries then aggregate the updated joint representations through masked cross-attention, producing group tokens. These group tokens then exchange information through inter-group self-attention. The resulting group representations are broadcast to its member joints. The updated joint tokens are passed to the next layer. The structures of ordinary layers and group layers are illustrated in Figure 1 (a).

Intra-group self-attention. Given the joint tokens $X \in \mathbb{R}^{N\times D}$ as input, the joint tokens first attend to both the VL memory and other joint tokens through self-attention with a VL key-value prefix. Queries are computed from the joint representations, while the keys and values concatenate the VL prefix with those computed from the joint representations:

$$J' = \text{Attention}\left(XW_{VL}^Q, [VL; X]W_{VL}^K, [VL; X]W_{VL}^V\right), \tag{7}$$

where $VL \in \mathbb{R}^{L\times D}$ represents the he observation memory, and $W_{VL}^{\{Q,K,V\}}$ are projection matrices in the VL-prefixed self-attention operation.

Then, the group layer applies the intra-group graph masked multi-head self-attention:

$$J = \text{Attention}\left(J'W_J^Q, J'W_J^K, J'W_J^V; M\right),\ M = B^\top B, \tag{8}$$

where $M \in \{0, 1\}^{N\times N}$ is the intra-group graph mask, B is the aforementioned joint-group mask and $W_J^{\{Q,K,V\}}$ are projection matrices in the intra-group self-attention operation. Within this operation, each joint therefore attends only to joints in the same group.

Joint-to-group aggregation. After the intra-group block, we perform joint-to-group aggregation using K learnable group query tokens $G \in \mathbb{R}^{K \times D}$, shared across all group-mediated layers. Each query token attends only to joint tokens within the corresponding group through masked cross-attention,

$$G = \text{Attention}\left(GW_A^Q, JW_A^K, JW_A^V; B\right), \tag{9}$$

where G is the aggregated group token matrix, $W_A^{\{Q,K,V\}}$ are projection matrices in the aggregation operation. The aggregation summarizes information from joints with similar motion patterns and functions. This operation maps N joint tokens to K group tokens for subsequent inter-group operation. The learnable group query tokens are randomly initialized.

Inter-group self-attention. After aggregation, we use self-attention to exchange information across groups. Each input group token summarizes its corresponding group, allowing self-attention to model inter-group interactions and produce the updated group tokens G':

$$\tilde{G} = \text{Attention}\left(GW_G^Q, GW_G^K, GW_G^V\right), \tag{10}$$

where $W_G^{\{Q,K,V\}}$ are projection matrices in the inter-group operation. This operation models inter-group dependencies at the group level, with each group represented by a single token regardless of its number of joints. The updated group tokens provide context for the subsequent group-to-joint broadcast.

Group-to-joint broadcast. After intra-group and inter-group interactions, we broadcast the group tokens back to their member joints to obtain the updated joint tokens $\tilde{X}$, which serve as input to the next layer.

The joint-group graph mask broadcasts each contextualized group representation exclusively to its member joints. A two-layer MLP fuses the resulting context with the joint-specific features,

$$\tilde{J} = \text{MLP}(B^\top G'). \tag{11}$$

The updated joint tokens $\tilde{J}$ pass through an feed forward network (FFN), whose output serves as input to the next layer.

$$\tilde{X} = \text{FFN}(\tilde{J}) \in \mathbb{R}^{N \times D}. \tag{12}$$

The structured communication path within each group-mediated layer is $J \to G \to \tilde{G} \to \tilde{J}$, where J and G denote the joint and group tokens, respectively, and $\tilde{J}$ and G' denote their updated representations. The joint-group graph mask restricts both aggregation and broadcast to member joints, while group-level self-attention enables communication across functional subsystems. This structure preserves joint-specific representations while routing cross-group communication through a group token representation per group. Interleaved ordinary layers complement this restricted path by providing direct joint-level communication across groups. The group summaries therefore impose a layer-specific inductive bias rather than a persistent bottleneck on cross-group communication. Each attention sublayer in both ordinary and group-mediated layers uses a residual connection followed by layer normalization. For clarity, the equations show only the core transformations, omitting these residual connections and normalization operations.

4 EXPERIMENTS

The central question is whether the proposed group-mediated structure improves action learning for robots with multiple functional subsystems beyond a general policy. We study this question on MS-HAB SetTable (Shukla et al., 2025), a mobile-manipulation benchmark and a real-world bimanual manipulation task.

We evaluate GroupACT on seven tasks from the MS-HAB SetTable benchmark. GroupACT achieves higher task success rates than the existing baselines. To isolate the contribution of the group-mediated structure, we conduct ablation studie in which the group layers in the action expert are progressively replaced with ordinary layers while keeping the remaining architecture. The results show that the structured group layers improve performance on mobile-manipulation tasks.

Table 1: MS-HAB SetTable results across different manipulation subtasks.

Method	Pick Apple	Place Apple	Pick Bowl	Place Bowl	Open Fridge	Open Drawer	Close Drawer	Average
ACT	28.0%	8.7%	28.0%	13.0%	2.0%	0.0%	85.7%	23.6%
DP	21.3%	28.0%	20.7%	69.3%	7.3%	0.0%	55.0%	28.8%
DP3	0.0%	31.0%	20.0%	32.0%	0.0%	0.0%	68.0%	21.6%
RDT	12.0%	32.0%	10.7%	18.7%	82.7%	44.0%	**100.0%**	42.9%
AC-DiT	33.3%	33.3%	36.0%	17.3%	90.7%	**81.3%**	97.3%	55.6%
WB-VIMA	1.6%	57.7%	1.6%	60.3%	27.0%	5.3%	87.3%	34.4%
AnchorVLA	22.7%	64.3%	**44.5%**	63.8%	88.9%	–	**100.0%**	64%
GroupACT	**47%**	**73%**	43%	**76%**	**92%**	80%	93%	**72.0%**

We further evaluate GroupACT on bimanual manipulation tasks using a real-world dual-arm platform. We collect demonstrations of language-conditioned tabletop storage tasks using a bimanual robot and evaluate GroupACT on the physical platform, comparing its performance with pretrained $\pi_{0.5}$.

4.1 MS-HAB SetTable Benchmark

The action space contains 11 DoFs: base forward velocity, base angular velocity, torso lift, seven arm joints, and one gripper command. These DoFs are organized into four groups of sizes 2, 1, 7, and 1, respectively. We use a prediction horizon of $H = 16$ at 20 Hz, corresponding to 0.8 s of future actions, and execute the first 4 steps of a predicted chunk before replanning.

A template converts each task name into a one-sentence instruction. We use RGB-D images at a resolution of 128×128 from the main and the wrist-mounted camera as visual inputs. We normalize the depth values to map physical distances measured by the depth sensors to image pixel intensities using a minus tanh mapping.

Evaluation covers seven subtasks: Pick Apple, Place Apple, Pick Bowl, Place Bowl, Open Fridge, Open Drawer, and Close Drawer. We evaluate GroupACT on 100 episodes per subtask, with no overlap between training and evaluation episodes.

As shown in Table 1, We compare GroupACT to previous methods including: ACT (Zhao et al., 2023), DP (Chi et al., 2025), DP3 (Ze et al., 2024), RDT (Liu et al., 2025), AC-DiT (Chen et al., 2026), WB-VIMA (Jiang et al., 2025), and AnchorVLA (Lim et al., 2026). GroupACT achieves a mean success rate of 72% across the seven subtasks. Among baselines with results for all seven subtasks, AC-DiT achieves the highest mean success rate of 55.6%, giving GroupACT an absolute advantage of 16.4 percentage points. AnchorVLA has no reported result for Open Drawer, so its average covers only six subtasks. On these six shared subtasks, GroupACT achieves a mean success rate of 70.6%, compared with 64% for AnchorVLA, a difference of 6.60 percentage points. GroupACT ranks first on Pick Apple, Place Apple, Place Bowl, and Open Fridge, with success rates of 47.0%, 73.0%, 76.0%, and 92.0%, respectively. Relative to the strongest baseline on each subtask, the corresponding improvements are 13.7, 8.7, 6.7, and 1.3 percentage points. GroupACT ranks second on Pick Bowl and Open Drawer, achieving success rates of 43.0% and 80.0%, respectively. These results are 1.5 and 1.3 percentage points below the strongest baselines on the corresponding subtasks.

4.2 Ablation Study

We conduct ablation experiments on the same MS-HAB SetTable subtasks to assess the contribution of the group-mediated blocks in GroupACT. The first variant removes the group query tokens, grouped cross-attention, and inter-group self-attention while retaining masked intra-group self-attention. This removes the group-mediated path for cross-group coordination while preserving intra-group communication. The second variant replaces all group-mediated layers with ordinary Transformer layers.

Table 2: Ablation results on MS-HAB SetTable. The term w/o GA means without group query tokens, grouped cross-attention, and inter-group self-attention. The term w/o GL means without full group layers.

Method	Pick Apple	Place Apple	Pick Bowl	Place Bowl	Open Fridge	Open Drawer	Close Drawer	Average
Full GroupACT	**47%**	**73%**	**43%**	**76%**	**92%**	**80%**	**93%**	**72.00%**
w/o GA	24%	53%	26%	69%	91%	46%	**93%**	57.42%
w/o GL	22%	56%	21%	53%	88%	39.8%	88%	52.54%

Table 2 shows that full GroupACT achieves a mean success rate of 72.00%. Removing group query tokens, grouped cross-attention, and inter-group self-attention reduces this rate to 57.43%, a decrease of 14.57 percentage points. Performance declines on six subtasks, with the largest reductions on Open Drawer, Pick Apple, and Place Apple, at 34.0, 23.0, and 20.0 percentage points, respectively. Close Drawer remains unchanged at 93%. Replacing all group-mediated layers with ordinary layers further reduces the mean success rate to 52.54%, which is 19.46 percentage points below full GroupACT. The full model outperforms this variant on all seven subtasks. The largest gains occur on Open Drawer, Pick Apple, Place Bowl, and Pick Bowl, at 40.2, 25.0, 23.0, and 22.0 percentage points, respectively. Improvements on Place Apple, Open Fridge, and Close Drawer are 17.0, 4.0, and 5.0 percentage points. Two invalid trajectories because of simulator failure appeared when evaluating the open drawer tasks of w/o GL variant and they are excluded from the success rate calculation. These results support the contribution of group layers within the structured blocks and the effectiveness of the group-mediated design as a whole.

4.3 Real-World Experiment

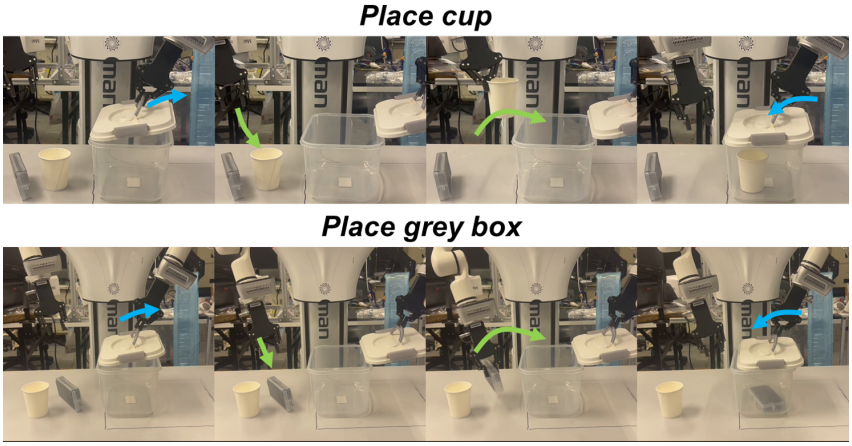

Figure 2: Bimanual manipulation on RealMan dual-arm robot platform. The blue arrows denote the left arm motion and the green arrows denote the right arm motion.

We collect teleoperated demonstrations for two long-horizon tabletop storage tasks on a RealMan dual-arm robot: storing cups and storing boxes. The dataset contains 64 valid demonstrations, with 29 for cup storage and 35 for box storage. Each arm has six revolute DoFs and a gripper. Both data collection and policy inference use images from one main-view camera and two wrist-mounted cameras. We evaluate both tasks on the physical robot. Figure 2 shows example task executions, and Appendix D.3 provides recordings from all three camera views.

We divide each task into three sequential stages based on the active arm. In R1, the left arm removes the container lid and holds it aside. In R2, the right arm grasps the object specified by the instruction and places it inside the container. In R3, the left arm replaces the lid to close the container. Both demonstration collection and evaluation execute each episode continuously across the three stages.

Table 3: Real-World Dual-Arm Tabletop Storage Experiments

Method	Open Container	Place Cup	Close Box	Open Container	Place Grey Box	Close Box
Pi0.5	**100%**	77.78%	**77.78%**	**100%**	55.56%	55.56%
GroupACT	**100%**	**88.89%**	66.67%	**100%**	**66.67%**	**66.67%**

During evaluation, we record stage outcomes in sequence and report stage-wise success rates. If stage R_t fails, all subsequent stages in that episode are also counted as failures.

These tasks require the policy to learn distinct behaviors for each arm, coordinate their actions, and adapt to the current task stage. We initialize GroupACT with the official Qwen3-VL-4B-Instruct checkpoint and a randomly initialized action expert, then train it on 64 demonstrations without prior robot image-action pretraining. We compare GroupACT with $\pi_{0.5}$, which is pretrained on large-scale robot data (Black et al., 2025a). As shown in Table 3, both policies achieve a 100% success rate in opening the container for both tasks. GroupACT achieves higher object-placement success rates, reaching 88.89% for the cup and 66.67% for the grey box, compared with 77.78% and 55.56% for $\pi_{0.5}$, respectively. Performance at the final box-closing stage varies by task: GroupACT achieves a lower success rate for the cup task (66.67% vs. 77.78%) but a higher success rate for the grey-box task (66.67% vs. 55.56%). Averaged across the two tasks, GroupACT therefore matches $\pi_{0.5}$ in box opening and closing, while improving object-placement success from 66.67% to 77.78%. Qualitative evaluation rollouts also illustrate online error correction by GroupACT. As shown in Figure 3, when the gripper is initially misaligned with the grey box, the policy makes a precise positional adjustment before successfully grasping it.

5 Conclusion

GroupACT introduces physical and functional priors through a joint-group graph and group-mediated layers, structuring communication within and across functional groups to model heterogeneous joint trajectories. On MS-HAB SetTable, GroupACT achieves 72.00% mean success across seven subtasks and ranks first on four. Replacing group-mediated layers with ordinary Transformer layers reduces mean success by 19.46 percentage points. On a physical dual-arm robot, GroupACT achieves 66.67% full-task success on both cup and box storage tasks using 64 teleoperated demonstrations without prior robot image-action pretraining. It outperforms $\pi_{0.5}$ at the object-placement stage of both tasks, with individual rollouts showing grasp retries and alignment corrections.

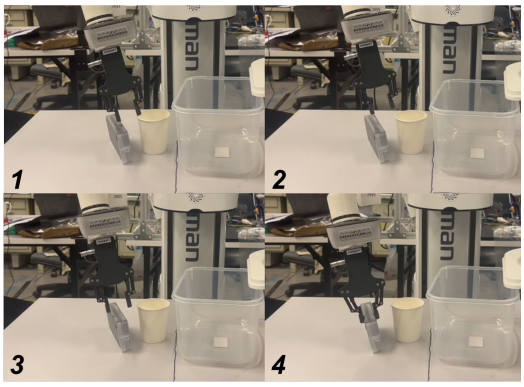

Figure 3: Misalignment correction and regrasping on the place grey box task. The time sequence of the four pictures are marked using numbers.

Computational and hardware constraints limit evaluation to the number of benchmarks and embodiments. Future work will extend evaluation to additional simulation benchmarks and real-world tasks. We will also collect demonstrations across embodiments and train shared policies to assess whether group-mediated coordination improves training efficiency and task performance across robots with different functional configurations.

AI USE STATEMENT

In this work, we used generative AI tools for code implementation, translation and grammar correction. We have not used generative AI tools for create or modify scientific figures or images. No parts of the experimental results, or conclusions were generated by LLMs.

REFERENCES

Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quan Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. RT-H: Action Hierarchies using Language. In *Proceedings of Robotics: Science and Systems*, 2024. doi: 10.15607/RSS.2024.XX.049. URL https://www.roboticsproceedings.org/rss20/p049.html.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael 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, 2025a. URL https://arxiv.org/abs/2504.16054.

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.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S Ryoo, Grecia Salazar, Pannag R Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan H Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics Transformer for Real-World Control at Scale. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.025. URL https://www.roboticsproceedings.org/rss19/p025.html.

Sixiang Chen, Jiaming Liu, Siyuan Qian, Han Jiang, Zhuoyang Liu, Chenyang Gu, Xiaoqi Li, Chengkai Hou, Pengwei Wang, Zhongyuan Wang, et al. Ac-dit: Adaptive coordination diffusion transformer for mobile manipulation. *Advances in Neural Information Processing Systems*, 38:64008–64036, 2026.

Nikita Chernyadev, Nicholas Backshall, Xiao Ma, Yunfan Lu, Younggyo Seo, and Stephen James. BiGym: A Demo-Driven Mobile Bi-Manual Manipulation Benchmark. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4201–4217. PMLR, 2025. URL https://proceedings.mlr.press/v270/chernyadev25a.html.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM 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.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-E: An Embodied Multimodal Language Model. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 8469–8488. PMLR, 2023. URL https://proceedings.mlr.press/v202/driess23a.html.

Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile ALOHA: Learning Bimanual Mobile Manipulation using Low-Cost Whole-Body Teleoperation. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4066–4083. PMLR, 2025. URL https://proceedings.mlr.press/v270/fu25b.html.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*, 2024. doi: 10.15607/RSS.2024.XX.090. URL https://www.roboticsproceedings.org/rss20/p090.html.

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://openreview.net/forum?id=Opmqtk_GvYL.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Wenlong Huang, Igor Mordatch, and Deepak Pathak. One policy to control them all: Shared modular policies for agent-agnostic control. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4455–4464. PMLR, 2020. URL https://proceedings.mlr.press/v119/huang20d.html.

Yunfan Jiang, Ruohan Zhang, Josiah Wong, Chen Wang, Yanjie Ze, Hang Yin, Cem Gokmen, Shuran Song, Jiajun Wu, and Li Fei-Fei. BEHAVIOR Robot Suite: Streamlining Real-World Whole-Body Manipulation for Everyday Household Activities. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 1246–1281. PMLR, 2025. URL https://proceedings.mlr.press/v305/jiang25b.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An Open-Source Vision-Language-Action Model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Vitaly Kurin, Maximilian Igl, Tim Rocktäschel, Wendelin Böhmer, and Shimon Whiteson. My body is a cage: the role of morphology in graph-based incompatible control. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=N3zUDGN5lO.

Xiaohan Lei, Min Wang, Bohong Weng, Wengang Zhou, and Houqiang Li. Structural action transformer for 3d dexterous manipulation. *IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR)*, 2026.

Hao Li, Ganlong Zhao, Yufei Liu, Haotian Hou, Guoquan Ye, Tongyan Fang, Chunxiao Liu, Siyuan Huang, Jianbo Liu, Xiaogang Wang, and Hongsheng Li. ACE-Ego-0: Unifying Egocentric Human and Robotic Data for VLA Pretraining, 2026a. URL https://arxiv.org/abs/2606.17200.

Runfa Blark Li, David Kim, Xinshuang Liu, Keito Suzuki, Dwait Bhatt, Nikola Raicevic, Xin Lin, Ki Myung Brian Lee, Nikolay Atanasov, and Truong Nguyen. PhysGraph: Physically-Grounded Graph-Transformer Policies for Bimanual Dexterous Hand-Tool-Object Manipulation, 2026b. URL https://arxiv.org/abs/2603.01436.

Jia Syuen Lim, Zhizhen Zhang, Peter Bohm, Brendan Tidd, Zi Huang, and Yadan Luo. AnchorVLA: Anchored Diffusion for Efficient End-to-End Mobile Manipulation, 2026. URL https://arxiv.org/abs/2604.01567.

Xuewu Lin, Tianwei Lin, Lichao Huang, Hongyu Xie, Yiwei Jin, Keyu Li, and Zhizhong Su. SEM: Enhancing Spatial Understanding for Robust Robot Manipulation, 2025. URL https://arxiv.org/abs/2505.16196.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: a Diffusion Foundation Model for Bimanual Manipulation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/49f80e4d2471ad4f2edf4f5f1ab62339-Abstract-Conference.html.

Ashish Malik. Zero-shot generalization using cascaded system-representations, 2019. URL https://arxiv.org/abs/1912.05501.

Chi Kit Ng, Long Bai, Guankun Wang, Yupeng Wang, Huxin Gao, Kun Yuan, Chenhan Jin, Tieyong Zeng, and Hongliang Ren. Endovla: Dual-phase vision-language-action model for autonomous tracking in endoscopy, 2025. URL https://arxiv.org/abs/2505.15206.

Open X-Embodiment Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models, 2023. URL https://arxiv.org/abs/2310.08864.

Austin Patel and Shuran Song. GET-Zero: Graph Embodiment Transformer for Zero-shot Embodiment Generalization. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025. URL https://get-zero-paper.github.io/.

Carmelo Sferrazza, Dun-Ming Huang, Fangchen Liu, Jongmin Lee, and Pieter Abbeel. Body transformer: Leveraging robot embodiment for policy learning. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3407–3424. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/sferrazza25a.html.

Arth Shukla, Stone Tao, and Hao Su. Maniskill-hab: A benchmark for low-level manipulation in home rearrangement tasks. In *International Conference on Learning Representations*, volume 2025, pp. 15288–15317, 2025.

Mingwu Su, Guankun Wang, Jinsong Lin, Rulin Zhou, Ziyi Hao, Zhiwei Fang, Huxin Gao, Jiewen Lai, Jiazheng Wang, Fan Zhang, and Hongliang Ren. Trajectory divergence horizon decision for reliable dual-arm surgical subtask manipulation, 2026. URL https://arxiv.org/abs/2608.09125.

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, 2026. URL https://arxiv.org/abs/2603.00182.

Vitalis Vosylius and Edward Johns. Instant Policy: In-Context Imitation Learning via Graph Diffusion. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://www.robot-learning.uk/instant-policy.

Tingwu Wang, Renjie Liao, Jimmy Ba, and Sanja Fidler. Nervenet: Learning structured policy with graph neural networks. In *ICLR (Poster)*, 2018.

Yuliang Wu, Yanhan Lin, WengKit Lao, Yuhao Lin, Yi-Lin Wei, Wei-Shi Zheng, and Ancong Wu. DexGrasp-Zero: A Morphology-Aligned Policy for Zero-Shot Cross-Embodiment Dexterous Grasping, 2026. URL https://arxiv.org/abs/2603.16806.

Zheng Xiong, Jacob Beck, and Shimon Whiteson. Universal morphology control via contextual modulation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38286–38300. PMLR, 2023. URL https://proceedings.mlr.press/v202/xiong23a.html.

Xiaomeng Xu, Jisang Park, Han Zhang, Eric Cousineau, Aditya Bhat, Jose Barreiros, Dian Wang, Jeannette Bohg, and Shuran Song. HoMMI: Learning Whole-Body Mobile Manipulation from Human Demonstrations. In *Proceedings of Robotics: Science and Systems*, 2026. doi: 10.15607/RSS.2026.XXII.205. URL https://www.roboticsproceedings.org/rss22/p205.html.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Haoqi Yuan, Bohan Zhou, Yuhui Fu, and Zongqing Lu. Cross-Embodiment Dexterous Grasping with Reinforcement Learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/ca8c6f28d8ba1e732e3f217ab05c4ec0-Paper-Conference.pdf.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

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

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

Table 4: Tensor interfaces of the implemented policy. The state array is stored as $[B, J]$ and receives its singleton feature axis inside the encoder. $H = 16$ for the MS-HAB configuration described here and $H = 25$ for RealMan. The action expert consists of 18 layers, with group layers replacing ordinary layers at positions 9, 12, and 15. It therefore contains 15 ordinary layers and three group layers.

Component	Operation	Output shape
Action trajectory	One horizon per action slot	$[B, J, H]$
Action encoder	Linear, $H \to 512$	$[B, J, 512]$
State encoder	Linear, $1 \to 512$	$[B, J, 512]$
Feature concatenation	Action state concatenation	$[B, J, 1024]$
Fusion MLP	Linear $1024 \to 512$, SiLU, Linear $512 \to 512$	$[B, J, 512]$
Qwen final hidden state	All nonpadding multimodal positions	$[B, L, 2560]$
VL projection	Linear, $2560 \to 512$	$[B, L, 512]$
Action Expert	Action generation	$[B, J, 512]$
Temporary group stream	Present only inside group-mediated layers	$[B, K, 512]$

A Architecture and Implementation Details

A.1 Action slots, groups, and tensor conventions

The details of GroupACT are listed in Table 4. We use B for batch size, J for the padded number of action slots, K for the padded number of groups, H for the action horizon, D for the Action Expert width, and L for the variable-length vision-language sequence. The implementation pads to $J = 20$ and $K = 6$. Its membership tensor $S \in \{0, 1\}^{K \times J}$ assigns each valid action slot to exactly one nonempty group.

A.2 Vision-language backbone and common memory

The policy uses the official Qwen3-VL-4B-Instruct model and processor. Images from the current observation are placed in a fixed order in one user message, followed by a text field of the form `Embodiment: {name}. Task: {instruction}.` The official chat template supplies the model-specific image and conversation tokens, including the generation prompt. Dataset-specific auxiliary text, when enabled, is appended to the task instruction before this formatting step.

All original Qwen parameters, including the vision encoder, are frozen. LoRA (Hu et al., 2021) is attached only to the decoder `q_proj`, `k_proj`, `v_proj`, and `o_proj` modules, with rank 16, scaling parameter 32, and dropout zero. The inspected backbone has 36 decoder layers, yielding 144 adapted projections and 288 low-rank parameter tensors. The Action Expert and other new policy modules are initialized independently of Qwen.

B Dataset Interfaces and Preprocessing

B.1 MS-HAB SetTable action and state interface

The simulation adapter uses the low-level controller demonstrations from ManiSkill-HAB, with control mode `pd_joint_delta_pos`. The model predicts recorded controller actions directly. In particular, an arm controller command is not converted a second time into a displacement by subtracting the observed joint position.

Group sizes are $[2, 1, 7, 1]$, with joint local index p_j: $[0, 1]$, $[0]$, $[0, \ldots, 6]$, and $[0]$ in each group. The motion category r_j for each group is $[0, 1, 2, 3]$, respectively. The gripper state is the mean of the two recorded finger positions. Nine slots have state measurements and all eleven have action targets. Both physical and normalized base states are explicitly zero-filled, while their action-valid bits remain true. Slots 11–19 and groups 4–5 are padding. The state excludes head pan and head tilt.

Table 5: Joint-Group labels. Raw action indices are zero-based. The two original head-controller DoFs are excluded from learning and restored as zeros when commands are sent to the environment.

Canonical slots	Group	State supplied to the model	Motion category ID
0–1: base velocity, base yaw	Base	Missing	0
2: torso lift	Body	`qpos[0]`	1
3–9: seven arm coordinates	Arm	`qpos[3:10]`	2
10: gripper	Gripper	Mean of `qpos[10]` and `qpos[11]`	3

B.2 Observations

MS-HAB visual observations are supplied in the order head RGB, head depth, wrist RGB, wrist depth. Each source view has spatial resolution 128×128. The depth stream contains millimeter-valued aligned depth rather than RGB colors. For a nonzero depth value, we construct

$$I_D = \mathrm{round}(255\,[1 - \tanh(D_{\mathrm{mm}}/1000)]) \tag{13}$$

and repeat the resulting uint8 channel three times. A zero depth value is treated as invalid and mapped to black. The transformation does not use per-frame minimum–maximum normalization. The Qwen processor subsequently performs its official image preprocessing, with minimum and maximum pixel settings 3,136 and 200,704. The task text includes the prefix `set table:` and the official subtask name, for example `set table:  pick the apple`.

B.3 Demonstration truncation and action windows

We used 1,000 demonstrations for each task to train the model. Each demonstration is truncated after its first successful transition, including the action at that transition. The retained dataset contain 365,533 transitions in total. Every retained transition is an eligible window start. For a start t and horizon offset $h = 0, \ldots, H-1$, the controller target is $a_{\min(t+h,T'-1)}$, where T' is the episode length. Windows shorter than $H = 16$ repeat the last retained action, and the repeated positions participate in the flow loss.

B.4 RealMan bimanual interface

The 14 action slots of RealMan robot are ordered as the six left-arm joints, the six right-arm joints, the left gripper, and the right gripper. The four groups have sizes $[6, 6, 1, 1]$. Arm slots use motion category $r_j = 2$ and grippers category $r_j = 3$. All fourteen slots have a state input. The two arms' recorded joint angles are converted from degrees to radians before normalization.

For frame t and $h = 1, \ldots, 25$, the targets are

$$\begin{aligned} a_{t,h}^{\mathrm{arm}} &= q_{\min(t+h,T-1)}^{\mathrm{arm}} - q_t^{\mathrm{arm}}, \\ a_{t,h}^{\mathrm{gripper}} &= g_{\min(t+h,T-1)}. \end{aligned} \tag{14}$$

Arm actions calculated using state delta. Gripper actions are absolute command-derived labels, with 0 denoting open and 1 denoting close.

The observation includes RGB views in the order `camera_high`, `camera_left_wrist`, and `camera_right_wrist`, and uses no depth or privileged information. No temporal interpolation or resampling is performed. The schema specifies a 30 Hz control rate, giving a $25/30 \approx 0.833$ s prediction horizon.

The cup instruction is: "Open the lid of the white box, pick up the water cup and place it inside the white box, then close the lid of the white box." The box instruction is: "Open the lid of the white box, pick up the gray box and place it inside the white box, then close the lid of the white box."

Table 6: Realman training sources and exact language instructions. A window count equals the number of native frames.

Task	Demonstrations	Windows
Place cup	29	41,237
Place gray box	35	46,429
Total	64	87,666

B.5 Quantile normalization

State and action have separate per-embodiment, per-coordinate normalization assets. For either quantity x, the forward and inverse transforms are

$$\begin{aligned} \widetilde{x} &= 2\frac{x - Q_{.01}}{Q_{.99} - Q_{.01} + 10^{-6}} - 1, \\ x &= \frac{\widetilde{x} + 1}{2}\left(Q_{.99} - Q_{.01} + 10^{-6}\right) + Q_{.01}. \end{aligned} \tag{15}$$

Neither transform clips to the quantile interval. Values outside the estimated percentiles can therefore map outside $[-1, 1]$. Missing-state and padding masks are applied separately from normalization.

C Training Configs

We use AdamW with peak learning rates of 10^{-4} for newly initialized modules and 10^{-5} for LoRA parameters. Each learning rate increases linearly during the first 5% of training and then follows a cosine decay schedule to 0.1 times its peak value. We apply a weight decay of 0.01 with the configured parameter exclusions and clip the gradient norm to a maximum of 1.0.

We train GroupACT and its ablation variants on eight MS-HAB tasks for 60,000 steps using a single NVIDIA H200 GPU and a batch size of 64. We exclude Close Fridge from the reported comparison because most baselines do not report results for this task. GroupACT achieves a success rate of 84% on Close Fridge. We train GroupACT and fine-tune $\pi_{0.5}$ on the RealMan dataset using a single NVIDIA A6000 GPU, a batch size of 16. Both models are trained for 10 epochs on the 64 demonstration trajectories.

D Evaluation Protocol

D.1 Action execution

The MS-HAB evaluations use $F = 4$, predict $H = 16$ controller commands, execute the first four steps of an action chunk, and then predict a new chunk. The real-world evaluations use $F = 15$, predict $H = 25$ controller commands.

Predictions are inverse-normalized and mapped back to the original controller order. For MS-HAB, the policy returns controller commands without adding the current joint positions. The official controller subsequently applies its own command clipping and scaling and, for delta-position coordinates, constructs targets relative to the current joints. Thus, unclipped quantile normalization does not remove the controller's action limits.

D.2 MS-HAB Task Evaluation

The evaluation covers the seven individual SetTable subtasks: pick apple, pick bowl, place apple, place bowl, open fridge, open drawer, and close drawer. It uses the official Pick, Place, Open, and Close subtask environments. For each subtask, 100 evaluation environments are sampled without replacement from the official training task-plan pool using base seed 42. The two ablation variants

use the same selected environments every shared task. Episodes have a 200-control-step limit, with automatic reset and termination ignoring disabled.

`success_at_end` is the official success flag at the first episode termination or truncation. The official environment terminates on success or failure in this configuration. Success is determined by the task's official compound predicate. Figure 4 shows the main camera view and wrist camera view of GroupACT executing the seven tasks, with frames sampled from selected episodes at 0%, 25%, 50%, 75%, and 100% of their duration. We evaluate GroupACT on MS-HAB using a host equipped with an NVIDIA RTX 5090 GPU.

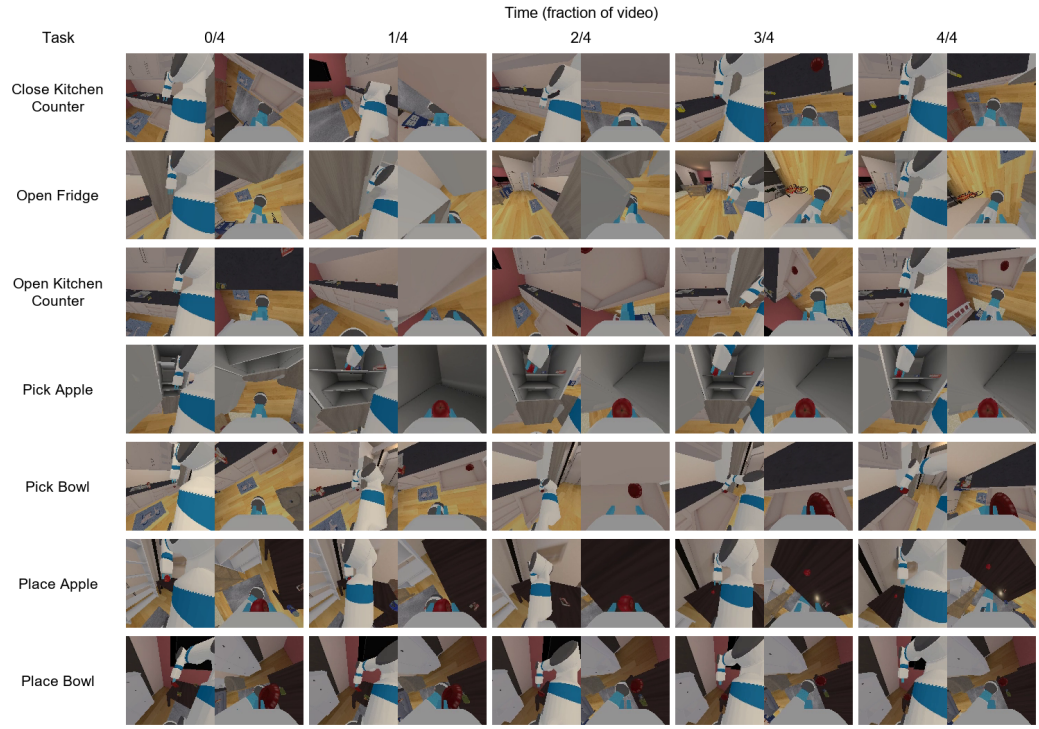

Figure 4: Dual camera view of seven tasks in MS-HAB SetTable.

D.3 Real-world Task Evaluation

Our RealMan dual-arm platform consists of two six-DoF arms, each equipped with a gripper. A RealSense D435 camera is mounted on each arm's wrist, and a third is mounted on the robot head. Although these cameras support depth sensing, we use only their RGB images. During evaluation, GroupACT runs on a local host equipped with an NVIDIA RTX 5090 GPU, with an inference latency of approximately 113 ms. The host connects directly to the robot controller via Ethernet and sends control commands. The controller transmits robot states and images from all three cameras to the host through Robot Operating System (ROS) interface. We randomize the positions of the objects to be grasped during both demonstration collection and evaluation. GroupACT and the $\pi_{0.5}$ baseline use the same task-specific training data. We use the following language instructions for the two tasks. For Place cup, the instruction is: Open the lid of the white box, pick up the water cup and place it inside the white box, then close the lid of the white box. For Place grey box, the instruction is: Open the lid of the white box, pick up the grey box and place it inside the white box, then close the lid of the white box. Figure 5 shows images from all three camera views during the execution of each task.

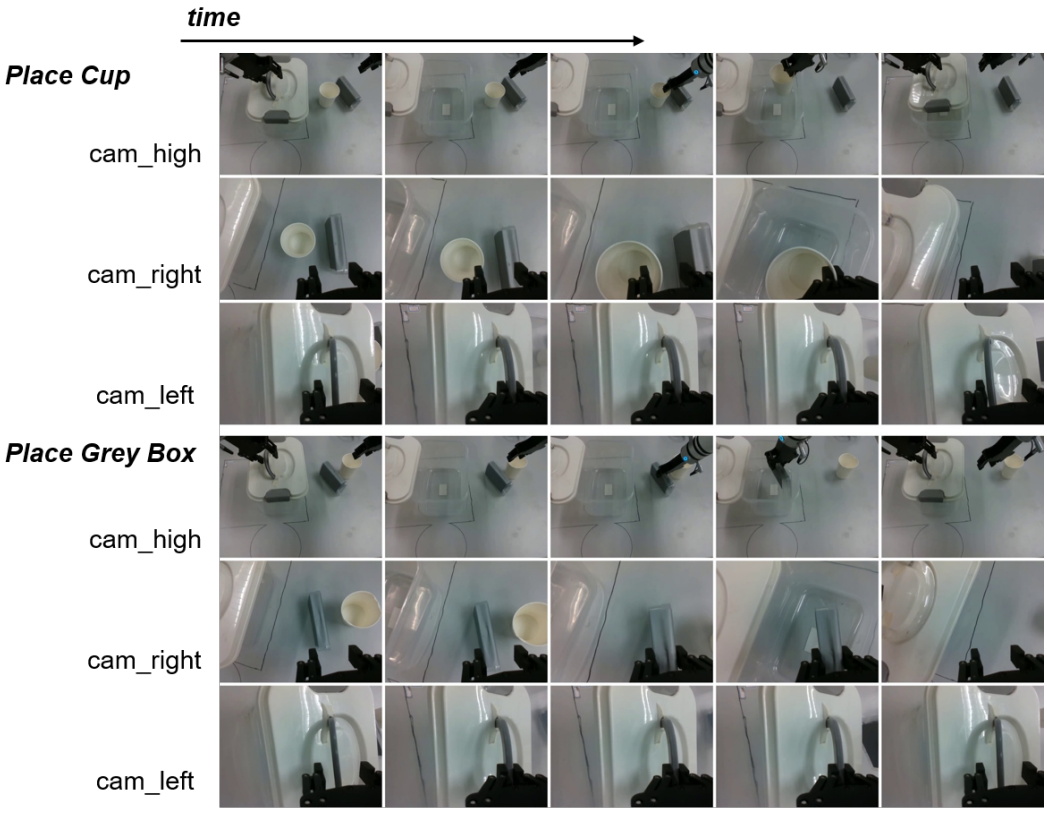

Figure 5: Three camera view of two real-world tasks.