PhysGraph: Physically-Grounded Graph-Transformer Policies for Bimanual Dexterous Hand–Tool–Object Manipulation

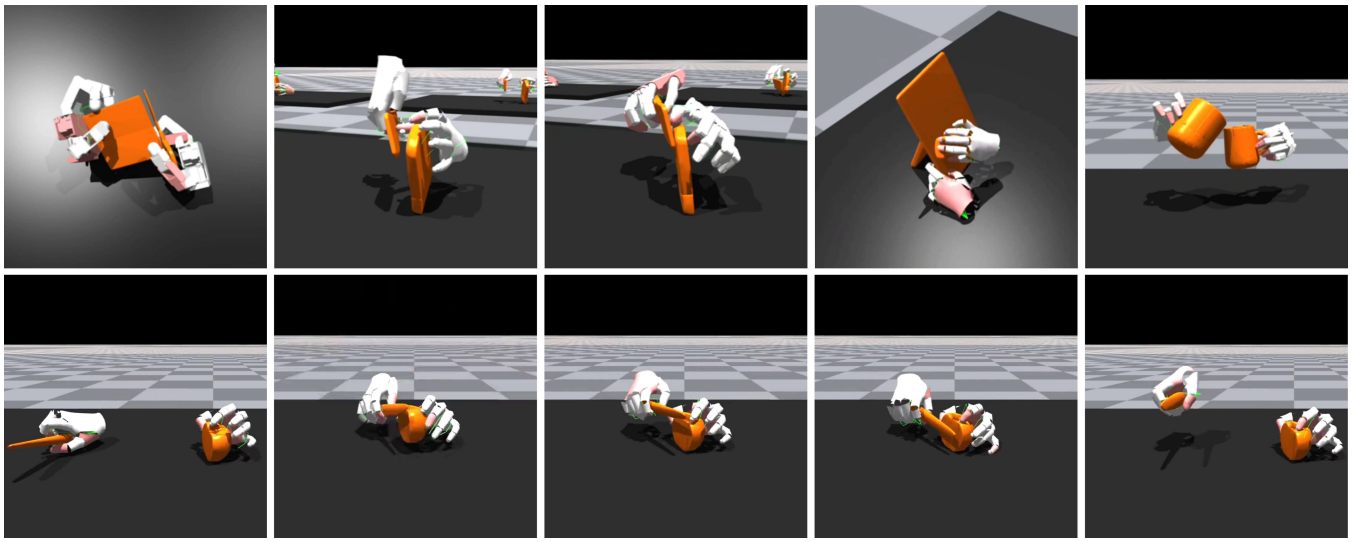


Fig. 1: ***PhysGraph*** is a physically-grounded graph transformer policy for bimanual hand-tool-object manipulation. **Top row**: PhysGraph policy trained and tested on diverse bimanual tool-use tasks with different robot hands (Allegro, ArtiMano, Shadow). **Bottom row**: Zero-shot policy transfer to unseen tool and object for similar tasks, e.g., trained on "slicing bread with chop knife" and tested on "cutting apple with fruit knife". Our code and additional results are available at: https://scientistrocknroll.github.io/PhysGraph_anonymous_website/

***Abstract*— Bimanual dexterous manipulation for tool use remains a formidable challenge in robotics due to the high-dimensional state space and complicated contact dynamics. Existing methods naively represent the entire system state as a single configuration vector, disregarding the rich structural and topological information inherent to articulated hands. We present PhysGraph, a physically-grounded graph transformer policy designed explicitly for challenging bimanual hand-tool-object manipulation. Unlike prior works, we represent the bimanual system as a kinematic graph and introduce per-link tokenization to preserve fine-grained local state information. We propose a physically-grounded bias generator that injects structural priors directly into the attention mechanism, including kinematic spatial distance, dynamic contact states, geometric proximity, and anatomical properties. This allows the policy to explicitly reason about physical interactions rather than learning them implicitly from sparse rewards. Extensive experiments show that PhysGraph significantly outperforms baseline - ManipTrans in manipulation precision and task success rates while using only 51% of the parameters of ManipTrans. Furthermore, the inherent topological flexibility of our architecture shows qualitative zero-shot transfer to unseen tool/object geometries, and is sufficiently general to be trained on three robotic hands (Shadow, Allegro, Inspire).**

I. INTRODUCTION

Bimanual dexterous manipulation is a long-standing goal in robotics aiming to enable robots to perform human-level multi-contact coordinated manipulation. Although recent learning-based approaches have made substantial progress in dexterous manipulation, bimanual tool-use remains particularly challenging because the robot must maintain stable grasps, regulate contact forces, and coordinate both hands while the tool mediates interaction between the hands and objects. Multi-fingered hands offer superior dexterity and robustness to variations in object geometry, compared to parallel-jaw grippers, but they come with high-dimensional state and action spaces and complex kinematic topology. For this reason, it remains difficult to learn reliable bimanual tool-use policies in practice.

In this paper, we postulate that a key bottleneck is related to how the control policy represents and reasons about the kinematic structure of the hand, tool, and object. Many recent dexterous manipulation policies treat the hand-tool-object states as a flattened single vector [1]–[4] fed into a multi-layer perceptron (MLP) to implicitly discover (i) the kinematic dependencies within each hand, (ii) cross-hand coordination patterns, and (iii) time-varying interactions among hand, tool, and object. The model must implicitly learn that the "index fingertip" is kinematically adjacent to the "index knuckle" from scratch. However, bimanual tool-use is inherently structured: the system is naturally described by a graph whose nodes (links) and edges (joints/contact) define feasible pathways for information flow and physical influence. Ignoring this structure forces the policy to learn

physical reasoning from sparse reward signals, often leading to brittle and physically implausible behaviors in challenging contact-rich tasks.

To address this, we propose PhysGraph, a physically-grounded graph-transformer policy for bimanual hand-tool-object manipulation, illustrated in Figure 1. Rather than flattening the state, we formulate a graph structure, where nodes represent individual rigid bodies (links, tools, objects) and edges represent physical couplings (joints, contact). Our approach introduces two key innovations: (i) a per-link tokenization strategy and (ii) a physically-grounded head-specific transformer bias generator. First, instead of pooling states into a global embedding, we process each link's states as a distinct token, preserving fine-grained local properties. Second, unlike generic Graph Transformers (e.g., Graphormer [5]) that utilizes abstract graph distances, we inject a learning-based head-specific composite bias directly into the attention mechanism. The composite bias includes *spatial bias* (kinematic chain distance), *edge bias* (joints and contact), *geometric bias* (positions proximity), and *anatomical priors* (serial/synergistic kinematics). These bias terms enable our policy to explicitly reason about the physical connectivity and contact logic, focusing attention on contacting fingers or coordinated joints. Our main contributions are summarized as follows.

- We propose PhysGraph, the first graph-transformer policy for high-DoF challenging bimanual dexterous tool-use tasks, which explicitly models the hand-tool-object interactions as a link-joint-contact graph and processes per-link tokenized observations.
- Structural priors are injected into the PhysGraph transformer via a novel physically-grounded bias generator, which provides topological, edge, distance, and anatomical bias via head-specific masking.
- Extensive experiments on challenging bimanual tool-use tasks demonstrate that PhysGraph significantly outperforms SOTA baseline - ManipTrans [1] in success rate and motion fidelity while using only ***51%*** of the parameters of ManipTrans, supporting zero-shot transfer to unseen tools/objects, and is compatible with popular robotic dexhands (Shadow [6], Allegro [7], Inspire [8]).

II. RELATED WORK

Bimanual Dexterous Manipulation. Recent works have focused heavily on learning diverse skills from human demonstrations. DexMimicGen [9] and DexMan [4] utilize automated data generation and video-based learning to solve basic bimanual tasks like handover and pick-and-place. Similarly, Bi-DexHands [2] and DexMachina [3] explore functional retargeting to map human hands key-points to robot hands. [10] further improves coordination by prioritizing object state representation. ManipTrans [1] stands out as the SOTA baseline, which employs a residual learning architecture to efficiently transfer human skills to robotic hands. However, these methods typically rely on global states, which often fail to capture the fine-grained dynamics required for the challenging tool-use tasks we address.

Dexterous Tool-Use & Sim-to-Real Challenges. While gripper-based tool use is well-explored, dexterous tool use remains nascent. Concurrent SOTA works have prioritized resolving sim-to-real gaps over task complexity. For instance, [11] successfully demonstrates in-hand manipulation of articulated tools on real hardware. However, to achieve sim-to-real, they simplify to a hand-tool single-hand task without tool-object interaction. Similarly, [12] achieves robust sim-to-real transfer for humanoid bimanual tasks but only focuses on grasp-and-reach behaviors rather than precise tool-object interaction. DexWild [13] and ClutterDexGrasp [14] addressed manipulation in unstructured, cluttered environments using scalable neural control. We take the complementary stance: *Rather than simplifying the task for sim-to-real, we focus on solving the complex bimanual tool-object interactions in simulation first*, such as precise paper shearing, providing a high-fidelity teacher policy. Our PhysGraph tackles the full complexity of high-DoF bimanual tool-object interaction, establishing a high-performance teacher policy that can serve as a foundation for future sim-to-real efforts.
Learning from References. A growing trend is to leverage human video data to synthesize physically plausible motions for humanoids and hands. SkillMimic [15] recently demonstrated the ability to learn highly dynamic skills - basketball from demonstrations. Similarly, TokenHSI [16] introduced a tokenized approach for unifying human-scene interactions, inspiring our per-link tokenization strategy. While full-body learning demonstrated that humanoid agents can learn diverse interactions often without explicit reference state inputs to the policy, this "reference-free" exploration is insufficient for dexterous hands due to the extremely higher complexity of finger-object contact dynamics. As highlighted by DexMan [4], the high-dimensional action space and dexterity of multi-fingered hands necessitates stronger guidance. Consequently, SOTA dexterous policies, including DexTrack [17], DexMachina [3], OmniGrasp [18], and ManipTrans [1], adopt reference-conditioning for tractable learning. PhysGraph aligns with this established paradigm by utilizing reference inputs to guide exploration, but distinguishes itself by explicitly modeling the dynamic topology of bimanual tool use through a topology-aware graph architecture rather than a flat vector focusing on single-agent interaction.
Structured and Graph-Based Policy Learning. To handle the complexity of high-DoF systems, there is a growing interest toward morphology-aware architectures. The use of Graph Neural Networks for robot control traces back to NerveNet [19], which explicitly embedded the robot's kinematic graph into the policy network to create morphology-aware controllers. Graphormer [5] later demonstrated the power of attention mechanisms for graph representation. GCNT [20] and GET-Zero [21] recently proposed Graph-Transformer policies to handle variable robot morphologies, proving that graph structures can generalize across embodiments. However, these methods defined their graph edges limited to static and unchangeable locomotion or kinematics, such as the static bones/limbs [20], [21] and chemical bonds [5]. PhysGraph builds upon this "Graph Story" but

evolves it for manipulation. We introduce dynamic contact edges and anatomical biases specifically for tool use. While these methods uses static graphs for morphology, PhysGraph dynamically rewires the graph topology based on contact states, allowing the policy to explicitly reason about the tool as a transient extension of the hand's kinematic chain.

III. Problem Statement

We study reference-conditioned dexterous hand-tool-object manipulation under contact-rich dynamics. At each timestep (t), the policy observes a state (s_t) containing robot proprioception and interaction-related signals together with a reference motion state (s_t^{ref}) derived from human demonstrations. Following prior dexterous manipulation methods, the policy outputs an action

$$a_t \sim \pi_\theta(\cdot \mid s_t) \tag{1}$$

and is trained using a reward combining task completion and reference-tracking objectives. The reference trajectory provides guidance for learning high-dimensional contact-rich manipulation, while the policy determines physically feasible actions under simulated dynamics. Our contribution focuses on the architecture used to parameterize (π_θ): rather than flattening the manipulation state, PhysGraph explicitly represents its kinematic and interaction structure.

IV. PhysGraph Architecture

Figure 2 summarizes PhysGraph. PhysGraph is a graph transformer policy that exploits the structure of hand-tool-object interaction. We design four novel physically-grounded composite biases terms that capture the kinematic, geometric and anatomical priors.

A. Kinematic Graph Transformer

Kinematic Graph Construction. We design a policy architecture that models the physical structure of bimanual manipulation as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, whose nodes $\mathcal{V}$ correspond to rigid bodies, such as hand links, tool and object, and edges $\mathcal{E}$ correspond to kinematic (joints) or dynamic (contact) interactions.

Tokenization and Transformer Backbone. We propose a novel per-link tokenization approach to preserve the local structure of interaction: contact occurs at hand links, stability depends on distal joints, and error correction propagates along kinematic chains. Representing rigid-body links as separate tokens enables a transformer to learn structured message passing, e.g., finger links $\leftrightarrow$ tool $\leftrightarrow$ object. Let $\{x_i\}_{i=1}^N$ denote per-token inputs extracted from the structured per-link state s_t, where tokens correspond to right-hand state, right-hand per-link states, left-hand state, left-hand per-link states, tool state, and object state. Each per-token input is embedded into a d-dimensional space using a tokenizer f_i followed by layer normalization:

$$h_i^{(0)} = \text{LN}(f_i(x_i)) \in \mathbb{R}^d, \qquad i = 1, \ldots, N. \tag{2}$$

We prepend a learnable policy token:

$$h_0^{(0)} = \mathbf{e}_{\text{POL}} \in \mathbb{R}^d, \tag{3}$$

and form the token matrix:

$$H^{(0)} = [h_0^{(0)}, h_1^{(0)}, \ldots, h_N^{(0)}]^\top \in \mathbb{R}^{(N+1)\times d}. \tag{4}$$

We employ an L-layer transformer encoder that maps $H^{(\ell-1)} \mapsto H^{(\ell)}$ for $\ell = 1, \ldots, L$. For each layer ℓ and attention head h, the attention logits are computed as:

$$A^{(\ell,h)} = \frac{Q^{(\ell,h)} K^{(\ell,h)^\top}}{\sqrt{d_h}} + B^{(h)}, \tag{5}$$

where, $Q^{(\ell,h)}$, $K^{(\ell,h)} \in \mathbb{R}^{(N+1)\times d_h}$ are the query and key matrices, and $B^{(h)} \in \mathbb{R}^{(N+1)\times(N+1)}$ is a *head-specific composite bias*. Attention weights and outputs follow standard transformers [22]. This design provides a useful separation: in (5), the content-based dot-product term captures general task-dependent relevance, while $B^{(h)}$ encodes task-agnostic physically-grounded plausibility, as we detail next.

B. Physically-Grounded Bias

We define the composite bias for each head as a weighted sum of four physically meaningful components:

$$\begin{aligned} B^{(h)} = \lambda_{\text{sp}} B_{\text{sp}}^{(h)} + \lambda_{\text{edge}} B_{\text{edge}}^{(h)} + \lambda_{\text{geo}} B_{\text{geo}}^{(h)} \\ + \lambda_{\text{anat}}^{(h)} B_{\text{anat}}^{(h)}. \end{aligned} \tag{6}$$

Figure 3 illustrates these four biases: spatial ($B_{\text{sp}}^{(h)}$), edge ($B_{\text{edge}}^{(h)}$), geometric ($B_{\text{geo}}^{(h)}$), and anatomical ($B_{\text{anat}}^{(h)}$). Throughout, biases are generated in *node space* on the physical graph $\mathcal{G}$, then lifted to *token space* via a fixed mapping to match the token sequence. All coefficients λ are learnable, so the network can adapt the reliance on each bias.

Spatial Bias (Kinematic Graph Distance). Kinematic influence is primarily local along joint chains: distal motion depends strongly on proximal joints, and control corrections often propagate through neighboring links. In sparse-reward RL, asking attention to discover this structure purely from data is inefficient. Spatial bias injects a prior that encourages communication between nodes that are close in the hand kinematic graph.

Let $d(u, v)$ be the shortest graph hop distance between nodes u and v on $\mathcal{G}$. The spatial bias is defined as:

$$B_{\text{sp}}^{(h)}(u, v) = b_{\text{sp}}^{(h)}(\min(d(u, v), D_{\max})), \tag{7}$$

where $b_{\text{sp}}^{(h)}(k) \in \mathbb{R}, k \in \{0, 1, \ldots, D_{\max}\}$ is a head-specific embedding over discrete hop distances, and $D_{\max}$ is a limit for clipping. This effectively provides a learnable "graph positional bias" that reflects the kinematic topology. Clipping the distances to D_{max} stabilizes training and prevents the bias vocabulary from growing with graph diameter.

Edge Bias (Joints + Contact). Not all node pairs at the same distance are equally important. Some relationships are special: self edges preserve identity, bone adjacency captures rigid kinematic coupling, and contact edges capture instantaneous physical constraints through which forces and torques transmit. Edge bias encodes these semantic edge types directly into attention.

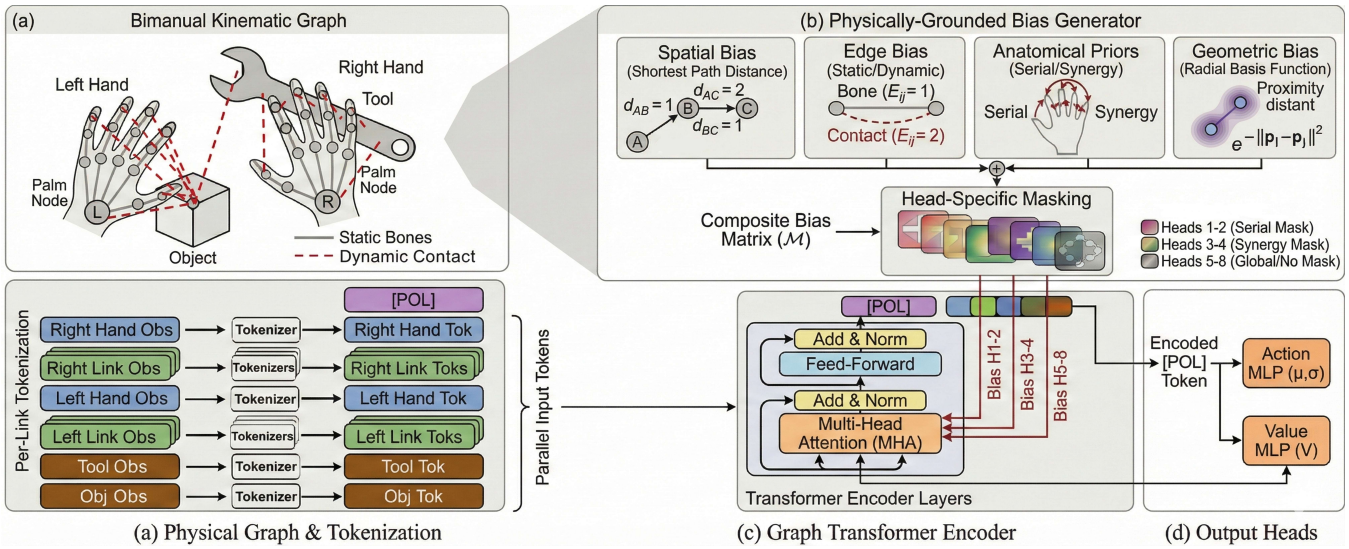


Fig. 2: **Overview of PhysGraph: (a) Physical Graph & Tokenization:** The bimanual workspace is modeled as a graph where nodes represent links of the left/right hands, tools, and objects and edges represent kinematic (joints) or dynamic (contact) interactions. State-based multi-modal observations for each link are processed into parallel input tokens. **(b) Physically-Grounded Bias Generator:** This module computes four distinct biases (detailed in Fig. 3), aggregated into a composite bias matrix $\mathcal{M}$. The biases are applied via head-specific masking, allowing different attention heads to focus on specific physical relationships. **(c) Graph Transformer Encoder:** The tokenized inputs are processed by a transformer encoder where the Multi-Head Attention (MHA) is modulated by the generated bias in (b). **(d) Output Heads:** The globally encoded $[POL]$ token is passed to MLP heads to predict the policy action distribution (μ, σ) and value function (V).

We define discrete edge types $\tau(u, v)$ such that: $\tau(u, v) \in \{0, 1, 2, 3\}$, representing (0) disconnected, (1) bone adjacency, (2) contact and (3) self. Concretely, using a binary contact indicator $C_t(u, v)$:

$$\tau(u,v) = \begin{cases} 3 & u = v, \text{self} \\ 2 & C_t(u,v) = 1, \text{contact} \\ 1 & (u,v) \in \mathcal{E}_{\text{bone}}, \text{adjacency} \\ 0 & \text{disconnected.} \end{cases} \tag{8}$$

Using the defined edge type, we define edge bias as:

$$B^{(h)}_{\text{edge}}(u,v) = b^{(h)}_{\text{edge}}(\tau(u,v)), \tag{9}$$

where $b^{(h)}_{\text{edge}}(c) \in \mathbb{R}, c \in \{0, 1, 2, 3\}$ is the head-specific edge embedding. For efficiency, static edge indices can be precomputed once and stored in a buffer, and dynamic contacts can be added as a sparse subset at runtime. This avoids constructing a full $N \times N$ adjacency matrix for the edges, which is important for high throughput.

During tool-use, contact is the dominant coupling mechanism: Once fingers/palm touch the tool, control depends on integrating local body states, tool/object motion, and response. Our proposed edge bias makes this pathway systematically accessible to attention.

Geometric Bias (Position Proximity). Kinematic graph topology alone cannot model interaction locality. Two fingertips from different fingers (or different hands) can become spatially adjacent during grasping even if they are far in the kinematic graph. Similarly, tool–object and hand–tool interactions depend on Euclidean proximity. We propose a geometric bias term that encourages attention to focus on spatially local pairs as a differentiable proxy for physical interaction likelihood.

Let $p_u \in \mathbb{R}^3$ be the position of node u. We compute an RBF kernel:

$$\kappa(u,v) = \exp\left(-\frac{\|p_u - p_v\|_2^2}{2\sigma^2}\right), \tag{10}$$

with learnable bandwidth σ. We project this proximity scalar into a head-specific bias with learned weight $w^{(h)}_{\text{geo}}$ as:

$$B^{(h)}_{\text{geo}}(u,v) = \mathbb{I}(d(u,v) > d_0)\, w^{(h)}_{\text{geo}}\, \kappa(u,v). \tag{11}$$

The first part of the expression (i.e., $d(u, v) > d_0$) gates geometric bias to pairs that are topologically distant to prevent redundancy with topological bias.

Rigid-body interactions are fundamentally driven by proximity. The RBF term offers a smooth inductive bias: nearby nodes in Cartesian position, regardless of a far kinematics distance, should exchange information, improving responsiveness to emerging contacts and bimanual coordination around the tool and the object.

Anatomical Priors (Serial + Synergy). Hands have anatomical structure beyond generic graphs. Each finger forms a kinematic serial chain, while fingers exhibit synergies at the same anatomical level, e.g., coordinated closing/opening and correlated motion of the MCP/PIP joints. Thus, a single attention head forced to represent all patterns can be inefficient. We instead specialize heads: some focus on serial kinematics,

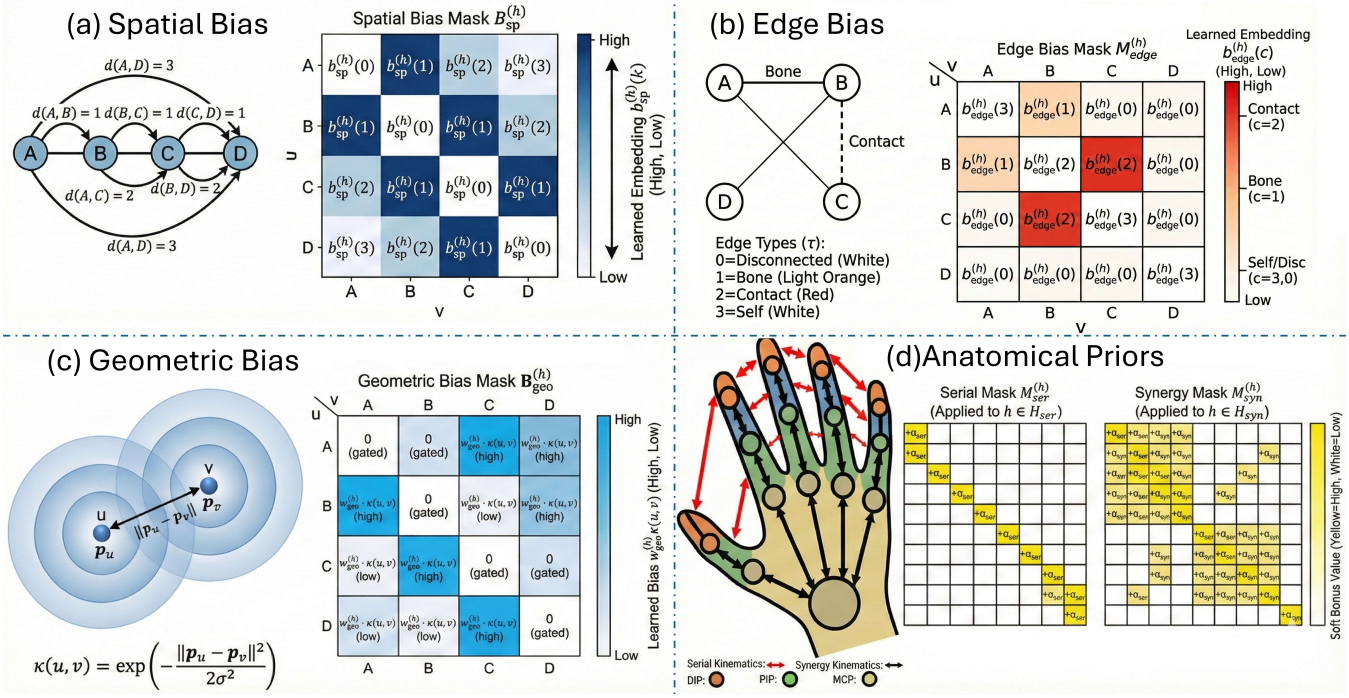


Fig. 3: **Details of the Physically-Grounded Biases. (a) Spatial Bias:** Encodes the topological structure of the hand graph by mapping the shortest path distance $d(u, v)$ between nodes to learned scalar embeddings $b_{sp}^{(h)}$, capturing the "hop" distance in the kinematic chain. **(b) Edge Bias:** Injects structural information by assigning distinct learned embeddings $b_{edge}^{(h)}$ indexed by the edge type τ connecting two nodes. **(c) Geometric Bias:** Incorporates spatial proximity in Cartesian space. It utilizes a Radial Basis Function (RBF) kernel $\kappa(u, v)$ to weight attention based on the Euclidean distance $\|\mathbf{p}_u - \mathbf{p}_v\|^2$ between node positions. **(d) Anatomical Priors:** Encodes knowledge regarding anatomical hand kinematics. The Serial Mask (M_{ser}) highlights dependencies along the kinematic chain of a single finger, while the Synergy Mask (M_{syn}) promotes attention between corresponding links across different fingers with the same anatomical levels.

some on synergy couplings, and others remain on global reasoning.

We define two binary relation masks in node space: (i) Serial validity $S(u, v)$: True for near-kinematic-neighbor interactions (e.g., $d(u, v) \leq 1$). (ii) Synergy validity $Y(u, v)$: True for anatomically corresponding groups (e.g., fingertip-to-fingertip across fingers; MCP-to-MCP; PIP-to-PIP) within each hand.

We allocate head subsets: $\mathcal{H}_{\text{ser}}$, $\mathcal{H}_{\text{syn}}$, $\mathcal{H}_{\text{glob}}$. Instead of hard masking ($-\infty$), we add soft bonuses that encourage—but do not force—these patterns:

$$B_{\text{anat}}^{(h)}(u, v) = \begin{cases} \alpha_{\text{ser}}\, S(u, v) & h \in \mathcal{H}_{\text{ser}}, \\ \alpha_{\text{syn}}\, Y(u, v) & h \in \mathcal{H}_{\text{syn}}, \\ 0 & h \in \mathcal{H}_{\text{glob}}. \end{cases} \tag{12}$$

While the hard constraints can harm dexterity, soft bonuses provide an inductive bias while preserving model flexibility.

V. EXPERIMENTS

We evaluate PhysGraph from four perspectives. In Sec. V-A, we compare against the SOTA baseline on challenging contact-rich bimanual tool-use tasks and ablate the proposed physically-grounded biases. Sec. V-B evaluates zero-shot generalization to unseen tool/object instances, while Sec. V-C examines compatibility across different dexterous hand embodiments. Finally, Sec. V-D validates PhysGraph on a physical xArm-Allegro platform.

Datasets. For training and evaluation, we utilize the OakInk2 dataset [23]. Bimanual manipulation is inherently more complex than single-hand tasks because it requires elaborate coordination between the dominant (tool-using) and non-dominant (object-stabilizing) hands in addition to complicated contact sequences between hand-tool, tool-object, and hand-object. We consider six highly challenging tasks that require sustained dynamic contacts and fine-grained force application. These tasks are described in Table I.

Metrics. We evaluate task success and motion fidelity using: (i) tool/object translation and rotation errors (E_t) and (E_r) relative to the expert trajectory; (ii) mean hand-joint error (E_j); (iii) mean fingertip error (E_{ft}), which measures contact-sensitive fingertip tracking; and (iv) Success Rate (SR), the percentage of trials satisfying the task-specific tracking thresholds.

Baselines. We benchmark PhysGraph against ManipTrans [1]. We leverage ManipTrans as the only baseline because, to the best of our knowledge, ManipTrans achieved the **best performance** on bimanual dexterous hands manipulation,

TABLE I: **Quantitative results on bimanual tool-use tasks of Oakink2 dataset [23].** SR: Success Rate; E_t: Tool & Object Translation Error; E_j: Hand Joint Error; E_ft: Fingertip Error. ArtiMANO Hand. Pink indicates the best performance.

Task ID	Description	Tool (R) / Object (L)	Methods	SR (%) ↑	E_t (cm) ↓	E_j (cm) ↓	E_ft (cm) ↓
083f7@0	Cut bread with knife	Knife / bread	Maniptrans	55.69	1.31	2.42	2.01
			PhysGraph (No Bias)	82.84	1.02	2.26	1.92
			PhysGraph (full)	90.05	0.69	2.17	1.42
97055@0	Scoop into bowl	Spoon / bowl	Maniptrans	74.99	1.76	3.10	2.16
			PhysGraph (No Bias)	82.58	1.58	2.70	2.48
			PhysGraph (full)	86.95	1.40	2.38	2.24
1292e@0	Pour mug to mug	mug / mug	Maniptrans	45.60	6.78	3.50	3.35
			PhysGraph (No Bias)	46.58	6.21	3.65	4.07
			PhysGraph (full)	49.77	5.93	3.07	4.33
817fb@0	Brush whiteboard	Brush / whiteboard	Maniptrans	50.05	1.55	2.17	2.24
			PhysGraph (No Bias)	58.84	1.60	1.70	1.94
			PhysGraph (full)	62.24	1.33	1.54	1.86
e1fa6@0	Shear paper	Scissors / paper	Maniptrans	15.16	1.18	3.31	2.25
			PhysGraph (No Bias)	27.16	1.43	2.92	1.97
			PhysGraph (full)	35.84	1.35	2.52	1.83
9b728@0	Spread on bread	Spoon / bread	Maniptrans	47.22	1.05	3.08	1.76
			PhysGraph (No Bias)	61.69	1.01	3.20	1.86
			PhysGraph (full)	86.64	0.98	3.17	1.52

using an MLP-based residual learning architecture that concatenates all observations into a single state vector.

Implementation Details. We use eight transformer heads, where two heads have the serial mask ($\mathcal{H}_{\text{ser}} \in \{1, 2\}$), two have the synergy mask ($\mathcal{H}_{\text{syn}} \in \{3, 4\}$), and the rest have the global mask ($\mathcal{H}_{\text{glob}} \in \{5, 6, 7, 8\}$). We train the policy using the PPO [24] algorithm, and use a PD controller for the forward dynamics. To ensure a strictly fair comparison of the policy architecture, all environmental configurations, reward functions and physical dynamics are kept *identical* to our baseline ManipTrans. All experiments run in the Isaac Gym [25], utilizing 4096 parallel environments per task at a time step of 1/60 s with an NVIDIA RTX 3090 GPU. For fairness in convergence comparison, all results are extracted from the best-performing checkpoint within a strict maximum of 8k training epochs, which is far more than sufficient to ensure convergence for all methods.

A. Evaluation on Bimanual Tool-Use

Quantitative results. Table I summarizes the evaluation. PhysGraph (Full) achieves consistently higher success rates than ManipTrans across all tasks, with particularly large gains on highly contact-sensitive behaviors such as cutting and spreading. PhysGraph also shows higher fidelity metrics in most settings, indicating more stable interaction trajectories and finer control around contact transitions. Importantly, the number of parameters for the ManipTrans policy for ArtiMANO is ***3785187***, whereas the counterpart of PhysGraph is ***1950792***. This means PhysGraph exhibits superior performance with much fewer model parameters ($1950792/3785187 = \mathbf{51\%}$), which strongly manifests PhysGraph's efficacy and efficiency.

Qualitative results. Figure 4 visualizes the rollouts. Compared to ManipTrans, PhysGraph exhibits more reliable bimanual coordination: ManipTrans frequently exhibits unnatural hand motions, leading to dropped tools (knife, scissors, mug), or misaligned interactions due to implausible force regulation (brush pushes the whiteboard to topple). PhysGraph produces highly natural, closed grasps, with physically plausible force maintaining tight contact throughout the tasks. ***Please check the corresponding videos at our website and the supplementary material for more details***.

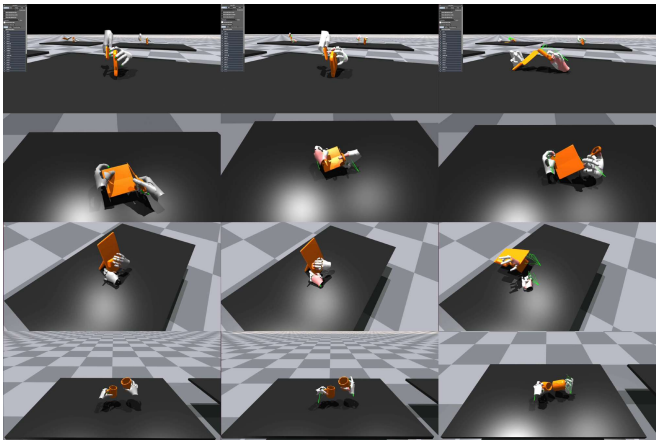


Fig. 4: **Qualitative results on bimanual tool-use tasks of Oakink2 dataset.** Left to right: Ground truth, PhysGraph (ours), ManipTrans. Top to bottom: 0837f@0, e1fa6@0, 817fb@0, 1292e@0. Please refer to the corresponding videos on our website.

Ablation on Physically Grounded Biases. To validate the contribution of our architectural design, we ablate the physically-grounded bias component in Table I. Notably, even without explicit physical biases, PhysGraph (No Bias) significantly outperforms the MLP-based ManipTrans across all tasks. This empirically validates the importance of our per-link tokenization strategy: unlike global MLPs that flatten structural information, representing the bimanual system as discrete link tokens preserves local geometric fidelity, enabling superior feature learning even with a standard transformer backbone. However, removing the biases causes

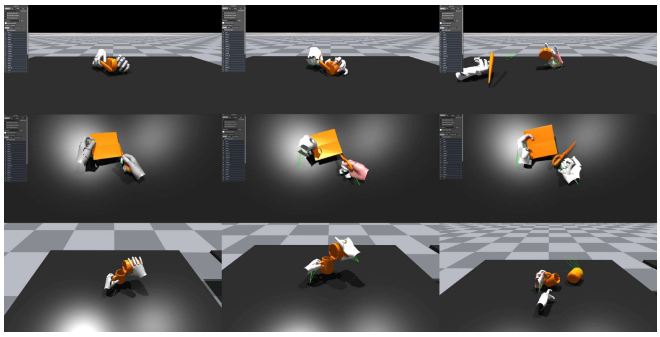


Fig. 5: **Zero-Shot Policy Transfer** *without finetuning*. Left to right: Ground truth, PhysGraph (ours), ManipTrans. Top to bottom (sequence trained → sequence deployed): 0837f@0 → 9fc3e@0 (chop knife → fruit knife; bread → apple), e1fa6@0 → 66c7f@0, 1292e@0 → b9695@0.

a sharp decline in all evaluation metrics. The no-bias variant struggles to maintain dynamic tool-use kinematics, forcing the attention mechanism to discover underlying kinematic structure purely from data, which is highly sample-inefficient. This shows that the proposed physical bias terms provide critical guidance, allowing the heads to communicate along valid physical pathways immediately. The experimental results support our initial motivation: explicitly encoding kinematic topology, edge semantics, geometric proximity, and anatomical insights provides effective inductive biases for contact-rich bimanual tool-use manipulation.

B. Zero-Shot Policy Transfer

We evaluate PhysGraph's capacity for zero-shot transfer by deploying frozen policies on unseen tool/object for different task instances ***without finetuning***. As shown in Fig. 5, PhysGraph is robust to instance changes, preserving correct interaction structure with more reliable task completion. As Fig. 5 shows, a policy trained exclusively to handle a large chop knife and a rectangular bread seamlessly transfers to a much smaller fruit knife and a spherical apple. This suggests that our physically-grounded bias is inherently spatially invariant; because we explicitly encode relative spatial proximity rather than relying on absolute state coordinates, the model achieves superior generalization beyond the specific instance geometries encountered during training.

C. Embodiment Compatibility Validation

We demonstrate that PhysGraph architecture is compatible with various robotic hand topologies. We train and evaluate PhysGraph across multiple dexterous hand embodiments (Shadow [6], Inspire [8], Allegro [7]). Table II shows that PhysGraph consistently improves success rate and interaction fidelity across different embodiments compared to the baseline. Since retargeting from MANO introduces additional mismatch for non-MANO embodiments, these results also indicate robustness to realistic trajectory noise and embodiment-specific discrepancies.

Fingertip Error (E_{ft}) Anomaly. In Tables I and II, we observe that on certain tasks (e.g., pour bottle and shear

TABLE II: **Quantitative Results of Cross Embodiments.** Pink indicates the best performance.

Task ID	Hand	Method	SR ↑	E_t ↓	E_j ↓	E_ft ↓
083f7@0	Inspire	ManipTrans	45.07	2.02	3.83	2.92
		PhysGraph	62.14	1.62	2.67	2.3
	Shadow	ManipTrans	22.34	1.58	3.54	2.67
		PhysGraph	54.77	1.25	3.03	2.26
	Allegro	ManipTrans	19.67	3.60	4.56	3.17
		PhysGraph	29.34	2.95	4.30	2.51
97055@0	Inspire	ManipTrans	52.95	2.16	4.20	2.91
		PhysGraph	78.45	1.17	3.24	1.97
	Shadow	ManipTrans	18.37	2.52	3.29	2.71
		PhysGraph	32.44	2.26	2.04	2.84
	Allegro	ManipTrans	23.37	3.51	4.86	3.25
		PhysGraph	36.28	2.85	4.08	2.41

paper), PhysGraph yields a higher fingertip error E_{ft} than ManipTrans. This is due to a limitation in the ground truth demonstration data rather than policy failure. We used [26] to extract expert trajectories following ManipTrans [1], aligning human keypoints to the MANO model and retargeting them to the robotic embodiments. However, this retargeting process inherently introduces kinematic noise and a loss of contact fidelity, frequently resulting in non-physical inter-penetrations between the robotic fingertips and the tool/object. ManipTrans blindly overfits to these erroneous target poses to minimize E_{ft} due to a lack of physical understanding. On the other hand, PhysGraph's physically-grounded biases penalize non-physical states, causing the policy to deviate from the noisy ground-truth keypoints to discover physically plausible actions. As shown in the shearing task (Fig. 4, 2nd row): the retargeted demonstration shows the thumb holding the scissor handles from the outside, which is implausible for human, However, PhysGraph successfully discovers a stable grasp where the thumb correctly inserts through the handle like human. By deviating from the flawed demonstration to respect physical constraints, PhysGraph achieves a higher SR despite the drop in E_{ft}.

D. Real-Robot Validation

To evaluate whether PhysGraph-generated tool-use behaviors can be realized on physical hardware, we deploy a reduced single-hand instance of PhysGraph on an xArm manipulator equipped with an Allegro dexterous hand. The original PhysGraph formulation models the dominant (tool-using) and non-dominant (object-stabilizing) hands together with the tool and object. Since our current hardware platform contains only one arm-hand pair, we remove the non-dominant-hand nodes and their associated edges, while retaining the dominant hand, tool, and object in the physical graph. In particular, the cup remains explicitly represented as the manipulated object although it is no longer stabilized by a second hand. The same per-link representation and physically-grounded bias formulation are then applied to this

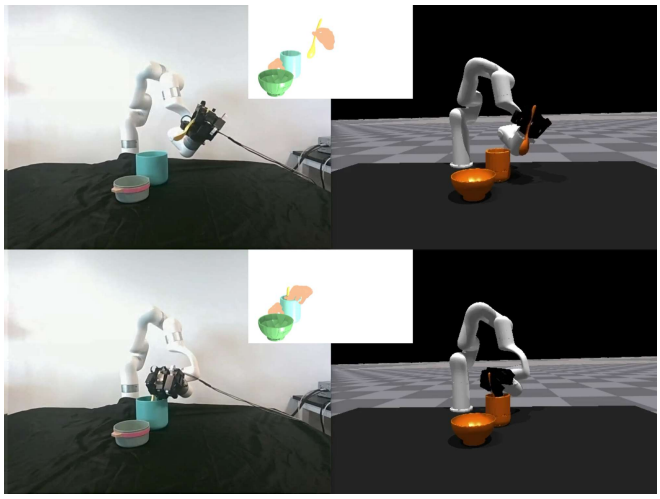


Fig. 6: Real-robot validation on task b2063. Synchronized views show the xArm-Allegro hardware execution (left), ground-truth demonstration replay (middle), and the corresponding PhysGraph physics rollout in Isaac Gym (right). Despite the single-hand hardware constraint, the dominant-hand-tool-object interaction preserves the approach, contact, stirring, and withdrawal phases of the reference behavior. **Check the video on the anonymous website**.

reduced topology.

We evaluate task b2063, in which the Allegro hand maintains a grasp on a spoon, reaches toward a cup, establishes tool-object contact, and performs a stirring motion. Fig. 6 compares the synchronized real-robot execution with the ground-truth demonstration and the corresponding physics rollout in Isaac Gym. The physical system reproduces the complete interaction sequence, including approach, spoon insertion, sustained stirring, and withdrawal, while maintaining a stable tool grasp. This experiment provides initial real-robot evidence that PhysGraph can execute contact-rich hand-tool-object behavior under physical robot dynamics, while also demonstrating that its graph-based formulation can be instantiated on a reduced manipulation topology when a full bimanual platform is unavailable.

VI. CONCLUSIONS

We proposed PhysGraph for bimanual dexterous hand–tool–object manipulation. Our results suggest that explicit physical structure is crucial for learning challenging contact-rich skills. More broadly, this work highlights that transformer policies for articulated physical systems can benefit from tokenization and attention mechanisms that better reflect their underlying topological, kinematic, and dynamic structure.

One key limitation of PhysGraph is that the policy requires a reference trajectory at inference time, which we share with other SOTA methods [2]–[4], [10]. Although conditioning the policy on reference stabilizes the learning of high-dimensional control actions with complex contact dynamics, it also constrains deployment when reference trajectories are unavailable. Future work should remove or reduce reliance on reference demonstrations.

REFERENCES

[1] K. Li, P. Li, T. Liu, Y. Li, and S. Huang, "ManipTrans: Efficient dexterous bimanual manipulation transfer via residual learning," in *CVPR*, 2025.

[2] B. Zhou, H. Yuan, Y. Fu, and Z. Lu, "Learning diverse bimanual dexterous manipulation skills from human demonstrations," in *AAAI*, 2026.

[3] Z. Mandi, Y. Hou, D. Fox, Y. Narang, A. Mandlekar, and S. Song, "Dexmachina: Functional retargeting for bimanual dexterous manipulation," 2025. [Online]. Available: https://arxiv.org/abs/2505.24853

[4] J. Hsieh, K.-H. Tu, K.-H. Hung, and T.-W. Ke, "Dexman: Learning bimanual dexterous manipulation from human and generated videos," 2025. [Online]. Available: https://arxiv.org/abs/2510.08475

[5] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T.-Y. Liu, "Do transformers really perform badly for graph representation?" in *NeuRIPS*, 2021.

[6] "ShadowRobot," https://www.shadowrobot.com/dexterous-hand-series, 2005.

[7] "Allegro Hands," https://www.allegrohand.com, 2013.

[8] "Inspire Hands," https://en.inspire-robots.com/product-category/the-dexterous-hands, 2019.

[9] Z. Jiang, Y. Xie, K. Lin, Z. Xu, W. Wan, A. Mandlekar, L. Fan, and Y. Zhu, "Dexmimicgen: Automated data generation for bimanual dexterous manipulation via imitation learning," in *ICRA*, 2025.

[10] Y. Chen, C. Wang, Y. Yang, and K. Liu, "Object-centric dexterous manipulation from human motion data," in *CoRL*, 2024.

[11] S. Atar, D. Huang, F. Richter, and M. Yip, "In-hand manipulation of articulated tools with dexterous robot hands with sim-to-real transfer," 2025. [Online]. Available: https://arxiv.org/abs/2509.23075

[12] T. Lin, K. Sachdev, L. Fan, J. Malik, and Y. Zhu, "Sim-to-real reinforcement learning for vision-based dexterous manipulation on humanoids," in *CoRL*, 2025.

[13] T. Tao, M. K. Srirama, J. J. Liu, K. Shaw, and D. Pathak, "Dexwild: Dexterous human interactions for in-the-wild robot policies," *RSS*, 2025.

[14] Z. Chen, Q. Yan, Y. Chen, T. Wu, J. Zhang, Z. Ding, J. Li, Y. Yang, and H. Dong, "ClutterDexGrasp: A sim-to-real system for general dexterous grasping in cluttered scenes," *CoRL*, 2025.

[15] Y. Wang, Q. Zhao, R. Yu, H. W. Tsui, A. Zeng, J. Lin, Z. Luo, J. Yu, X. Li, Q. Chen, J. Zhang, L. Zhang, and P. Tan, "Skillmimic: Learning basketball interaction skills from demonstrations," in *CVPR*, 2025.

[16] L. Pan, Z. Yang, Z. Dou, W. Wang, B. Huang, B. Dai, T. Komura, and J. Wang, "Tokenhsi: Unified synthesis of physical human-scene interactions through task tokenization," in *CVPR*, 2025, pp. 5379–5391.

[17] X. Liu, J. Adalibieke, Q. Han, Y. Qin, and L. Yi, "DexTrack: Towards generalizable neural tracking control for dexterous manipulation from human references," in *ICLR*, 2025.

[18] Z. Luo, J. Cao, S. Christen, A. Winkler, K. Kitani, and W. Xu, "Omnigrasp: Grasping diverse objects with simulated humanoids," in *NeuRIPS*, 2024.

[19] T. Wang, R. Liao, J. Ba, and S. Fidler, "Nervenet: Learning structured policy with graph neural networks," in *ICLR*, 2018.

[20] Y. Luo, M. Yao, and X. Xiao, "Gcnt: graph-based transformer policies for morphology-agnostic reinforcement learning," in *IJCAI*, 2025.

[21] A. Patel and S. Song, "GET-Zero: Graph embodiment transformer for zero-shot embodiment generalization," in *ICRA*, 2025.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeuRIPS*, 2017.

[23] X. Zhan, L. Yang, Y. Zhao, K. Mao, H. Xu, Z. Lin, K. Li, and C. Lu, "OAKINK2: A dataset of bimanual hands-object manipulation in complex task completion," in *CVPR*, June 2024, pp. 445–456.

[24] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[25] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021.

[26] Y. Qin, W. Yang, B. Huang, K. Van Wyk, H. Su, X. Wang, Y.-W. Chao, and D. Fox, "Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system," in *RSS*, 2023.