∂O_θ: ACTION-DIFFERENTIABLE NEURAL OBJECTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Learned 3D object representations are typically evaluated by reconstruction accuracy. However, accurate geometry does not guarantee that the representation provides useful gradients for optimizing physical interaction. We ask whether a learned object representation can serve directly as a differentiable interface from geometry to action. We introduce ∂O_θ, Action-Differentiable Neural Objects, a structured neural representation that supports diverse articulated, skinned, and deformable object identities. Conditioned on sparse keypoints, ∂O_θ provides continuous occupancy, signed-distance, and surface-normal queries at arbitrary 3D locations. The same neural object is used in two directions: geometry reconstruction given keypints during representation learning, while after training, task gradients could flow backward through frozen geometry to optimize keypoints for specific interaction. ∂O_θachieves 94.2% simulated grasp success and 78.8% OOD real-world success, while representing up to 100 objects with a compact 1.9M-parameter shared model. These results suggest that neural object representations can be designed not only to reconstruct geometry, but also to expose useful derivatives for physical interaction.

1 INTRODUCTION

Robots interact with the physical world through objects, from grasping and manipulation to articulation and deformation. A useful object representation should therefore capture not only what an object looks like, but also the structure needed to act upon it. Traditional representations such as meshes, point clouds, SDFs, FK, LBS, and FEM explicitly encode geometry, structure, or physical state, making them directly useful for collision, contact, and control.

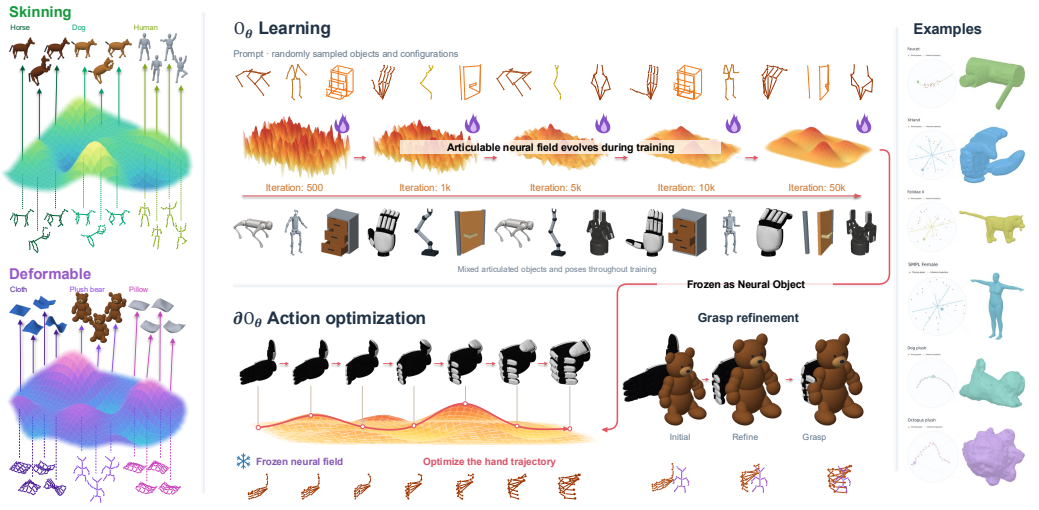

Figure 1: ∂O_θ: A single Neural Object Model represents diverse identities across skinned, deformable, and articulated objects. Once learned, the model weights are frozen and interaction objectives optimize the keypoints through the neural object. On the right, we visualize reconstructed neural objects; dots indicate training samples, while lines illustrate the inference space.

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.

Data-driven methods instead learn geometry from observations. Neural implicit fields, NeRFs, and 3D Gaussian Splatting show that complex geometry and appearance can be represented with learned or optimized representations (Park et al., 2019; Mildenhall et al., 2020; Kerbl et al., 2023). More recent work further models articulated, deformable, and state-conditioned objects; for example, WorldString represents geometry as a neural function of structural keypoints across configurations (Xu et al., 2026a).

However, accurate reconstruction does not necessarily imply usefulness for physical interaction. Reconstruction measures whether geometry can be reproduced, but not whether the learned representation itself provides useful gradients for action. We therefore ask a stronger question: can a learned object representation itself serve as a differentiable model through which configurations and actions are optimized? Differentiable simulation has shown the value of such gradients for grasp optimization (Turpin et al., 2023); here, we ask whether the learned neural object itself can provide this interface.

We introduce ∂O_θ, **Action-Differentiable Neural Objects**. ∂O_θ separates persistent object structure from current configuration and exposes occupancy, signed-distance, and surface-normal queries. Once learned, the neural object is frozen while task gradients propagate through its geometry to action-relevant variables. Thus, representation learning and action optimization use the same model in opposite directions: the former optimizes network parameters given configurations, while the latter fixes the network and optimizes configurations and actions.

This makes ∂O_θ a differentiable interface between representation and action. The same representation supports reconstruction across many objects, direct geometry-to-action optimization, and optimized trajectories for policy learning. Our contributions are threefold:

- *Generalization across many objects.* A compact 1.9M shared model represents 100 rigid, 70 skinned, or 30 soft objects while preserving object-specific geometry.
- *Geometry-to-action optimization.* Beyond reconstruction, ∂O_θ serves as a differentiable engine for dexterous grasp optimization, improving success rate from 67.3% to 94.2% with FastGrasp'd.
- *Real-world actionability.* ∂O_θ-trained policies transfer effectively to the real world, achieving 78.8% OOD success versus 19.2% for ACT, 17.3% for $\pi_{0.5}$, and below 10% for simulator policies.

2 Related Works

Data-Driven 3D Reconstruction. Neural 3D representations span occupancy and signed-distance fields (Mescheder et al., 2019; Park et al., 2019), radiance fields and Gaussian primitives (Mildenhall et al., 2020; Kerbl et al., 2023), and dynamic representations modeling temporal deformation or sparse controls (Pumarola et al., 2021; Wu et al., 2024; Huang et al., 2024). Beyond time conditioning, shape-, pose-, and handle-conditioned methods represent articulated and deformable geometry (Mu et al., 2021; Heppert et al., 2023; Chen et al., 2021; Mihajlovic et al., 2021; Tang et al., 2022). More closely related, WorldString models keypoint-conditioned object geometry (Xu et al., 2026a), NeuROK learns kinematic state spaces for latent simulation (Geng et al., 2026), and neural scene representations support visuomotor control through learned latent dynamics (Li et al., 2022). We achieve multi-object shape conditioning and signed-distance readouts, using the resulting geometric queries for differentiable interaction objectives.

Simulation and Dynamics. Simulation specifies how geometric states evolve under forces and actions. Differentiable physics exposes derivatives through prescribed models and numerical solvers (Hu et al., 2019; 2020; Du et al., 2021), while learned approaches model state transitions using particle interactions, graph networks, latent dynamics, or adaptive-resolution representations (Li et al., 2019; Sanchez-Gonzalez et al., 2020; Li et al., 2020; Wang et al., 2023b). Hybrid methods combine physical solvers with learned constitutive responses (Ma et al., 2023) or reconstructed neural representations (Li et al., 2023; Xie et al., 2024; Jiang et al., 2025; Xu et al., 2026b). These approaches model physical evolution for tasks such as system identification and planning. In contrast, we evaluate a neural geometric model at supplied configurations rather than modeling action-conditioned state transitions, differentiating through geometric queries and kinematic updates but not contact dynamics.

Table 1: Comparison of representative geometry and interaction methods.

Method	Shared Geometry	SDF Output	Skeleton-Free	Neural Interaction Optimization	Policy / Grasp Learning
WorldString (Xu et al., 2026a)	✗	✗	✓	✗	✗
NeuROK (Geng et al., 2026)	✓	✗	✓	✗	✗
A-SDF (Mu et al., 2021)	✓	✓	✓	✗	✗
LEAP (Mihajlovic et al., 2021)	✓	✗	✗	✓	✗
RNDF (Chen et al., 2025)	✗	✓	✗	✓	✗
Dr. Robot (Liu et al., 2025b)	✗	✗	✗	✓	✗
Fast-Grasp'D (Turpin et al., 2023)	–	–	✗	✗	✓
$\partial\mathcal{O}_\theta$ **(Ours)**	✓	✓	✓	✓	✓

Differentiable Optimization and Control. Differentiable hand and robot representations support interaction optimization through neural occupancy, configuration-dependent distance fields, and polynomial distance fields (Karunratanakul et al., 2021; Koptev et al., 2023; Chen et al., 2025; Li et al., 2024). Differentiable rendering further enables action optimization and collision-aware control (Liu et al., 2025b; Ruan et al., 2025). Object SDF functionals and grasp-distance fields provide learned objectives for manipulation planning (Driess et al., 2022; Weng et al., 2023), while grasp synthesis methods optimize contact simulation or force-closure objectives (Turpin et al., 2023; Wang et al., 2023a; Zurbrügg et al., 2025). Differentiable models also support controllers, skill learning, policy optimization, and morphology–control co-design (Amos et al., 2018; Lin et al., 2022; Xu et al., 2022; 2021). These methods differ in what is differentiated and which variables receive task gradients. We instead use frozen, state-conditioned neural object geometry to evaluate contact and penetration objectives, propagating gradients through spatial queries and configuration-dependent features to optimize hand configurations directly or train policies through kinematic rollouts.

3 Method

We represent an actionable object as a *Structured Neural Object*, $\mathcal{O}_\theta$, conditioned on sparse structural keypoints and queried continuously in 3D space. We also extend the representation from a single object to multiple object identities by separating persistent object structure from configuration, and enrich its geometric outputs from occupancy to signed distance and surface normals. We denote the resulting action-differentiable formulation as $\partial\mathcal{O}_\theta$: once learned, $\mathcal{O}_\theta$ is kept frozen, while gradients through its geometric queries are propagated to robot configuration and action variables, producing optimized interaction trajectories that are subsequently used for policy learning.

3.1 Preliminaries: One Model, One Object

Prior Work. We begin with the single-object formulation of WorldString Xu et al. (2026a). Let $\mathbf{K} = \{\mathbf{k}_j\}_{j=1}^N$ denote an ordered set of structural keypoints describing one object configuration, and let $\mathbf{x} \in \mathbb{R}^3$ be a spatial query. We denote the corresponding Structured Neural Object by $\mathcal{O}_\theta$, which predicts occupancy at $\mathbf{x}$ as $p_\theta^{\text{occ}}(\mathbf{x} \mid \mathbf{K}) = \mathcal{O}_\theta(\mathbf{x}; \mathbf{K})$.

WorldString factorizes $\mathcal{O}_\theta$ into three stages. The State Transformer conditions learnable canonical object embeddings $\boldsymbol{\omega}_0$ on the keypoints as $\mathbf{Z}_s = \Phi_s(\boldsymbol{\omega}_0, \mathbf{K})$, and the Object Transformer further integrates these features as $\mathbf{Z}_{\text{obj}} = \Phi_o(\mathbf{Z}_s)$. Finally, the Voxel Transformer decodes geometry at an arbitrary spatial query:

$$\mathcal{O}_\theta(\mathbf{x}; \mathbf{K}) = \Phi_v(\text{Pos}(\mathbf{x}), \mathbf{Z}_{\text{obj}}) .$$

where $\text{Pos}(\mathbf{x})$ denotes the positional encoding of the continuous query coordinate $\mathbf{x}$. The model is trained across different configurations of the same object using sampled spatial queries and occupancy supervision with verfication of learning from both simulation and real data.

Extension to global position. To support global motion, we extend $\mathcal{O}_\theta(\mathbf{x}; \mathbf{K})$ the Structured Neural Object to $\mathcal{O}_\theta(\mathbf{x}; \mathbf{K}, \mathbf{c})$, where $\mathbf{K}$ is the keypoint configuration and $\mathbf{c} \in \mathbb{R}^3$ is the global object position. We canonicalize both the query and keypoints as $\widetilde{\mathbf{x}} = \mathbf{x} - \mathbf{c}$ and $\widetilde{\mathbf{K}} = \mathbf{K} - \mathbf{c}$, while

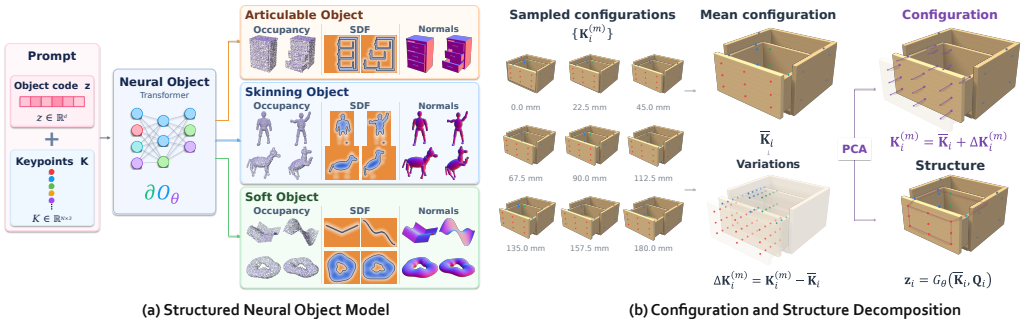

Figure 2: **Structured Neural Object representation and structure–configuration decomposition. (a)** A shared neural object represents articulable, skinned, and soft objects from an object code $\mathbf{z}$ and keypoints $\mathbf{K}$, with occupancy, SDF, and surface-normal outputs. **(b)** PCA separates sampled configurations $\mathbf{K}_i^{(m)}$ into configuration variation and a persistent structural descriptor $\mathbf{z}_i$.

$\mathbf{c}$ is separately encoded as a position token for the State Transformer. During training, objects are observed under varying global translations, allowing the model to separate local geometry from global positions.

3.2 Generalization: One Model, Multiple Objects

Figure 2 summarizes the multi-object representation and the decomposition of structure and configuration. In extending to a generalization form, across objects, geometry depends on both persistent structure and current configuration (e.g. joint positions, keypoints). We condition $\mathcal{O}_\theta$ on a structural descriptor $\mathbf{z}_i$, keypoints $\mathbf{K}_i \in \mathbb{R}^{N_i \times 3}$, and global position $\mathbf{c}_i$:

$$\mathcal{O}_\theta(\mathbf{x}; \mathbf{z}_i, \mathbf{K}_i, \mathbf{c}_i),$$

where $\mathbf{x} \in \mathbb{R}^3$ is a spatial query and θ is shared within each training setting. We denote the keypoints of object i in the m-th observed configuration as $\mathbf{K}_i^{(m)}$, where m indexes the observed configurations.

Decomposing structure and configuration. Fine-grained kinematic annotations are not available for every object; typically, for each object i, we observe its keypoints $\mathbf{K}_i^{(m)}$ in different states. Although each keypoint could in principle move anywhere, a real object moves in only a few ways: the keypoints of a cabinet, for example, change only when its drawer slides. We use PCA to find these ways of moving directly from data.

We first average the keypoints over all observed states to obtain a mean shape $\overline{\mathbf{K}}_i$, so that every state becomes the mean shape plus an offset. PCA then finds the R_i offset patterns $\{\mathbf{B}_{ir}\}_{r=1}^{R_i}$ that explain most of how these offsets vary. Each pattern $\mathbf{B}_{ir} \in \mathbb{R}^{N_i \times 3}$ moves all keypoints together in one coordinated way, and a number $\alpha_{ir}^{(m)}$ says how far the object has moved along it in configuration m. A state is thus summarized by only R_i numbers:

$$\mathbf{K}_i^{(m)} \approx \overline{\mathbf{K}}_i + \sum_{r=1}^{R_i} \alpha_{ir}^{(m)} \mathbf{B}_{ir}.$$

For the cabinet, a single pattern suffices: $R_i = 1$, $\mathbf{B}_{i1}$ slides the drawer's keypoints outward, and $\alpha_{i1}^{(m)}$ measures how far the drawer is open in configuration m.

The motion subspace thus captures how the keypoints move across states, while its complement captures what stays fixed: the directions in which the keypoints never move. We turn this into a structural condition with a learnable encoder G_θ, shared across all objects. For each keypoint, G_θ takes its reference position together with its features from the PCA complement and outputs a structural embedding; collecting these over all keypoints gives

$$\mathbf{z}_i = G_\theta\big(\overline{\mathbf{K}}_i, \mathbf{Q}_i\big),$$

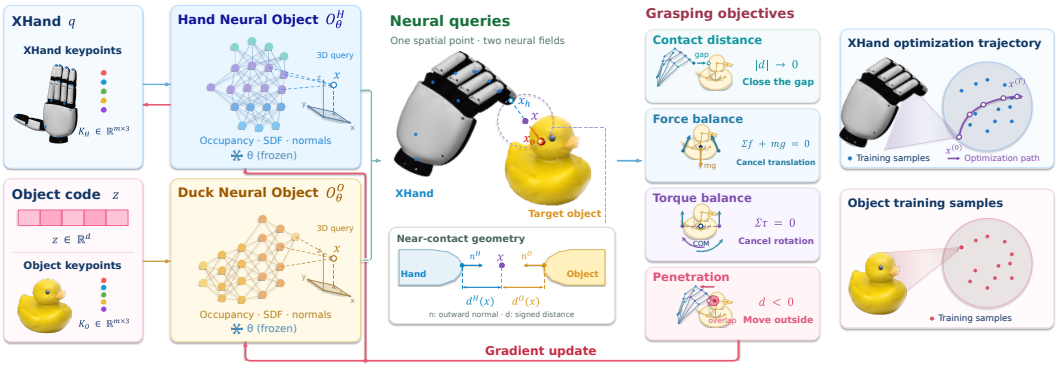

Figure 3: **Gradient flow through frozen hand and object models.** The frozen hand and object models provide SDF and surface-normal queries for grasping objectives. Their gradients propagate through the query coordinates and keypoint-dependent features to update the hand configuration $\mathbf{u} = (\mathbf{q}, \mathbf{t}, \boldsymbol{\omega})$, while the target object's configuration remains fixed.

where $\mathbf{Q}_i$ denotes the per-keypoint complement features.

The resulting decomposition therefore provides

$$\{\mathbf{K}_i^{(m)}\} \xrightarrow{\text{PCA}} \begin{cases} \mathbf{z}_i, & \text{persistent structure,} \\ \mathbf{K}_i^{(m)} = \overline{\mathbf{K}}_i + \Delta\mathbf{K}_i^{(m)}, & \text{current configuration.} \end{cases}$$

Both $\mathbf{z}_i$ and the current keypoints $\mathbf{K}_i$ condition the shared Structured Neural Object.

Decoding occupancy, SDF, and surface normals. We extend WorldString's occupancy decoder with an SDF head on the same query feature $\mathbf{h}_i(\mathbf{x})$. The heads predict occupancy $p_i^{\text{occ}}(\mathbf{x})$ and signed distance $d_i(\mathbf{x})$ under their corresponding supervision. Surface normals follow from the spatial SDF gradient:

$$\mathbf{n}_i(\mathbf{x}) = \frac{\nabla_{\mathbf{x}} d_i(\mathbf{x})}{\max(\|\nabla_{\mathbf{x}} d_i(\mathbf{x})\|_2, \epsilon)}.$$

The resulting interface, $\mathcal{O}_\theta(\mathbf{x}; \mathbf{z}_i, \mathbf{K}_i, \mathbf{c}_i) \mapsto \{p_i^{\text{occ}}, d_i, \mathbf{n}_i\}$, supplies geometric signals for action optimization through the motion coordinates.

3.3 Action: Model as Differentiable Neural Object

From neural object to action optimization. The key advantage of a Structured Neural Object is that the same representation can be used in two complementary directions. During representation learning, the object configuration is given and the model parameters are optimized to reconstruct geometry. During action optimization, we reverse this process: the learned parameters are frozen, while the structured object state is optimized to satisfy an interaction objective. Formally,

$$\theta^\star = \arg\min_\theta \mathcal{L}_{\text{geo}}(\mathcal{O}_\theta(\mathbf{x}; \mathbf{z}, \mathbf{K}, \mathbf{c})) \qquad \longrightarrow \qquad (\mathbf{K}_H^\star, \mathbf{c}_H^\star) = \arg\min_{\mathbf{K}_H, \mathbf{c}_H} \mathcal{L}_{\text{task}}\left(\mathcal{O}_{\theta_H}^{\text{hand}}, \mathcal{O}_{\theta_O}^{\text{obj}}\right),$$

where θ_H and θ_O remain fixed during action optimization.

We represent the dexterous hand and target object as two frozen Structured Neural Objects, $\mathcal{O}_{\theta_H}^{\text{hand}}$ and $\mathcal{O}_{\theta_O}^{\text{obj}}$. The target structure and configuration remain fixed, while the hand keypoints $\mathbf{K}_H$ and global position $\mathbf{c}_H$ are updated through the differentiable geometry. In practice, the task objective is constructed from SDF-based contact, penetration, fingertip/palm, and pose terms.

The gradient flow for action optimization. Freezing the neural-object parameters does not remove derivatives with respect to its structured inputs. Let d_H^r denote the hand SDF evaluated at an object-surface sample $\mathbf{y}_r$, and d_O^j the object SDF evaluated at a hand probe $\mathbf{p}_j(\mathbf{K}_H)$. The grasp

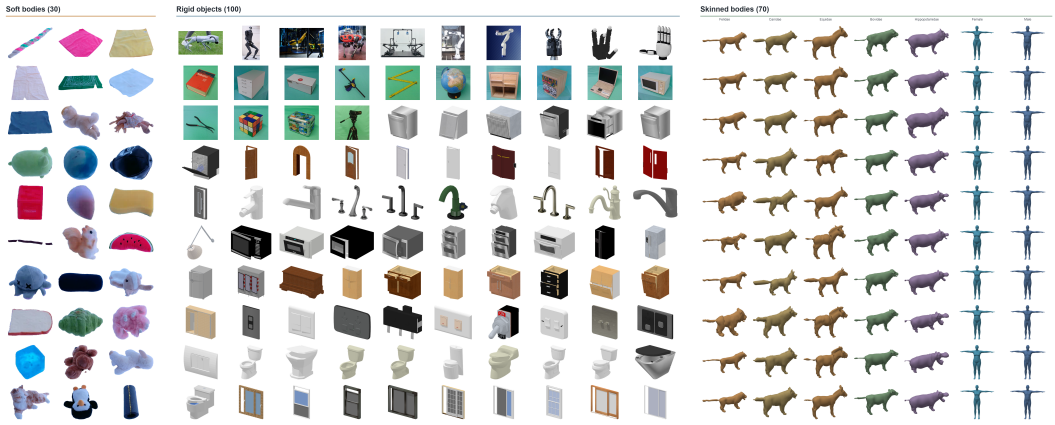

Figure 4: Training Asset samples visualization for Soft, Articulable and Skinning object.

objective propagates to the hand keypoints through

$$\frac{\partial \mathcal{L}_{\text{grasp}}}{\partial \mathbf{K}_H} = \sum_r \frac{\partial \mathcal{L}_{\text{grasp}}}{\partial d_H^r} \frac{\partial d_H^r}{\partial \mathbf{K}_H} + \sum_j \frac{\partial \mathcal{L}_{\text{grasp}}}{\partial d_O^j} \frac{\partial d_O^j}{\partial \mathbf{p}_j} \frac{\partial \mathbf{p}_j}{\partial \mathbf{K}_H}.$$

The first term differentiates through the hand representation, while the second differentiates through the target object as the hand probes move. Together, they provide gradients for updating the hand configuration while keeping the neural-object parameters fixed. Figure 3 illustrates these gradient paths and the resulting interaction objectives. This is the core of $\partial\mathcal{O}_\theta$: differentiating through geometry to guide action.

Policy learning from optimized trajectories. Repeated action optimization produces a sequence of progressively refined hand states, $\mathbf{u}^{(0)} \to \mathbf{u}^{(1)} \to \cdots \to \mathbf{u}^{(T)}$, which we treat as an optimized interaction trajectory. By repeating this process across different objects, placements, and initial configurations, we obtain a collection of keypoint–action demonstrations for policy learning. Given the structured keypoints $\mathbf{K}_t$, the policy predicts an action

$$\mathbf{a}_t = \pi_\phi(\mathbf{K}_t), \qquad \phi^\star = \arg\min_\phi \mathbb{E}_{(\mathbf{K}_t, \mathbf{a}_t^\star)\sim\mathcal{D}} \left[\|\pi_\phi(\mathbf{K}_t) - \mathbf{a}_t^\star\|_2^2 \right],$$

where $\mathbf{a}_t^\star$ is derived from the optimized trajectory. The learned policy therefore maps structured object configurations directly to actions, amortizing the iterative optimization into a feedback controller that is subsequently evaluated zero-shot on the real robot.

4 Experiments

Dataset and Baselines. Our reconstruction datasets cover 200 training objects shown in Fig. 4: 100 articulated objects from PartNet-Mobility, RBO, and robot assets; 70 skinned identities spanning 7 SMAL and SMPL categories, with ten identities per category; and 30 deformable objects from Deform360. Rigid and soft-body evaluation uses held-out poses of training objects, whereas skinned evaluation uses poses of unseen identities. Reconstruction baselines include NN and Optim.NN, the original WorldString, specialized methods for articulated and deformable geometry, and general image-to-3D models, including TRELLIS.2 and Meshy. Manipulation experiments use four objects—a duck, piggy bank, trophy, and apple—with policy-learning and grasp-optimization baselines introduced in the corresponding sections.

Implementation Details. We train one shared network per object family in PyTorch using AdamW. Training proceeds in three stages: occupancy training, SDF post-training, and surface-normal fine-tuning through the SDF gradients. Keypoints and query locations are expressed in a common normalized coordinate system. Reconstruction metrics are averaged first over configurations within each object and then over objects. During manipulation, the learned geometry remains frozen, and optimized trajectories provide supervision for policy learning. Detailed configurations are provided in Appendix A.

Table 2: Multi-field reconstruction quality across three object families.

Family	Method	Occupancy			SDF (voxels)		Normal	Parameter Number
		P ↑	R ↑	F1 ↑	MAE ↓	CD ↓	Error (°) ↓	
Rigid	NN	88.15	88.15	88.15	0.64	0.95	10.58	/
	Optim.NN	89.34	91.25	89.98	0.43	0.71	**9.71**	/
	A-SDF (Mu et al., 2021)	60.22	**97.98**	71.58	0.61	3.23	28.40	2.0M
	CARTO (Heppert et al., 2023)	88.64	91.73	89.91	1.96	2.11	25.73	0.8M
	TRELLIS.2 (Xiang et al., 2026)	11.08	22.33	11.32	/	/	89.75	4.0B
	Meshy (Meshy AI)	28.87	20.80	23.09	/	/	76.65	/
	WorldString (Xu et al., 2026a)	93.59	95.78	94.22	/	/	/	1.4M*100
	∂O_θ **(Ours)**	**96.56**	96.54	**96.54**	**0.26**	**0.54**	18.53	1.9M
Skinning	NN	68.67	67.40	67.91	5.88	7.86	56.00	/
	Optim.NN	84.71	86.66	85.63	3.17	2.24	32.04	/
	NSDP (Tang et al., 2022)	**93.84**	87.41	90.43	/	/	29.73	9.0M
	TRELLIS.2 (Xiang et al., 2026)	33.05	36.83	34.15	/	/	91.46	4.0B
	Meshy (Meshy AI)	33.73	29.53	31.43	/	/	68.63	/
	WorldString (Xu et al., 2026a)	91.61	91.88	91.74	/	/	/	1.4M*70
	∂O_θ **(Ours)**	93.71	**96.37**	**95.02**	**0.46**	**0.59**	**15.70**	1.9M
Skinning-	LEAP (Mihajlovic et al., 2021)	95.98	94.47	95.21	/	/	24.28	5.5M
	∂O_θ **(Ours)**	**96.00**	**97.69**	**96.83**	**0.27**	**0.31**	**9.23**	1.9M
Soft	NN	91.67	92.11	91.88	1.35	1.10	28.71	/
	Optim.NN	**94.43**	94.84	94.64	0.72	0.55	26.67	/
	NSDP (Tang et al., 2022)	91.55	80.26	85.29	/	/	32.81	9.0M
	DynoSurf (Yao et al., 2024)	79.89	85.27	82.34	/	/	38.52	0.2M*30
	DG-Mesh (Liu et al., 2025a)	57.71	64.45	58.87	/	/	58.14	2.6M*30
	TRELLIS.2 (Xiang et al., 2026)	26.34	46.42	32.09	/	/	90.34	4.0B
	Meshy (Meshy AI)	42.73	34.87	37.23	/	/	106.74	/
	WorldString (Xu et al., 2026a)	92.68	93.82	93.24	/	/	/	1.4M*30
	∂O_θ **(Ours)**	94.18	**98.92**	**96.47**	**0.37**	**0.44**	**19.22**	1.9M

4.1 Main Results

Occupancy Reconstruction. Table 2 compares reconstruction across articulated, skinned, and deformable objects. Our method achieves the highest occupancy F1 in all three families, improving over the original WorldString by 2.32, 3.28, and 3.23 percentage points, respectively. These gains are obtained with one shared model per family. On the SMPL subset, our method also outperforms LEAP, reaching an F1 of 96.83 compared with 95.21.

Geometric Accuracy. Across the three object families, our method achieves mean absolute distance errors of 0.26–0.46 voxels and surface Chamfer distances of 0.44–0.59 voxels, the lowest reported values in each family. It also achieves the lowest normal errors on skinned and deformable objects, at $15.70°$ and $19.22°$, and on the SMPL subset, at $9.23°$. The lower rigid-object normal errors of NN and Optim.NN are consistent with their advantage in interpolating unchanged local part surfaces. The learned SDF supplies both distance values and spatial gradients for the subsequent manipulation experiments.

Parameter Efficiency. Sharing parameters substantially reduces the total model size. For the 100 articulated objects, independently trained WorldString models require approximately 140M parameters in total, whereas our shared model uses approximately 1.9M while achieving higher reconstruction F1. Qualitative reconstructions across articulated motion, skinning, and soft-body deformation are provided in Appendix C.

4.2 Real-World Deployment

Evaluation Setup. Figure 5 shows the hardware setup and sensing interfaces. Table 3 compares Real2Real policies ACT (Zhao et al., 2023) and $\pi_{0.5}$ (Physical Intelligence et al., 2025) with

Table 3: Real-world grasp success rates (%) on four objects under in-distribution and out-of-distribution placements.

Method	Success Rate (%) ↑							
	Duck		Piggy Bank		Trophy		Apple	
	In-distribution	Out-of-distribution	In-distribution	Out-of-distribution	In-distribution	Out-of-distribution	In-distribution	Out-of-distribution
Real2Real Policy								
ACT	46.15	23.08	53.85	15.38	61.54	23.08	76.92	15.38
$\pi_{0.5}$	69.23	15.38	46.15	7.69	69.23	23.08	69.23	23.08
Sim2Real Policy								
MuJoCo-trained policy	23.08	0.00	38.46	7.69	38.46	7.69	61.54	15.38
Isaac Lab-trained policy	15.38	7.69	76.92	23.08	15.38	0.00	30.77	7.69
∂O_θ **(Ours)**	**100.00**	**100.00**	**84.62**	**76.92**	**76.92**	**69.23**	**84.62**	**69.23**

Table 4: Grasp optimization quality and convergence on four objects. Pen. is penetration depth in millimeters; SR is grasp success rate (%) in simulator. Convergence reports capped mean steps to the shared definition of convergence, averaged equally over objects.

Method	Duck			Piggy Bank			Trophy			Apple			Convergence
	Loss ↓	Pen. ↓	SR ↑	Loss ↓	Pen. ↓	SR ↑	Loss ↓	Pen. ↓	SR ↑	Loss ↓	Pen. ↓	SR ↑	Steps ↓
Fast-Grasp'D	44.96	16.18	69.23	24.75	13.78	30.77	21.61	7.74	84.62	40.94	12.69	84.62	1334
DexGraspNet	231.76	20.85	7.69	111.38	14.24	7.69	3.07	5.70	61.54	4.51	4.41	30.77	2687
∂O_θ **(Ours)**	**1.11**	**4.71**	**100.00**	**0.70**	**1.16**	**84.6**	**1.42**	**1.35**	**92.30**	**0.71**	**0.54**	**100.00**	**1084**

Sim2Real policies trained on trajectories generated and validated in MuJoCo (Todorov et al., 2012) or Isaac Lab (Mittal et al., 2025). Our policy and the two simulator-trained policies share the policy architecture, observation interface, and training budget, and are evaluated on the same placements. In-distribution (ID) positions lie within the 8×8 cm training square; out-of-distribution (OOD) positions lie outside it but within the concentric 24×24 cm test square. A grasp succeeds when the hand lifts the object and maintains a stable hold. Appendix B details the evaluation protocol.

Real-World Performance. Our policy achieves the highest success rate in all eight object–placement settings, averaging 86.54% under ID placements and 78.85% under OOD placements. The strongest baseline averages are 63.46% for $\pi_{0.5}$ under ID placements and 19.23% for ACT under OOD placements. Our success rate decreases by only 7.7 percentage points under the placement shift, showing that policies learned from geometry-optimized trajectories remain effective beyond the training region.

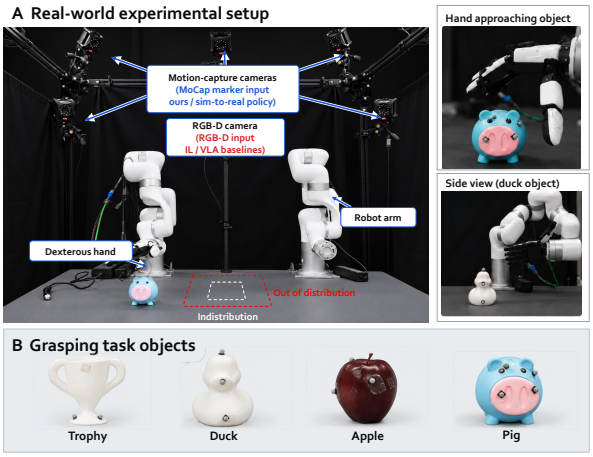

Figure 5: Real-world setup and grasping objects.

4.3 Grasp Optimization

Evaluation Setup. Table 4 compares our grasp optimizer with Fast-Grasp'D (Turpin et al., 2023) and the grasp-synthesis optimizer of DexGraspNet (Wang et al., 2023a), using the same hand, initial configurations, and maximum iteration budget. We measure a common evaluation loss, hand–object penetration, simulated grasp success, and convergence steps; Appendix B defines the metrics and their aggregation.

Grasp Quality. Our method achieves the lowest loss and penetration depth and the highest simulated success rate on all four objects. Its mean success rate is 94.23%, compared with 67.31% for Fast-Grasp'D and 26.92% for DexGraspNet. On the apple, penetration decreases from 12.69 mm and 4.41 mm for these baselines to 0.54 mm with our method.

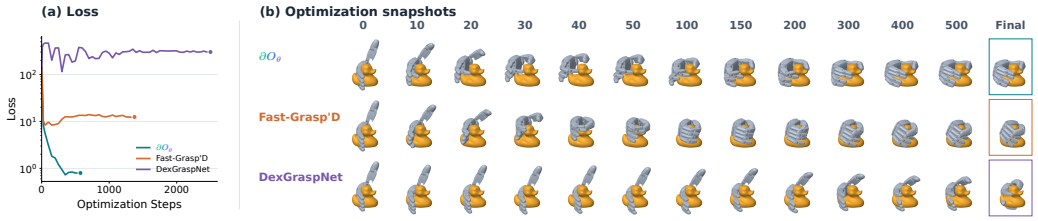

Figure 6: **Grasp optimization on the duck.** Comparison of ∂O_θ, Fast-Grasp'D, and DexGraspNet from the same initial hand configuration. (a) Common evaluation loss using ground-truth geometry (log scale). (b) Hand configurations at the indicated optimization steps.

Table 5: Ablations of shape-code representations on ten rigid objects (left) and cumulative training stages on all 100 rigid objects (right).

(a) Shape-code representations.

Shape code	Occupancy (%)		
	P ↑	R ↑	F1 ↑
None	60.52	67.93	64.01
Learned ID embedding	89.21	89.33	89.03
∂O_θ **(Ours)**	**93.17**	**98.65**	**95.60**

(b) Training stages.

Training stage	Occ. F1 (%) ↑	SDF MAE (vox.) ↓	Normal Err. (°) ↓
Occ only	92.17	/	/
+ SDF	93.41	0.44	49.11
+ Normal (Full)	**96.54**	**0.26**	**18.53**

Optimization Convergence. The capped mean step count is 1084 for our method, compared with 1334 for Fast-Grasp'D and 2687 for DexGraspNet, corresponding to reductions of 18.7% and 59.7%, respectively. This demonstrates that our method achieves more efficient convergence while requiring substantially fewer optimization iterations. Figure 6 further illustrates this behavior on a representative duck-grasping example, where ∂O_θ decreases the evaluation loss more rapidly and reaches a lower final loss in fewer optimization steps than both baselines.

4.4 Ablation Studies

Shape-code Representation. On ten rigid objects, we replace the PCA-complement shape code with a learned identity embedding or remove it from the shared reconstruction model (Table 5a). Our shape code achieves an occupancy F1 of 95.60, exceeding the identity embedding (89.03) and no-code variant (64.01) by 6.57 and 31.59 percentage points. These results show that structural statistics extracted from training configurations provide more effective conditioning than object identity alone.

Training Stages. Table 5b follows the three training stages on all 100 rigid objects. SDF post-training raises occupancy F1 from 92.17 to 93.41, with a distance MAE of 0.44 voxels and a normal error of 49.11°. Subsequent normal fine-tuning reduces the angular error to 18.53°, a 62.3% reduction, while lowering SDF MAE to 0.26 voxels and increasing F1 to 96.54. The large normal error after SDF training shows why distance accuracy and gradient accuracy must be evaluated separately; explicit normal supervision improves both gradient direction and field reconstruction.

5 Conclusion

We presented ∂O_θ, a shared neural object representation that connects geometric reconstruction with action optimization. A PCA decomposition of observed keypoint variation provides structural conditioning for reconstruction and motion coordinates for optimization. The learned representation supplies occupancy, signed distances, and surface normals, enabling task gradients to update object configurations while model parameters remain fixed. Optimized interactions then provide demonstrations for policy learning. Experiments demonstrate accurate reconstruction across 200 articulated, skinned, and deformable objects with compact shared models, alongside improved grasp optimization and effective real-world policy deployment.

AI USE STATEMENT

We used generative AI primarily to assist manuscript drafting and language editing, and to retrieve relevant information and literature. AI tools also supported discussions of experimental design and interpretation of results. Team members reviewed and verified the AI-assisted content and retrieved information. The authors take responsibility for the final text, claims, and research artifacts.

ETHICS STATEMENT

This work studies object reconstruction and robotic manipulation using existing datasets, model assets, and tabletop experiments. We attribute datasets and visual materials to their sources; their use and redistribution remain subject to the respective licenses. Physical interaction can involve unintended contact, so deployment requires appropriate workspace and collision safeguards. Our evaluation concerns the specified objects and experimental conditions.

REPRODUCIBILITY STATEMENT

Appendix A documents the object collection, pose budgets, data splits, training details, and QPose–FK parameterization. Appendix B specifies the training seeds, reconstruction aggregation, method-specific grasp objectives, stopping rule, independent evaluation loss, and real-world trial protocol, including success counts and confidence intervals.

REFERENCES

Brandon Amos, Ivan Jimenez, Jacob Sacks, Byron Boots, and J. Zico Kolter. Differentiable MPC for end-to-end planning and control. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://github.com/locuslab/differentiable-mpc.

Xu Chen, Yufeng Zheng, Michael J. Black, Otmar Hilliges, and Andreas Geiger. SNARF: Differentiable forward skinning for animating non-rigid neural implicit shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. URL https://xuchen-ethz.github.io/snarf/.

Yiting Chen, Xiao Gao, Kunpeng Yao, Loïc Niederhauser, Yasemin Bekiroglu, and Aude Billard. Implicit articulated robot morphology modeling with configuration space neural signed distance functions. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025. URL https://arxiv.org/abs/2309.16085.

Danny Driess, Jung-Su Ha, Marc Toussaint, and Russ Tedrake. Learning models as functionals of signed-distance fields for manipulation planning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 245–255. PMLR, 2022. URL https://proceedings.mlr.press/v164/driess22a.html.

Tao Du, Kui Wu, Pingchuan Ma, Sebastien Wah, Andrew Spielberg, Daniela Rus, and Wojciech Matusik. DiffPD: Differentiable projective dynamics. *ACM Transactions on Graphics*, 41(2), 2021. doi: 10.1145/3490168. URL https://doi.org/10.1145/3490168.

Chen Geng, Guangzhao He, Yue Gao, Yunzhi Zhang, Shangzhe Wu, and Jiajun Wu. NeuROK: Generative 4D neural object kinematics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 39239–39251, 2026. URL https://chen-geng.com/neurok.

Nick Heppert, Muhammad Zubair Irshad, Sergey Zakharov, Katherine Liu, Rares Andrei Ambrus, Jeannette Bohg, Abhinav Valada, and Thomas Kollar. Carto: Category and joint agnostic reconstruction of articulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Heppert_CARTO_Category_and_Joint_Agnostic_Reconstruction_of_ARTiculated_Objects_CVPR_2023_paper.html.

Yuanming Hu, Jiancheng Liu, Andrew Spielberg, Joshua B. Tenenbaum, William T. Freeman, Jiajun Wu, Daniela Rus, and Wojciech Matusik. ChainQueen: A real-time differentiable physical simulator for soft robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019. URL https://arxiv.org/abs/1810.01054.

Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. DiffTaichi: Differentiable programming for physical simulation. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://arxiv.org/abs/1910.00935.

Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. SC-GS: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. URL https://yihua7.github.io/SC-GS-web/.

Hanxiao Jiang, Hao-Yu Hsu, Kaifeng Zhang, Hsin-Ni Yu, Shenlong Wang, and Yunzhu Li. PhysTwin: Physics-informed reconstruction and simulation of deformable objects from videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7219–7230, 2025. URL https://jianghanxiao.github.io/phystwin-web/.

Korrawe Karunratanakul, Adrian Spurr, Zicong Fan, Otmar Hilliges, and Siyu Tang. A skeleton-driven neural occupancy representation for articulated hands. In *International Conference on 3D Vision (3DV)*, 2021. URL https://korrawe.github.io/HALO/HALO.html.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

Mikhail Koptev, Nadia Figueroa, and Aude Billard. Neural joint space implicit signed distance functions for reactive robot manipulator control. *IEEE Robotics and Automation Letters*, 8(2): 480–487, 2023. doi: 10.1109/LRA.2022.3227860. URL https://doi.org/10.1109/LRA.2022.3227860.

Xuan Li, Yi-Ling Qiao, Peter Yichen Chen, Krishna Murthy Jatavallabhula, Ming Lin, Chenfanfu Jiang, and Chuang Gan. PAC-NeRF: Physics augmented continuum neural radiance fields for geometry-agnostic system identification. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://arxiv.org/abs/2303.05512.

Yiming Li, Yan Zhang, Amirreza Razmjoo, and Sylvain Calinon. Representing robot geometry as distance fields: Applications to whole-body manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024. URL https://arxiv.org/abs/2307.00533.

Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B. Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. In *International Conference on Learning Representations (ICLR)*, 2019. URL https://arxiv.org/abs/1810.01566.

Yunzhu Li, Hao He, Jiajun Wu, Dina Katabi, and Antonio Torralba. Learning compositional Koopman operators for model-based control. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://koopman.csail.mit.edu/.

Yunzhu Li, Shuang Li, Vincent Sitzmann, Pulkit Agrawal, and Antonio Torralba. 3D neural scene representations for visuomotor control. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 112–123. PMLR, 2022. URL https://proceedings.mlr.press/v164/li22a.html.

Xingyu Lin, Zhiao Huang, Yunzhu Li, Joshua B. Tenenbaum, David Held, and Chuang Gan. DiffSkill: Skill abstraction from differentiable physics for deformable object manipulations with tools. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://arxiv.org/abs/2203.17275.

Isabella Liu, Hao Su, and Xiaolong Wang. Dynamic gaussians mesh: Consistent mesh reconstruction from dynamic scenes. In *International Conference on Learning Representations*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/057779005710fdbca7be97667d16047d-Abstract-Conference.html.

Ruoshi Liu, Alper Canberk, Shuran Song, and Carl Vondrick. Differentiable robot rendering. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 117–129. PMLR, 2025b. URL https://proceedings.mlr.press/v270/liu25a.html.

Pingchuan Ma, Peter Yichen Chen, Bolei Deng, Joshua B. Tenenbaum, Tao Du, Chuang Gan, and Wojciech Matusik. Learning neural constitutive laws from motion observations for generalizable PDE dynamics. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 23279–23300. PMLR, 2023. URL https://proceedings.mlr.press/v202/ma23a.html.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. URL https://arxiv.org/abs/1812.03828.

Meshy AI. Meshy: Ai 3d model generator. https://www.meshy.ai/. Accessed: 2026-09-24.

Marko Mihajlovic, Yan Zhang, Michael J. Black, and Siyu Tang. Leap: Learning articulated occupancy of people. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Mihajlovic_LEAP_Learning_Articulated_Occupancy_of_People_CVPR_2021_paper.html.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, 2020. URL https://www.matthewtancik.com/nerf.

Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, et al. Isaac Lab: A GPU-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025. URL https://arxiv.org/abs/2511.04831.

Jiteng Mu, Weichao Qiu, Adam Kortylewski, Alan Yuille, Nuno Vasconcelos, and Xiaolong Wang. A-sdf: Learning disentangled signed distance functions for articulated shape representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. URL https://openaccess.thecvf.com/content/ICCV2021/html/Mu_A-SDF_Learning_Disentangled_Signed_Distance_Functions_for_Articulated_Shape_Representation_ICCV_2021_paper.html.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. URL https://arxiv.org/abs/1901.05103.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. URL https://www.albertpumarola.com/research/D-NeRF/index.html.

Quanyuan Ruan, Jiabao Lei, Wenhao Yuan, Yanglin Zhang, Dekun Lu, Guiliang Liu, and Kui Jia. Prof. Robot: Differentiable robot rendering without static and self-collisions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22562–22572, 2025. URL https://www.qrcat.cn/prof-robot/.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8459–8468. PMLR, 2020. URL https://proceedings.mlr.press/v119/sanchez-gonzalez20a.html.

Jiapeng Tang, Lev Markhasin, Bi Wang, Justus Thies, and Matthias Niessner. Neural shape deformation priors. In *Advances in Neural Information Processing Systems*, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6d09ef61aeb76be676b358f6f87b3484-Abstract-Conference.html.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109. URL https://github.com/google-deepmind/mujoco.

Dylan Turpin, Tao Zhong, Shutong Zhang, Guanglei Zhu, Eric Heiden, Miles Macklin, Stavros Tsogkas, Sven Dickinson, and Animesh Garg. Fast-Grasp'D: Dexterous multi-finger grasp generation through differentiable simulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023. URL https://dexgrasp.github.io/.

Ruicheng Wang, Jialiang Zhang, Jiayi Chen, Yinzhen Xu, Puhao Li, Tengyu Liu, and He Wang. DexGraspNet: A large-scale robotic dexterous grasp dataset for general objects based on simulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023a. URL https://pku-epic.github.io/DexGraspNet/.

Yixuan Wang, Yunzhu Li, Katherine Driggs-Campbell, Li Fei-Fei, and Jiajun Wu. Dynamic-resolution model learning for object pile manipulation. In *Proceedings of Robotics: Science and Systems*, 2023b. doi: 10.15607/RSS.2023.XIX.047. URL https://www.roboticsproceedings.org/rss19/p047.html.

Thomas Weng, David Held, Franziska Meier, and Mustafa Mukadam. Neural grasp distance fields for robot manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023. URL https://arxiv.org/abs/2211.02647.

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4D gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. URL https://guanjunwu.github.io/4dgs/.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, and Jiaolong Yang. Native and compact structured latents for 3d generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. PhysGaussian: Physics-integrated 3D gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. URL https://arxiv.org/abs/2311.12198.

Jie Xu, Tao Chen, Lara Zlokapa, Michael Foshey, Wojciech Matusik, Shinjiro Sueda, and Pulkit Agrawal. An end-to-end differentiable framework for contact-aware robot design. In *Proceedings of Robotics: Science and Systems*, 2021. doi: 10.15607/RSS.2021.XVII.008. URL https://www.roboticsproceedings.org/rss17/p008.html.

Jie Xu, Viktor Makoviychuk, Yashraj Narang, Fabio Ramos, Wojciech Matusik, Animesh Garg, and Miles Macklin. Accelerated policy learning with parallel differentiable simulation. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=ZSKRQMvttc.

Kunqi Xu, Jitao Li, Jianglong Ye, Tianshu Tang, Isabella Liu, Sifei Liu, and Xueyan Zou. Worldstring: Actionable world representation, 2026a. URL https://arxiv.org/abs/2605.18743.

Qingshan Xu, Jiao Liu, Shangshu Yu, Yuxuan Wang, Yuan Zhou, Junbao Zhou, Jiequan Cui, Yew-Soon Ong, and Hanwang Zhang. NeuSpring: Neural spring fields for reconstruction and simulation of deformable objects from videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 11361–11369, 2026b. doi: 10.1609/aaai.v40i14.38117. URL https://ojs.aaai.org/index.php/AAAI/article/view/38117.

Yuxin Yao, Siyu Ren, Junhui Hou, Zhi Deng, Juyong Zhang, and Wenping Wang. Dynosurf: Neural deformation-based temporally consistent dynamic surface reconstruction. In *European Conference on Computer Vision*, 2024. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/html/4886_ECCV_2024_paper.php.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.016. URL https://www.roboticsproceedings.org/rss19/p016.html.

René Zurbrügg, Andrei Cramariuc, and Marco Hutter. GraspQP: Differentiable optimization of force closure for diverse and robust dexterous grasping. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 2583–2602. PMLR, 2025. URL https://proceedings.mlr.press/v305/zurbrugg25a.html.

A METHOD IMPLEMENTATION DETAILS

Object collection and data splits. The articulated objects comprise 76 PartNet-Mobility objects, 14 RBO objects, and ten robot assets. Soft-object images come from Deform360; articulated panels use source photographs where available and renderings otherwise. Skinned panels render the saved identity meshes with display materials. Training uses 1,000 poses per training object; validation and evaluation each use 100 poses per object in the corresponding split. Rigid and soft-body evaluation uses new poses of training objects. Skinned evaluation uses poses of identities excluded from model training.

Training regimes and coordinate conventions. The rigid, soft-body, and skinned settings train separate shared models. Rigid descriptors use training-set mean keypoints and complement-projector rows; soft-body descriptors use a fixed training-reference pose and PCA of rigidly aligned training configurations. Reference alignment transforms keypoints and spatial queries together. The soft-body voxel-query wrapper accounts for the cell-center offset before applying this transform. The 100-object recipe uses continuous float keypoints. Some skinned occupancy and SDF recipes instead quantize keypoints onto the 256-cell lattice; their spatial query gradients remain available, but keypoint derivatives must not be inferred from the quantized preprocessing. The downstream XHand checkpoint explicitly uses continuous conditioning.

For occupancy training, let n_{ic} be the batch count of class c for object i. The rigid training weights are $w_{ic} = (n_{i0} + n_{i1})s_c/(2\max(n_{ic}, 1))$, where $s_0 = 1$ and s_1 is the configured foreground scale. If either class is absent, both weights become one. Counts aggregate across distributed workers. Object-macro training uses uniform α_i over represented objects; the 100-object recipe uses presentation-proportional weights because exposure quotas already control object sampling. The soft-body trainer balances classes per object, and the skinned trainer balances sampled categories and global occupancy classes.

SDF post-training. Three quarters of distance queries are sampled near surfaces and one quarter uniformly. In raw readout units, the default objective is

$$\mathcal{L}_{\text{post}} = 0.9\langle \rho_{1/8}(\widehat{s} - s^*)\rangle_{\text{near}} + 0.1\langle \rho_{1/8}(\widehat{s} - s^*)\rangle_{\text{uniform}} + \lambda_o \mathcal{L}_{\text{occ}}, \tag{1}$$

where $\rho_{1/8}$ is Smooth L1 with transition $1/8$ and averages include only valid targets. Sampling proportions and loss weights are distinct. For these generalization checkpoints, $d_{\text{vox}} = 8\widehat{s}$ and $d_{\text{norm}} = \widehat{s}/32$ on a 256-cell normalization cube. Frozen-backbone runs omit occupancy queries; joint runs retain an occupancy objective on additional labeled queries, warm up the readout, and then train the trunk at a smaller learning rate. These scale factors are specific to these checkpoints. Grasping uses each hand/object wrapper's stored distance normalization before conversion to common world units.

QPose parameterization for grasp optimization. In the XHand experiments, QPose provides the joint parameters $\mathbf{q}$ used to generate hand keypoints by forward kinematics. Let $\mathbf{k}_H = \text{vec}(\mathbf{K}_H)$ denote the stacked keypoint coordinates and F_{KP} the FK mapping, with the wrist pose applied to express keypoints in the common coordinate frame. The grasp objective propagates through the frozen neural geometry and this mapping to QPose:

$$\nabla_{\mathbf{q}} \mathcal{L}_{\text{grasp}} = \left(\frac{\partial \mathbf{k}_H}{\partial \mathbf{q}}\right)^{\top} \nabla_{\mathbf{k}_H} \mathcal{L}_{\text{grasp}}.$$

After each QPose update, FK supplies the next keypoint configuration, $\mathbf{K}_H^{(\ell+1)} = F_{\text{KP}}(\mathbf{q}^{(\ell+1)})$, which is passed back to the neural geometry for the next objective evaluation. The geometry-network parameters remain fixed. Here ℓ indexes optimizer updates; the demonstration control index t follows the execution timing and action mapping used to convert the optimized trajectory into actions.

Policy learning for real-world evaluation. For the four-object evaluation, we freeze the learned hand and object geometry and optimize hand configurations under the grasp objective. The resulting QPose trajectories provide keypoint–action demonstrations that supervise a policy through action regression, as described in Section 3.3. At deployment, the policy maps keypoint observations to

actions. The MuJoCo and Isaac Lab baselines use simulator-generated demonstrations with the same policy architecture, observation interface, and training budget.

B Experimental Evaluation Details

Reconstruction runs and aggregation. We train our reconstruction models with seeds 0 and 1, evaluate each run separately, and average their results. Let m_{sit} be a reconstruction metric for seed s, object i, and test pose t. With N evaluated objects and $T_i = 100$ poses per object, the reported mean is

$$\overline{m} = \frac{1}{2} \sum_{s \in \{0,1\}} \frac{1}{N} \sum_{i=1}^{N} \frac{1}{T_i} \sum_{t=1}^{T_i} m_{sit}. \tag{2}$$

Each seed therefore evaluates 10,000 rigid-object poses and 3,000 soft-object poses. The ten-object shape-code ablation evaluates 1,000 poses per seed, and the 100-object training-stage ablation evaluates 10,000. For skinned objects, the number is 100 times the number of held-out identities. Evaluating two seeds doubles the number of model evaluations, while preserving the test-set size. Precision, recall, and F1 are aggregated separately; mean F1 is not recomputed from mean precision and recall.

Method-specific grasp objectives. Each method optimizes its own objective. The common ground-truth geometry loss defined below is used only for evaluation and comparison curves. For the duck, ∂O_θ uses

$$\mathcal{L}_{\text{duck}} = 0.002\mathcal{L}_{\text{grasp}} + \mathcal{L}_{\text{collision}} + 5\mathcal{L}_{\text{floor}}. \tag{3}$$

The grasp term combines object-surface contact and penetration against the hand SDF, finger-pad and palm proximity, fingertip attraction, joint priors, and closure after contact. The additional collision term covers the hand and flange: fingers allow approximately 3 mm of penetration, while the palm and flange require 1 mm clearance. The floor term requires 3 mm clearance from the table. This is the static terminal-state version of the policy teacher objective, without successful-trajectory guidance or trajectory-optimization terms.

For the piggy bank, trophy, and apple, ∂O_θ uses the corresponding policy teacher's terminal loss. Its terms cover finger-pad contact, whole-hand and flange penetration, table collision, thumb opposition, grasp center and height, enclosure, and pose priors. Object-specific settings are retained, including the trophy's allowed grasp region. Thus, the four objects share an optimization framework but retain their own terminal objectives.

Fast-Grasp'D optimizes contact-response stability together with penetration, joint-midrange, and joint-limit penalties. It samples 432 hand-surface points and evaluates seven virtual perturbation conditions: rest and the positive and negative directions of each Cartesian axis. Contact forces, friction, and torque responses determine the stability objective. Contact inflation is gradually removed during the first 64 steps. This optimization uses differentiable contact-response calculations; the full MuJoCo grasp rollout is reserved for execution-based evaluation.

DexGraspNet minimizes

$$E = E_{\text{fc}} + 100E_{\text{dis}} + 100E_{\text{pen}} + 10E_{\text{spen}} + E_{\text{joints}}, \tag{4}$$

where the terms represent force closure, contact distance, object-to-hand penetration, self-penetration, and joint-limit violations. It maintains four contact points, proposes random contact replacements, and uses an annealing rule to accept or reject proposed grasps. Penetration queries use 2,000 object-surface samples. The optimizer requires no additional model training.

Stopping rule and computation budget. All methods use the same native-loss plateau criterion, with complete gradient evaluations as the computation counter. Let a_g and b_g be the medians of the retained grasp's native loss in the previous and most recent windows of 50 gradient evaluations. Every 25 evaluations, we test

$$\frac{|b_g - a_g|}{\max(|a_g|, |b_g|, 10^{-8})} < 0.01. \tag{5}$$

Three consecutive passing checks confirm convergence. An objective-phase change clears the windows and passing-check count while retaining the total computation count. The common cap is

6,000 full gradient evaluations; rejected proposals and L-BFGS line-search evaluations also count. Reaching the cap or exiting the solver before the plateau is confirmed is recorded as unconfirmed convergence. Evaluation always uses the last accepted grasp, without selecting the best-scoring intermediate pose. The independent evaluation loss does not determine stopping or output selection.

Independent grasp evaluation. The Loss entries in Table 4 and the comparison curves in Figure 6 use the same proxy objective,

$$\mathcal{L}_{\text{eval}} = \mathcal{L}_{\text{contact}} + 0.1\mathcal{L}_{\text{wrench}} + 10\big(\mathcal{L}_{\text{penetration}} + \mathcal{L}_{\text{self}} + \mathcal{L}_{\text{floor}} + \mathcal{L}_{\text{joint}}\big). \tag{6}$$

This evaluator uses ground-truth meshes independently of the optimized neural fields, sampling 256 surface points per hand link and 4,096 points on the object. Contact measures finger-pad proximity, wrench measures force and torque imbalance, and the remaining terms penalize bidirectional hand–object penetration, self-penetration, table penetration, and joint-limit violations. Contact distances are normalized by 10 mm and penetration distances by 1 mm before squaring; torque uses the object radius to normalize its moment arm.

For each trial, Pen. is the largest sampled penetration depth across both hand-to-object and object-to-hand queries. Table 4 reports its mean across trials in millimeters. Loss likewise averages the proxy score across trials. SR measures the fraction of final grasps that lift the object and maintain a stable hold under the common simulation execution protocol. Object-level success rates are weighted equally when reporting the four-object mean.

Real-world grasping protocol. The 8×8 cm training square is centered within the 24×24 cm test square. Taking their common center as the origin, ID positions have both horizontal coordinates in $[-4, 4]$ cm. OOD positions lie within $[-12, 12] \times [-12, 12]$ cm but outside the training square. Methods are evaluated on the same placements. A trial succeeds when the dexterous hand lifts the object and maintains a stable hold. Success rates are computed separately for each object and placement split, with 13 trials per method, object, and split. Thus, each method has 52 ID and 52 OOD trials. Means across the four objects use equal object weights.

Observation modalities and scope of the real-world comparison. The real-world methods in Table 3 do not all use the same observation modality. Our policy and the simulator-trained policies receive structured object-state observations derived from motion-capture markers, whereas ACT and $\pi0.5$ operate from RGB-D observations, as illustrated in Fig. 5. Therefore, the comparison with ACT and $\pi0.5$ should be interpreted as an end-to-end comparison of the evaluated systems under their respective sensing interfaces, rather than as a controlled comparison that isolates the effect of the policy-learning method alone. In particular, the performance gap cannot be attributed solely to the learned representation or policy, since differences in state estimation may also contribute. In contrast, our method, the MuJoCo-trained policy, and the Isaac Lab-trained policy share the same policy architecture, observation interface, training budget, and evaluation placements. These comparisons therefore more directly isolate the effect of the demonstration-generation procedure while controlling for the policy and sensing interfaces. A fully matched comparison in which ACT or a VLA baseline receives the same structured observations, or our policy operates from the same RGB-D perception pipeline, would further disentangle the effects of sensing and policy learning.

Figure 7 shows example real-world duck-grasping rollouts under ID and OOD placements.



Figure 7: Real-world duck-grasping rollouts of $\pi_{0.5}$, ∂O_θ (WorldString Policy in the figure), and a MuJoCo-trained policy under ID and OOD placements. Each rollout is normalized to a 10 s display timeline; the asterisk marks the final frame.

C Reconstruction under Continuous Pose Variation

We visualize shared Occupancy reconstruction on seven articulated objects, seven skinned categories, and seven soft objects. Each row shows four configurations of the same identity, with camera orientation and display scale fixed to facilitate comparison. The identity condition remains unchanged within each row, allowing the visualization to focus on how reconstructed geometry varies with the input keypoint configuration. The left panels show PCA projections of the input keypoints for visualization only.

Articulated objects. Figure 8 shows seven articulated objects. For each object, we fix the identity condition and vary one joint coordinate, displaying four predictions along a monotonic motion segment. This setting changes the relative arrangement of rigid parts while preserving their individual shapes. The sequences illustrate how the shared reconstruction model responds to configuration changes, including the movement of articulated components and the resulting changes in the object's overall geometry. Keeping the viewpoint fixed makes these changes directly comparable across the four displayed configurations.

Skinned objects. Figure 9 shows one identity from each of the seven skinned categories, covering animal and human body models. Within each row, the identity condition remains fixed while one joint coordinate varies. The four predictions illustrate how the reconstructed surface changes with articulation, including deformation around joints and transitions between connected body regions. Across rows, the examples span different body structures and proportions, illustrating the range of geometry represented by the shared model. Together, the sequences show how identity conditioning and the current keypoint configuration contribute to reconstruction across these categories.

Soft-body deformation. Figure 10 shows seven soft objects with the highest mean test F1 in the 30-object model. Each row contains four held-out states from one recording, selected for keypoint variation and ordered by source time. Here, configuration changes follow the recorded deformation sequence and capture changes in the object's shape. Keypoints and predicted geometry share the dataset's rigid alignment to the training reference, making the displayed states comparable in a common coordinate frame. The projection uses the first two principal components of the aligned training keypoints, providing a compact view of the pose variation accompanying the reconstructed surfaces.

Additional demonstrations. Additional demonstrations and accompanying information are available on our anonymous project page: `https://anonymous.4open.science/r/Action-Differentiable-Neural-Objects`.

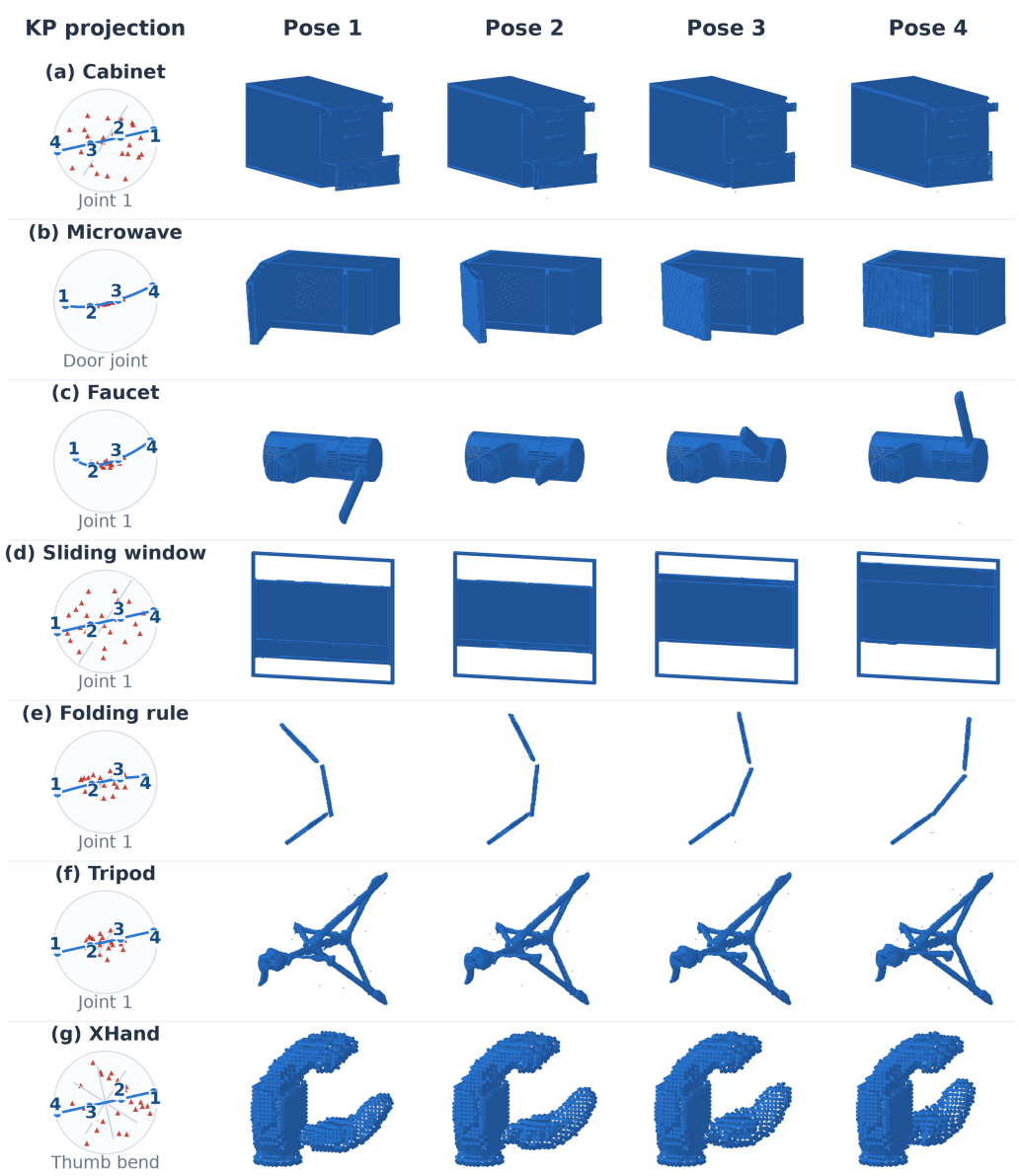

Figure 8: **Pose variation for seven articulated objects.** Red triangles mark sampled training configurations in the keypoint projection. The blue curve and numbered markers identify the four reconstructed states shown to the right.

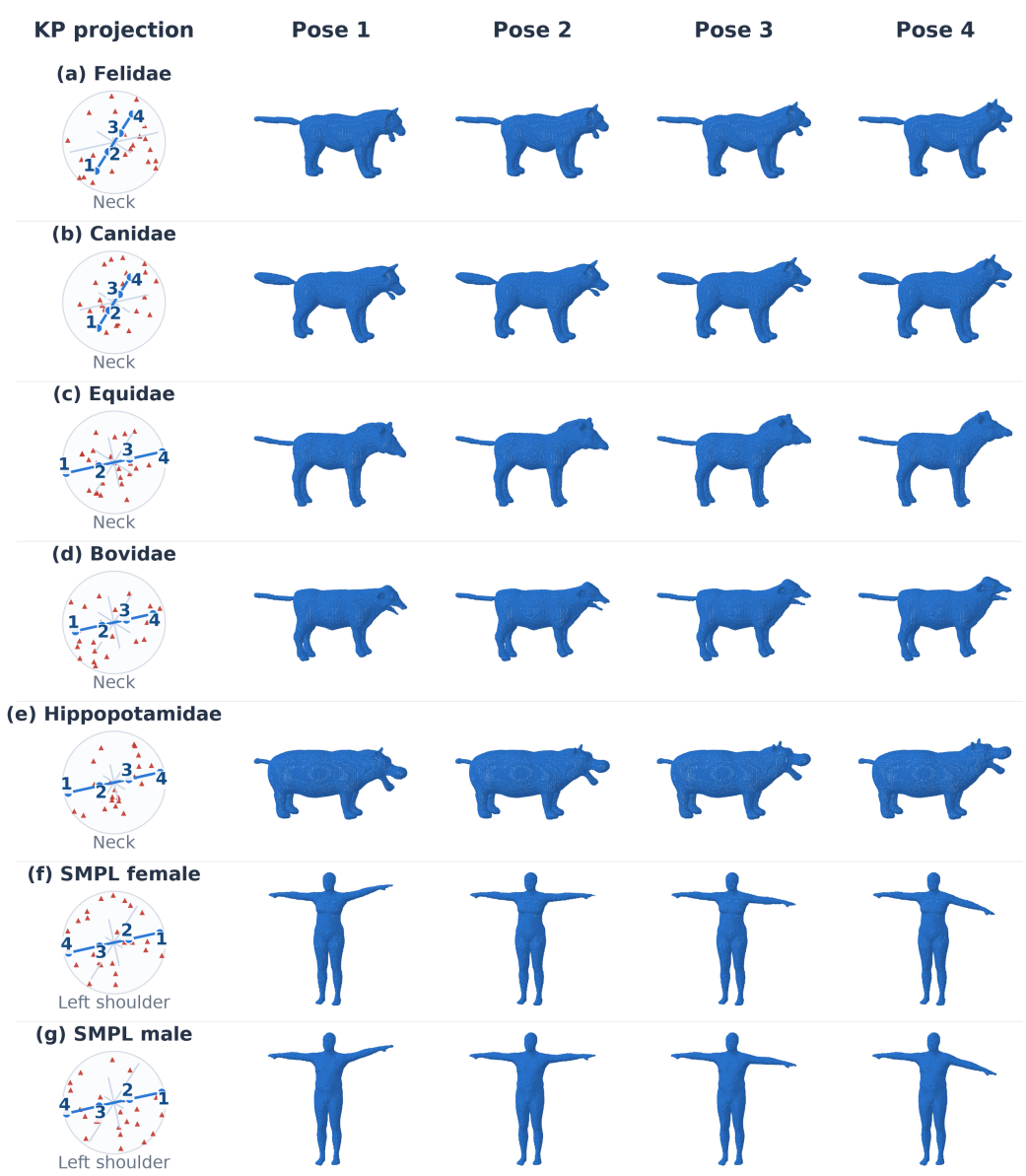

Figure 9: **Pose variation across seven skinned categories, with one identity per category.** Each animal varies its neck joint; each human varies the left shoulder. Projection markers and column numbering follow Figure 8.

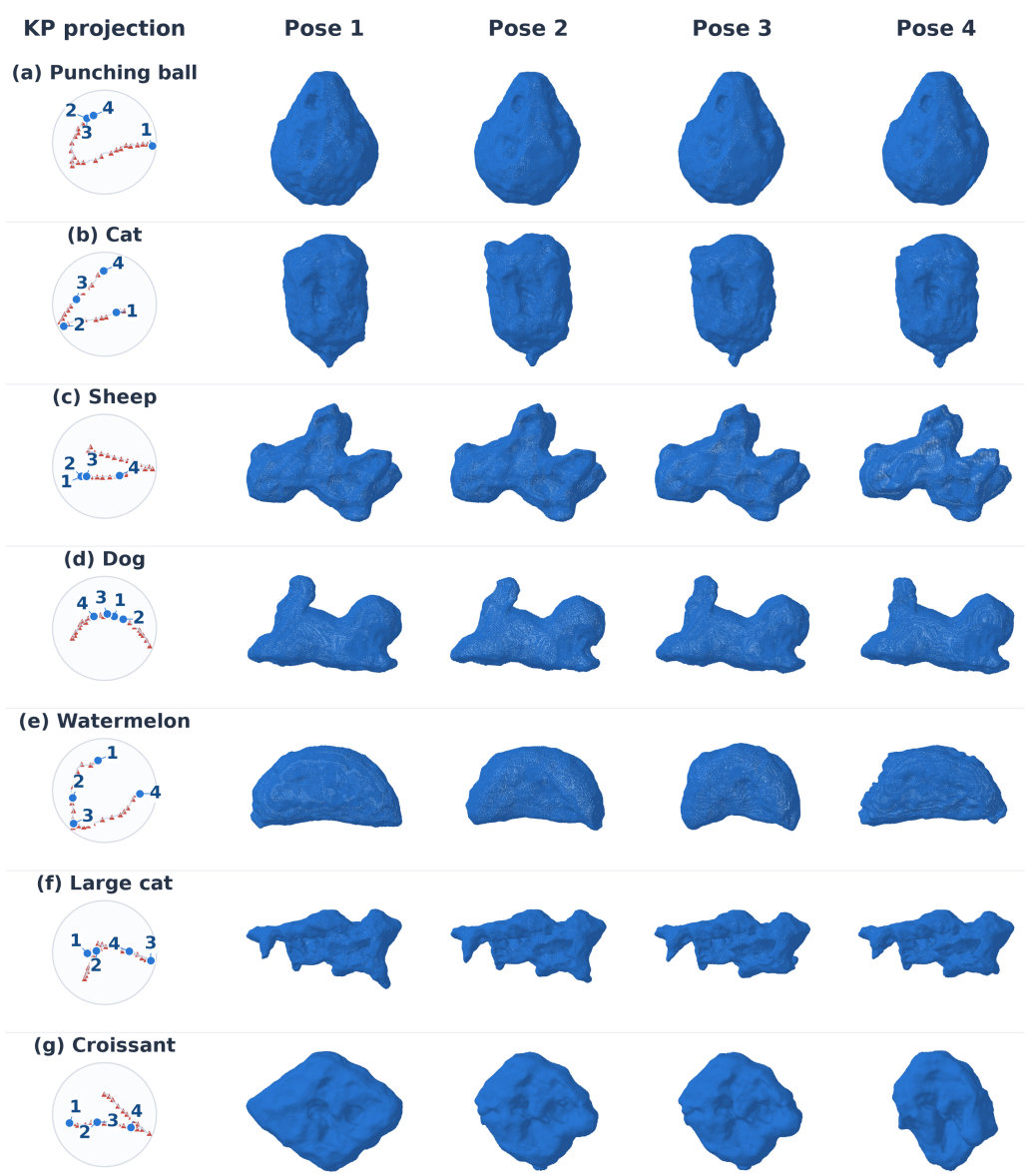

Figure 10: **Reconstruction under soft-body deformation.** Red triangles mark sampled training configurations; the gray curve traces the recorded keypoint sequence. Numbered blue markers correspond to the four reconstructed states.