Beyond Point-Attached Semantics: Object-Centric Semantic Fields for Generalizable Manipulation

Zheng SUN
Department of Mechanical of Automation Engineering
The Chinese University of Hong Kong China
zhengsun@link.cuhk.edu.hk

Lerong ZHANG
Department of Mechanical of Automation Engineering
The Chinese University of Hong Kong China
lrzhang@cuhk.edu.hk

Zhihao LI
Department of Mechanical of Automation Engineering
The Chinese University of Hong Kong China
zhli@cuhk.edu.hk

Zhuo LI
Department of Mechanical of Automation Engineering
The Chinese University of Hong Kong China
zhuoli@cuhk.edu.hk

Quentin ROUXEL
Department of Mechanical of Automation Engineering
The Chinese University of Hong Kong China
quentinrouxel@cuhk.edu.hk

Fei CHEN
Department of Mechanical of Automation Engineering
The Chinese University of Hong Kong China
f.chen@ieee.org

Abstract: Generalizable robot manipulation requires stable 3D understanding of functional object parts, such as handles, tool heads, openings, and graspable regions. Raw point clouds provide geometry but lack explicit part semantics, and their sampled points vary with viewpoint, sensor configuration, and object instance. Existing 2D feature lifting and discrete 3D point-wise features enrich point clouds with semantics, but the resulting features remain attached to observation-dependent samples. We propose an object-centric continuous semantic field that conditions on an object point cloud and reads part-aware semantic embeddings at explicit 3D query locations. The field is trained from part-annotated object models and then frozen to generate semantic point clouds as object-level conditioning for manipulation policies. Experiments on RoboTwin simulation tasks and real-world bimanual object manipulation show that our representation provides more stable functional-part cues and improves policy performance over raw point-cloud, 2D feature lifting, and 3D point-wise feature baselines. Project Page: https://zainzh.github.io/beyond-point-attached-semantics.

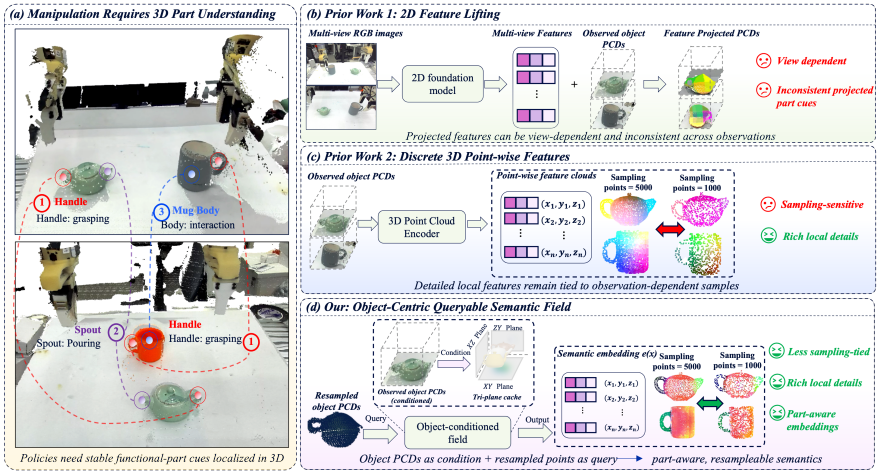

Figure 1: Teaser of our object-centric continuous semantic field. Existing 2D feature lifting and discrete 3D point-wise features attach semantics to observation-dependent samples. We instead condition a queryable semantic field on the object point cloud and read part-aware embeddings at explicit 3D query locations, producing semantic point clouds for downstream policy learning.

Keywords: Object-Centric Representation Learning; Continuous Semantic Fields; Generalizable Robotic Manipulation

1 Introduction

Robot manipulation is executed in 3D: grasps, contacts, and tool motions must be generated relative to object geometry observed through depth sensors. To generalize across object instances, a policy must understand functional object parts rather than object categories alone. Parts such as handles, tool heads, openings, and contact regions can vary in shape, scale, and location, while preserving their interaction roles [1, 2, 3]. Learning stable part-level 3D object representations is therefore important for cross-instance generalization and for manipulation policies that can operate beyond the training object distribution.

Raw point clouds are a natural 3D input for manipulation policies [4, 5], but they are an unstable basis for learning such part-level cues. First, raw points encode geometry without explicit part semantics, forcing policies to infer task-relevant parts and cross-instance correspondences from demonstrations. Second, the sampled points themselves depend on viewpoint, sensor configuration, and instance geometry [6]. As a result, policies trained directly on raw point clouds must learn action generation while also handling semantic ambiguity and sampling variability.

A common strategy is to enrich point clouds with semantic features. Prior works project features from 2D vision or vision-language models into 3D [7, 8, 9], or predict dense point-wise semantic and affordance features on the input points [10, 11]. These methods improve the semantic content of point clouds, but the features usually remain attached to the observed discrete samples. When the raw point distribution changes, the semantic representation presented to the policy changes with it. Moreover, features obtained from view-dependent images or local 3D neighborhoods can still vary with viewpoint, projection quality, and local sampling.

Our key insight is that object representations for manipulation should not merely attach semantic features to the currently observed points. Instead, they should separate object conditioning from semantic readout. We use the observed object point cloud as a geometric condition for the current instance, and use explicit 3D query positions to read out semantic embeddings. This condition-query separation allows semantic features to be generated at controlled object locations rather than only at raw sensor samples. Fig. 1 illustrates this shift from point-attached semantic features to an object-conditioned queryable semantic field.

Based on this idea, we propose an object-centric continuous semantic field for part-aware object representation learning. The field is trained using part-annotated object models from PartNext [12]. In our experiments, we train one category-level field for each object family, with part labels shared

across instances of that family. Given support points from an object instance, the model builds an object-conditioned tri-plane feature cache and predicts semantic embeddings and training-time part logits at explicit query locations. We supervise the queried outputs with part anchoring, cross-instance part alignment, and augmentation stability, which respectively ground predictions in part labels, align corresponding parts across instances, and improve embedding stability under support perturbations.

After training, the semantic field is frozen and used as an object-level representation module for downstream policy learning. At each policy step, we query the field at resampled object locations and pair each query coordinate with its semantic embedding to form a semantic point cloud. This representation preserves 3D spatial structure while providing functional-part cues to the policy. We use it as an additional point-cloud modality without changing the policy objective or action representation.

We evaluate our approach on four RoboTwin simulation tasks and four real-world manipulation tasks. Compared with raw point-cloud policies, 2D feature lifting, and discrete 3D point-wise features, our representation provides more stable part-level conditioning and improves policy performance, especially on tasks that require functional-part localization. Our main contributions are:

- We propose an object-centric continuous semantic field that represents functional object parts as queryable 3D semantic embeddings, enabling semantic readout at explicit object locations rather than only at observed sensor points.
- We train category-level fields from part-annotated object models using part anchoring, cross-instance part alignment, and augmentation stability, producing embeddings that are stable and aligned across object instances.
- We export the learned field as policy-ready semantic point clouds and validate its benefits on multi-task and cross-instance manipulation in both RoboTwin simulation and real-world bimanual robot experiments.

2 Related Work

2.1 3D Policy Learning for Robotic Manipulation

Imitation learning policies, including diffusion-based policies, have achieved strong performance in robotic manipulation by learning action distributions from demonstrations [13, 14, 15]. To better capture spatial interactions, recent methods condition policies on 3D observations such as point clouds [4, 5], voxel grids [16], or multi-view representations [6, 17, 18, 19]. These methods encode raw geometry using PointNets [20], sparse 3D representations, or 3D Transformers [21]. However, the object representation provided to the policy is often still tied to observation-dependent samples, leaving the policy to infer functional parts and cross-instance correspondences from task demonstrations. Our work focuses on this representation bottleneck: instead of changing the policy backbone, we provide a queryable, part-aware object representation as policy conditioning.

2.2 Part-aware and Semantic Object Representations

A large body of work enriches 3D observations with semantic or actionable features. Some methods lift features from 2D vision foundation models, such as DINOv2 [22], into 3D for open-vocabulary or language-aware perception [23, 24, 7, 8, 9]. Other methods predict dense point-wise semantic, part, or affordance features directly on the input geometry [25, 26, 11]. A closely related concurrent work learns part-aware 3D features for generalizable manipulation through geometric contrastive learning and semantic alignment [10]. These methods improve the semantic expressiveness of 3D observations, but their features are primarily attached to observed discrete points.

Continuous object representations provide a queryable alternative. Neural descriptor fields and related object-centric fields have been used for correspondence, pose alignment, and category-level

manipulation [1, 2]. However, these fields are typically designed for descriptor matching or planning, rather than for producing part-level semantic embeddings as direct policy inputs. In contrast, our method learns an object-conditioned continuous semantic field from part-annotated object models and exports queried embeddings as semantic point clouds for downstream policy learning.

3 Method

We propose an object-centric continuous semantic field for learning part-aware object representations and using them as policy conditioning for robotic manipulation. As illustrated in Fig. 2, our method treats the observed object point cloud as a geometric condition and reads semantic embeddings at explicit 3D query locations. This separates where object geometry is observed from where semantic features are evaluated, allowing the downstream policy to condition on resampled, part-aware semantic point clouds rather than only on raw observation samples.

3.1 Problem Setup and Training Assumptions

We learn a category-level object semantic field for each object family used in our experiments. Given an object point cloud, the field is conditioned on the geometry of the current object instance and can be queried at arbitrary 3D locations to produce part-aware semantic embeddings. In this work, we train the semantic field separately for each object family, while requiring the part labels to be consistent across instances within that family. The semantic field is trained using part-annotated object models from PartNext [12], while the downstream policy is trained separately from task demonstrations collected in simulation or on the real robot.

For each object instance, we separate the object representation into *support points* and *query points*. The support set $\mathcal{P}_{sup} = \{p_i \in \mathbb{R}^3\}_{i=1}^{N_s}$ provides the geometric condition of the current object instance. During semantic field training, support points are sampled from the object surface and used only to construct the object-conditioned field. During policy learning and deployment, support points are sampled from the observed object point cloud. The support points themselves are not the final representation consumed by the policy; they serve as the condition from which the field is built.

The query set $\mathcal{Q} = \{x_j \in \mathbb{R}^3\}_{j=1}^{N_q}$ specifies the 3D locations where semantic outputs are evaluated. During semantic field training, query points are sampled from labeled object surfaces, and each valid query point has a part label $y_j \in \{1, \dots, C\}$ used for supervision. During policy learning and deployment, part labels are not required. We resample object locations as query points and use the field to produce semantic embeddings at these locations. Thus, support points describe the observed object instance, while query points define where semantic information is read out. This separation allows semantic features to be generated at controlled query locations rather than being restricted to the original sensor samples. Formally, the semantic field is defined as

$$f_\theta(x \mid \mathcal{P}_{sup}) \to (e_x, \ell_x), \tag{1}$$

where $x \in \mathbb{R}^3$ is a query coordinate, $e_x \in \mathbb{R}^d$ is an L2-normalized semantic embedding, and $\ell_x \in \mathbb{R}^C$ denotes logits over the part label set of the corresponding object category. The logits are used only for training supervision, while the semantic embedding is used as the object-level representation for downstream policy learning.

3.2 Object-Conditioned Continuous Semantic Field

Support encoding. Given the support set $\mathcal{P}_{sup}$, we extract point-wise geometric features using a frozen pre-trained 3D backbone and map them into the semantic field space with a lightweight adapter:

$$\mathcal{F}_{sup} = E(\mathcal{P}_{sup}), \qquad \tilde{\mathcal{F}}_{sup} = g_\phi(\mathcal{F}_{sup}), \tag{2}$$

where $\mathcal{F}_{sup} = \{f_i \in \mathbb{R}^{d_u}\}_{i=1}^{N_s}$ and $\tilde{\mathcal{F}}_{sup} = \{\tilde{f}_i \in \mathbb{R}^{d_s}\}_{i=1}^{N_s}$. We use Utonia [27] as $E(\cdot)$ and keep it frozen, so that semantic field learning builds on general 3D geometric priors rather than relearning low-level geometry.

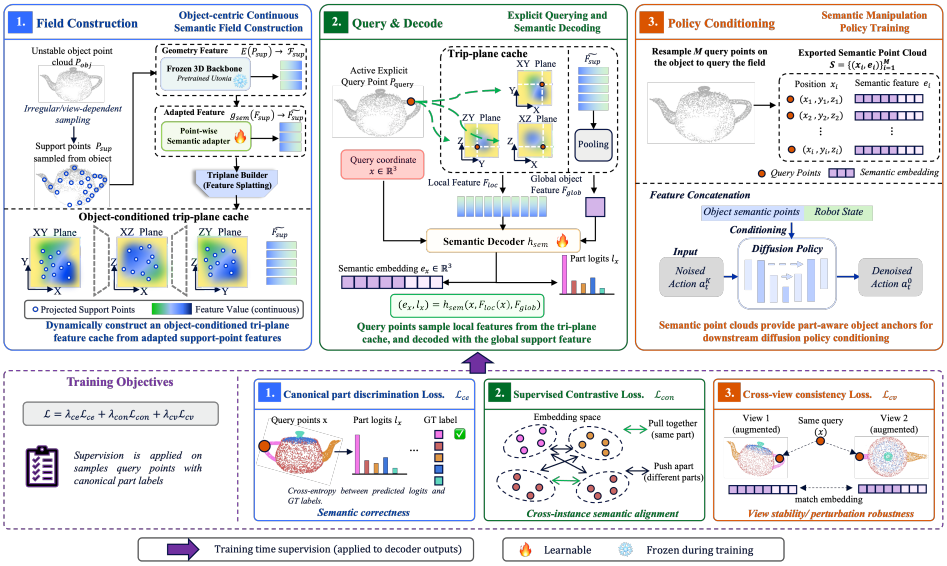

Figure 2: Overview of our object-centric continuous semantic field. Support points condition an object-specific tri-plane cache; 3D query locations read out part-aware embeddings and training-time part logits. The frozen field is then queried at resampled object locations to export semantic point clouds for downstream manipulation policies.

Tri-plane feature cache. To make the representation queryable, we aggregate the adapted support features into an object-conditioned tri-plane cache:

$$\mathcal{T}_{obj} = \Pi(\mathcal{P}_{sup}, \tilde{\mathcal{F}}_{sup}) = \{T_{xy}, T_{xz}, T_{yz}, F_{glob}\}. \tag{3}$$

Before aggregation, support coordinates are normalized to the object frame. The three planes $T_{xy}, T_{xz}, T_{yz} \in \mathbb{R}^{H \times W \times d_t}$ are 2D feature maps defined on the xy, xz, and yz projections, respectively. The global feature $F_{glob} \in \mathbb{R}^{d_g}$ is obtained by pooling support features. This cache is built separately for each object instance and stores the support-conditioned geometry and semantic cues used for querying.

Query decoding. Given the tri-plane cache $\mathcal{T}_{obj}$ and a query coordinate x, we project x onto the three planes and bilinearly sample the corresponding plane features. The sampled features are fused into a local feature $F_{loc}(x)$, which is decoded together with the global feature and normalized coordinate:

$$(e_x, \ell_x) = h_\theta\left(x, F_{loc}(x), F_{glob}\right). \tag{4}$$

The local feature provides support-conditioned information around the query location, while F_{glob} summarizes the object instance. We include the normalized coordinate x as positional context in the object frame, which helps disambiguate spatially similar local patterns.

3.3 Semantic Field Learning Objectives

We supervise only the queried outputs of the field; support points are used to condition the object instance. Let $\mathcal{Q}_{valid}$ be the set of labeled query points and $y_x \in \{1, \ldots, C\}$ be the part label of query point x. For each training object, we generate two augmented support conditions $\mathcal{V}$ and optimize a weighted sum of three objectives:

$$\mathcal{L} = \lambda_{part}\mathcal{L}_{part} + \lambda_{align}\mathcal{L}_{align} + \lambda_{stab}\mathcal{L}_{stab}. \tag{5}$$

Part anchoring. We implement this objective with cross-entropy supervision on the part logits:

$$\mathcal{L}_{part} = \frac{1}{|\mathcal{V}|}\sum_{v \in \mathcal{V}} \frac{1}{|\mathcal{Q}_{valid}|} \sum_{x \in \mathcal{Q}_{valid}} \mathrm{CE}(\ell_x^{(v)}, y_x). \tag{6}$$

This objective anchors the queried outputs to the predefined part labels of the object family, making each field prediction semantically identifiable.

Table 1: Simulation success rates on four RoboTwin manipulation tasks.

Method	Hang Mug	Beat Hammer	Open Micro.	Put Cabinet
DP3 [5]	24%	74%	22%	72%
2D Lifting [22]	23%	78%	21%	60%
3D Point-wise [27]	31%	77%	26%	76%
Ours	**37**%	**84**%	**35**%	**83**%

Cross-instance part alignment. Part anchoring supervises logits but does not directly organize the embedding space used by the policy. We implement this objective with supervised contrastive learning on the normalized embeddings. For an anchor query i, let $P(i)$ denote valid queries with the same part label, excluding the anchor:

$$\mathcal{L}_{align} = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \frac{1}{|\mathcal{I}_v|} \sum_{i \in \mathcal{I}_v} \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp((e_i^{(v)})^\top e_p^{(v)}/\tau)}{\sum_{a \neq i} \exp((e_i^{(v)})^\top e_a^{(v)}/\tau)}, \tag{7}$$

where $\mathcal{I}_v$ contains anchors with at least one positive sample, $P(i)$ excludes the anchor, the denominator is computed over valid query embeddings in the same batch and augmentation, and τ is a temperature parameter. This objective pulls together embeddings of the same part and separates different parts. Since part labels are consistent across instances within each object family, it encourages corresponding parts from different instances to align in the embedding space.

Augmentation stability. To improve stability, we enforce embedding consistency for the same query coordinate under two augmented support conditions while keeping the query coordinate fixed in the normalized object frame. Let $e_x^{(1)}$ and $e_x^{(2)}$ be the embeddings predicted at query point x when conditioning on two augmentations of the same object instance. We minimize

$$\mathcal{L}_{stab} = \frac{1}{|\mathcal{Q}_{valid}|} \sum_{x \in \mathcal{Q}_{valid}} \left(1 - \langle e_x^{(1)}, e_x^{(2)} \rangle\right). \tag{8}$$

This objective does not add new semantic labels; it regularizes the field to produce stable embeddings under support perturbations and small geometric transformations. Together, $\mathcal{L}_{part}$ anchors queried outputs to part semantics, $\mathcal{L}_{align}$ aligns corresponding parts across instances, and $\mathcal{L}_{stab}$ improves representation stability.

3.4 Semantic Point Clouds for Policy Learning

After semantic field training, we freeze the field and use it as an object-level representation module for policy learning. At each policy step, an observed object point cloud $\mathcal{P}_{obj}$ is used to construct the support set $\mathcal{P}_{sup}$ and the corresponding tri-plane cache. We then resample M object query locations $\{x_i\}_{i=1}^M$ from the observed object region and evaluate the embedding branch of the field, producing a semantic point cloud

$$\mathcal{S}_{obj} = \{(x_i, f_\theta^{emb}(x_i \mid \mathcal{P}_{sup}))\}_{i=1}^M. \tag{9}$$

where each point stores its 3D location and queried semantic embedding. The part logits are discarded at this stage; the policy only receives semantic embeddings. For downstream manipulation, we provide these semantic point clouds as additional object-level observations. The policy observation at time t is

$$o_t = (\mathcal{C}_t, \{\mathcal{S}_t^k\}_{k=1}^K, q_t), \tag{10}$$

where $\mathcal{C}_t$ is the scene point cloud, $\mathcal{S}_t^k$ is the semantic point cloud of the k-th target object, and q_t is the robot proprioceptive state. The scene point cloud preserves global geometry and spatial context, while the semantic point clouds provide part-aware object cues. We use DP3 as the point-cloud policy backbone and treat $\mathcal{S}_t^k$ as an additional point-cloud modality. The training objective and action representation are unchanged. Thus, the policy input is generated by an explicit resampling-and-querying step, rather than being limited to features attached to the original observed points.

Table 2: Real-world success rates on four manipulation tasks.

Method	Grasp Mug	Beat Cube	Stir Mug	Pour Water
DP3 [5]	7/20	8/20	3/20	4/20
3D Point-wise [27]	7/20	9/20	7/20	5/20
Ours	**17/20**	**17/20**	**10/20**	**10/20**

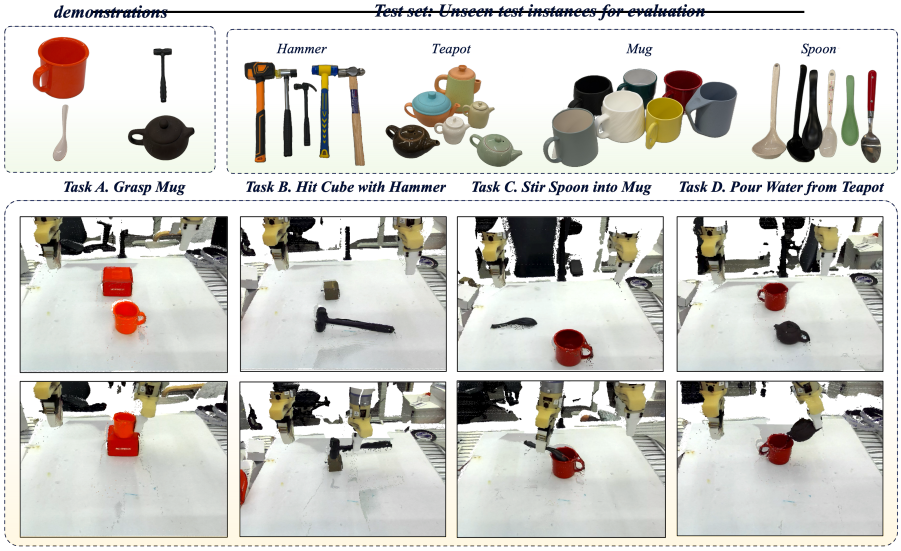

Figure 3: Real-world tasks and object splits. Policies are trained on training object instances and evaluated on held-out test instances for mug grasping, tool-object contact, grasping and stirring, and pouring-related manipulation.

4 Experiments

We evaluate the proposed object-centric continuous semantic field through both simulation and real-world robotic experiments. Our experiments are designed to answer three questions: (1) whether the proposed representation improves the success rate of multi-task manipulation policies; (2) whether it improves policy generalization to unseen object instances; and (3) whether the learned semantic embeddings exhibit stronger cross-instance part consistency than 2D feature lifting and discrete 3D point-wise features.

4.1 Experiment Setup

We evaluate our method on four RoboTwin simulation tasks [28] and four real-world bimanual manipulation tasks. The tasks cover mug hanging or grasping, tool-object contact, stirring, and pouring-related manipulation, all of which require localizing functional object parts such as handles, tool heads, openings, or graspable regions.

We use DP3 [5] as the point-cloud imitation learning policy backbone in all experiments. In simulation, task demonstrations are generated in RoboTwin, and we compare four object representations: raw point-cloud **DP3**, **2D Feature Lifting**, **3D Point-wise Features**, and **Ours**. The DP3 baseline uses raw point clouds as 3D observations. The 2D Feature Lifting baseline projects DINOv2 [22] features from RGB observations onto 3D points. The 3D Point-wise Features baseline extracts dense point-wise features from observed points using the same frozen Utonia encoder [27] as our method.

For real-world experiments, demonstrations are collected on our bimanual robot using training object instances, and policies are evaluated on held-out instances with different geometry and appearance. We compare **DP3**, **3D Point-wise Features**, and **Ours**. Across all comparisons, methods share the same demonstrations, policy backbone, training pipeline, and success criteria; only the object-level visual representation differs. Detailed task definitions, object splits, hyperparameters, and success criteria are provided in the appendix.

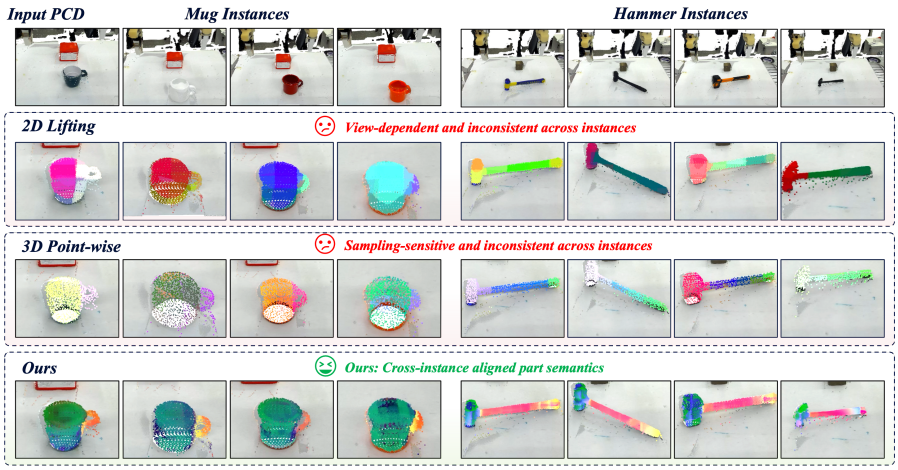

Figure 4: Cross-instance feature visualization on real observations. Colors are obtained from a shared PCA per method and category. Compared with 2D lifting and 3D point-wise features, our field yields more consistent part-level colors across mug and hammer instances.

4.2 Quantitative Results

Simulation Results. Table 1 reports success rates on four RoboTwin simulation tasks. Compared with the raw point-cloud DP3 baseline, adding semantic features through 2D feature lifting or frozen 3D point-wise features does not lead to consistent improvements across tasks. For example, 2D lifting improves performance on Beat Block Hammer but degrades performance on Put Object Cabinet, while 3D point-wise features provide moderate gains but remain limited on tasks requiring precise functional-part localization. These results are consistent with our hypothesis that semantic enrichment alone is insufficient when the features remain attached to observation-dependent point samples. In contrast, our method improves performance on all four tasks, suggesting that querying an object-conditioned semantic field at resampled object locations provides more stable and policy-useful part-aware conditioning.

Real-World Evaluation. Table 2 reports real-world success rates, and Figure 3 shows the corresponding task setup and grouped object instances. Real-world observations introduce stronger sampling variation than simulation due to sensor noise, partial views, calibration error, and appearance and geometry changes. Under these conditions, the gap between our method and the baselines becomes more pronounced. While DP3 and 3D Point-wise Features achieve limited success on several tasks, our method substantially improves performance across all four real-world tasks. This suggests that the learned semantic field provides more stable functional-part cues under real observation noise. We further analyze the cross-instance stability of the learned embeddings in Section 4.3.

4.3 Representation Analysis

To qualitatively analyze the learned representations, Figure 4 visualizes feature embeddings on four real mug instances and four real hammer instances. All features are computed from real robot observations; for clarity, we crop the target object regions for visualization. For each method and object category, we fit a shared PCA projection over embeddings from the four instances and map the first three principal components to RGB. Therefore, colors are comparable across instances within the same method-category row, but not necessarily across different methods.

The 2D Feature Lifting and 3D Point-wise Feature baselines produce meaningful local feature variations on individual instances, but the color patterns of corresponding functional parts are less consistent across instances. For example, mug handles and hammer heads do not always preserve stable colors across different object instances. In contrast, our method produces more consistent feature patterns for mug bodies and handles, as well as for hammer heads and handles. This qualitative result suggests that the learned continuous semantic field better aligns part-level embeddings across instances, providing more stable functional-part cues for downstream policy learning.

5 Limitations

Our method assumes rigid objects with stable geometry and semantically meaningful functional parts, and is not directly applicable to deformable objects, soft bodies, or objects whose topology changes substantially during interaction. It also relies on discrete canonical part supervision, which may be limited for ambiguous, continuous, or task-dependent functional regions. Finally, the current field mainly captures part-level semantics and does not explicitly model fine-grained geometry or physical properties. Future work could integrate geometric representations and physical attributes to support more general object understanding and manipulation.

6 Conclusion

We presented an object-centric continuous semantic field for learning queryable part-level object representations for manipulation. By treating the object point cloud as a geometric condition and reading semantic embeddings at explicit query positions, our method converts observation-dependent point clouds into resampleable semantic point clouds with cross-instance aligned functional-part cues. The learned field is frozen and integrated into downstream manipulation policies as an object-level conditioning module. Experiments in simulation and the real world show improved multi-task performance and cross-instance generalization over raw point-cloud and point-wise semantic baselines, suggesting that queryable object-centric semantics provide an effective representation foundation for generalizable robotic manipulation.

References

[1] A. Simeonov, Y. Du, A. Tagliasacchi, J. B. Tenenbaum, A. Rodriguez, P. Agrawal, and V. Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6394–6400. IEEE, 2022.

[2] C. Pan, B. Okorn, H. Zhang, B. Eisner, and D. Held. Tax-pose: Task-specific cross-pose estimation for robot manipulation. In *Conference on Robot Learning*, pages 1783–1792. PMLR, 2023.

[3] C. Tang, A. Xiao, Y. Deng, T. Hu, W. Dong, H. Zhang, D. Hsu, and H. Zhang. Functo: Function-centric one-shot imitation learning for tool manipulation. *arXiv preprint arXiv:2502.11744*, 2025.

[4] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

[5] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. *arXiv preprint arXiv:2402.10885*, 2024.

[6] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y.-W. Chao, and D. Fox. Rvt: Robotic view transformer for 3d object manipulation. In *Conference on Robot Learning*, pages 694–710. PMLR, 2023.

[7] T. Chen, Y. Mu, Z. Liang, Z. Chen, S. Peng, Q. Chen, M. Xu, R. Hu, H. Zhang, X. Li, et al. G3flow: Generative 3d semantic flow for pose-aware and generalizable object manipulation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1735–1744, 2025.

[8] X. Fan, S. Deng, X. Wu, Y. Lu, Z. Li, M. Yan, Y. Zhang, Z. Zhang, H. Wang, and H. Zhao. Any3d-vla: Enhancing vla robustness via diverse point clouds. *arXiv preprint arXiv:2602.00807*, 2026.

[9] Y. Wang, G. Yin, B. Huang, T. Kelestemur, J. Wang, and Y. Li. Gendp: 3d semantic fields for category-level generalizable diffusion policy. In *CoRL*, pages 4866–4878, 2024.

[10] Y. Chen, M. Jiang, K. Zheng, J. Liang, C. Tie, H. Lu, R. Wu, and H. Dong. Learning part-aware dense 3d feature field for generalizable articulated object manipulation. *arXiv preprint arXiv:2602.14193*, 2026.

[11] C. Xu, H. Li, S. Cheng, J. Hu, H. Fan, Z. Feng, and S. Liu. Action-geometry prediction with 3d geometric prior for bimanual manipulation. *arXiv preprint arXiv:2602.23814*, 2026.

[12] P. Wang, Y. He, X. Lv, Y. Zhou, L. Xu, J. Yu, and J. Gu. Partnext: A next-generation dataset for fine-grained and hierarchical 3d part understanding. *Advances in Neural Information Processing Systems*, 38, 2026.

[13] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

[14] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[15] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson. Implicit behavioral cloning. In *Conference on robot learning*, pages 158–168. PMLR, 2022.

[16] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

[17] H. Chen, C. Zhu, Y. Li, and K. Driggs-Campbell. Tool-as-interface: Learning robot policies from human tool usage through imitation learning. *arXiv e-prints*, pages arXiv–2504, 2025.

[18] Z. Li, J. Liu, Z. Li, Z. Dong, T. Teng, Y. Ou, D. Caldwell, and F. Chen. Language-guided dexterous functional grasping by llm generated grasp functionality and synergy for humanoid manipulation. *IEEE Transactions on Automation Science and Engineering*, 22:10506–10519, 2025.

[19] T. Tian, X. Kang, and Y.-L. Kuo. O3Afford: One-shot 3d object-to-object affordance grounding for generalizable robotic manipulation. *arXiv preprint arXiv:2509.06233*, 2025.

[20] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[21] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021.

[22] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[23] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola. Distilled feature fields enable few-shot language-guided manipulation. *arXiv preprint arXiv:2308.07931*, 2023.

[24] Y. Wang, M. Zhang, Z. Li, K. R. Driggs-Campbell, J. Wu, L. Fei-Fei, and Y. Li. D3Fields: Dynamic 3d descriptor fields for zero-shot generalizable robotic manipulation. In *ICRA 2024 Workshop on 3D Visual Representations for Robot Manipulation*, 2023.

[25] K. Mo, L. J. Guibas, M. Mukadam, A. Gupta, and S. Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6813–6823, 2021.

[26] R. Wu, Y. Zhao, K. Mo, Z. Guo, Y. Wang, T. Wu, Q. Fan, X. Chen, L. Guibas, and H. Dong. Vat-mart: Learning visual action trajectory proposals for manipulating 3d articulated objects. *arXiv preprint arXiv:2106.14440*, 2021.

[27] Y. Zhang, X. Wu, Y. Yang, X. Fan, H. Li, Y. Zhang, Z. Huang, N. Wang, and H. Zhao. Utonia: Toward one encoder for all point clouds. *arXiv preprint arXiv:2603.03283*, 2026.

[28] T. Chen, Z. Chen, B. Chen, Z. Cai, Y. Liu, Z. Li, Q. Liang, X. Lin, Y. Ge, Z. Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

[29] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. URL https://arxiv.org/abs/2408.00714.

Figure 5: Simulation task examples used in our evaluation. These tasks require localizing functional object parts, such as mug handles, hammer heads, articulated handles, and placement regions, to complete object-centric manipulation.

A Task Definitions and Success Criteria

We provide additional details for the simulation and real-world experiments. All tasks require the robot to identify and interact with task-relevant functional parts, rather than relying only on object-level category recognition. We use the same task initialization protocol, maximum execution horizon, and success criteria for all compared methods.

Simulation tasks.

- **Hanging Mug.** The robot grasps the rim or another stable graspable region of a mug and hangs the mug by placing its handle onto a target rack. This task requires identifying both a reliable grasping region and the mug handle used for hanging. A trial is successful if the mug is stably hung on the target rack without falling.
- **Beat Block with Hammer.** The robot grasps the handle of a hammer and uses the hammer head to hit a target block. The task requires distinguishing the hammer handle from the hammer head and generating effective contact with the hammer head. A trial is successful if the hammer head contacts the block and moves it to the target state.
- **Open Microwave.** The robot opens a microwave door by interacting with the door handle or another valid interactive region. The task requires localizing the articulated interaction region and applying motion in the correct direction. A trial is successful if the microwave door is opened beyond the predefined angle threshold.
- **Put Object into Cabinet.** The robot first interacts with the cabinet handle to open the cabinet, and then places the target object into the cabinet or target storage region. This task requires identifying the cabinet handle as the functional interaction region and completing the subsequent placement. A trial is successful if the cabinet is opened and the object is placed inside the target cabinet region without falling.

Real-world tasks.

- **Grasp Mug.** The robot grasps a mug from the table and lifts it stably. This task requires identifying a reliable graspable region, such as the mug handle, rim, or body. A trial is successful if the robot lifts the mug above a height threshold without slipping or dropping it.
- **Beat Cube.** The robot grasps a hammer and uses the hammer head to hit a target cube. This task requires distinguishing the hammer handle from the hammer head and generating contact with the hammer head rather than the handle or side surface. A trial is successful if the hammer head contacts the cube and causes visible displacement or reaches the target state.
- **Stir Mug.** The robot grasps a stirring tool and inserts its tip into a mug to perform a stirring motion. This task requires identifying the graspable region and functional tip of the tool, as well as the opening of the mug. A trial is successful if the tool tip enters the mug and completes the stirring motion without losing control or colliding severely with the mug.

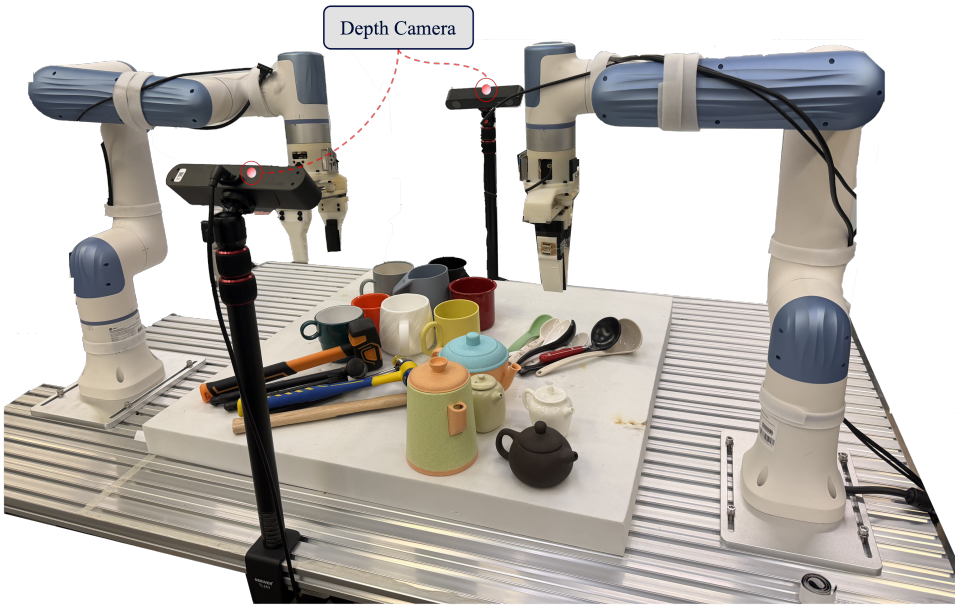

Figure 6: Real-world experimental setup. The bimanual robot platform is equipped with two calibrated ZED 2i RGB-D cameras, and the evaluated objects include mugs, hammers, stirring tools, and containers used in the real-world manipulation tasks.

- **Pour Water.** The robot grasps a container and pours its contents into a target cup or target region. This task requires identifying a stable grasp region, the opening or pouring direction of the container, and the target container location. A trial is successful if the container is tilted toward the target and the contents enter the target cup or region.

B Real-World Setup and Point-Cloud Processing

For the real-world experiments, we use a bimanual robot platform equipped with two calibrated ZED 2i RGB-D cameras. Figure 6 shows the robot setup and the object instances used in our real-world evaluation. At each policy step, we reconstruct the scene point cloud from calibrated RGB-D observations and extract object-level point clouds for the task-relevant objects.

To obtain object point clouds, we use text-prompted real-time tracking and segmentation with SAM2 [29]. For each target object, a text prompt specifies the object of interest in the RGB observations. SAM2 tracks the corresponding object masks over time and across camera views. The segmented multi-view RGB-D observations are then back-projected into 3D using camera intrinsics and extrinsics, producing object-specific point clouds in the world coordinate frame. These object point clouds are used both as raw object observations for point-wise baselines and as the support condition for our semantic field. All compared methods share the same RGB-D observations, calibration, segmentation masks, and point-cloud extraction pipeline.

C Representation and Policy Implementation Details

Semantic field training. We train the semantic field from part-annotated PartNext object models [12]. For each object family, all training instances use a consistent part label set, such as handle/body for mugs or head/handle for hammers. The Utonia backbone is initialized from the public check-

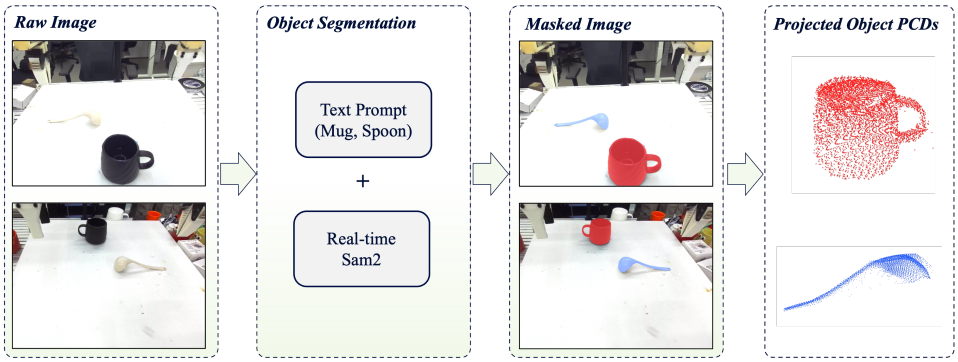

Figure 7: Object point-cloud extraction pipeline. Text prompts specify task-relevant objects in multi-view RGB observations; SAM2 tracks and segments the objects over time; the masked RGB-D observations are back-projected with camera calibration to obtain object-specific point clouds for policy input and semantic-field conditioning.

point and kept frozen; the trainable components are the lightweight feature adapter, tri-plane feature fusion module, and query decoder. During training, support points are sampled from the object surface to condition the field, and query points are sampled from labeled surface locations to provide part supervision. For augmentation stability, we generate two augmented support conditions from the same object using independent SO(3) rotations and Gaussian coordinate jitter, while supervising corresponding query points across the two views. Table 3 summarizes the main representation training and architecture settings.

Table 3: Semantic field training and architecture settings used in our experiments.

Component	Setting
Dataset	PartNext meshes with part annotations
Trainable components	Feature adapter, tri-plane fusion, and query decoder
Backbone	Frozen Utonia checkpoint
Support points	5000 surface points per object
Query points	2048 labeled surface points per object
Coordinate augmentation	SO(3) rotation and Gaussian jitter, $\sigma = 0.005$
Tri-plane cache	Resolution 64, padding 0.05
Adapter / branch dimension	256 / 256
Semantic embedding dimension	128, L2-normalized
Query decoder	Two LayerNorm–GELU MLP layers with embedding and logit heads
Optimizer	AdamW, learning rate 10^{-3}, weight decay 10^{-4}
Batch size / epochs	6 / 4000
Loss weights	$\lambda_{part} = 0.5$, $\lambda_{align} = 0.2$, $\lambda_{stab} = 0.1$
Mixed precision	Enabled

Semantic point-cloud export. After semantic field training, the field is frozen and used only as a representation module. For each observed object point cloud, we first remove invalid zero points and use the remaining object points as the support condition. We then sample a fixed number of object query locations and evaluate the embedding output of the field. In the default policy preprocessing, we export 256 query points per semantic object, and each exported point stores its world-frame coordinate concatenated with a 128-dimensional semantic embedding, yielding an object-level semantic point cloud of shape $256 \times (3 + 128)$. Part logits are not passed to the policy; they are used only during representation training and optional visualization.

Policy integration. We use DP3 as the policy backbone and keep its diffusion objective unchanged. In the observation encoder the scene point cloud and each exported semantic point cloud are treated as separate point-cloud modalities. The scene point cloud provides global geometry, while each semantic point cloud provides XYZ coordinates concatenated with semantic embeddings

for a target object. The encoded point-cloud features are concatenated with the robot proprioceptive feature to form the conditioning vector for the diffusion policy. Table 4 summarizes the downstream policy and semantic point-cloud preprocessing settings.

Table 4: Downstream policy and semantic point-cloud preprocessing settings.

Component	Setting
Policy backbone	DP3 point-cloud diffusion policy
Scene point cloud	1024 points
Semantic point clouds	Up to two target objects, 256 points each
Semantic point dimension	$3 + 128$ channels
Observation horizon	3 steps
Action horizon	6 steps, with total horizon 8
Action / proprioception dimension	14 / 14
Point-cloud encoder	PointNet encoder, output dimension 128 per modality
Diffusion scheduler	DDIM, 100 training steps and 10 inference steps
Policy optimizer	AdamW, learning rate 10^{-4}, weight decay 10^{-6}
Batch size / epochs	256 / 3000

Hardware. All representation and policy models are trained on a desktop workstation with a single NVIDIA RTX 4090 GPU and Ubuntu 22.04. Real-robot deployment and online policy inference are run on a laptop with an NVIDIA RTX 4070 GPU and Ubuntu 22.04. The semantic field is frozen during deployment, so online computation only requires object point-cloud extraction, field querying for the semantic point clouds, and DP3 policy inference.

Runtime and asynchronous execution. During real-world deployment, the perception and policy pipeline includes object point-cloud construction, SAM2-based object segmentation, semantic/observation encoding, and DP3 policy inference. Table 5 reports the measured latency of the main computation stages on the RTX 4070 laptop used for deployment. The end-to-end wall-clock time of a full inference update can be higher than the sum of these stages due to camera synchronization, robot I/O, data transfer, and action post-processing.

Table 5: Runtime breakdown of the main computation stages during real-world deployment.

Stage	Latency
Object point-cloud construction	$\sim$100 ms
SAM2 preprocessing and segmentation	$\sim$200 ms
Semantic / observation encoding	$\sim$170 ms
DP3 policy inference	$\sim$155 ms

Since the full perception-to-policy pipeline is slower than the low-level robot command rate, we use asynchronous execution instead of blocking robot control on every policy query. A low-frequency inference thread reads RGB-D observations, extracts object point clouds, queries the semantic field, and predicts an action chunk from the policy. In parallel, a high-frequency control thread consumes the latest available action chunk and sends smooth joint commands to the robot at a fixed servo rate. When a new chunk is not yet available, the controller holds or repeats the most recent target command as a keep-alive action. We further apply command smoothing and per-step motion limits before execution. This design decouples expensive semantic perception and policy inference from low-level command streaming, enabling continuous robot motion while using the proposed semantic field in the real-world control loop.

D Additional Semantic Feature Visualizations

Figure 8 provides additional qualitative visualizations of the semantic embeddings produced by our object-centric field on different object families. Colors are obtained by projecting the queried

Figure 8: Additional qualitative visualizations of queried semantic embeddings on mugs, hammers, microwaves, spoons, and teapots. Similar colors indicate similar embedding responses after RGB projection, highlighting consistent part-aware patterns across object instances.

embeddings to RGB for visualization only; consistent color patterns across instances indicate that the learned representation captures stable part-aware semantic structure.