GRIM: Task-Oriented Grasping with Conditioning on Generative Examples

Shailesh*, Alok Raj*, Nayan Kumar*, Priya Shukla†,
Andrew Melnik‡, Micheal Beetz‡, Gora Chand Nandi†
*IIT ISM Dhanbad, India
†IIIT Allahabad, India
‡University of Bremen, Germany

***Abstract*—Task-Oriented Grasping (TOG) presents a significant challenge, requiring a nuanced understanding of task semantics, object affordances, and the functional constraints dictating how an object should be grasped for a specific task. To address these challenges, we introduce GRIM (Grasp Re-alignment via Iterative Matching), a novel training-free framework for task-oriented grasping. Initially, a coarse alignment strategy is developed using a combination of geometric cues and principal component analysis (PCA)-reduced DINO features for similarity scoring. Subsequently, the full grasp pose associated with the retrieved memory instance is transferred to the aligned scene object and further refined against a set of task-agnostic, geometrically stable grasps generated for the scene object, prioritizing task compatibility. In contrast to existing learning-based methods, GRIM demonstrates strong generalization capabilities, achieving robust performance with only a small number of conditioning examples. Project Page**

I. Introduction

Robotic manipulation remains a fundamentally challenging problem, particularly when it involves grasping objects in a manner that is appropriate for a specific task. The ability to reliably grasp a wide variety of objects is essential—not merely for achieving geometric stability, but for enabling purposeful interaction. Task-Oriented Grasping (TOG) goes beyond conventional grasping strategies by requiring a deeper understanding of object affordances, task semantics, and the functional requirements that dictate how an object should be held to effectively accomplish a given task. Despite growing interest in Task-Oriented Grasping (TOG), the scarcity of task-annotated grasping datasets (e.g., Murali et al. [16]) limits the scalability of training-based methods. These approaches also struggle to generalize to novel object instances and categories, posing a major challenge for real-world deployment. To overcome these limitations, we present *GRIM* (Grasp Re-alignment via Iterative Matching), a novel training-free framework that adopts a retrieve-align-transfer (RAT) strategy. *GRIM* builds a dynamic memory of object-task interactions using **purely** synthetic data, in-the-wild images, and human demonstrations, enabling scalable and data-efficient task-oriented grasping. Given a novel scene object and a target task, GRIM first retrieves a semantically and visually similar object-task example from its memory. This retrieval is driven by a joint similarity metric that integrates learned visual representations from DINO embeddings [18] and semantic embeddings of task descriptions from CLIP [20]. Once a relevant memory instance is retrieved, its object point cloud is aligned to the scene object using a hybrid alignment strategy. This process begins with a coarse alignment based on geometric cues and PCA-reduced DINO feature scoring, followed by fine-grained refinement using the classical Iterative Closest Point (ICP) algorithm [2]. Finally, the grasp pose associated with the retrieved memory instance is transferred to the aligned scene object and further refined by evaluating it against a set of task-agnostic, geometrically stable grasps generated for the scene object, prioritizing task compatibility.

The main contributions of this research are:

1) A flexible memory construction pipeline that integrates object-task experiences from diverse data sources, including AI-generated videos, web images, and human demonstrations.
2) A robust and training-free alignment strategy that leverages learned dense features for semantically-aware coarse alignment, followed by precise ICP refinement, suitable for aligning novel objects where only partial or noisy observations may be available.

II. Related Works

As robots are increasingly expected to interact meaningfully with their environments, Task-Oriented Grasping (TOG) has emerged as a vital research direction—focusing not just on grasp stability, but on enabling task-relevant manipulation. Research in this area has largely followed two paths: analytical methods and increasingly dominant data-driven techniques. Murali et al. [16]

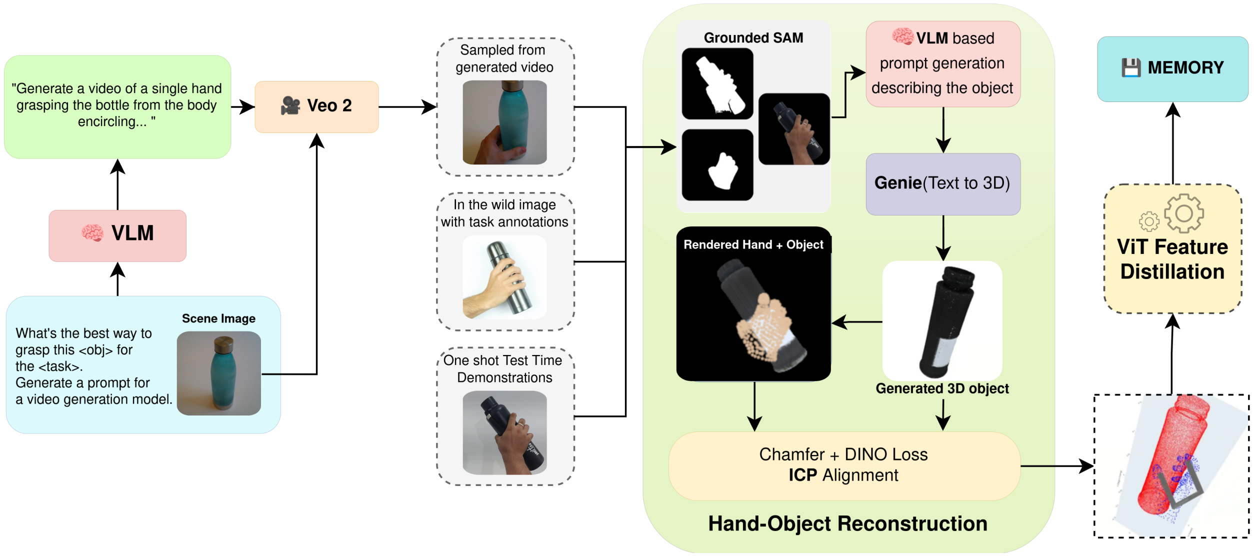

"Generate a video of a single hand grasping the bottle from the body encircling... "
Veo 2
VLM
Scene Image
What's the best way to grasp this <obj> for the <task>. Generate a prompt for a video generation model.
Sampled from generated video
In the wild image with task annotations
One shot Test Time Demonstrations
Grounded SAM
VLM based prompt generation describing the object
Genie(Text to 3D)
Rendered Hand + Object
Generated 3D object
Chamfer + DINO Loss
ICP Alignment
Hand-Object Reconstruction
MEMORY
ViT Feature Distillation

Fig. 1: The figure shows our memory creation pipeline. The Hand-Object Reconstruction block is built on Wu et al. [27]

A. *Data-Driven Approaches*

Early data-driven TOG efforts, such as those by Dang and Allen [3] and Liu et al. [12], focused on learning class-task-grasp relationships directly from data. However, as noted by Tang et al. [23], these methods often yielded unsatisfying performance due to the absence of external knowledge. Recognizing this, a significant body of work has explored integrating semantic knowledge. For instance, Song et al. [22] and Huang et al. [6] employed Bayesian Networks over constructed semantic KBs, while Ardón et al. [1], Zese et al. [28] and Liu et al. [13] utilized probabilistic logic for reasoning over semantic attributes. These approaches often necessitate grounding geometric information to pre-defined semantic representations and grapple with the scalability of their knowledge bases.

A pivotal challenge, as highlighted by multiple sources Tang et al. [23], Murali et al. [16], is the scarcity of large-scale, diverse TOG datasets. Murali et al. [15] addressed this by contributing the TaskGrasp dataset and the GCNGrasp algorithm. GCNGrasp leverages the Knowledge Graph built from this dataset but struggles with generalizing to concepts outside this graph. More recently, Tang et al. [23] proposed leveraging LLMs to inject open-ended semantic knowledge, aiming to improve generalization to novel concepts, though it remains a training-based method. Other training-dependent works like Tang et al. [24] and those by Jin et al. [7] and Nguyen et al. [17] also rely on manually annotated datasets, underscoring the persistent data acquisition bottleneck.

Our work, GRIM, diverges from these training-centric paradigms. While we acknowledge the importance of semantic understanding, we eschew the need for extensive pre-training on task-specific grasp annotations or reliance on structured KBs. Instead, GRIM champions a training-free approach by dynamically constructing a memory from heterogeneous data sources, including synthetic data, in-the-wild images, and human demonstrations, thereby directly addressing the data scarcity and annotation burden that encumbers many prior systems.

B. *Training-Free Approaches*

The emergence of powerful foundation models has catalyzed training-free TOG methodologies. Approaches like Li et al. [10], Rashid et al. [21], and Mirjalili et al. [14] utilize LLMs or VLMs to map semantic knowledge to target objects for grasp region selection. As Dong et al. [4] highlights, while these methods eliminate the need for model training and manual annotation, they typically produce only coarse spatial priors for grasping, lacking the precision required for generating directly executable grasp poses.

RTAGrasp also explores the training-free approach by learning TOG constraints from human demonstration videos. Like us, it avoids training and uses demonstrations, but GRIM stands out with a more diverse memory construction pipeline—drawing not only from human videos, but also from AI-generated content and web images. GRIM also introduces a unique retrieval-alignment-transfer process. The work most similar to GRIM in terms of retrieval is Ju et al. [8], which uses CLIP to retrieve contact points. However, as RTAGrasp points out, RoboABC struggles with selecting grasps that

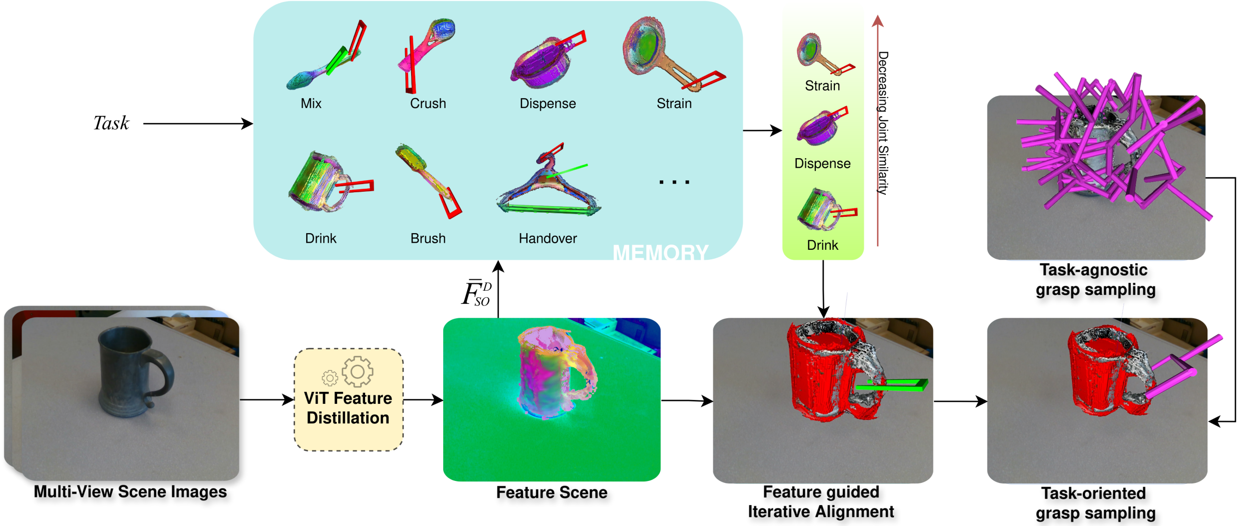


Fig. 2: The figure describes our retrieval, alignment and transfer process. The feature scene and memory objects are shown with DINOv2 PCA features as color representation. In the feature guided iterative alignment phase, the red point cloud is retrieved from memory, overlaid with the scene object point cloud.

match specific tasks and with figuring out the correct grasp orientation (”how to grasp”).

GRIM builds upon the strengths of retrieval but significantly extends them. Our retrieval leverages a joint visual (Oquab et al. [18]) and task-semantic (Radford et al. [20]) similarity, moving beyond simple contact points. Crucially, we introduce a robust, semantically-aware alignment strategy using PCA-reduced DINO features followed by ICP, designed to handle novel objects with partial observations. This fine-grained alignment facilitates a more precise transfer of the full grasp pose from a memory instance, which is then further refined against task-agnostic geometric stability criteria for the scene object. This contrasts with methods providing only regional guidance or relying solely on transferred poses without scene-specific geometric validation. Thus, GRIM offers a comprehensive training-free solution addressing both ”where” and ”how” to grasp with enhanced precision and adaptability to novel instances by sidestepping the constraints of pre-defined datasets and explicit knowledge engineering.

III. Methodology

We introduce GRIM (Grasp Re-alignment via Iterative Matching), a training-free framework for TOG. To achieve this, we adopt a retrieval-alignment-and-transfer approach. Our framework is divided into primarily into two steps: Memory Creation (Fig. 1) and Retrieval and Transfer (Fig. 2). We further describe our method in detail.

A. Memory Creation

For generalized semantic alignment and transfer to novel scenes and objects, we create a memory $\mathcal{M}$ of seen objects extracted from diverse data sources. Each instance in $\mathcal{M}$ contains a feature mesh F_M of the object, a 6D grasp pose G_t in the mesh’s coordinate frame, the corresponding task T, and object name O. To construct a single memory instance, we begin with an image I_{HO} depicting a human hand performing a grasp on a target object, annotated with the corresponding task T. From this image, we extract the object mesh, the hand mesh, and their relative pose. For this step, we build upon and refine the approach of Wu et al. [27], adapting it to our specific use case (details in Appendix C). Once we have the hand mesh, we simplify the mesh to extract a 6D parallel gripper grasp pose G_t for the task T. For an object O, we have multiple grasp poses in $\mathcal{M}$, and for a particular task, multiple grasp poses may be valid.

We use the extracted object mesh from the previous step to create F_M. Following the descriptor field representation of Wang et al. [25], we construct a feature mesh by associating DINO-based embeddings with mesh vertices. Therefore, the constructed memory can be represented as:

$$\mathcal{M} = \{(F_M, G_t, T, O)\} \tag{1}$$

All we need is a single frame to create an instance in our memory. We construct our memory from different data sources. For each source, the data extraction method varies slightly:

1) AI Generated Videos: With the rise of generative AI, SOTA (state-of-the-art) video generation models are highly capable at generating accurate videos following a prompt. We leverage one such SOTA model, Veo 2, for sourcing generated videos. We first make a list of objects, their images, along with their task images using the TaskGrasp [16] dataset. We use the object image, and name to prompt a VLM (Gemini) to describe the best way of grasping the object for the particular task and to generate a prompt for a video generation model accurately describing about the details of the video depicting the grasping action. This prompt is fed into the video generative model. Once we have the video, we sample the middle frame of the video, since the depiction of grasping remains consistent throughout the generated videos. This is our primary data generation method owing to its inherent scalability. For details, refer to the Appendix B.

2) In-the-Wild Web Images: The Internet has abundance an of images that can be scraped for learning useful grasping skills. We use human-sampled images from the internet and annotate the depicted task by leveraging a VLM. Then any web image with grasping demonstration can easily be integrated with our framework.

3) Test-Time Expert Demonstrations: At some point, an agent with existing memory might not perform well because that memory could not generalize sufficiently. So with our method, we can easily append the memory with just a single test-time image of grasp demonstration by a human and update the memory.

B. Memory Retrieval

When encountering a novel object or task, humans often draw upon past experiences, recalling the most analogous situations from memory. Inspired by this, our system implements a similarity search mechanism within its memory database $\mathcal{M}$. Consider a scenario where the robot encounters a novel scene containing a target object, represented by its point cloud P_{SO} and associated per-point DINO features F^D_{SO}. The robot is assigned a current task T_S for this object. The features F^D_{SO} are extracted from the scene, akin to dense descriptor fields Wang et al. [25]. Following segmentation of the target object P_{SO}, its per-point DINO features F^D_{SO} are averaged to yield a global object descriptor $\bar{F}^D_{SO}$. Similarly, the current scene task T_S is encoded using a text encoder (e.g., CLIP [20]) to obtain its embedding E_{T_S}.

The memory database $\mathcal{M}$ contains a set of stored objects. Each memory object $i \in \mathcal{M}$ is represented by its point cloud $P_{MO,i}$, its per-point DINO features $F^D_{MO,i}$, and an associated global DINO descriptor $\bar{F}^D_{MO,i}$ (obtained by averaging $F^D_{MO,i}$). Each memory object i is also associated with a set of tasks $\{T_{M,i,j}\}$, where each task $T_{M,i,j}$ has a corresponding CLIP embedding $E_{T_{M,i,j}}$ and an associated grasp pose $G_{M,i,j}$.

To retrieve the most relevant memory instance, we compute a joint similarity score $S_{\text{joint}}(i,j)$ for each memory object i and its associated task j:

$$S_{\text{joint}}(i,j) = \text{sim}_{\cos}(\bar{F}^D_{SO}, \bar{F}^D_{MO,i}) \cdot \text{sim}_{\cos}(E_{T_S}, E_{T_{M,i,j}}) \tag{2}$$

where $\text{sim}_{\cos}(\cdot,\cdot)$ denotes the cosine similarity. This score is computed over all memory object-task pairs in $\mathcal{M}$. We retrieve the memory instance with the highest joint similarity.

C. Alignment Module

After the semantic memory retrieval, we have a source memory object (point cloud P_{MO} with DINO features F^D_{MO}) similar to the masked scene object (P_{SO} with DINO features F^D_{SO}). A PCA model, M_{PCA}, is trained on the original F^D_{MO} and F^D_{SO} to project DINO features into a lower D_{PCA}-dimensional space as F'^D_{MO} and F'^D_{SO}. We begin the alignment process by computing centroids c_{MO} and c_{SO}, and an initial scale factor s_g by comparing eigenvalues along the principal geometric components of P_{MO} and P_{SO}. The P_{MO} point clouds often have very different size than that of P_{SO}; applying the scale factor s_g helps in matching their sizes. Then we do a grid search over Euler angles, generating candidate rotation matrices $\{R_i\}$. Each R_i forms an initial transformation:

$$T_{\text{init},i}(p) = s_g R_i(p - c_{MO}) + c_{SO} \tag{3}$$

We aim to find the closest initial coarse alignment here, so we calculate a score for each candidate. One might use Chamfer distance between the $T_{\text{init},i}(P_{MO})$ and P_{SO} but here we do not aim to find the candidates with the best geometric match but with the best feature match. For better generalization we want feature alignment of O_M with O_S, as the memory might not always contain the exact same object as the scene (e.g., a spoon handle with a spatula handle). For each transformed source point p_m of P_{MO}, we find its K_{eval} nearest neighbors $\{p_{s,k}\}$ in P_{SO}. The cost for each pair $(p_m, p_{s,k})$ is a weighted sum as shown in Eq. 4

$$C_{\text{pair}} = w_g \|p_m - p_{s,k}\|^2 + w_f(1 - \cos(F'^D_{M,p_m}, F'^D_{S,p_{s,k}})) \tag{4}$$

where $F'^D_{X,p}$ denotes the PCA-DINO feature of point p in dataset X. w_g and w_f are the weights assigned to geometric distance and feature distance. The minimum C_{pair} over K_{eval} neighbors gives the point's cost, and the average of these point costs determines $\text{Score}(T_{\text{init},i})$. The top K_{orient} initial transformations $\{T^*_{\text{init}}\}$ undergo ICP refinement, yielding refined poses $\{T_{\text{ref},j}\}$. Finally, these refined poses are re-evaluated using the same

TABLE I: Comparison of Precision with different methods

Method	Novel Instances								
	Paint roller	Brush	Tongs	Strainer	Frying Pan	Fork	Mortar	Ice Scrapper	Pizza Cutter
Random	0.30	0.66	0.23	0.24	0.32	0.26	0.31	0.60	0.50
RTAGrasp	0.39	**0.93**	0.28	0.55	0.42	0.35	0.37	**0.91**	0.57
GRIM(Ours)	**0.89**	0.90	**0.58**	**0.58**	**0.60**	**0.40**	**0.72**	0.71	**0.92**

combined score metric, with a potentially tighter distance threshold. The $T_{\text{ref},j}$ yielding the lowest final score is selected as the optimal transformation T_{final}.

D. Grasp Transfer

Following alignment, the retrieved memory grasp G_M is transformed into the scene using the final alignment T_{final} to yield the scene grasp $G_S = T_{\text{final}} \cdot G_M$, appropriately scaled for the target. However, G_S might not represent an optimal or directly executable grasp pose for the scene object geometry and robot gripper. To address this, we adopt a sampling-and-evaluation strategy inspired by prior work RTAGrasp [4]. We first sample N task-agnostic, geometrically feasible grasp poses $\{G_{A,i}\}_{i=1}^{N}$ using AnyGrasp [5]. Each candidate grasp $G_{A,i} = (R_{A,i}, \mathbf{t}_{A,i})$ is associated with a geometric stability score $S_{\text{geo},i}$.

To evaluate the suitability of each candidate $G_{A,i}$ with respect to the intent captured by the transferred memory grasp G_S, we define a task-compatibility score. Let $\mathbf{p}_{\text{target}} = \mathbf{t}_S$ be the target position derived from G_S, and $\mathbf{v}_{\text{target}} = R_S \mathbf{e}_z$ be its primary approach direction (where $\mathbf{e}_z = [0, 0, 1]^T$). For each candidate grasp $G_{A,i}$, let its approach direction be $\mathbf{o}_{z,i} = R_{A,i} \mathbf{e}_z$. The task-compatibility score $S_{\text{task},i}$ for $G_{A,i}$ is then computed as:

$$S_{\text{task},i} = \frac{\mathbf{v}_{\text{target}} \cdot \mathbf{o}_{z,i}}{\|\mathbf{v}_{\text{target}}\| \|\mathbf{o}_{z,i}\|} + \exp\left(-\frac{\|\mathbf{t}_{A,i} - \mathbf{p}_{\text{target}}\|^2}{2\sigma^2}\right) \tag{5}$$

where $\sigma = 0.1$ is a scaling factor. The first term in Eq. (5) measures the cosine similarity between the candidate grasp's approach direction and the target direction derived from the memory grasp. The second term is a Gaussian decay function that penalizes positional deviation from the target position. Since $\|\mathbf{v}_{\text{target}}\| = 1$ and $\|\mathbf{o}_{z,i}\| = 1$ (as they are column vectors from rotation matrices or normalized direction vectors), the first term simplifies to $\mathbf{v}_{\text{target}} \cdot \mathbf{o}_{z,i}$.

The final score S_i for each candidate grasp $G_{A,i}$ combines task-compatibility and geometric stability:

$$S_i = w_{\text{task}} S_{\text{task},i} + w_{\text{geo}} S_{\text{geo},i} \tag{6}$$

Following RTAGrasp [4], we prioritize task-compatibility by setting $w_{\text{task}} = 0.95$ and $w_{\text{geo}} = 0.05$, given that most candidates generated by the sampler are already geometrically stable. This sampling-and-evaluation approach allows us to leverage robust task-agnostic grasp generation techniques while effectively aligning the selected grasp with the task context inferred from the memory system, without requiring intricate hand-to-gripper re-targeting. The robot then selects the candidate grasp $G_A^* = \arg\max_i S_i$ for execution. In our implementation, we use AnyGrasp [5] as a grasp sampler, although other stable grasp synthesis methods could be used.

IV. Experiments and Results

A. Baselines

We compare GRIM with the following methods: (1) **Random**, which is Task-Agnostic and focuses only on grasp stability. (2) **RTAGrasp** [4]is a training-free method that, like our approach, employs a memory retrieval approach but differs from ours in that it uses 2D feature matching for memory transfer. For a fair comparison, we use the same data source and amount to create memory for RTAGrasp.

B. Dataset

We extensively test our framework on the TaskGrasp [16] dataset, and compare the results with the baselines. Since both the approaches use out-of-domain data, we evaluate on all of positive example data of TaskGrasp. We also deliberately modify and shorten the memory and create two splits: held-out objects and held-out tasks. For the held-out object split, there is no identical memory object and for the held-out task split, there might be an identical object but never the same task present in the memory.

C. Memory

GRIM's memory buffer contains data from 180 generated videos, 15 internet-sampled images and 15 self-demonstrated images, totaling to 210 grasp data instances. We use the same images to create a memory buffer for RTAGrasp, we estimate the 2D grasp point and 3D direction vector from our 6D grasp pose.

D. Evaluation Metric

For our approach and all the baselines, at the end, we classify the 25 annotated grasp poses present in the TaskGrasp dataset for all object instances. For GRIM and RTAGrasp, we sample from the task-agnostic grasp poses available in TaskGrasp, and these sampled poses are labeled as true. We use Average Precision over classification of grasp poses as our evaluation metric.

E. Results

TABLE II: Average Precision calculated over all data, held-out objects, and held-out tasks.

Method	All Data	Held-out Objects	Held-out Tasks
Random	0.49	0.41	0.43
RTAGrasp	0.58	0.52	0.51
GRIM (Ours)	**0.67**	**0.65**	**0.64**

Table II presents the average precision across the complete TaskGrasp dataset (All Data), the Held-Out Objects split, and the Held-Out Tasks split. In the held-out settings, we exclude all object/task data from the memory that appears in the inference split. Our method demonstrates strong generalization to unseen objects and tasks, outperforming other approaches. These results highlight the effectiveness of our 3D feature-guided alignment and transfer strategy over traditional 2D feature matching and transfer methods. Performance on a subset of individual objects is shown in Table I.

V. Conclusion

In this research, we present a novel training-free framework for task-oriented grasping. Experimental results demonstrate that our approach generalizes more effectively than 2D feature matching and transfer-based methods. By leveraging DINO-learned visual features, our method achieves robust semantic alignment that cannot be achieved through geometric cues alone. Although our memory module plays a crucial role in overall effectiveness, we demonstrate strong performance even on synthetic data, which is typically noisy and challenging. For feature representation, we currently rely on visual features that lack geometric understanding; incorporating geometric information could potentially further enhance the effectiveness of our approach.

References

[1] Paola Ardón, Éric Pairet, Ronald P. A. Petrick, Subramanian Ramamoorthy, and Katrin Solveig Lohan. Learning grasp affordance reasoning through semantic relations. *IEEE Robotics and Automation Letters*, 4:4571–4578, 2019. URL https://api.semanticscholar.org/CorpusID:195345691.

[2] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14:239–256, 1992. URL https://api.semanticscholar.org/CorpusID:21874346.

[3] Hao Dang and Peter K. Allen. Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1311–1317, 2012. doi: 10.1109/IROS.2012.6385563.

[4] Wenlong Dong, Dehao Huang, Jiangshan Liu, Chao Tang, and Hong Zhang. Rtagrasp: Learning task-oriented grasping from human videos via retrieval, transfer, and alignment. *arXiv preprint arXiv:2409.16033*, 2024.

[5] Hao-Shu Fang, Chenxi Wang, Hongjie Fang, Minghao Gou, Jirong Liu, Hengxu Yan, Wenhai Liu, Yichen Xie, and Cewu Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains, 2023. URL https://arxiv.org/abs/2212.08333.

[6] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models. In *arXiv preprint arXiv:2207.05608*, 2022.

[7] Shiyu Jin, Jinxuan Xu, Yutian Lei, and Liangjun Zhang. Reasoning grasping via multimodal large language model. *ArXiv*, abs/2402.06798, 2024. URL https://api.semanticscholar.org/CorpusID:267627619.

[8] Yuanchen Ju, Kaizhe Hu, Guowei Zhang, Gu Zhang, Mingrun Jiang, and Huazhe Xu. Robo-abc: Affordance generalization beyond categories via semantic correspondence for robot manipulation. *arXiv preprint arXiv:2401.07487*, 2024.

[9] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, Piotr Doll'ar, and Ross Girshick. Segment anything. In *arXiv preprint arXiv:2304.02643*, 2023.

[10] Samuel Li, Sarthak Bhagat, Joseph Campbell, Yaqi Xie, Woojun Kim, Katia Sycara, and Simon Stepputtis. Shapegrasp: Zero-shot task-oriented grasping with large language models through geometric decomposition. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10527–10534, 2024. doi: 10.1109/IROS58592.2024.10801661.

[11] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Su, Lin Zhu, Lei Zhang, and Yu Qiao. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *arXiv preprint arXiv:2303.05499*, 2023.

[12] Weiyu Liu, Angel Andres Daruna, and S. Chernova. Cage: Context-aware grasping engine. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2550–2556, 2019. URL https://api.semanticscholar.org/CorpusID:202750339.

[13] Weiyu Liu, Angel Daruna, Maithili Patel, Kartik Ramachandruni, and Sonia Chernova. A survey of semantic reasoning frameworks for robotic systems. *Robotics and Autonomous Systems*, 159:104294, 2023. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2022.104294. URL https://www.sciencedirect.com/science/article/pii/S092188902200183X.

[14] Reihaneh Mirjalili, Michael Krawez, Simone Silenzi, Yannik Blei, and Wolfram Burgard. Langrasp: Using large language models for semantic object grasping, 2024. URL https://arxiv.org/abs/2310.05239.

[15] Adithyavairavan Murali, Weiyu Liu, Kenneth Marino, S. Chernova, and Abhinav Kumar Gupta. Same object, different grasps: Data and semantic knowledge for task-oriented grasping. In *Conference on Robot Learning*, 2020. URL https://api.semanticscholar.org/CorpusID:226306649.

[16] Adithyavairavan Murali, Weiyu Liu, Kenneth Marino, Sonia Chernova, and Abhinav Gupta. Same object, different grasps: Data and semantic knowledge for task-oriented grasping. In *Conference on Robot Learning*, 2020.

[17] Toan Nguyen, Minh N. Vu, Baoru Huang, Tuan Van Vo, Vy Truong, Ngan Le, Thi DK Vo, Bac Le, and Anh Nguyen. Language-conditioned affordance-pose detection in 3d point clouds. *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3071–3078, 2023. URL https://api.semanticscholar.org/CorpusID:262063614.

[18] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193.

[19] Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing hands in 3D with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[20] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

[21] Adam Rashid, Satvik Sharma, Chung Min Kim, Justin Kerr, Lawrence Yunliang Chen, Angjoo Kanazawa, and Ken Goldberg. Language embedded radiance fields for zero-shot task-oriented grasping. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=k-Fg8JDQmc.

[22] D. Song, K. Huebner, V. Kyrki, and D. Kragic. Learning task constraints for robot grasping using graphical models. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1579–1585, 2010. doi: 10.1109/IROS.2010.5649406.

[23] Chao Tang, Dehao Huang, Wenqi Ge, Weiyu Liu, and Hong Zhang. Graspgpt: Leveraging semantic knowledge from a large language model for task-oriented grasping. *arXiv preprint arXiv:2307.13204*, 2023.

[24] Chao Tang, Dehao Huang, Lingxiao Meng, Weiyu Liu, and Hong Zhang. Task-oriented grasp prediction with visual-language inputs. *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4881–4888, 2023. URL https://api.semanticscholar.org/CorpusID:257233075.

[25] Yixuan Wang, Mingtong Zhang, Zhuoran Li, Tarik Kelestemur, Katherine Driggs-Campbell, Jiajun Wu, Li Fei-Fei, and Yunzhu Li. D^{3}fields: Dynamic 3d descriptor fields for zero-shot generalizable rearrangement. *arXiv preprint arXiv:2309.16118*, 2023.

[26] Yixuan Wang, Mingtong Zhang, Zhuoran Li, Tarik Kelestemur, Katherine Driggs-Campbell, Jiajun Wu, Li Fei-Fei, and Yunzhu Li. D^{3}fields: Dynamic 3d descriptor fields for zero-shot generalizable rearrangement. In *8th Annual Conference on Robot Learning*, 2024.

[27] Jane Wu, Georgios Pavlakos, Georgia Gkioxari, and Jitendra Malik. Reconstructing hand-held objects in 3d from images and videos. *arXiv preprint arXiv:2404.06507*, 2024.

[28] Riccardo Zese, Elena Bellodi, Evelina Lamma,

Fabrizio Riguzzi, and Fabiano Aguiari. Semantics and inference for probabilistic description logics. In Fernando Bobillo, Rommel N. Carvalho, Paulo C.G. Costa, Claudia d'Amato, Nicola Fanizzi, Kathryn B. Laskey, Kenneth J. Laskey, Thomas Lukasiewicz, Matthias Nickles, and Michael Pool, editors, *Uncertainty Reasoning for the Semantic Web III*, pages 79–99, Cham, 2014. Springer International Publishing. ISBN 978-3-319-13413-0.

Appendix

A. VLM-Based Reasoning and Video Prompt Generation

For the goal of generating a video depicting a particular task, we first prompt a VLM to describe the best way of grasping and generate a prompt for the same. We use Gemini-2.5-Flash as our VLM. This task requires the VLM to reason about the object and task semantics. We also put the scene image as reference for scene-conditioned reasoning. The prompt we use:

VLM Prompt

```
For an object {OBJ}, I want you to describe the best way a single human hand can hold this object for the
task of {TASK}. The {OBJ}'s image is given, please refer to the image while reasoning about the grasping
way for the given task.

For the holding method, provide:
1. A concise, single-line description of the holding method. (e.g., "Holding the knife by its handle for
cutting.")
2. A detailed text-to-video generation prompt (single paragraph, 7-8 lines). This prompt must clearly
describe the grasping method, the hand's position relative to the object/parts. It also must specify that
the video should feature a single hand, the object, and the hand must be completely visible throughout the
video, and the entire object must be in frame at all times.
3. There must be only the right hand in the video prompt. Never use left hand or both hands in the prompt.

Your response should be in JSON format, where each element of the array is an object.
For the object-task pair, the output JSON must have exactly two string keys: "way_to_hold" and
"video_prompt".
Do not include any other text, explanations, or markdown formatting like ```json ... ``` outside of the JSON
array itself.

Example of the JSON array structure for a "cup" and task of "drink":

  {
    "way_to_hold": "Holding a ceramic cup firmly by its D-shaped handle.",
    "video_prompt": "Generate a video depicting a single human hand securely gripping the D-shaped handle
    of a standard ceramic coffee cup. The fingers should be visibly wrapped through the handle's opening,
    with the thumb pressing firmly against the top curve of the handle for stability, ensuring the cup is
    held upright. The palm is not touching the body of the cup. The hand must be completely visible
    throughout the video, and the entire cup must be in frame at all times. The video should focus on the
    hand-object interaction, showing the grip and the cup's details clearly."
  }
Now, generate this JSON for the object {OBJ}.
```

We notice that for many cases the grasp pose described by the VLM remains fairly the same. So, in order to be efficient with the number of generated videos, we use a slightly different approach. We first prompt the VLM to generate K (3 in our case) distinct ways of grasping the object and then map these three ways of grasping to all the tasks. This way is much more efficient as we are generating three videos per object, and these can be mapped to all the tasks present for that object.

K Grasping Ways Prompt

```
For an object "{OBJ}", I want you to describe multiple ways (3 ways preferable) a single human hand can
hold this object.
Ensure the holding/grasping methods are distinct, primarily differing in the grasping location on the
object. Assume I will also provide an image of the scene with the video generation prompt.

For each holding method, provide:
1. A concise, single-line description of the holding method. (e.g., "Holding the knife by its handle for
cutting.")
2. A detailed text-to-video generation prompt (single paragraph, 7-8 lines). This prompt must clearly
describe the grasping method, the hand's position relative to the object/parts. It also must specify that
the video should feature a single hand, the object, and The hand must be completely visible throughout the
video, and the entire object must be in frame at all times.
3. There MUST be only the right hand in the video prompt. Never use left hand or both hands in the prompt.

Your response MUST be a JSON array, where each element of the array is an object.
Each object in the array must have exactly two string keys: "way_to_hold" and "video_prompt".
Do not include any other text, explanations, or markdown formatting like ```json ... ``` outside of the JSON
array itself.

Example of the JSON array structure for a "cup":
[
```

```
  {
    "way_to_hold": "Holding a ceramic cup firmly by its D-shaped handle.",
    "video_prompt": "Generate a video depicting a single human hand securely gripping the D-shaped handle
    of a standard ceramic coffee cup. The fingers should be visibly wrapped through the handle's opening,
    with the thumb pressing firmly against the top curve of the handle for stability, ensuring the cup is
    held upright. The palm is not touching the body of the cup. The hand must be completely visible
    throughout the video, and the entire cup must be in frame at all times. The video should focus on the
    hand-object interaction, showing the grip and the cup's details clearly."
  },
  {
    "way_to_hold": "Cradling the body of a warm ceramic cup with one hand.",
    "video_prompt": "Create a video showcasing a single human hand gently cradling the main cylindrical
    body of a warm ceramic cup. The fingers should be spread slightly, conforming to the curve of the cup,
    with the palm providing broad support from underneath and the side. The thumb might rest along the
    upper rim or side, opposite the fingers. The hand must be completely visible throughout the video, and
    the entire cup must be in frame at all times. The video should highlight the hand's gentle grip and the
    cup's surface texture."
  },
  {
    "way_to_hold": "Pinching the rim of an empty teacup with thumb and index finger.",
    "video_prompt": "Generate a video that illustrate a single human hand delicately holding an empty,
    lightweight teacup by its rim. The grasp involves the thumb pressing on the outer surface of the rim
    and the index finger (and possibly middle finger) supporting it from the inner surface, a precise pinch
    grip. The remaining fingers might be curled or extended gracefully away from the cup body. The hand
    must be completely visible throughout the video, and the entire cup must be in frame at all times. The
    video should focus on the hand's dexterity and the teacup's delicate design."
  }

Now, generate this JSON array for the object "{OBJ}".
```

Task-Video Mapping Prompt

```
You are an expert in robotics and human-object interaction with a focus on practicality.
Your task is to identify ALL suitable ways a single human hand can hold an object to perform a specific
task.
Prioritize inclusivity: if a holding method is **possible or doable** for the task, even if not the
absolute most optimal or common way, it should be considered valid. We want to ensure we capture at least
one plausible holding method if any exists.

Object: "{OBJ}" (original ID: "{XXX_OBJ}")
Task to perform: "{task_name}"

Consider the following predefined ways to hold the object "{OBJ}", including their descriptions and
intended video visualizations:
{holding_options_str}
Reason deeply about the physical requirements of the task "{TASK}" when performed with the object "{OBJ}".
Consider factors like:
- Stability needed for the task.
- Precision required.
- Force application (if any).
- Necessary orientation of the object.
- Freedom of movement for the hand or object parts.
- Safety and realism of the hold for the given task.

Based on your reasoning, identify **ALL holding methods from the list above that are possible or doable**
for a single human hand to effectively and realistically perform the task. A task can have multiple valid
ways to hold the object. Your goal is to be comprehensive.

Your response MUST be a JSON object containing a single key "valid_indices".
The value for "valid_indices" must be a list of integers, where each integer is an index from the provided
list of holding methods.
For example:
If methods 0 and 2 are suitable:
{
  "valid_indices": [0, 2]
}
If only method 1 is suitable:
{
  "valid_indices": [1]
}
If all methods (0, 1, and 2) are considered possible or doable:
{
  "valid_indices": [0, 1, 2]
}
There must always be at least one index in the list.
Do not include any other text, explanation, or markdown formatting outside of this JSON object.
```

B. AI Generated Video

A significant portion of our memory dataset (86%) is constructed using AI-generated videos. For this purpose, we leverage the capabilities of the Veo 2 generative model. While image-based generative models often struggle with interpreting complex textual prompts, we found that video generation models exhibit better fidelity in this regard. Specifically, generated videos demonstrate improved performance in adhering to grasping instructions, such as those provided by a large language model like Gemini.

However, these models can still struggle with non-intuitive scenarios or when requiring nuanced object interaction. For instance, if an object possesses a prominent handle, the generated video might default to a grasp on the handle, even if the prompt specifies a different interaction point. Examples illustrating the outputs from our video generation pipeline, including variations based on different task prompts given a reference image, are presented in Figure 3. We anticipate that continued advancements in such generative models will directly translate to enhanced capabilities and performance for our overall framework, further improving its ability to learn from diverse and complex interactions.

C. 3D Hand and Object Reconstruction from Images

To populate our grasp memory $\mathcal{M}$ with task-oriented 6-DOF parallel gripper poses, we process single images depicting human hands interacting with objects. This process leverages and adapts the MCC-HO framework presented by Wu et al. [27] for hand-object 3D reconstruction. When processing AI-generated videos (as detailed further in Appendix B, if applicable, or simply "from AI-generated videos"), a representative frame is typically selected by sampling from the middle of the video, as grasping actions are often consistently depicted there. For other image sources, a single static image is used directly.

The pipeline begins with segmenting the hand and object from the input image. For this, we employ Grounding SAM, which typically combines a text-promptable object detector (such as Grounding DINO by Liu et al. [11]) with the Segment Anything Model (SAM) by Kirillov et al. [9]. In our implementation, we utilize a SAM model with a ViT-Base backbone (`facebook/sam-vit-base`) for segmentation, guided by prompts to acquire precise masks of the interacting entities. These masks guide the subsequent 3D reconstruction.

Following segmentation, the MCC-HO framework is used to jointly reconstruct the 3D geometry of both the hand and the held object from the single view. A critical part of the object reconstruction module, adapted for our memory creation, involves an iterative alignment procedure. This alignment optimizes the fit of a retrieved or generated object model to the visual and geometric cues from the image. The optimization function for this alignment, L_{align}, is a weighted sum of a Chamfer loss (L_{CD}) and a DINO PCA-based feature similarity loss ($L_{\text{DINO\_PCA}}$):

$$L_{\text{align}} = L_{\text{CD}}(P_{\text{target}}, P_{\text{cand}}(R, T, s)) + w_{\text{DINO}} \cdot L_{\text{DINO\_PCA}} \tag{7}$$

where:

- P_{target} is the combined target point cloud (from the initial object reconstruction and the known hand geometry).
- $P_{\text{cand}}(R, T, s)$ is the candidate object point cloud, transformed by rotation R, translation T, and scale s.
- $L_{\text{CD}}(P_1, P_2) = \sum_{x\in P_1} \min_{y\in P_2} \|x - y\|_2^2 + \sum_{y\in P_2} \min_{x\in P_1} \|y - x\|_2^2$ is the Chamfer distance between two point sets P_1 and P_2.
- $L_{\text{DINO\_PCA}} = 1 - \text{sim}_{\cos}(\bar{f}_{\text{PCA}}(D(I_{\text{target}})), \bar{f}_{\text{PCA}}(D(I_{\text{cand}})))$ measures the cosine dissimilarity between the mean PCA-projected DINOv2 features. $D(I)$ represents the DINOv2 features extracted from an image I (`facebook/dinov2-small-patch14-224`, which corresponds to ViT-S/14), $\bar{f}_{\text{PCA}}$ denotes the mean of these features after PCA projection, I_{target} is the input image patch, and I_{cand} is the rendered image of the candidate object.
- w_{DINO} is the weight for the DINO loss component, set to 0.005 in our setup.

The alignment proceeds through several stages: an initial alignment of principal axes, followed by coarse rotational adjustments via flips about these axes, then fine-grained rotational refinement, and finally, fine-tuning of the translation. The entire pipeline, from image input to the reconstructed hand and object, takes approximately 7 minutes per image to process on an Nvidia RTX4060 laptop GPU.

Once the 3D point cloud of the human hand is accurately reconstructed by the MCC-HO module, we convert this detailed five-fingered representation into a simplified 6-DOF parallel gripper pose. This conversion is achieved using our algorithm, which first identifies key segments of the hand—specifically the thumb, index finger, middle finger, and the palm/back of the hand—by processing the hand vertices. The centroids of these segments are then used to define the gripper's characteristics. The midpoint between the thumb centroid and the combined centroid of the index and middle fingers defines the gripper's center (translation). The vector connecting the thumb and opposing fingers establishes the primary axis for gripper width and one component of its orientation.

The palm centroid provides a reference point to better estimate the approach vector and thus the complete 3D orientation (rotation matrix) of the gripper. The distance between the opposing finger segments determines the gripper width, and an estimated gripper finger length is derived based on the hand's overall dimensions and the relative positions of the segments. This method robustly extracts a functional parallel gripper pose suitable for robotic execution.

D. Feature Guided Alignment

The most crucial part of our grasp transfer framework lies in Feature Guided 3D alignment. We use DINOv2-vitl14's visual features for creating our feature-rich point cloud, both for the memory object and the scene. Subsequently, we segment the target object using Grounded-SAM to obtain its feature-rich point cloud, a process similar to that described by Wang et al. [26]. We explored various algorithms for source and target point cloud alignment, including pure geometric alignment and pure feature-based alignment. However, we found that neither performs optimally in isolation. Pure geometric alignment necessitates that the target and source point clouds possess roughly similar shapes; even with complete point clouds, it frequently converges to a flipped orientation of the correct one. Furthermore, this method suffers particularly in cases involving noisy or partial point clouds. As for purely feature-based matching, we observe that methods effective in 2D image domains—such as those in Murali et al. [16]—do not translate well to 3D. This is primarily because DINO features, being trained on 2D images, capture only visual information. When these features are distilled into 3D, they suffer from object symmetry, often leading to incorrect correspondences such as matching features from the right side of an object to its left, and vice versa.

To this end, we designed a hybrid alignment algorithm that synergistically leverages both visual features and geometric cues. This approach is formalized by a cost function for each potential point pair $(p_m, p_{s,k})$ between the memory point cloud (m) and a scene point cloud (s), calculated as a weighted sum:

$$C_{\text{pair}} = w_g \|p_m - p_{s,k}\|^2 + w_f(1 - \cos(F'^{D}_{M,p_m}, F'^{D}_{S,p_{s,k}})) \tag{8}$$

where p_m is a point from the memory object, $p_{s,k}$ is a point from the scene object, and $F'^{D}_{X,p}$ denotes the PCA-DINO feature of point p in dataset X. The terms w_g and w_f represent the weights assigned to the geometric and feature similarity components, respectively. Our Feature Guided Iterative Alignment approach is able to perform well even in cases where pure geometric methods fail, demonstrating significant robustness and accuracy.

The generalization of our feature-guided alignment is particularly evident when aligning objects of different categories, as illustrated in the second section of Figure 5 (”Alignments between Objects of Different Category”). For instance, our framework demonstrates that an object in memory possessing a handle, such as a Ladle, can successfully generalize its alignment to various other objects in the scene that also feature handles, like a Grater or a Whisk. This ability to identify and match salient functional parts like handles across diverse object types underscores the semantic understanding embedded within our hybrid approach, facilitated by the DINO features guiding the geometric alignment.

Further highlighting the advantages of our method, Figure 6 provides a direct visual comparison between pure geometric alignment and our feature-guided alignment for several challenging pairs. For the pure geometric alignment results shown, we effectively set the feature weight $w_f = 0$ in Equation 8, relying solely on geometric proximity (w_g maintained). As can be observed, the pure geometric approach often misaligns, converges to local minima, or results in flipped orientations. In contrast, our feature-guided alignment consistently produces more accurate and semantically correct alignments. With these results, it becomes apparent that our Feature Guided Iterative Alignment stands superior, offering a more robust and generalizable solution for 3D object alignment in complex scenarios.

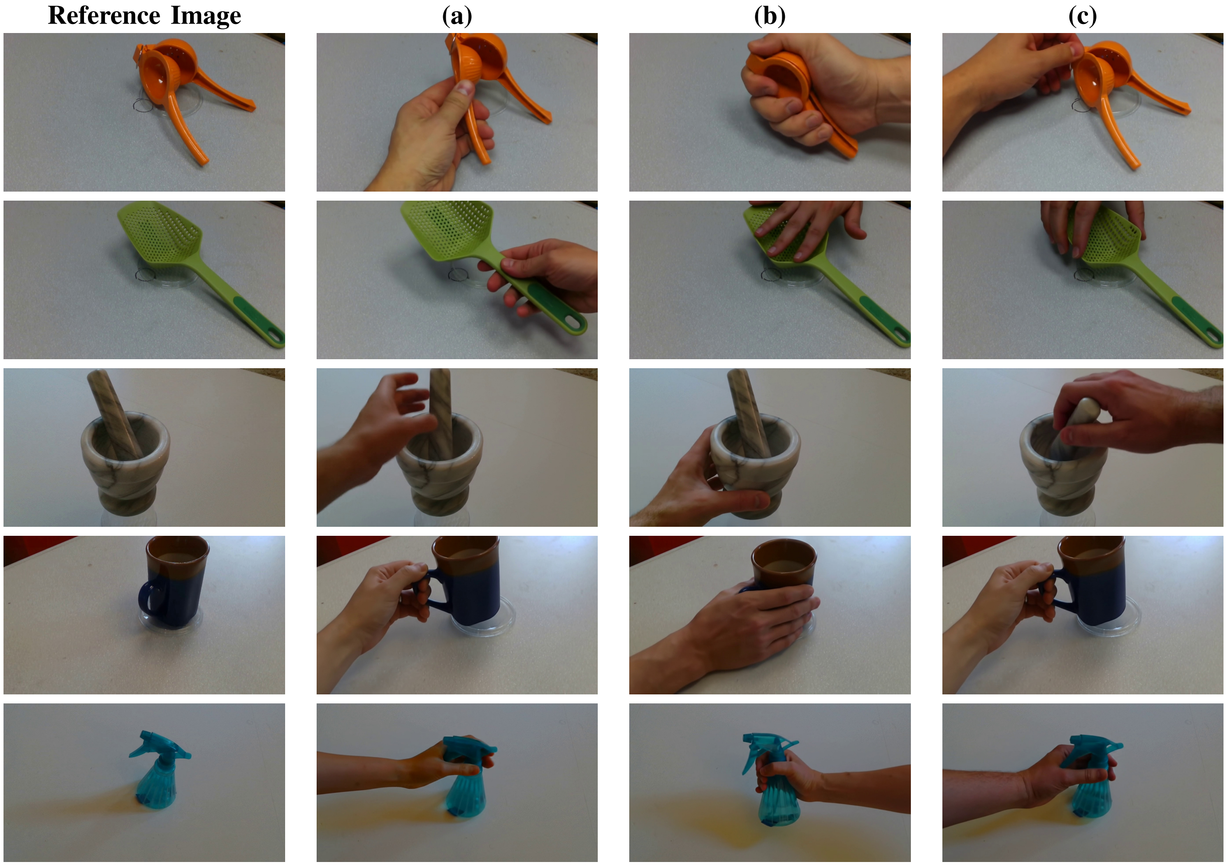
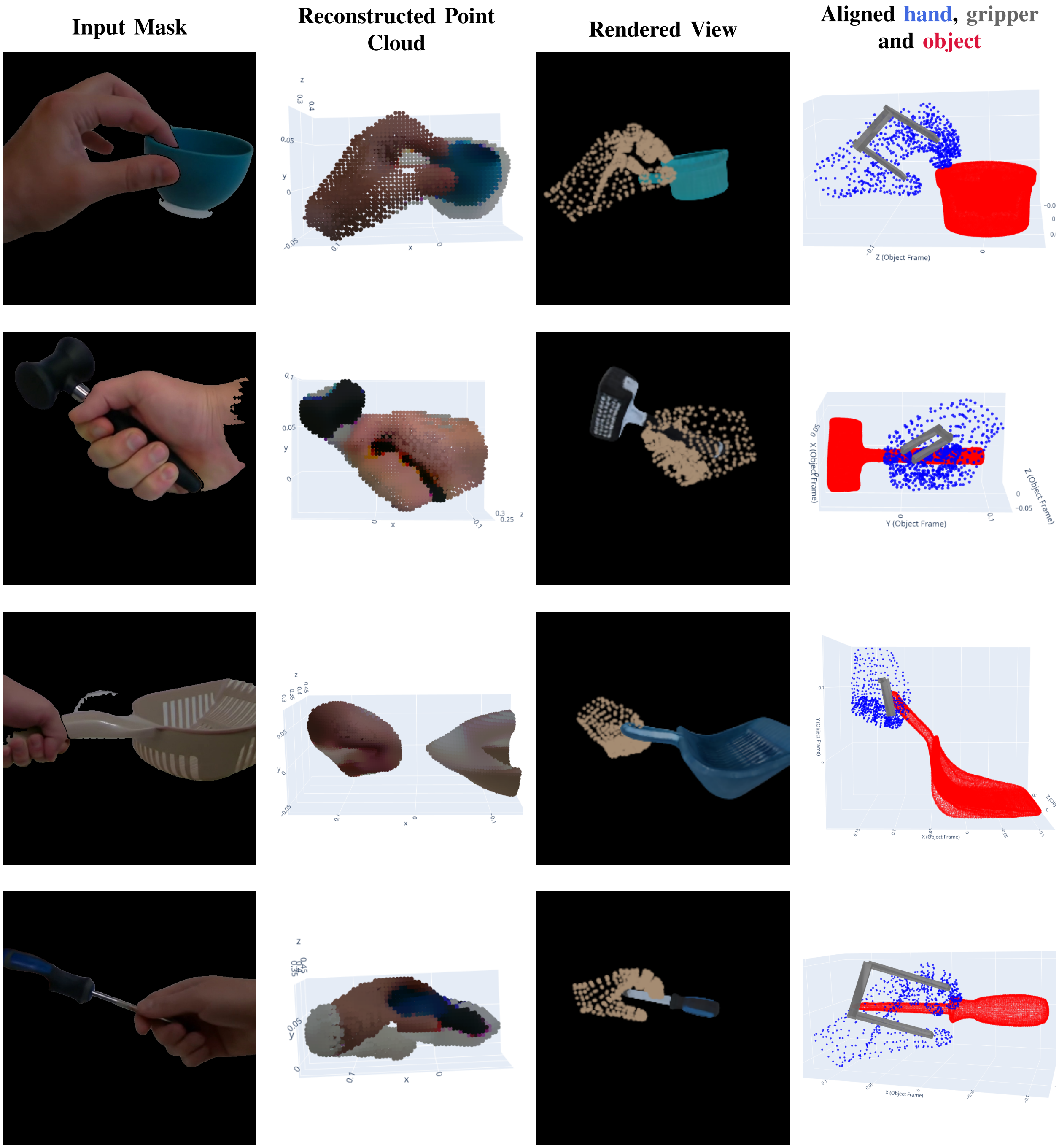


Fig. 3: On the left we have the reference image used for video generation. (a), (b) and (c) are sampled frames from the generated videos using different task prompts.



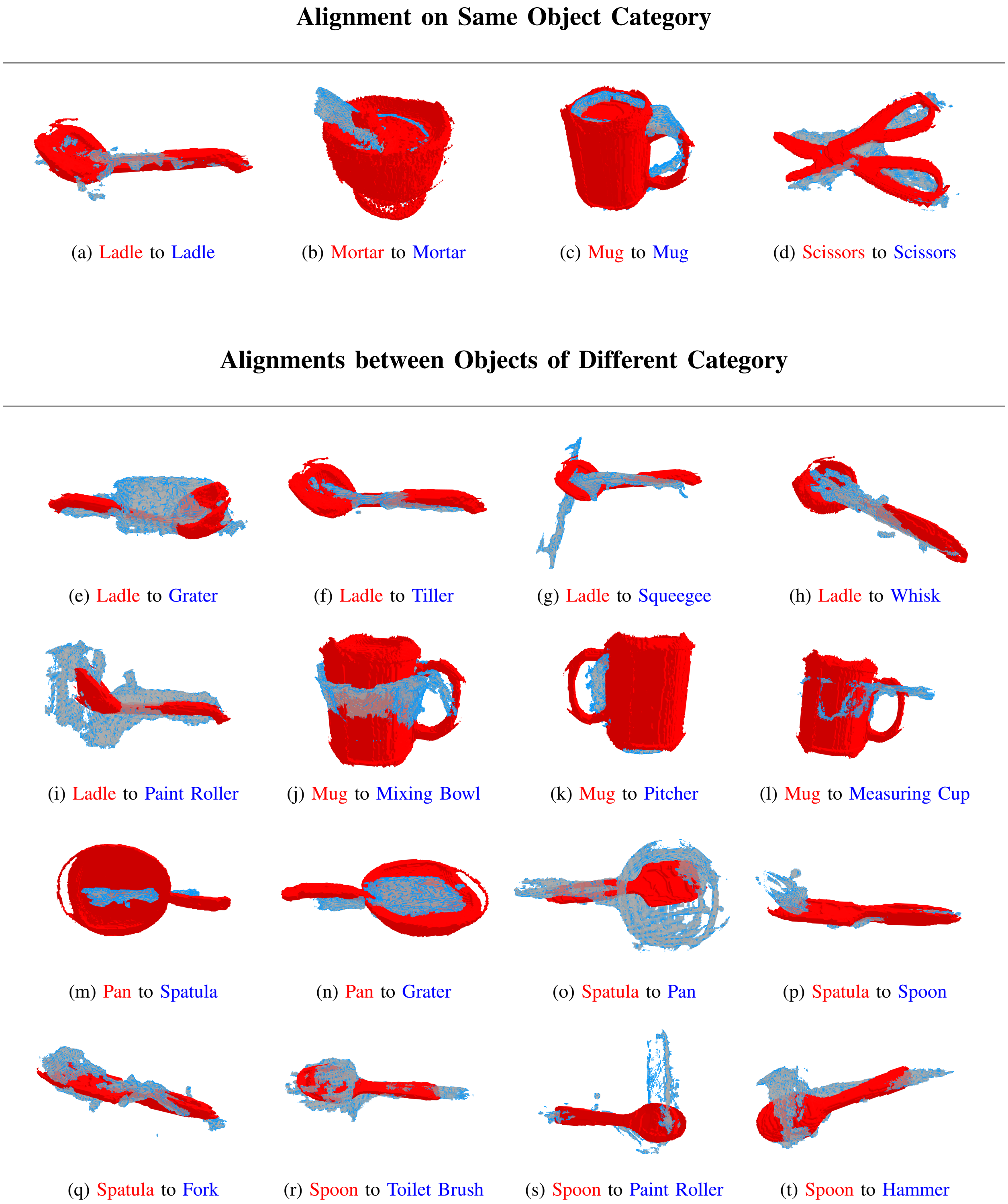
Input Mask

Reconstructed Point Cloud

Rendered View

Aligned hand, gripper and object

Fig. 4: Visualization of the 3D hand-object reconstruction and grasp pose derivation pipeline for various objects. Each section shows four stages from left to right: Input image masked by Grounding SAM; Reconstructed 3D Point Cloud (PCD) of the object; Rendered view of the reconstructed object geometry used for DINO feature alignment; and Final aligned pose showing the hand (obtained using HaMeR by Pavlakos et al. [19]) and the derived parallel gripper relative to the target object.



Alignment on Same Object Category

(a) Ladle to Ladle

(b) Mortar to Mortar

(c) Mug to Mug

(d) Scissors to Scissors

Alignments between Objects of Different Category

(e) Ladle to Grater

(f) Ladle to Tiller

(g) Ladle to Squeegee

(h) Ladle to Whisk

(i) Ladle to Paint Roller

(j) Mug to Mixing Bowl

(k) Mug to Pitcher

(l) Mug to Measuring Cup

(m) Pan to Spatula

(n) Pan to Grater

(o) Spatula to Pan

(p) Spatula to Spoon

(q) Spatula to Fork

(r) Spoon to Toilet Brush

(s) Spoon to Paint Roller

(t) Spoon to Hammer

Fig. 5: Examples of **Feature Guided Iterative Alignment**. In the sub-captions, the source object (retrieved from memory, often visualized as a red point cloud) is aligned to the target object (from the scene, often visualized as a blue point cloud). The first section shows alignments where source and target objects are of the same category (e.g., Ladle to Ladle). The second section demonstrates alignments between objects of different categories (e.g., Ladle to Grater), indicating the framework's ability to generalize across diverse pairings.

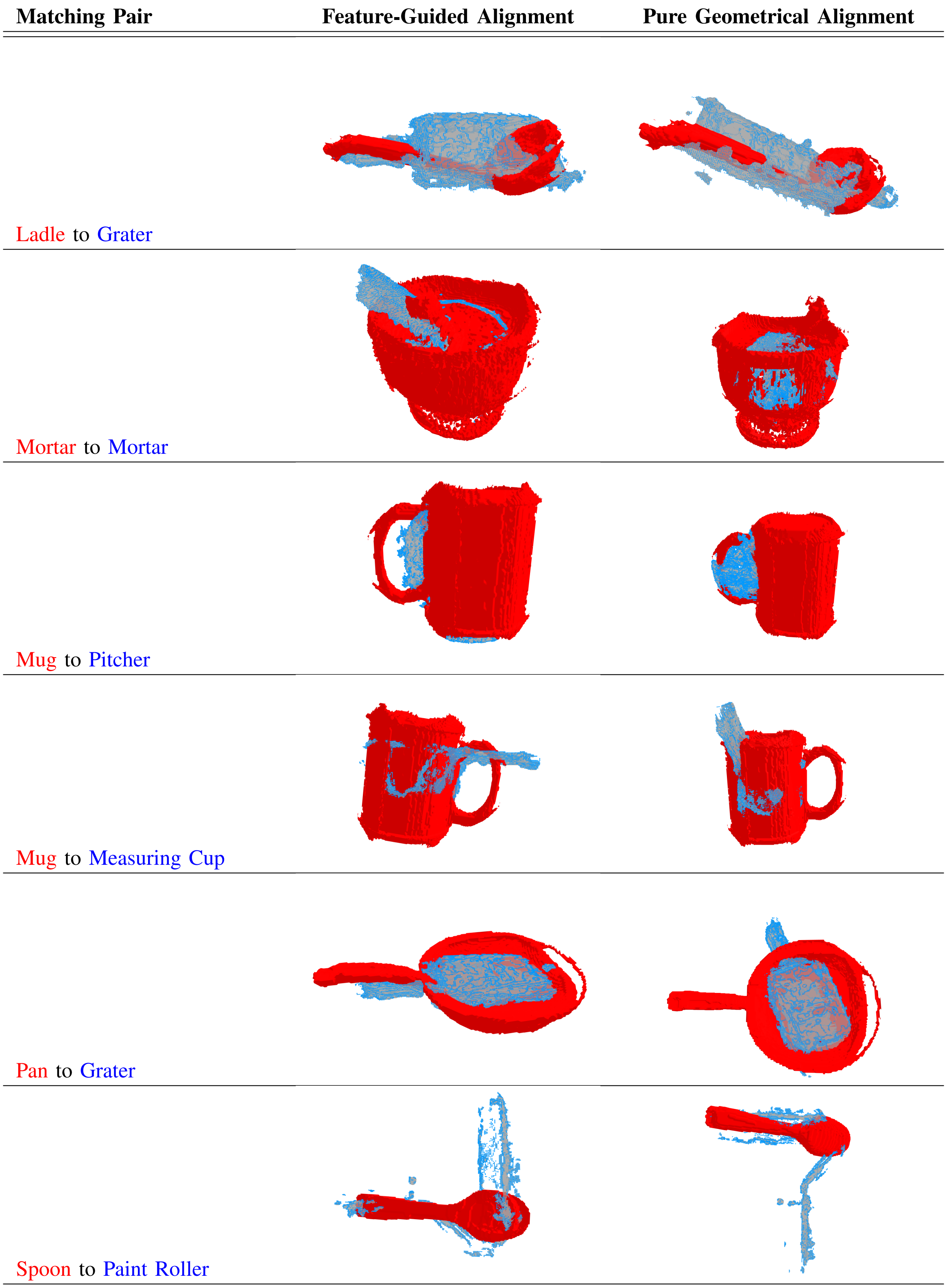


Fig. 6: Comparison of object alignments. Column 1 describes the matching pair (Source in red, Target in blue). Column 2 shows results from Feature-Guided Alignment, and Column 3 shows results from Pure Geometrical Alignment. Each row displays a corresponding pair.