GRIM: Task-Oriented Grasping with Conditioning on Generative Examples

Shailesh[1]
Alok Raj[1] Nayan Kumar[1] Priya Shukla[2]
Andrew Melnik[3] Micheal Beetz[3] Gora Chand Nandi[2]

[1] IIT ISM Dhanbad, India
[2] IIIT Allahabad, India
[3] University of Bremen, Germany

Abstract: Task-Oriented Grasping (TOG) requires a robot to select grasps that are not only stable but also functionally appropriate for a given task. This presents a significant challenge, demanding a nuanced understanding of task semantics, object affordances, and functional constraints. Existing learning-based approaches often struggle with generalization due to the scarcity of large-scale, task-annotated grasp datasets. To overcome these limitations, we introduce GRIM (Grasp Realignment via Iterative Matching), a novel, training-free framework for TOG. GRIM operates on a retrieve, align, and transfer paradigm. It first queries a memory of object-task examples, built from diverse sources including generative AI, web images, and human demonstrations. Given a new scene object, GRIM retrieves a semantically similar example and aligns its 3D geometry to the scene object using a robust coarse-to-fine strategy. This alignment is guided by a combination of geometric cues and a semantic similarity score over dense DINO features. Finally, the task-oriented grasp from the memory instance is transferred to the scene object and refined against a set of geometrically stable grasps to ensure task compatibility and physical feasibility. By eschewing task-specific training, GRIM demonstrates strong generalization, achieving state-of-the-art performance on benchmark datasets with only a small number of conditioning examples. Project Page.

Keywords: Task-Oriented Grasping, Robotic Manipulation, Foundation Models

1 Introduction

The ability for robots to physically interact with the world is fundamental to their utility. While grasp synthesis has made significant strides in achieving geometric stability, true manipulation intelligence lies in selecting grasps that are functionally suitable for a specific goal. This problem, known as Task-Oriented Grasping (TOG), moves beyond the question of "Can I pick this up?" to "How should I pick this up to complete task X?". For example, a hammer must be grasped by its handle to be used for hammering, not by its head. This requires a deep understanding of object affordances, task semantics, and the functional constraints they impose [1]. A primary bottleneck for progress in TOG is the data-scarcity problem. Supervised learning methods [1, 2] are powerful but depend on large, manually annotated datasets that specify which grasps are suitable for which tasks. Creating such datasets is labor-intensive and scales poorly, limiting the ability of these models to generalize to novel objects and tasks not seen during training.

To address these challenges, we propose **GRIM** (Grasp Re-alignment via Iterative Matching), a novel **training-free** framework that leverages the power of pre-trained foundation models in a retrieve-align-transfer pipeline [3, 4]. Instead of training a model on a fixed dataset, GRIM builds a dynamic, evergreen memory of object-task interactions from diverse and easily accessible sources:

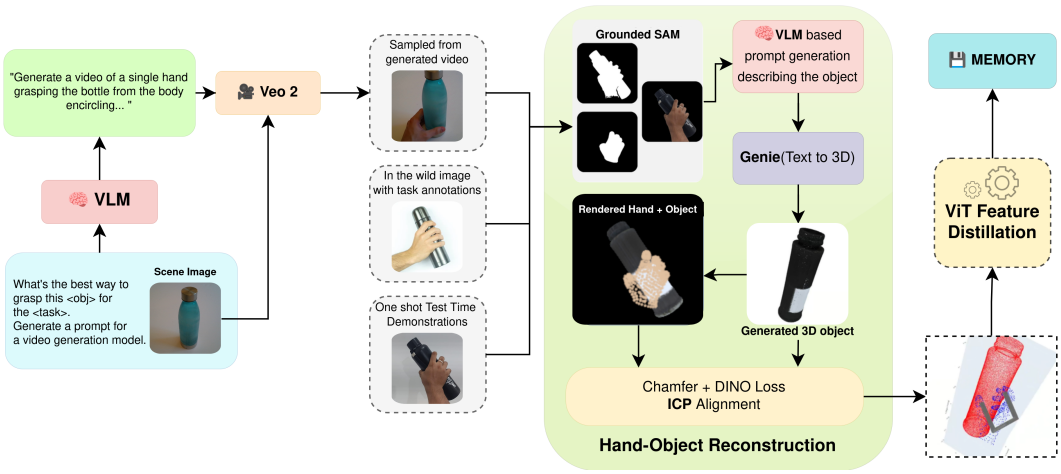


Figure 1: Our memory creation pipeline. A diverse set of inputs (AI-generated video frames, web images, human demonstrations) are processed by a hand-object reconstruction module [9]. This yields an object mesh and a corresponding task-oriented grasp pose. We enrich the object mesh with dense DINO features to create a feature mesh, which is stored in memory alongside the task label and grasp pose.

synthetic data from generative models, in-the-wild images from the web, and on-demand human demonstrations. This diverse memory provides a rich source of functional priors, inspired by the cognitive concept of the world as an external memory [5].

When faced with a new object and task, GRIM's workflow is as follows (Figure 2):

1. **Retrieve:** It queries its memory to find the most relevant prior experience, using a joint similarity metric that considers both the visual appearance of the object (via DINO embeddings [6]) and the semantics of the task description (via CLIP embeddings [7]).
2. **Align:** It robustly aligns the 3D point cloud of the retrieved memory object with the scene object. This is a key contribution, employing a coarse-to-fine strategy that first uses PCA-reduced DINO features for a semantically-aware coarse alignment, followed by a precise ICP [8] refinement.
3. **Transfer & Refine:** The task-specific grasp pose from the memory instance is transferred to the aligned scene object. This transferred pose then serves as a powerful prior to select and refine the best grasp from a set of pre-computed, geometrically stable candidates for the scene object.

Our main contributions are:

- A flexible and scalable memory construction pipeline that integrates object-task experiences from diverse sources, including a novel application of generative AI, circumventing the need for manually annotated datasets.
- A novel 3D alignment strategy that prioritizes semantic correspondence over geometric shape. By matching dense DINO features, our method works effectively even with sparse, partial point clouds where traditional geometry-based alignment techniques often fail.
- A complete training-free framework that shows generalization to both novel objects and novel tasks, validated through extensive experiments and real-world robot demonstrations.

2 Related Work

Task-Oriented Grasping (TOG) research has evolved from analytical methods to data-driven techniques, with a recent shift towards leveraging large-scale pre-trained models.

2.1 Data-Driven Approaches

Early data-driven methods learned direct mappings from object classes and tasks to grasps [10, 11]. However, these approaches often lacked semantic understanding and struggled to generalize [2]. To inject semantic knowledge, subsequent works utilized knowledge bases (KBs) and probabilistic logic [12, 13, 14, 15, 16], but these systems often require significant engineering to construct and scale the KBs.

The release of the TaskGrasp dataset by Murali et al. [1] was a significant step, enabling methods like GCNGrasp which uses a Graph Convolutional Network. However, such methods are inherently limited by the contents of their training data and knowledge graph, struggling to generalize to concepts unseen during training. More recent works like GraspGPT [2] and GraspMolmo [17] leverage Large Language Models (LLMs) and Vision-Language Models (VLMs) to incorporate open-world knowledge, improving generalization. Nevertheless, these models still rely on a foundational training phase on task-specific datasets [18, 19, 20], inheriting the associated data acquisition bottleneck.

GRIM diverges fundamentally from these paradigms. It is entirely training-free, obviating the need for task-specific grasp annotations. By dynamically building a memory from heterogeneous data, it directly tackles the data scarcity and annotation challenges that constrain prior methods.

2.2 Training-Free and Retrieval-Based Approaches

The advent of powerful foundation models has spurred the development of training-free TOG methods. Many approaches use LLMs or VLMs to provide semantic guidance, mapping a language command to a region on an object where a grasp should be executed [21, 22, 23]. While these methods avoid training, they typically only provide coarse spatial priors (e.g., "grasp the handle"), not directly executable 6D grasp poses.

Closer to our work are retrieval-based methods. RTAGrasp [24] also proposes a training-free approach using a memory of human demonstrations. It retrieves a relevant video and uses 2D feature matching to transfer a grasp point. While effective, its reliance on 2D matching can be ambiguous and less robust to viewpoint changes. RoboABC [25] uses CLIP to retrieve contact points but struggles to determine the full 6D grasp pose, particularly the crucial grasp orientation.

GRIM builds upon the strengths of retrieval but makes several key improvements. Our retrieval is guided by a joint 3D visual (DINO [6]) and task-semantic (CLIP [7]) similarity. Crucially, we introduce a robust, semantically-aware 3D alignment strategy that aligns entire object point clouds, not just 2D features [4]. This allows for a more precise transfer of the full 6D grasp pose, which is then further refined against the scene object's specific geometry. This holistic process addresses both "where" and "how" to grasp with high precision and adaptability, without the limitations of pre-defined datasets or explicit training.

3 Methodology

We introduce GRIM (Grasp Re-alignment via Iterative Matching), a training-free framework for TOG. Our approach follows a retrieve-align-transfer pipeline, detailed below.

3.1 Memory Creation

To generalize to novel scenes, we construct a memory $\mathcal{M}$ of object-task experiences from diverse data sources. Each instance in $\mathcal{M}$ is a tuple (F_M, G_t, T, O), containing the object's feature mesh F_M, a 6D task-oriented grasp pose G_t, the corresponding task description T, and the object name O.

The pipeline to create a single memory instance (Figure 1) begins with an image or video frame I_{HO} depicting a functional grasp. We use a hand-object reconstruction model [9] to extract the object mesh and hand mesh. We then derive a 6D parallel-jaw gripper pose G_t from the hand

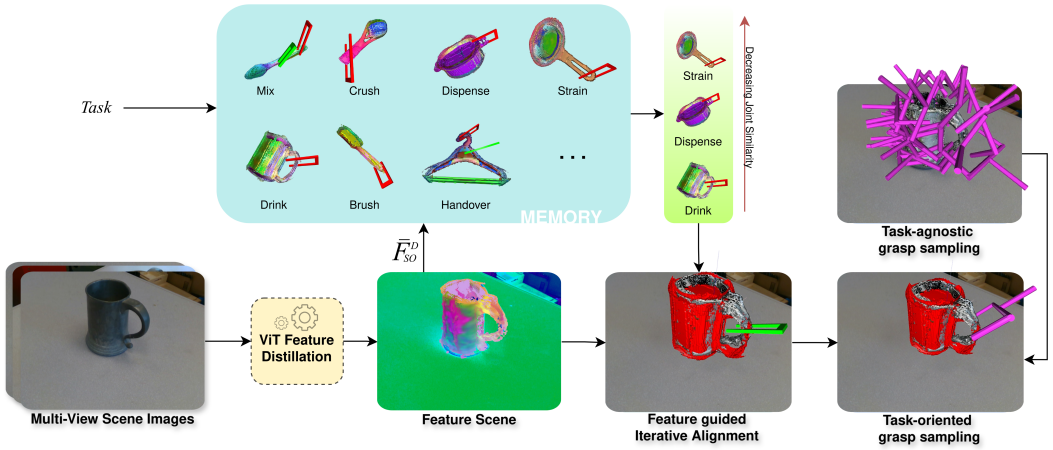


Figure 2: The GRIM pipeline for a given scene object and task. **(1) Retrieval:** The system queries its memory using joint visual and task similarity to find the best matching prior experience (a cup for the task 'drink'). **(2) Alignment:** The retrieved memory object (red point cloud) is aligned with the scene object (grey point cloud) using our feature-guided iterative alignment. The colors on the objects represent PCA-reduced DINO features, showing semantic correspondence. **(3) Transfer & Refine:** The grasp from the memory object is transferred to the scene object and used to select the best among a set of task-agnostic, stable grasp candidates (cluster of purple grasps), resulting in the final task-oriented grasp (single purple grasp).

mesh. This conversion is done by first identifying the centroids of hand segments: the thumb, the combined index and middle fingers, and the palm. The gripper's center (translation) is defined as the midpoint between the centroid of the thumb and the combined centroid of the opposing fingers. The vector connecting these centroids establishes the closing direction, and the palm's centroid provides a reference point to determine the approach vector.

To create the feature mesh F_M, we sample points from the object mesh and compute a dense DINOv2 feature vector for each point, effectively creating a semantic descriptor field on the object's surface, similar to Wang et al. [26] and PS et al. [27].

A key strength of our framework is its ability to ingest data from varied sources:

3.1.1 AI-Generated Videos.

To create a scalable and diverse data source, we leverage generative AI [28]. For an object and task from a source like TaskGrasp [1], we prompt a VLM (Gemini Pro) to generate a detailed textual description of a video showing the correct grasp. This description, along with a starting image frame, is then used as a prompt for a video generation model such as VEO2 [29] to generate a short video. We sample a frame from this video to serve as I_{HO}. This process allows for cheap, large-scale creation of functionally-grounded grasp data.

3.1.2 In-the-Wild Web Images.

We use images scraped from the web that show human grasping actions. For each image, we use a VLM to generate a plausible task description T.

3.1.3 Test-Time Expert Demonstrations.

Our framework supports lifelong learning. If the robot fails on a task, a human can provide a single-image demonstration, which is seamlessly processed and added to the memory $\mathcal{M}$, improving future performance on similar tasks [30].

3.2 Memory Retrieval

Given a novel scene containing a target object (represented by its point cloud P_{SO} with per-point DINO features F^D_{SO}) and a task command T_S, we retrieve the most relevant memory instance.

First, we compute a global visual descriptor $\bar{F}^D_{SO}$ for the scene object by averaging its per-point DINO features. We encode the task command T_S into a text embedding E_{T_S} using CLIP's text encoder.

For each memory instance $i \in \mathcal{M}$ with its global object descriptor $\bar{F}^D_{MO,i}$ and task embedding $E_{T_{M,i}}$, we compute a joint similarity score:

$$S_{\text{joint}}(i) = \alpha \cdot \text{sim}_{\cos}(\bar{F}^D_{SO}, \bar{F}^D_{MO,i}) + (1 - \alpha) \cdot \text{sim}_{\cos}(E_{T_S}, E_{T_{M,i}}) \tag{1}$$

where $\text{sim}_{\cos}(\cdot, \cdot)$ is the cosine similarity and α is a hyperparameter balancing visual and task similarity (we use $\alpha = 0.5$). The memory instance (F^*_M, G^*_t, T^*, O^*) with the highest S_{joint} is selected for the next stage.

3.3 Semantic 3D Alignment

After retrieving a memory object (source point cloud P_{MO}, DINO features F^D_{MO}), we must align it to the scene object (target point cloud P_{SO}, features F^D_{SO}). A purely geometric alignment like standard ICP would fail if the objects have different shapes (e.g., aligning a metal spatula to a plastic one). We therefore propose a coarse-to-fine alignment strategy guided by semantic features.

Coarse Alignment: To reduce the dimensionality and noise of the DINO features, we apply PCA, projecting both F^D_{MO} and F^D_{SO} into a lower 4-dimensional space ($D_{\text{PCA}} = 4$). We then perform a grid search over a discretized set of initial rotations to find a promising coarse alignment. Specifically, we sample 8 steps for each of the three Euler angles (roll, pitch, yaw), resulting in $8^3 = 512$ candidate rotations $\{R_i\}$. For each candidate, we compute a transformation $T_{\text{init},i}$ that aligns the point cloud centroids and applies the rotation. The quality of this initial transformation is evaluated using a joint feature-geometric score. For each point in the transformed source cloud, we find its $K = 3$ nearest neighbors in the target cloud and compute a cost based on a weighted sum of the squared Euclidean distance ($w_g = 10$) and the feature dissimilarity ($w_f = 100$). By heavily weighting the feature component, we prioritize finding a semantically meaningful alignment over a purely geometric one. The top 10 transformations with the lowest cost are selected as candidates for the fine refinement stage.

Fine Refinement: The best coarse alignment is then used to initialize the Iterative Closest Point (ICP) algorithm. This standard ICP step refines the alignment to be geometrically precise. This two-step process, where semantics guide the initial guess and geometry refines it, allows for robust alignment even between objects that are semantically similar but geometrically distinct. The final output is a transformation T_{final} that maps points from the memory object's coordinate frame to the scene object's frame.

3.4 Grasp Transfer and Refinement

With the alignment T_{final}, we transfer the task-oriented grasp G_M from memory to the scene object: $G_S = T_{\text{final}} \cdot G_M$. However, due to small alignment errors or geometric differences, G_S may not be perfectly stable or executable.

To find an optimal, executable pose, we follow a sample-and-refine strategy inspired by Dong et al. [24]. First, we use a task-agnostic grasp sampler, AnyGrasp [31], to generate a set of N geometrically stable grasp candidates $\{G_{A,i}\}_{i=1}^N$ on the scene object, each with a geometric quality score $S_{\text{geo},i}$.

We then re-rank these candidates based on their compatibility with our transferred task-oriented grasp $G_S = (R_S, \mathbf{t}_S)$. We define a task-compatibility score $S_{\text{task},i}$ for each candidate grasp $G_{A,i} =$

$(R_{A,i}, \mathbf{t}_{A,i})$:

$$S_{\text{task},i} = \underbrace{(\mathbf{v}_{\text{target}} \cdot \mathbf{v}_{A,i})}_{\text{Orientation Sim.}} + \underbrace{\exp\left(-\frac{\|\mathbf{t}_{A,i} - \mathbf{t}_S\|^2}{2\sigma^2}\right)}_{\text{Position Sim.}} \tag{2}$$

where $\mathbf{v}_{\text{target}}$ and $\mathbf{v}_{A,i}$ are the approach vectors of the grasps (e.g., the z-axis of the gripper frame), and σ is a bandwidth parameter (set to 0.02m). This score rewards candidates that are close in both position and orientation to the transferred task-centric pose.

The final score for each candidate is a weighted sum of its task compatibility and geometric quality:

$$S_i = w_{\text{task}} S_{\text{task},i} + w_{\text{geo}} S_{\text{geo},i} \tag{3}$$

We heavily prioritize task-compatibility by setting $w_{\text{task}} = 0.95$ and $w_{\text{geo}} = 0.05$, as AnyGrasp already ensures candidates have high geometric quality. The grasp candidate G_A^* with the highest final score S_i is selected for execution.

4 Experiments and Results

We conduct extensive experiments to evaluate GRIM's performance, focusing on its ability to generalize to novel objects and tasks.

4.1 Experimental Setup

Baselines: We compare GRIM against three representative baselines:

- **Random:** A task-agnostic baseline that randomly selects a geometrically stable grasp from the candidates provided by AnyGrasp.
- **RTAGrasp [24]:** A state-of-the-art training-free method that uses 2D feature matching to transfer grasps from a video memory.
- **GraspMolmo [17]:** A state-of-the-art learning-based VLM, which was fine-tuned on a mixture of its primary synthetic dataset (PRISM, 379k examples) and a portion of the TaskGrasp.

Dataset: We evaluate all methods on the TaskGrasp dataset [1], which provides object point clouds and annotated task-oriented grasps. To rigorously test generalization, we use two challenging splits:

- **Held-out Objects:** The memory contains no objects of the same category as the test object.
- **Held-out Tasks:** The memory contains no examples of the task being performed, even if it has seen the object category before.

Memory: Our memory for GRIM and RTAGrasp is constructed from a combination of 180 AI-generated video frames, 15 web images, and 15 human demonstrations, totaling 210 instances. This small size highlights the data efficiency of our approach. To ensure a fair comparison with RTA-Grasp, we build its memory from the same source images and derive its required 2D grasp points from our 6D poses.

Evaluation Metric: Following standard practice, we evaluate the methods on their ability to identify the correct task-oriented grasps from a set of proposals. We use the 25 annotated grasps for each object instance in TaskGrasp as candidates. A predicted grasp is considered correct if it is one of the positive examples for the given task. We report the Mean Average Precision (mAP) over all object-task pairs.

4.2 Quantitative Results

GRIM's effectiveness and data efficiency are demonstrated in our quantitative evaluations (Table 2). On the full TaskGrasp dataset, GRIM achieves a Mean Average Precision (mAP) of 0.67. This result

Table 1: Per-category Average Precision on novel object instances from the TaskGrasp dataset.

Method	Novel Instances							
	Paint roller	Brush	Tongs	Strainer	Pan	Fork	Mortar	Ice Scrapper
Random	0.30	0.66	0.23	0.24	0.32	0.26	0.31	0.60
RTAGrasp	0.39	**0.93**	0.28	0.55	0.42	0.35	0.37	**0.91**
GraspMolmo	0.56	0.73	0.55	0.44	0.46	0.53	0.66	0.65
GRIM(Ours)	**0.89**	0.90	**0.58**	**0.58**	**0.60**	**0.76**	**0.72**	0.71

not only surpasses the state-of-the-art training-free method, RTAGrasp (0.58), but also, remarkably, outperforms GraspMolmo (0.62). This comparison is particularly significant: GraspMolmo is a powerful VLM trained on a massive dataset of 379,000 synthetic task-oriented grasp examples, whereas GRIM's memory contains only 210 instances from heterogeneous, un-curated sources. This result strongly validates our central thesis: by effectively retrieving and re-aligning functional priors from a small but diverse memory, it is possible to achieve superior generalization without relying on vast, expensive, and potentially biased training datasets.

Furthermore, GRIM's performance advantage is most pronounced in the challenging generalization splits. On held-out objects and tasks, GRIM's mAP degrades by only 3%, whereas RTAGrasp's performance drops by over 10%. This underscores the robustness of our 3D semantic alignment strategy, which successfully transfers functional knowledge even without direct categorical or task precedents—a scenario where 2D feature matching proves less effective.

To understand why our method works well, we tested it without its key parts in an ablation study (Table 3). The results clearly show that semantic alignment is the most critical component. Without it (GRIM w/o Semantic Alignment), performance drops to 0.50 mAP, which is nearly as poor as the random baseline. This confirms that using features is crucial for aligning objects for a task, especially when their shapes differ. The grasp refinement step is also important. Without it (GRIM w/o Grasp Refinement), performance falls to 0.59 mAP. This means the transferred grasp is a good starting point for the task, but it must be fine-tuned to the scene object's geometry to be successful. In summary, both components are vital: semantic alignment provides the correct functional idea, and refinement makes that idea physically work.

A qualitative analysis further illuminates the behavior of the semantic alignment module, particularly its failure modes. Its performance is intrinsically linked to the fidelity of the input point cloud. In scenarios with severe sensor noise or extreme sparsity, the process of establishing dense feature correspondences can break down. This corrupts geometric priors like the centroid and leads to a flawed coarse alignment from which the local ICP refinement cannot recover. The final transferred grasp is consequently misplaced and functionally irrelevant. This underscores a key dependency: while GRIM is robust to partial views, its ability to reason functionally is contingent on receiving a partial point cloud of sufficient quality to support the crucial semantic alignment stage.

Table 2: Mean Average Precision (mAP) on the TaskGrasp dataset. GRIM consistently outperforms all baselines, with particularly strong performance on the held-out splits, demonstrating superior generalization.

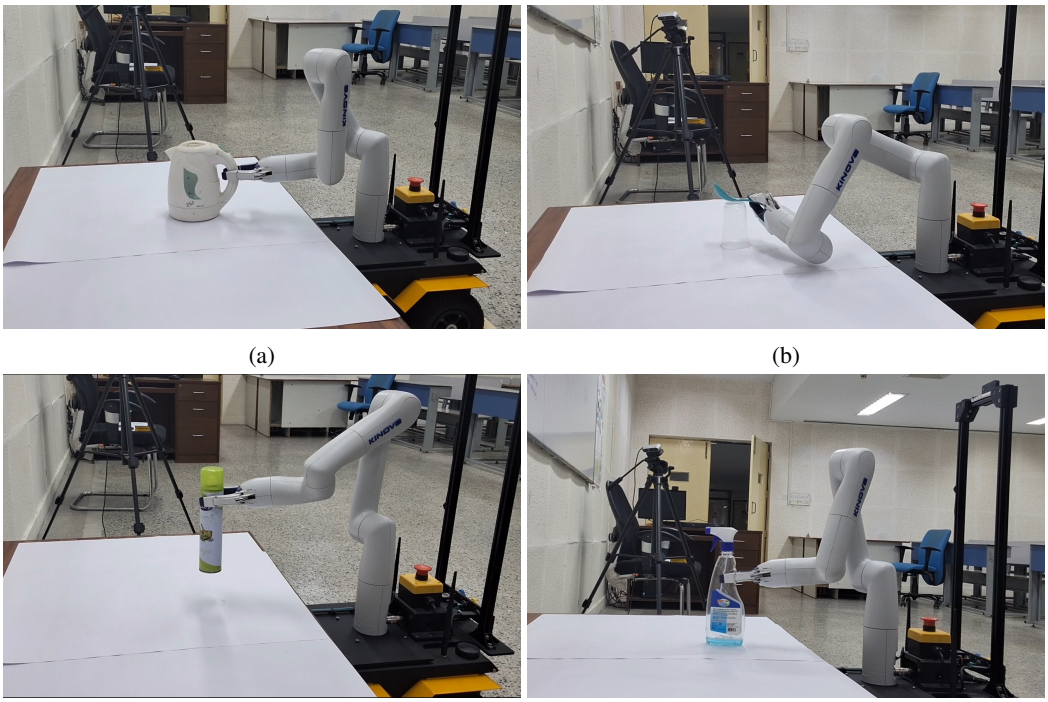
Method	**All Data**	**Held-out Obj.**	**Held-out Tasks**
Random	0.49	0.41	0.43
RTAGrasp	0.58	0.52	0.51
GraspMolmo	0.62	0.57	0.55
GRIM (Ours)	**0.67**	**0.65**	**0.64**

Table 3: Ablation study of GRIM's key components. Results are reported as Mean Average Precision (mAP) on the full TaskGrasp dataset, demonstrating the critical role of both semantic alignment and grasp refinement.

Configuration	**mAP (All Data)**
Ablations:	
GRIM w/o Semantic Alignment	0.50
GRIM w/o Grasp Refinement	0.59
GRIM (Full Model)	**0.67**

4.3 Real-World Robot Validation

To demonstrate the practical applicability of GRIM, we deployed it on a Kinova Gen3 Lite manipulator. The scene is captured by two RGB-D cameras. We used the same 210-instance memory from our simulation experiments, containing no instances of the test objects. We evaluated GRIM on 5 novel objects with associated tasks: a mallet ('hammer'), a kettle ('pour'), a spray bottle ('spray'), an aerosol can ('spray'), and a spoon ('scoop'). For each object-task pair, we performed 10 trials. GRIM achieved a high success rate, successfully executing the task-oriented grasp in 39 out of 50 trials. Failures were not due to flawed grasp selection but were instead traced to perception errors; specifically, noise in point cloud reconstruction and calibration inaccuracies were able to disrupt the subsequent 3D alignment stage. Figure 3 shows qualitative examples of successful executions.



(a) (b) (c) (d)

Figure 3: **Real-world deployment of GRIM with novel objects.** The system correctly plans and executes task-oriented grasps. The Kinova Gen3 Lite robot successfully executing the planned grasp.

5 Conclusion

We have presented GRIM, a training-free framework for task-oriented grasping that demonstrates remarkable generalization capabilities by retrieving and re-aligning functional priors from a diverse

memory. Our key innovation is a robust 3D alignment process guided by semantic features, which allows for effective knowledge transfer between objects that are functionally similar but geometrically different. By leveraging generative models and other readily available data sources, GRIM circumvents the data bottleneck that plagues traditional supervised methods. Our extensive experiments show that GRIM significantly outperforms existing training-free and learning-based approaches, particularly in its ability to handle novel objects and tasks.

Future work could explore incorporating explicit geometric reasoning, perhaps through the generation of digital twins [32], to further refine the alignment and grasp transfer process. Nevertheless, GRIM represents a significant step towards building more general, adaptable, and data-efficient robotic manipulation systems.

References

[1] A. Murali, W. Liu, K. Marino, S. Chernova, and A. Gupta. Same object, different grasps: Data and semantic knowledge for task-oriented grasping. In *Conference on Robot Learning*, 2020.

[2] C. Tang, D. Huang, W. Ge, W. Liu, and H. Zhang. Graspgpt: Leveraging semantic knowledge from a large language model for task-oriented grasping. *arXiv preprint arXiv:2307.13204*, 2023.

[3] Y. Kuang, J. Ye, H. Geng, J. Mao, C. Deng, L. Guibas, H. Wang, and Y. Wang. Ram: Retrieval-based affordance transfer for generalizable zero-shot robotic manipulation. *arXiv preprint arXiv:2407.04689*, 2024.

[4] N. Di Palo and E. Johns. Dinobot: Robot manipulation via retrieval and alignment with vision foundation models. *arXiv preprint arXiv:2402.13181*, 2024.

[5] A. Melnik, F. Schüler, C. A. Rothkopf, and P. König. The world as an external memory: The price of saccades in a sensorimotor task. *Frontiers in behavioral neuroscience*, 12:253, 2018.

[6] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193.

[7] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

[8] P. J. Besl and N. 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.

[9] J. Wu, G. Pavlakos, G. Gkioxari, and J. Malik. Reconstructing hand-held objects in 3d from images and videos. *arXiv preprint arXiv:2404.06507*, 2024.

[10] H. Dang and P. 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.

[11] W. Liu, A. A. 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.

[12] 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.

[13] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, P. Sermanet, N. Brown, T. Jackson, L. Luu, S. Levine, K. Hausman, and B. Ichter. Inner monologue: Embodied reasoning through planning with language models. In *arXiv preprint arXiv:2207.05608*, 2022.

[14] W. Liu, A. Daruna, M. Patel, K. Ramachandruni, and S. 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.

[15] P. Ardón, É. Pairet, R. P. A. Petrick, S. Ramamoorthy, and K. S. Lohan. Learning grasp affordance reasoning through semantic relations. *IEEE Robotics and Automation Letters*, 4: 4571–4578, 2019. URL https://api.semanticscholar.org/CorpusID:195345691.

[16] R. Zese, E. Bellodi, E. Lamma, F. Riguzzi, and F. Aguiari. Semantics and inference for probabilistic description logics. In F. Bobillo, R. N. Carvalho, P. C. Costa, C. d'Amato, N. Fanizzi, K. B. Laskey, K. J. Laskey, T. Lukasiewicz, M. Nickles, and M. Pool, editors, *Uncertainty Reasoning for the Semantic Web III*, pages 79–99, Cham, 2014. Springer International Publishing. ISBN 978-3-319-13413-0.

[17] A. Deshpande, Y. Deng, A. Ray, J. Salvador, W. Han, J. Duan, K.-H. Zeng, Y. Zhu, R. Krishna, and R. Hendrix. Graspmolmo: Generalizable task-oriented grasping via large-scale synthetic data generation, 2025. URL https://arxiv.org/abs/2505.13441.

[18] C. Tang, D. Huang, L. Meng, W. Liu, and H. 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.

[19] S. Jin, J. Xu, Y. Lei, and L. Zhang. Reasoning grasping via multimodal large language model. *ArXiv*, abs/2402.06798, 2024. URL https://api.semanticscholar.org/CorpusID:267627619.

[20] T. Nguyen, M. N. Vu, B. Huang, T. V. Vo, V. Truong, N. Le, T. D. Vo, B. Le, and A. 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.

[21] A. Rashid, S. Sharma, C. M. Kim, J. Kerr, L. Y. Chen, A. Kanazawa, and K. 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] R. Mirjalili, M. Krawez, S. Silenzi, Y. Blei, and W. Burgard. Lan-grasp: Using large language models for semantic object grasping, 2024. URL https://arxiv.org/abs/2310.05239.

[23] S. Li, S. Bhagat, J. Campbell, Y. Xie, W. Kim, K. Sycara, and S. 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.

[24] W. Dong, D. Huang, J. Liu, C. Tang, and H. Zhang. Rtagrasp: Learning task-oriented grasping from human videos via retrieval, transfer, and alignment. *arXiv preprint arXiv:2409.16033*, 2024.

[25] Y. Ju, K. Hu, G. Zhang, G. Zhang, M. Jiang, and H. Xu. Robo-abc: Affordance generalization beyond categories via semantic correspondence for robot manipulation. *arXiv preprint arXiv:2401.07487*, 2024.

[26] Y. Wang, M. Zhang, Z. Li, T. Kelestemur, K. Driggs-Campbell, J. Wu, L. Fei-Fei, and Y. Li. D^3fields: Dynamic 3d descriptor fields for zero-shot generalizable rearrangement. *arXiv preprint arXiv:2309.16118*, 2023.

[27] A. PS, A. Melnik, G. C. Nandi, et al. Splatr: Experience goal visual rearrangement with 3d gaussian splatting and dense feature matching. *arXiv preprint arXiv:2411.14322*, 2024.

[28] A. Melnik, M. Ljubljanac, C. Lu, Q. Yan, W. Ren, and H. Ritter. Video diffusion models: A survey. *Transactions on Machine Learning Research*, 2024.

[29] Google. Veo 2. via Google AI Studio, 2025. Accessed in May 2025. Model available at https://aistudio.google.com/.

[30] F. Malato, F. Leopold, A. Melnik, and V. Hautamäki. Zero-shot imitation policy via search in demonstration dataset. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7590–7594. IEEE, 2024.

[31] H.-S. Fang, C. Wang, H. Fang, M. Gou, J. Liu, H. Yan, W. Liu, Y. Xie, and C. Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains, 2023. URL https://arxiv.org/abs/2212.08333.

[32] A. Melnik, B. Alt, G. Nguyen, A. Wilkowski, Q. Wu, S. Harms, H. Rhodin, M. Savva, M. Beetz, et al. Digital twin generation from visual data: A survey. *arXiv preprint arXiv:2504.13159*, 2025.

[33] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Su, L. Zhu, L. Zhang, and Y. Qiao. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *arXiv preprint arXiv:2303.05499*, 2023.

[34] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Doll'ar, and R. Girshick. Segment anything. In *arXiv preprint arXiv:2304.02643*, 2023.

[35] Y. Wang, M. Zhang, Z. Li, T. Kelestemur, K. Driggs-Campbell, J. Wu, L. Fei-Fei, and Y. Li. D^3fields: Dynamic 3d descriptor fields for zero-shot generalizable rearrangement. In *8th Annual Conference on Robot Learning*, 2024.

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-Pro 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.

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. The prompts we use are detailed below.

VLM Prompt: Single Task to Video 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}.
```

VLM Prompt: K Distinct Grasping Ways

```
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
    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}".
```

VLM Prompt: Task-Video Mapping

```
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 4. 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.

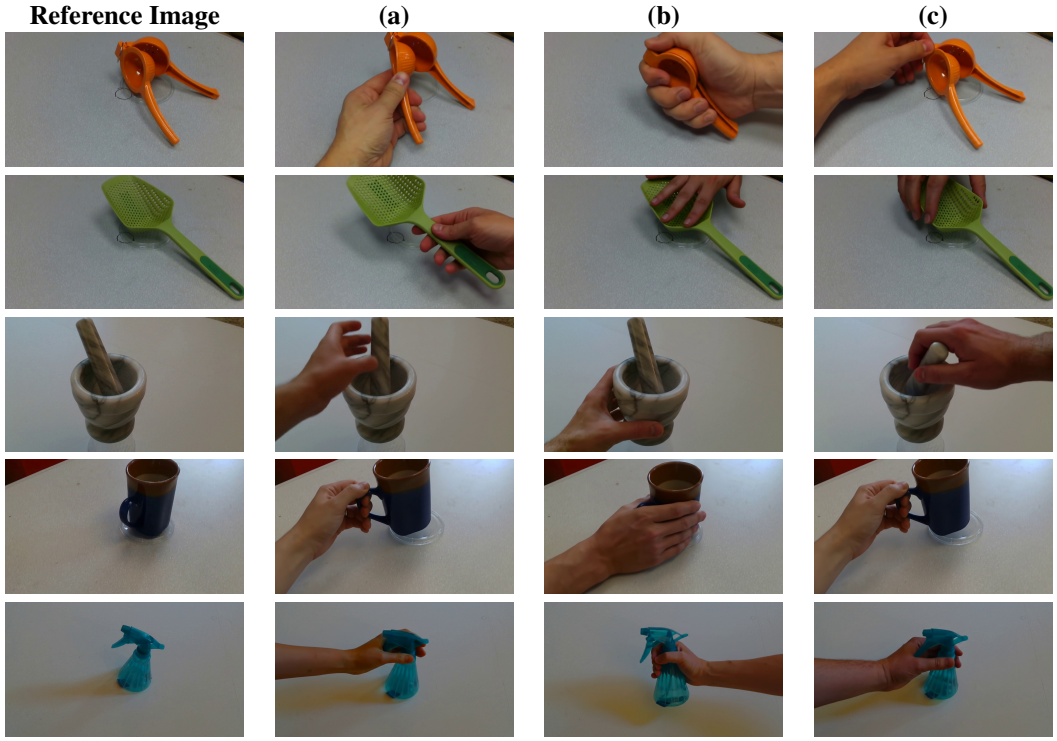


Figure 4: On the left we have the reference image used for video generation. (a), (b) and (c) are sampled frames from the generated videos using different task prompts.

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. [9] for hand-object 3D reconstruction. When processing AI-generated videos (as detailed in Appendix B), 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. [33]) with the Segment Anything Model (SAM) by Kirillov et al. [34]. In our implementation, we utilize a SAM model with a ViT-Base backbone (`facebook/sam-vit-base`)

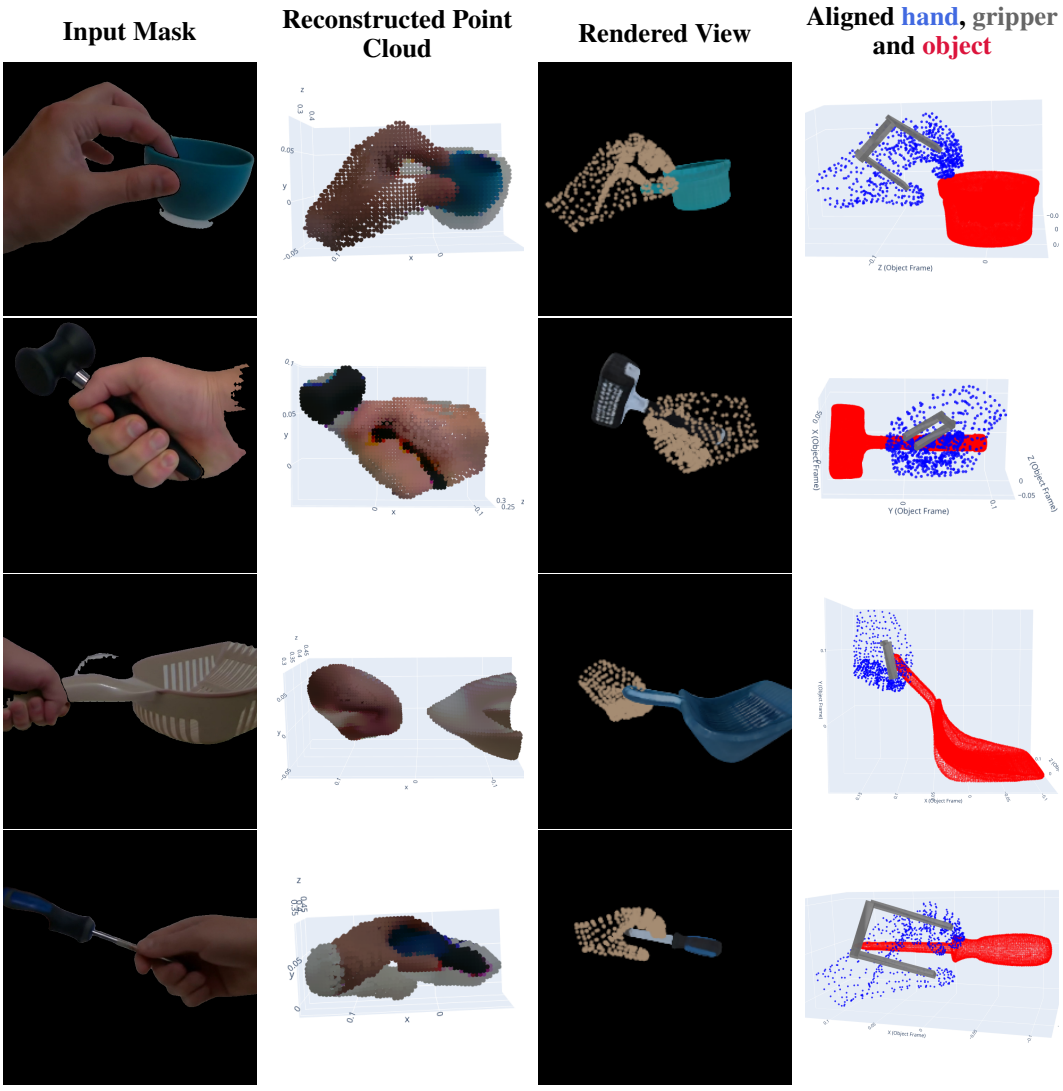


Figure 5: Visualization of the 3D hand-object reconstruction and grasp pose derivation pipeline. Each row shows four stages: Input image masked by Grounding SAM; Reconstructed 3D Point Cloud (PCD); Rendered view for DINO feature alignment; and Final aligned pose showing the reconstructed hand, the derived parallel gripper, and the target object.

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 ($L_{\text{DINO\_PCA}}$):

$$L_{\text{align}} = L_{\text{CD}}(P_{\text{target}}, P_{\text{cand}}(R, T, s)) + w_{\text{DINO}} \cdot L_{\text{DINO\_PCA}} \tag{4}$$

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

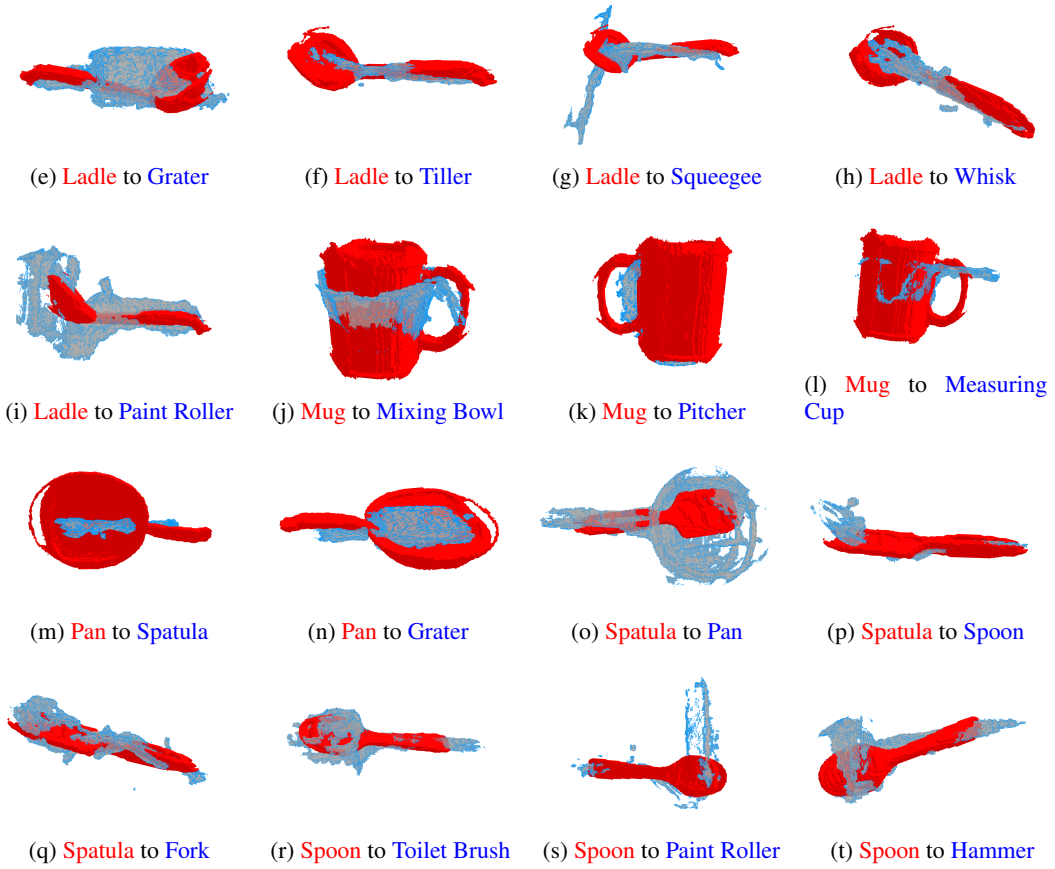
The most crucial part of our grasp transfer framework lies in Feature Guided 3D alignment. We use DINOv2-vitl14's visual features for creating our feature-rich point cloud, both for the memory object and the scene. Subsequently, we segment the target object using Grounded-SAM to obtain its feature-rich point cloud, a process similar to that described by Wang et al. [35]. 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. [1]—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

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

Figure 6: Examples of Feature Guided Iterative Alignment. The source object (from memory) is aligned to the target object (from the scene). The framework demonstrates robust alignment both within the same object category and across different categories, highlighting its generalization capabilities.

weighted sum:

$$C_{\text{pair}} = w_g \|p_m - p_{s,k}\|^2 + w_f (1 - \cos(F'^D_{M,p_m}, F'^D_{S,p_{s,k}})) \tag{5}$$

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 Iter-

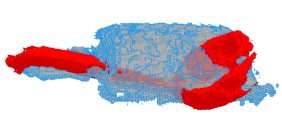
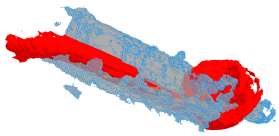
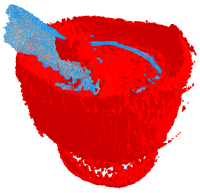
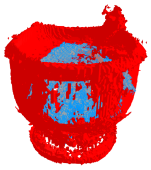
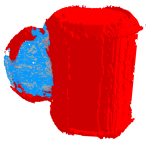
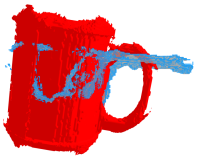
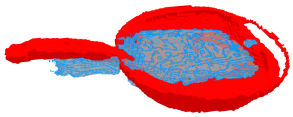
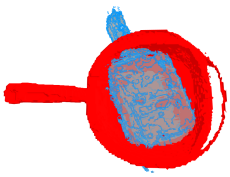
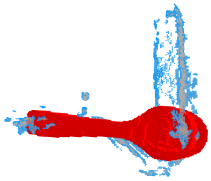
Matching Pair	Feature-Guided Alignment	Pure Geometrical Alignment
Ladle to Grater		
Mortar to Mortar		
Mug to Pitcher		
Mug to Measuring Cup		
Pan to Grater		
Spoon to Paint Roller		

Figure 7: Comparison of object alignments. Column 1 describes the matching pair (Source in red, Target in blue). Column 2 shows results from our Feature-Guided Alignment, which consistently produces semantically correct poses. Column 3 shows results from a Pure Geometrical Alignment, which often fails by converging to incorrect local minima or flipped orientations.

ative 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 6. 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 7 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 5, 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.