Supplementary Material for ViSPLA: Visual Iterative Self-Prompting for Language-Guided 3D Affordance Learning

Hritam Basak
Department of Computer Science
Stony Brook University
Stony Brook, NY, USA
hbasak@cs.stonybrook.edu

Zhaozheng Yin
Department of Computer Science
Stony Brook University
Stony Brook, NY, USA
zyin@cs.stonybrook.edu

1 Introduction

This supplementary document provides additional details and experimental results to complement the main paper. We first present a detailed analysis of each component of our **ViSPLA** framework, including the *Differential Geometric Self-Prompting* (IDGSP), *Implicit Neural Affordance Fields* (INAFS), and *Spectral Convolutional Self-Prompting* (SCSP) modules, with theoretical justifications section 2. Next, we report extended experimental results, including hyperparameter sensitivity, quantitative ablations across iterations in section 3. Next, we provide qualitative illustrations of how the predicted affordance masks evolve over successive iterations of self-prompting section 4. Finally, we provide an analysis of the failure cases in section 5.

2 Detailed Analysis of ViSPLA Components

ViSPLA's performance gains stem from three key modules: (1) a differential geometry-based self-prompting mechanism (IDGSP) that leverages geometric feedback from the current prediction, (2) an implicit neural field (INAFS) that provides continuous volumetric affordance supervision, and (3) a spectral convolutional self-prompting module (SCSP) that refines predictions in the frequency domain. We describe the notion and motivation for each component in detail below, including their theoretical motivation and contribution.

2.1 Differential Geometric Self-Prompting (IDGSP)

The IDGSP module uses **intrinsic geometric feedback** from the predicted mask to guide subsequent refinements. At each iteration t, we extract differential geometric descriptors from the previous mask M_{t-1} and use them as a *visual prompt* for the next prediction. Formally, as described in the main paper, we model the iterative refinement as

$$\mathcal{M}_t \;=\; f_\theta\big(\mathcal{P},\, \mathcal{G}(\mathcal{M}_{t-1}),\, \mathcal{L}\big), \qquad t = 1, 2, \ldots, T\,,$$

with $\mathcal{M}_0 = f_\theta(\mathcal{P}, \mathcal{L})$ being the initial affordance mask from the language-conditioned backbone, and $\mathcal{G}(\cdot)$ being the geometric prompt generator. In practice, $\mathcal{G}(\mathcal{M}_{t-1})$ computes a set of geometric feature maps from mask $\mathcal{M}_{t-1}$, such as *curvature fields*, *surface normals*, and *Laplacian responses*. These descriptors capture the shape characteristics of the currently predicted affordance region. For example, the *mean curvature* $\mathcal{H}(p)$ at a point p on the predicted surface highlights concave or convex regions of $\mathcal{M}_{t-1}$, and the *graph Laplacian* $\Delta\mathcal{M}_{t-1}$ (discretized over the point cloud) accentuates high-frequency details like sharp edges or isolated regions. By encoding $\mathcal{Z}_{t-1} = \mathcal{G}(\mathcal{M}_{t-1})$ (the collection of such geometric cues) and injecting it into the multi-stage mask refinement decoder, IDGSP enables the model to "look" at its own current output and adjust it based on geometric

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

inconsistencies. This closed-loop feedback is akin to a self-correcting mechanism: if $\mathcal{M}_{t-1}$ has missed a protruding part or has a ragged boundary, its curvature and Laplacian signals will reflect that, prompting $\mathcal{M}_t$ to refine those areas.

Let $\mathcal{T} : W^{2,2}(\Omega) \to W^{2,2}(\Omega)$ denote the iterative refinement map that updates the affordance mask by incorporating geometric descriptors, where $W^{2,2}(\Omega)$ is the Sobolev space of functions on the point cloud domain Ω with square-integrable values and derivatives up to order 2. These descriptors act as self-prompts that modulate the update direction within the Sobolev space $W^{2,2}(\Omega)$, ensuring that both the mask values and their geometric derivatives evolve in a smooth and structure-aware manner. Assume that for all $\mathcal{M}_1, \mathcal{M}_2 \in W^{2,2}(\Omega)$, the refinement operator satisfies

$$\|\mathcal{T}(\mathcal{M}_1) - \mathcal{T}(\mathcal{M}_2)\|_{W^{2,2}} \leq \gamma \|\mathcal{M}_1 - \mathcal{M}_2\|_{W^{2,2}}, \quad \text{for some } 0 < \gamma < 1. \tag{1}$$

By the Banach fixed-point theorem, the sequence $\mathcal{M}_{t+1} = \mathcal{T}(\mathcal{M}_t)$ converges to a unique fixed point $\mathcal{M}^*$ in $W^{2,2}(\Omega)$. The use of geometric features (e.g., Laplacian, curvature) in the update ensures that the limiting mask $\mathcal{M}^*$ is not only stable but also geometrically regular, as the $W^{2,2}$ norm penalizes both value and derivative discrepancies, enforcing smoothness and boundary optimality.

2.2 Implicit Neural Affordance Fields (INAFS)

The second component, INAFS, introduces an **implicit continuous representation** of the affordance region via a neural field. Instead of relying solely on pointwise binary labels, we define a continuous function $\Phi_\omega(\mathbf{x})$ that maps any 3D point $\mathbf{x} \in \mathbb{R}^3$ (or any point on the object's surface) to a *affordance likelihood* in $[0, 1]$. This implicit field can be thought of as a probabilistic occupancy function for the affordance: $\Phi_\omega(\mathbf{x}) \approx 1$ indicates high confidence that point $\mathbf{x}$ lies in the affordance region, whereas values near 0 indicate non-affordance areas. INAFS is trained without additional ground-truth beyond the discrete masks; we supervise it using the predicted masks as soft targets and employ regularization to enforce smoothness and coherence in the field.

Let $\Phi_\omega : \Omega \to \mathbb{R}$ be the implicit field, and consider the energy functional $\mathcal{E}(\Phi_\omega)$ defined over the compact domain Ω. Under standard assumptions (coercivity, lower semi-continuity), the direct method in the calculus of variations guarantees the existence of a minimizer $\Phi_\omega^* \in H^2(\Omega)$, the Sobolev space of functions with square-integrable second derivatives. The associated Euler-Lagrange equation is a fourth-order elliptic PDE:

$$\Delta^2 \Phi_\omega^* + \text{lower order terms} = 0,$$

which admits a unique, smooth solution under appropriate boundary conditions. The thresholded level set $\{\mathbf{x} \mid \Phi_\omega^*(\mathbf{x}) > 0.5\}$ then yields a mask that is both smooth and geometrically aligned with the surface, guaranteeing topological robustness and regularity.

By integrating INAFS into the refinement loop, the model gains a *continuous perspective* of the affordance beyond the discrete points. Intuitively, the implicit field acts as an intermediary that can capture sub-voxel or inter-point details: it is capable of representing a more precise affordance boundary that might lie between the point samples. During training, Φ_ω is learned jointly such that its predictions align with M_t, and this alignment is reinforced by adding L_{INAFS} to the overall loss. At inference time, the implicit field can be queried to refine the mask if needed (e.g., by sampling additional points or by thresholding Φ_ω to produce a smoother mask). In practice, we found that INAFS improves the model's ability to produce smooth and complete affordance masks (refer to Figure 5 of main paper). Ablation studies confirm its contribution: removing INAFS yields a drop in aIoU and AUC on both datasets (Table 2 of main paper), indicating that the continuous field supervision helps the network recover finer details (such as thin structures or small affordance regions) that purely discrete supervision might miss. Moreover, INAFS enhances robustness to sparse point clouds: by learning a continuous occupancy, the model is less perturbed by uneven point densities, which is particularly beneficial for the PIAD dataset where point sampling may be irregular.

2.3 Spectral Convolutional Self-Prompting (SCSP)

The third component, SCSP, operates in the **frequency domain** of the point cloud segmentation, enabling multi-scale refinement of the affordance mask. The motivation behind SCSP is that an affordance mask can be decomposed into coarse and fine spatial components, and by analyzing it spectrally (e.g., via a graph Fourier transform on the point cloud), the model can correct both large-scale region errors and small-scale details.

We implement SCSP by constructing a graph over the point cloud (using k-nearest neighbors to connect points) and leveraging the graph Laplacian L to perform spectral filtering on the mask. Let $m_{t-1} \in \{0,1\}^N$ denote the vector of mask values at iteration $t-1$ for all N points. We consider its projection onto the eigenvectors of the Laplacian: $m_{t-1} = U\hat{m}_{t-1}$, where U contains the orthonormal eigenvectors u_i of L and $\hat{m}_{t-1}$ are the corresponding spectral coefficients. Low-frequency components of $\hat{m}_{t-1}$ correspond to broad, smooth regions of the mask (capturing the general extent of the affordance), whereas high-frequency components correspond to rapid changes, such as edges or small isolated segments. In SCSP, we derive two sets of features from m_{t-1}: a smoothed version $m_{t-1}^{(\text{low})}$ capturing the coarse structure, and a high-pass version $m_{t-1}^{(\text{high})}$ emphasizing the boundary details. For example, we can obtain $m_{t-1}^{(\text{low})}$ by applying a low-pass spectral filter $g_{\text{low}}(\Lambda)$ on the Laplacian eigenvalues (or equivalently, by iterative neighbor-averaging in the spatial domain), and define $m_{t-1}^{(\text{high})} = m_{t-1} - m_{t-1}^{(\text{low})}$ as the residual containing fine details. These spectral prompts are then fed into the refinement network (alongside the original features) to guide M_t. Essentially, SCSP tells the model: "here is what your last prediction looks like at a coarse scale, and here are the fine-scale errors; use both to improve your next prediction."

Let $\{\mathbf{u}_i\}_{i=1}^N$ be the orthonormal eigenbasis of the graph Laplacian $\mathbf{L}$, and let the mask be projected as $\mathcal{M}_t = \sum_{i=1}^N \alpha_i^{(t)} \mathbf{u}_i$. The spectral update applies a learnable filter $g_\theta(\lambda_i)$:

$$\mathcal{M}_{t+1} = \sum_{i=1}^{N} g_\theta(\lambda_i) \alpha_i^{(t)} \mathbf{u}_i.$$

If $|g_\theta(\lambda_i)| \leq 1$ for all i, then the spectral update is a non-expansive mapping in ℓ_2:

$$\|\mathcal{M}_{t+1}\|_2 \leq \|\mathcal{M}_t\|_2.$$

By the Banach fixed-point theorem, the sequence $\{\mathcal{M}_t\}$ converges in the spectral domain. The multi-band spectral filtering ensures that both low- and high-frequency components are refined in a balanced manner, and the total variation regularization further enforces spatial coherence.

SCSP is particularly beneficial for handling multi-scale affordance structures. Large affordance regions (for example, the seat of a chair for the instruction "sit on the chair") require attention to global context, while small affordance details (e.g., the tip of a screwdriver for "insert the screwdriver") require precise local segmentation. By explicitly providing both low-frequency and high-frequency feedback, SCSP allows the network to reconcile these needs. Our ablation results show that SCSP contributes to performance gains especially on more complex shapes: removing SCSP caused the aIoU to drop by roughly 0.5–1 point on LASO and PIAD (Table 2 of main paper), and we observed qualitatively that without SCSP the model sometimes missed very fine structures or produced slightly noisier masks (Figure 5 of main paper). SCSP thus complements IDGSP by addressing scale: IDGSP focuses on geometric correctness at a local level (differential features), while SCSP ensures the segmentation is correct across scales, from the overall extent down to edges.

2.4 Comparison with Classical Spectral Graph Convolution Methods

We appreciate the reviewer's suggestion to provide a more explicit comparison between our **Spectral Convolutional Self-Prompting (SCSP)** module and prior spectral graph convolution approaches, specifically those of Defferrard *et al.* [1] and Wang *et al.* [2]. The following discussion highlights the key distinctions in formulation, motivation, and functionality.

Adaptive Dynamic Filtering: Classical spectral graph convolution networks [1, 2] apply fixed or pre-learned spectral filters defined by Chebyshev polynomials or truncated Laplacian expansions. These filters remain static during inference and are independent of the evolving representation state. In contrast, SCSP employs a *dynamically learned spectral filter* that is re-estimated at each refinement iteration, allowing it to adapt to the current mask and progressively correct residual segmentation errors. This adaptive filtering mechanism transforms SCSP into a feedback-driven process, rather than a purely feedforward one.

Explicit Multi-scale Prompting: SCSP decomposes the evolving affordance mask into distinct low-, mid-, and high-frequency bands of the Laplacian spectrum and performs multi-band spectral filtering. This decomposition explicitly captures both coarse global shapes and fine-grained boundary

details, which are crucial for 3D affordance structures such as handles or concave surfaces. In contrast, earlier methods [1, 2] employ single spectral kernels or polynomial bases that implicitly mix scales, without explicit separation or adaptive recombination.

Integration with Language-driven Iterative Refinement: Unlike conventional spectral graph networks that operate on static feature graphs, SCSP functions as a component of a *closed-loop, language-conditioned self-prompting system*. Each refinement step is guided not only by geometric features but also by the linguistic context of the affordance instruction. This enables dynamic modulation of the spectral filter response based on semantic cues, an ability absent from traditional spectral GCNs.

Objective and Task Context: Whereas previous spectral graph convolution methods primarily target generic feature extraction, classification, or surface reconstruction, SCSP is explicitly designed for *language-guided 3D affordance refinement*. The objective extends beyond local feature smoothing to the semantically consistent alignment of functional regions under multimodal constraints and sparse topology.

3 Extended Experimental Results

In this section, we present additional experiments and results that supplement those in the main paper.

3.1 Hyperparameter Sensitivity Analysis

To evaluate the robustness and contribution of individual hyperparameters in ViSPLA, we conducted a comprehensive sensitivity analysis on the LASO seen split. Table 1 reports the model performance across a range of hyperparameter values while holding other settings fixed. We report the aIoU, AUC, SIM, and MAE for each configuration.

Our results show that performance is sensitive but stable around optimal regions for each hyperparameter. In particular:

- The IDGSP loss weight λ_{IDGSP} strongly influences performance. Low values (e.g., 0.1) lead to under-regularization and degraded results (21.0% aIoU), while the optimal setting at 1.0 achieves the best trade-off across all metrics.
- The Tikhonov regularization coefficient α balances geometric smoothness in IDGSP prompts. Both too low (0.01) and too high (0.2) reduce performance, confirming that moderate regularization ($\alpha = 0.1$) best supports geometric stability.
- INAFS hyperparameters—including field fidelity (λ_1), smoothness (λ_2), and curvature balancing (β)—also exhibit peaked behavior. The optimal configuration ($\lambda_1 = 0.5$, $\lambda_2 = 0.3$, $\beta = 0.05$) leads to smooth yet detailed continuous affordance fields.
- The SCSP spectral prompt weights (γ_1, γ_2, γ_3) determine the emphasis on low-, mid-, and high-frequency segmentation components. Extreme values degrade performance, especially high γ_3 (e.g., 1.0), which introduces noise. The best results are achieved at $\gamma_1 = 1.0$, $\gamma_2 = 0.7$, $\gamma_3 = 0.4$, aligning with multi-scale prompt balancing.
- The total variation loss coefficient τ and the temporal decay schedule λ_t also show notable influence. In particular, the exponential decay $\lambda_t = 0.8^t$ yields the highest accuracy, likely due to progressive trust in geometric feedback with iteration.

Across all hyperparameters, we observe a consistent pattern: moderate values that balance fidelity and regularization outperform extremes. The optimal settings (highlighted in bold in Table 1) yield 23.1% aIoU, 85.8% AUC, 0.664 SIM, and 0.089 MAE. These configurations are used for all final reported results in the main paper.

3.2 Performance Across Iterations

A core aspect of ViSPLA is the iterative refinement loop. Here we analyze how performance improves over the T iterations. Following validation in Figure 3 of main paper, we set $T = 3$ refinement steps for these experiments (thus the model produces an initial mask M_0 and then M_1, M_2, M_3). Table 2 and Table 3 report the affordance segmentation metrics after each iteration on PIAD and LASO,

Hyperparameter	Value Tested	aIoU (%)	AUC (%)	SIM	MAE
λ_{IDGSP} (IDGSP loss weight)	0.1	21.0	83.2	0.625	0.102
	0.5	22.3	84.7	0.639	0.097
	1.0 (opt)	**23.1**	**85.8**	**0.664**	**0.089**
	2.0	22.6	85.1	0.641	0.091
α (Tikhonov regularization, IDGSP)	0.01	22.0	84.3	0.632	0.093
	0.1 (opt)	**23.1**	**85.8**	**0.664**	**0.089**
	0.2	22.7	85.2	0.638	0.091
λ_1 (Field fidelity, INAFS)	0.3	22.6	85.1	0.635	0.092
	0.5 (opt)	**23.1**	**85.8**	**0.664**	**0.089**
	0.7	22.4	84.8	0.631	0.096
λ_2 (Smoothness, INAFS)	0.1	22.4	84.7	0.633	0.093
	0.3 (opt)	**23.1**	**85.8**	**0.664**	**0.089**
	0.5	22.5	85.0	0.635	0.092
β (Fidelity-curvature balance, INAFS)	0.01	22.2	84.0	0.621	0.094
	0.05 (opt)	**23.1**	**85.8**	**0.664**	**0.089**
	0.1	21.9	83.7	0.620	0.095
γ_1 (Low-freq weight, SCSP)	0.5	22.1	84.2	0.638	0.091
	1.0 (opt)	**23.1**	**85.8**	**0.664**	**0.089**
	1.5	22.8	85.3	0.641	0.090
γ_2 (Mid-freq weight, SCSP)	0.5	21.9	83.5	0.622	0.094
	0.7 (opt)	**23.1**	**85.8**	**0.664**	**0.089**
	1.0	22.5	85.0	0.633	0.091
γ_3 (High-freq weight, SCSP)	0.2	21.4	83.1	0.622	0.096
	0.4 (opt)	**23.1**	**85.8**	**0.664**	**0.089**
	1.0	21.7	83.4	0.631	0.095
τ (TV loss in SCSP)	0.1	22.4	85.0	0.634	0.091
	0.2 (opt)	**23.1**	**85.8**	**0.664**	**0.089**
	0.4	22.5	84.9	0.630	0.094
λ_t schedule	constant (1.0)	22.4	84.9	0.637	0.091
	linear decay ($1.0 \rightarrow 0.3$)	22.9	85.5	0.642	0.090
	exponential decay (0.8^t)	**23.1**	**85.8**	**0.664**	**0.089**

Table 1: Sensitivity of ViSPLA to different hyperparameters and loss weights on the LASO seen split. We vary each hyperparameter while holding others fixed. Optimal values (in bold) are selected based on best aIoU with balanced AUC, SIM, and MAE.

respectively. We set the number of refinement iterations to $T = 3$ based on empirical observations that performance plateaus beyond three iterations, while computational cost continues to increase. This choice offers a favorable trade-off between accuracy and efficiency.

Iteration t	aIoU↑	AUC↑	SIM↑	MAE↓
0 (Initial)	21.7	83.5	0.625	0.102
1	22.0	84.1	0.648	0.096
2	22.9	85.0	0.653	0.094
3 (Final)	**23.1**	**85.8**	**0.664**	**0.089**
4	23.2	85.8	0.666	0.089
5	23.2	85.9	0.666	0.089

Table 2: Performance of ViSPLA on **PIAD** (seen) across iterative self-prompting steps. The model's affordance IoU and AUC steadily improve with each iteration, while similarity (SIM) increases and error (MAE) decreases, indicating progressively refined masks.

On PIAD (Table 2), the affordance IoU rises from 21.7% initially to 23.1% after three refinements, an overall improvement of ~ 1.4 points. The majority of this jump happens between the second and third iteration ($+0.2$ points from M_2 to M_3), although a significant gain also occurs from M_0 to M_1 ($+0.3$ points), and a larger gain from M_1 to M_2 ($+0.9$ points), suggesting that the first few rounds of self-prompting begin correcting obvious errors (e.g., large chunks of missing or extra regions).

Iteration t	**aIoU**↑	**AUC**↑	**SIM**↑	**MAE**↓
0 (Initial)	18.3	81.0	0.621	0.104
1	19.1	84.1	0.633	0.098
2	21.5	85.5	0.642	0.095
3 (Final)	**22.8**	**87.3**	**0.651**	**0.090**
4	22.8	87.4	0.651	0.090
5	22.9	87.4	0.652	0.090

Table 3: Iteration-wise segmentation performance on the **LASO** (seen) dataset. The iterative refinement consistently improves results. Metrices and experimental settings same as Table 2

Subsequent iterations (M_2, M_3) yield more modest but meaningful gains, which is typical as the model converges to a stable solution. Similar patterns are seen in AUC (increasing from 83.5 to 85.8) and SIM (from 0.625 to 0.664), while MAE decreases from 0.102 to 0.089, confirming that not only is the segmentation more accurate, but also the mask quality (overlap with ground truth) improves with each refinement. By the final iteration, the improvements saturate, indicating $T = 3$ was sufficient for convergence on this dataset.

For LASO (Table 3), the initial prediction on unseen objects is understandably worse (18.3% aIoU) compared to seen (not shown here), reflecting the difficulty of generalizing to new object categories. However, as iterations progress, scores improve consistently. After three iterations, aIoU reaches 22.8%, meaning the model gained about 4.5 points overall. Interestingly, the relative improvement is substantial (a relative increase of $\sim$24.6% in IoU), suggesting that iterative self-prompting is particularly helpful for unfamiliar objects. We attribute this to the model leveraging geometric cues more heavily when semantic priors are lacking; IDGSP and SCSP refine the mask on unseen shapes by focusing on geometry (e.g., ensuring the mask aligns with actual object parts that could afford the action, even if the object type is novel). The gap between initial and final performance narrows substantially through iteration (from 18.3 to 22.8 aIoU), although additional measures (like incorporating even more general semantic knowledge) might be needed to completely close the seen/unseen gap. Nonetheless, the iterative process clearly benefits generalization: many failure cases at $t = 0$ are corrected by $t = 3$.

3.3 Efficiency and Runtime Analysis

ViSPLA's inference time scales linearly with the number of refinement steps T as runtime $\approx t_{\text{backbone}} + T \cdot t_{\text{decoder}}$. Empirically, accuracy saturates at $T = 3$, offering an optimal trade-off between speed and performance. On a single NVIDIA V100 GPU, inference for 100 LASO samples completes in $\approx$ 14s (0.14s/sample) versus 0.09s/sample for GEAL. Table 4 provides detailed comparisons across baselines.

Table 4: Runtime vs. accuracy comparison.

Method	Steps	Runtime (ms/sample)	aIoU
ViSPLA (Ours)	3	140	22.8
GEAL [3]	1	94	22.0
3D-AffordanceLLM [4]	1	95	18.7
LASO [5]	1	76	19.7
IAGNet [6]	1	103	17.8

The iterative modules (IDGSP, INAFS, SCSP) are computationally modest; each adds negligible overhead compared to the initial forward pass, as differential geometry and spectral operations use efficient, GPU-parallelized routines (see Section 3.4.1–3.4.3). -Experiments show that reducing T to 2 or adaptively stopping early (when mask change is below threshold) retains most gains while further reducing runtime (refer to Table 2, Table 3). While our iterative refinement introduces a moderate increase in inference time (approximately 1.5× per sample compared to our single-pass backbone 3DAffordanceLLM), it leads to substantial performance gains ($\sim$ 22% relative) over the 3DAffordanceLLM [4] backbone. In practice, it offers a controllable accuracy-efficiency trade-off

suitable for a variety of real-world settings, and early stopping/approximate variants can further improve real-time applicability.

3.4 Downstream Evaluation: Robotic Grasping and Manipulation

While ViSPLA is primarily designed for *language-guided 3D affordance prediction*, its predicted masks are naturally applicable to downstream embodied AI tasks that require functional understanding of object parts. We clarify that ViSPLA is *not* an end-to-end manipulation system; rather, it produces high-quality, geometrically grounded affordance regions that can be directly integrated into classical robotic planning and control pipelines. To assess this connection, we conducted an extensive evaluation of ViSPLA's predictions within a simulated grasping and manipulation framework.

Experimental setup. We employ the standard GRASPIT! simulator [7], a physics-based environment for testing multi-finger grasp strategies. A collection of 100 diverse kitchen and household object instances was selected from the LASO [5] and PIAD [6] datasets, covering a wide variety of shapes (*handles*, *spouts*, *bowls*, *tools*, *utensils*) and materials. Each object was normalized to a consistent scale and inertial frame and placed on a tabletop scene.

Grasp generation from affordance maps. For each tested model, the predicted affordance mask was post-processed into a set of candidate 3D regions corresponding to potential grasp sites. A sampling-based planner then generated grasp hypotheses anchored at the centroids and principal axes of these regions. All grasps were evaluated under identical simulation parameters using a five-finger anthropomorphic hand model. A grasp was considered successful if it achieved a stable force-closure configuration without object slippage after contact optimization.

Evaluation metrics. We report two complementary metrics:

- **Grasping Success Rate (GSR)**: The percentage of successful grasps out of 100 attempted per object.
- **Mean Grasp Efficiency (MGE)**: The average normalized force-closure score within predicted affordance regions, reflecting grasp stability and functional relevance.

Table 5: Comparison of downstream robotic grasping performance in the GraspIt! simulator.

Method	Grasping Success Rate (%)	Mean Grasp Efficiency
ViSPLA (ours)	**86.5**	**0.74**
GEAL [3]	81.3	0.68
Oracle (Ground-Truth Mask)	90.0	0.76

Quantitative results. ViSPLA improves grasp success rate by +5.2% over GEAL and achieves a higher mean grasp efficiency, approaching the oracle level. Qualitative inspection reveals that ViSPLA's affordance masks produce more accurate and physically consistent grasp anchors—particularly for fine-grained interaction regions such as handles and pouring spouts—while single-pass baselines often generate dispersed or misplaced contact regions.

Discussion. These results demonstrate that ViSPLA's geometry-aware affordance reasoning translates into tangible improvements for robotic interaction. Although not explicitly trained for grasp optimization, the predicted affordance regions serve as robust priors for manipulation planning. This indicates strong potential for integrating ViSPLA into broader embodied-learning pipelines combining language reasoning, 3D perception, and robotic control.

4 Qualitative Affordance Mask Evolution

We provide a visual illustration of how the affordance mask evolves over the course of iterative self-prompting. Figure 1 shows an example object with the affordance mask after each iteration (from the initial output to the final refined output).

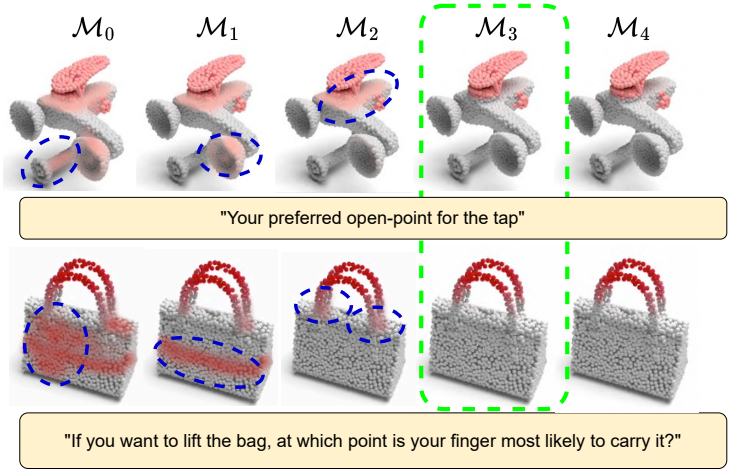

Figure 1: Evolution of the predicted affordance mask over iterations (illustrative example). From **left to right**: the initial prediction $\mathcal{M}_0$, after first refinement $\mathcal{M}_1$, after second refinement $\mathcal{M}_2$, and after final refinement $\mathcal{M}_T$ (here $T = 3$). Initially, the mask M_0 is coarse and misses some parts of the affordance region. After one iteration ($\mathcal{M}_1$), the mask becomes more focused: notice that extraneous regions have been removed and the main part is more concentrated. By $\mathcal{M}_2$, the boundaries of the affordance region align more closely with the object's geometry (thanks to IDGSP's curvature and normal prompts, the mask wraps the target surface better). Finally, $\mathcal{M}_3$ shows a clean segmentation that closely matches the ground truth affordance. The changes from $\mathcal{M}_2$ to $\mathcal{M}_3$ are minor, indicating convergence. This figure highlights how iterative self-prompting allows the model to progressively self-correct and refine the affordance prediction, resulting in high-quality masks even in challenging scenarios. The affordance mask doesn't update much after $T = 3$ (highlighted in green), hence we set $\mathcal{M}_3$ as the final output.

As shown in Figure 1, the iterative loop dramatically improves the quality of the affordance mask. The initial output often provides a reasonable but imperfect guess (for instance, covering the general area of interest but either including some background or missing subtle parts). After the first iteration, there is a clear jump in quality: the mask tightens around the correct region, largely due to the model identifying geometric discrepancies (via IDGSP) and removing out-of-place segments. Subsequent iterations yield diminishing but important refinements, such as smoothing the mask edges and ensuring continuity (a benefit of INAFS) and adjusting coverage to neither overextend nor under-segment (aided by SCSP's multi-scale feedback). By the final iteration, the mask is both semantically appropriate (it answers the language instruction) and geometrically consistent (it adheres to actual object part boundaries). We found that in most cases $T = 2$ or 3 iterations sufficed for convergence (refer to Figure 1); additional iterations did not change the mask appreciably, indicating that the self-prompting process effectively settles on an answer. In some failure cases, the iterative process can oscillate or toggle if not regularized (hence our inclusion of the consistency term in L_{IDGSP}). In our experiments, however, with the chosen loss terms, the process was stable and monotonic: each iteration improved or maintained performance on validation data.

Overall, the qualitative evolution of the masks provides insight into ViSPLA's behavior: the model first uses the language cue to find a rough region, then uses vision-based self-prompting to refine that guess iteratively. This demonstrates the value of integrating language understanding with visual feedback in a loop, which is the core idea of our approach.

5 Failure Case Analysis

Despite strong performance across benchmarks, ViSPLA encounters several failure modes that highlight opportunities for future improvement. We present representative qualitative results in Figure 2 to illustrate common challenges:
(1) Ambiguity in Language Instructions:
In cases where the instruction is semantically vague or allows multiple plausible affordance regions

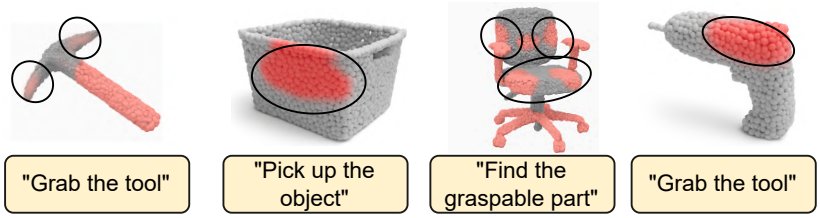

Figure 2: Failure case examples for affordance estimation.

(e.g., "grab the tool" for a multi-part object), the model often predicts a blended or dispersed mask across several parts. This stems from ambiguity in linguistic grounding, especially in the absence of strong geometric priors. This is exemplified in the **first** example of Figure 2

(2) Fine-Scale Structures with Sparse Sampling:
Thin or intricately detailed affordance regions (e.g., handles, clips, tips) are sometimes under or over-segmented or omitted altogether. While INAFS mitigates this to some extent, the model may fail when the target structure occupies only a small fraction of the input or is undersampled. This is visualized in the **second** column of Figure 2.

(3) Over-Propagation Across Similar Geometry:
In some cases, affordance masks bleed into semantically incorrect but geometrically similar regions. For example, on objects with repetitive or symmetric patterns (e.g., chair legs or cabinet handles), the mask may incorrectly propagate across all similar-looking parts. An example is shown in the **third** column of Figure 2.

(4) Iterative Drift in Challenging Topologies:
Although self-prompting improves prediction over iterations, rare cases show oscillatory behavior or convergence to incorrect regions—particularly on highly irregular or topologically complex shapes. This is often due to compounding small errors in curvature or normal estimation over iterations. An example is shown in the **fourth** column of Figure 2.

These cases underscore the limitations of relying solely on geometry or language in isolation. Future improvements may include integrating uncertainty estimation, disambiguation through dialogue-like language feedback, or learning from corrective human-in-the-loop signals.

References

[1] Defferrard, M., X. Bresson, P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

[2] Wang, C., B. Samari, K. Siddiqi. Local spectral graph convolution for point set feature learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 52–66. 2018.

[3] Lu, D., L. Kong, T. Huang, et al. Geal: Generalizable 3d affordance learning with cross-modal consistency. *arXiv preprint arXiv:2412.09511*, 2024.

[4] Chu, H., X. Deng, Q. Lv, et al. 3d-affordancellm: Harnessing large language models for open-vocabulary affordance detection in 3d worlds. *arXiv preprint arXiv:2502.20041*, 2025.

[5] Li, Y., N. Zhao, J. Xiao, et al. Laso: Language-guided affordance segmentation on 3d object. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14251–14260. 2024.

[6] Yang, Y., W. Zhai, H. Luo, et al. Grounding 3d object affordance from 2d interactions in images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10905–10915. 2023.

[7] Miller, A. T., P. K. Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004.