FAILURE-AWARE ADAPTATION FOR TRANSPARENT OBJECTS IN FEED-FORWARD 3D RECONSTRUCTION

Anonymous authors
Paper under double-blind review

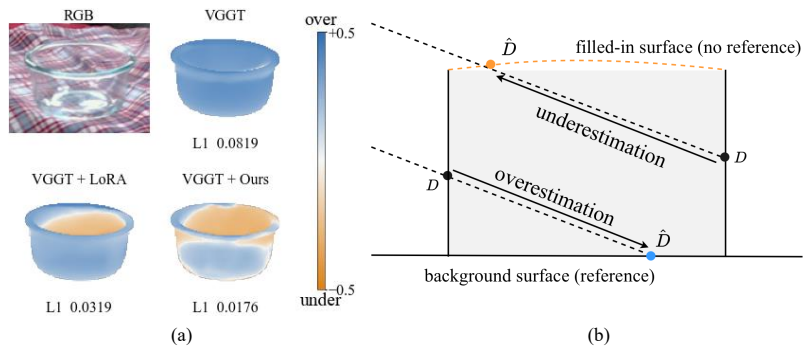


Figure 1: Feed-forward 3D foundation models fail on transparent objects in two opposite directions. (a) L1 depth error for a transparent object under zero-shot prediction, LoRA, and Ours. The zero-shot prediction is dominated by overestimation associated with background leakage. LoRA reduces the error, but underestimation around the concave interior remains, whereas our method corrects both error directions. (b) The two error directions differ in reference availability. Background leakage can be related to an observable background surface, whereas a filled-in concavity has no analogous geometric reference for recovering the missing depth.

ABSTRACT

Feed-forward 3D foundation models reconstruct scene geometry from a set of input views without per-scene optimization, yet remain unreliable for transparent objects. We analyze these failures and find that transparent-object errors concentrate among a small fraction of high-error predictions and separate into two asymmetric modes, *overestimation* and *underestimation*. Overestimation is primarily associated with background leakage, whereas underestimation concentrates around concave structures, revealing distinct geometric characteristics between the two failure modes. Motivated by this asymmetry, we introduce a failure-aware parameter-efficient adaptation framework with two complementary modules. Background-aware LoRA selectively adapts regions associated with background leakage using background geometric cues. Curvature-Informed Corrector infers error directions from local geometric cues and corrects overestimation and underestimation through separate correction paths. Our method achieves state-of-the-art performance on depth estimation benchmarks for transparent objects while consistently improving reconstruction accuracy across different feed-forward 3D foundation models.

1 INTRODUCTION

Transparent object perception is important for reliable geometric understanding in real-world environments, particularly for indoor navigation and robotic manipulation (Fang et al., 2022; Jiang et al., 2023). Among the required geometric cues, depth is critical for determining where transparent objects are located in 3D space. However, light transmission and refraction obscure visual cues to the object surface, making its depth difficult to recover (Jiang et al., 2023). Existing approaches rely on RGB-D measurements (Zhu et al., 2021; Fang et al., 2022; Chen et al., 2022a), per-scene

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

optimization (Li et al., 2023; Agrawal et al., 2024; Kim et al., 2025), or transformations that make transparent objects appear opaque before depth estimation (Wang et al., 2026a; Zhao et al., 2026).

Recent feed-forward 3D foundation models such as DUSt3R (Wang et al., 2024), VGGT (Wang et al., 2025), and π^3 (Wang et al., 2026b) offer a promising alternative by reconstructing scene geometry from multiple input views in a single forward pass without per-scene optimization. Despite strong performance across diverse scenes, they still produce severe errors on transparent objects, as shown in the Figure 1 (a). This raises a natural question. *How does transparency-induced ambiguity manifest in feed-forward 3D foundation models, and how can its structure guide efficient adaptation?*

We find that transparent-object errors exhibit a distinct structure in both magnitude and direction. A disproportionate share of the error is concentrated among a small fraction of high-error predictions, where low-rank adaptation (Hu et al., 2021) becomes progressively less effective. These errors further separate into two modes, **overestimation** and **underestimation**, as illustrated in the Figure 1 (b). Overestimation is primarily associated with background leakage, where the prediction approaches the background visible through the transparent object. Underestimation instead concentrates around concave structures that the model tends to flatten, consistent with a convexity prior (Langer & Bülthoff, 2001; Liu & Todd, 2004). The two modes also differ in the geometric information available for correction. Background leakage can be related to the visible background, whereas errors around concave structures must instead be inferred from local object geometry. This asymmetry suggests that allocating additional adaptation capacity alone is insufficient and that the two failure modes require different geometric cues for correction.

Motivated by this structure, we introduce two parameter-efficient modules on top of a frozen feed-forward 3D foundation model. *Background-aware LoRA* uses geometric cues from the visible background to identify regions associated with background leakage and selectively applies an additional low-rank update to these regions. *Curvature-Informed Corrector* uses local geometric cues derived from the predicted depth to infer error directions and corrects overestimation and underestimation through separate correction paths. Together, the two modules form a sequential correction framework that first suppresses large overestimation errors associated with background leakage and subsequently corrects smaller errors in both directions.

We evaluate our method on depth estimation benchmarks for transparent objects. Our method achieves state-of-the-art performance against existing approaches while using a small number of trainable parameters. Applying the same adaptation framework to three feed-forward 3D foundation models consistently improves reconstruction performance across different backbone architectures. The performance gains are largely maintained when the adapted models are evaluated on other datasets without additional training.

Our contributions are as follows.

- **Structured analysis of depth estimation failures on transparent objects.** We show that depth estimation errors on transparent objects concentrate among a small fraction of high-error pixels and separate into two asymmetric modes. Overestimation is primarily associated with background leakage, whereas underestimation concentrates around concave structures. We further identify a key asymmetry in the geometric information available for correcting the two error modes.
- **Failure-aware parameter-efficient correction.** Guided by this analysis, we introduce Background-aware LoRA to selectively adapt regions associated with background leakage and Curvature-Informed Corrector to infer error directions from local geometric cues and apply direction-specific correction. The proposed framework outperforms parameter-matched LoRA baselines and consistently improves performance across different feed-forward 3D foundation models.

2 Related Work

Transparent Object Geometry Estimation Estimating the geometry of transparent objects has been studied from several directions. Depth completion methods restore corrupted RGB-D measurements using geometric cues (Sajjan et al., 2020), synthetic data and sensor simulation (Dai et al.,

2022; Fang et al., 2022), or implicit representations (Zhu et al., 2021). Other approaches explicitly model transparent geometry through refracted ray tracing or differentiable rendering with per-scene optimization (Li et al., 2023; Dai et al., 2023; Agrawal et al., 2024; Kim et al., 2025). More recent methods build on pretrained depth models by transforming transparent regions into opaque appearances (Wang et al., 2026a; Zhao et al., 2026), directly adapting the model to transparent-object data (Costanzino et al., 2023; Liu et al., 2025), or repurposing diffusion priors for depth estimation (Xu et al., 2025; Verma et al., 2026). Prior analyses have also reported large geometric errors around container interiors and openings (Jiang et al., 2022; Wang et al., 2026a) and examined whether depth models recover the transparent surface or the visible background (Xu et al., 2026). Our work extends this line of analysis to feed-forward 3D foundation models by characterizing transparent-object errors along both magnitude and direction.

Feed-forward 3D Reconstruction and Adaptation Feed-forward 3D reconstruction has emerged as an alternative to scene-specific optimization. Early methods directly regress point maps from image pairs (Wang et al., 2024; Leroy et al., 2024), while recent foundation models extend this formulation to arbitrary numbers of views (Wang et al., 2025), permutation-equivariant reconstruction (Wang et al., 2026b), and unified monocular and multi-view reconstruction (Lin et al., 2025). Unlike optimization-based representations such as radiance fields (Mildenhall et al., 2021) and Gaussian splatting (Kerbl et al., 2023), these models recover geometry in a single forward pass. Recent work further suggests that feed-forward models can produce model-dependent fallback geometry when visual evidence is insufficient (Paul et al., 2026).

Adapting large pretrained models while preserving their general capabilities has motivated parameter-efficient methods such as LoRA (Hu et al., 2021). Subsequent approaches increase effective update rank through Khatri–Rao products (Albert et al., 2025b) or random bases (Albert et al., 2025a), while LieRA (Si et al., 2025) extends parameter-efficient adaptation to higher-order tensors through Lie-group parameterization. These methods provide general-purpose mechanisms for efficient adaptation. We instead study how parameter-efficient adaptation can be conditioned on the geometric failure structure specific to transparent objects.

3 MOTIVATION

We analyze the failure of feed-forward 3D foundation models on transparent objects along two complementary dimensions. We first examine how the error is distributed and then study how the remaining errors differ in direction and geometric structure. These observations reveal a distinction between errors that can be corrected using an observable geometric reference and those for which such a reference is unavailable.

3.1 CONCENTRATION OF LARGE ERRORS

Depth errors on transparent objects are highly concentrated among a small fraction of high-error regions. To characterize this concentration, we report RMSE over the top-k object pixels ranked by error in the Figure 2 (a). A small fraction of high-error pixels accounts for a large portion of the total error. This concentration is substantially stronger for transparent objects than for opaque objects in the same scenes. We refer to the majority of predictions as the *body* of the distribution and the sparse high-error region as the *tail*. A natural approach is to adapt the frozen backbone using LoRA (Hu et al., 2021). LoRA substantially reduces errors across the body, but its effectiveness progressively decreases toward the tail, as shown in the Figure 2 (b). The remaining challenge is therefore not simply to increase adaptation capacity, but to selectively address the high-error predictions.

3.2 ASYMMETRIC FAILURE

The tail does not represent a single failure mode. We define the signed depth error at pixel p as

$$e(p) = \hat{D}(p) - D(p),$$

where $\hat{D}$ and D denote predicted and ground-truth depth. Positive errors correspond to *overestimation*, where the prediction lies behind the true surface, while negative errors correspond to *underestimation*, where the prediction lies in front of it. Overestimation is primarily associated with

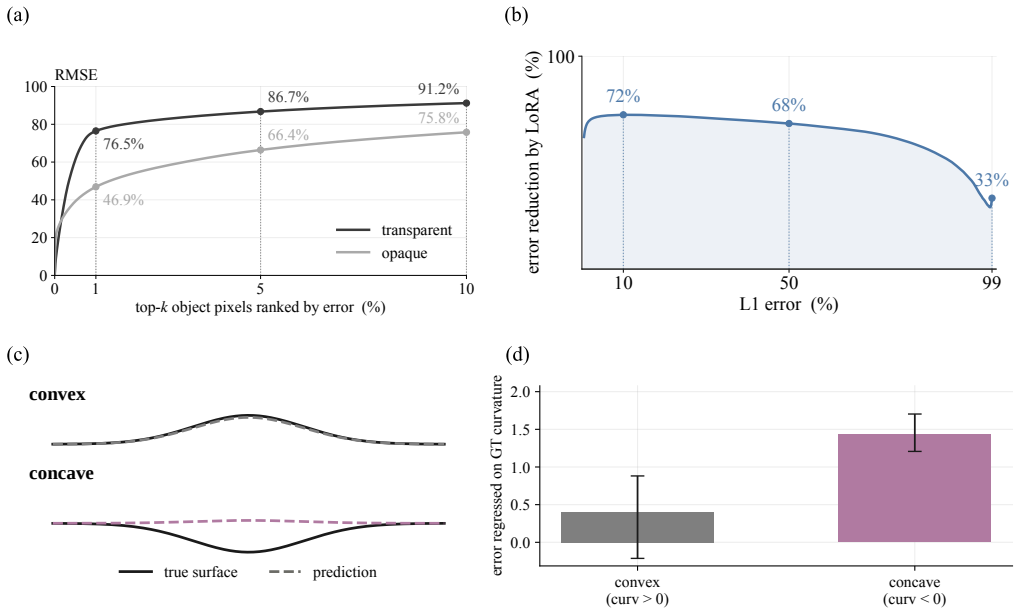


Figure 2: **Transparent-object errors concentrate in the tail and exhibit asymmetric geometric structure.** (a) Cumulative contribution to depth error from the top-k object pixels ranked by error. The errors are substantially more concentrated in a small high-error tail than opaque-object errors. (b) Error reduction from LoRA. The improvement progressively decreases toward the high-error tail. (c) Illustration of the geometric asymmetry, where predictions tend to preserve convex geometry but flatten concave structures. (d) Regression of depth error on GT curvature. The substantially larger coefficient for concave regions confirms a stronger tendency toward flattening.

background leakage, where the prediction follows a surface visible through the transparent object. Underestimation instead concentrates around concave and open structures.

We examine the geometric structure of underestimation through local curvature. Figure 2 (c) illustrates how the prediction can either follow the underlying geometry or flatten it. We quantify this behavior by regressing signed depth error against ground-truth curvature at the scale of object parts. Figure 2 (d) reveals a clear asymmetry. Predictions largely preserve convex geometry, while concave geometry shows a much stronger tendency toward flattening. This result indicates that the two error directions are associated with different geometric failure mechanisms rather than forming a homogeneous high-error population.

More importantly, the two directions differ in the geometric information available for correction. Background leakage can be related to the observable background surface behind the transparent object. This relation provides a reference for identifying where leakage occurs and how the prediction deviates from the object surface. A filled-in concavity provides no analogous reference. Once the concavity is flattened, the prediction forms a plausible surface without an observable counterpart that reveals the missing depth.

These observations suggest that the two error modes call for different correction strategies, one that exploits an observable geometric reference where available and one that infers the error direction from local geometry where no such reference exists.

4 METHOD

Our method builds on a feed-forward 3D foundation model that is first adapted using LoRA (Hu et al., 2021) on transparent-object data, while keeping the backbone frozen. Starting from this adapted model, we introduce two sequential stages motivated by the failure structure identified in

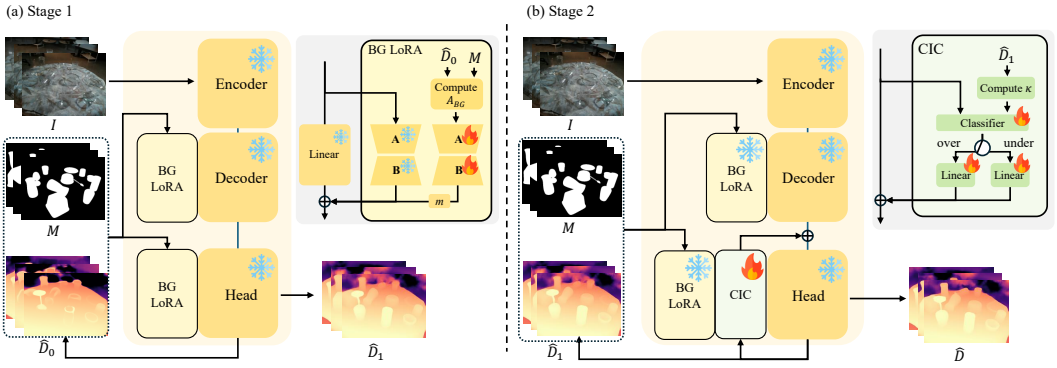


Figure 3: **Overview of our two-stage failure-aware adaptation framework.** Starting from a frozen backbone with adapted LoRA parameters, (a) Stage 1 trains Background-aware LoRA to correct background leakage, and (b) Stage 2 freezes the preceding components and trains CIC to correct the remaining directional errors.

Section 3. In the first stage, we freeze the backbone f and the pretrained LoRA parameters and train only Background-aware LoRA. Using the initial prediction $\hat{D}_0$ and object mask M, Background-aware LoRA identifies regions associated with background leakage and produces a refined prediction $\hat{D}_1$. In the second stage, we further freeze BG LoRA and train only the Curvature-Informed Corrector. The corrector infers the direction of the errors from local geometric cues in $\hat{D}_1$ and applies direction-specific correction to produce the final depth map $\hat{D}$.

4.1 Stage 1: Background-aware LoRA

Background leakage is a major source of overestimation in the high-error tail and provides an observable geometric reference through the background visible behind the transparent object. Background-aware LoRA exploits this reference to identify leakage and allocate additional adaptation capacity to affected regions.

Background affinity We first measure how strongly each object pixel is associated with the visible background. For each transparent object, we construct two coarse geometric references, S_{obj} and S_{bg}, using local planar fitting. The object-side reference S_{obj} is fitted to $\hat{D}_0$ within the object region, while the background reference S_{bg} is fitted to the surrounding background. We then compare predicted depth $\hat{D}_0$ with the two references to measure its affinity to the background. We define background affinity at an object pixel p as

$$A_{\text{BG}}(p) = \left|\hat{D}_0(p) - S_{\text{obj}}(p)\right| - \left|\hat{D}_0(p) - S_{\text{bg}}(p)\right|. \tag{1}$$

A positive value indicates that the predicted depth is closer to the background reference than to the object-side reference. We apply min-max normalization within each object instance to get A_{BG}.

Background-aware adaptation We augment the LoRA with an additional low-rank update dedicated to regions associated with background leakage. To identify these regions, we construct a soft mask m_{BG} with the highest background affinity A_{BG}. The additional details are provided in the Appendix B.

For a linear layer with frozen weight W and input x, the adapted output is

$$y = Wx + \Delta_{\text{LoRA}}(x) + m_{\text{BG}}\,\Delta_{\text{BG}}(x), \tag{2}$$

where Δ_{LoRA} denotes the LoRA update and Δ_{BG} denotes the additional background-aware update.app:bglora

Training objective We first convert the prediction and the target to disparity and align them by a global scale and shift estimated with least-squares fitting. We then compute the per-pixel absolute error $\ell(p)$ in disparity and discard a fraction τ_{trim} of pixels with the largest errors. To emphasize errors associated with background leakage, we optimize the following weighted term.

$$\mathcal{L}_w = \frac{\sum_p w_{\text{BG}}(p)\,\ell(p)}{\sum_p w_{\text{BG}}(p)}, \qquad w_{\text{BG}}(p) = 1 + m_{\text{BG}}(p). \tag{3}$$

We additionally apply a multi-scale gradient matching loss $\mathcal{L}_{\text{grad}}$ to preserve local geometric structure. The complete objective is

$$\mathcal{L}_{\text{BG}} = \mathcal{L}_w + \alpha\mathcal{L}_{\text{grad}}. \tag{4}$$

4.2 Stage 2: Curvature-Informed Corrector

Background-aware LoRA substantially reduces large errors associated with background leakage, yet smaller residual errors persist in both directions. Underestimation is frequently observed around concave structures without an observable geometric reference, while overestimation can persist where background affinity does not provide sufficient evidence for correction. To address these residual errors, we exploit local geometric cues to infer each direction. Specifically, the Curvature-Informed Corrector combines multi-scale curvature with complementary geometric features to distinguish overestimation from underestimation and routes each direction through a dedicated correction path.

Direction assignment Let $\mathcal{O}$ the set of object pixels. We estimate the direction from local geometric cues available directly from the prediction, without requiring ground-truth geometry at inference time. We characterize local geometry at scale s as

$$\kappa_s(p) = B_s[\hat{D}](p) - \hat{D}(p), \tag{5}$$

where $B_s[\hat{D}]$ represents the local average depth around each pixel at scale s. The signed deviation from the local average provides a geometric cue for distinguishing locally convex and concave structures. To infer the error direction, we use a classifier that combines multi-scale curvature with complementary local geometric features. Detailed feature definitions are provided in the appendix. The classifier outputs a direction score $\sigma(p)$, which assigns each object pixel to one of two error directions

$$m_{\text{over}}(p) = \mathbf{1}\left[\sigma(p) \geq \tau_{\text{sign}}\right], \qquad m_{\text{under}}(p) = \mathbf{1}\left[\sigma(p) < \tau_{\text{sign}}\right]. \tag{6}$$

The two assignments are mutually exclusive. During training, the classifier is supervised by the sign of the ground-truth depth error, whereas at inference time it relies solely on geometric features derived from the model prediction.

Direction-specific correction We introduce two lightweight correction paths, C_{over} and C_{under}, for the two error directions. The predicted direction assignments modulate the corresponding correction paths, allowing overestimation and underestimation to be corrected separately. Implementation details for mapping the pixel-level assignments to the intermediate feature representation are provided in the appendix.

Training objective We therefore optimize a L1 objective over object pixels

$$\mathcal{L}_{\text{CIC}} = \frac{1}{|\mathcal{O}|}\sum_{p\in\mathcal{O}}\left|\hat{D}(p) - D(p)\right|, \tag{7}$$

where $\hat{D}$ denotes the corrected prediction.

5 EXPERIMENTS

5.1 SETUP

Datasets. We use ClearPose (Chen et al., 2022b) for training and in-domain evaluation, with a scene-level split of 30 training, 5 validation, and 15 test scenes. ClearGrasp (Sajjan et al., 2020) and TRansPose (Kim et al., 2024) are used for cross-dataset evaluation. All models are trained on ClearPose only and evaluated on them without retraining.

Baselines. We compare with parameter-efficient adaptation methods and with methods designed for transparent objects. The PEFT baselines are LoRA (Hu et al., 2021), DoRA (Liu et al., 2024), AdaLoRA (Zhang et al., 2023), LieRA (Si et al., 2025), and KRAdapter (Albert et al., 2025b), all on the same frozen backbone. For transparent-object methods, we evaluate SeeClear (Wang et al., 2026a) with our implementation on the VGGT and reproduce MODEST (Liu et al., 2025). We additionally quote the results of Depth4ToM (Costanzino et al., 2023), MODEST, D4RD (Tosi et al., 2024), and DKT (Xu et al., 2025) as reported by SeeClear. DKT is fine-tuned on ClearGrasp, whereas our method is trained only on ClearPose and transferred without retraining.

Evaluation metrics. We evaluate predicted depth $\hat{D}$ within the ground-truth transparent-object mask using RMSE and MAE in millimeters, AbsRel as a fraction, and threshold accuracy δ at 1.01, 1.03, 1.05, 1.10, and 1.25, measuring the percentage of predictions within each threshold.

Table 1: **Quantitative results on the ClearPose dataset.** Add. Params denotes the number of trainable parameters. All methods are evaluated using the same alignment protocol unless otherwise noted. † indicates evaluation without alignment.

Method	Add. Params	RMSE ↓	AbsRel ↓	MAE ↓	$\delta_{1.01}$ ↑	$\delta_{1.03}$ ↑	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑
VGGT	-	155.4	0.1053	86.7	7.21	22.41	38.97	70.18	93.25
PEFT									
LoRA r16	6.1M	75.8	0.0452	38.4	26.02	62.13	77.58	89.84	97.72
DoRA	6.4M	75.8	0.0470	40.0	24.73	60.08	76.08	89.14	97.50
AdaLoRA	6.1M	84.3	0.0459	39.2	21.47	55.40	75.24	91.58	98.28
LieRA r16	6.1M	96.8	0.0614	51.2	14.35	41.38	62.58	85.68	97.42
KRAdapter	11.1M	87.4	0.0459	39.1	20.93	55.84	75.89	91.64	98.27
Ours (VGGT)	5.9M	**40.6**	**0.0289**	**24.7**	**36.42**	**73.28**	**85.37**	**94.35**	**98.92**
Transparent-aware									
SeeClear	-	133.5	0.0808	69.7	9.24	28.75	48.32	78.58	96.19
MODEST †	-	101.0	0.0744	70.0	14.50	37.03	51.56	73.81	93.01

5.2 QUANTITATIVE RESULTS

As shown in Table 1, our method improves over all PEFT approaches on every metric on ClearPose. PEFT methods substantially improve the frozen backbone, but their gains remain limited for transparent objects, suggesting that adaptation capacity alone does not address the structured errors in this domain. Our method also improves over the transparent-object baselines on the same backbone, indicating that explicitly accounting for the failure structure adds a benefit beyond adaptation. MODEST predicts metric depth, so reproducing its reported ClearPose results requires evaluation without alignment. The further details are provided in the Appendix E.

Table 2 evaluates cross-dataset generalization on ClearGrasp without additional training. The improvement transfers, and our method performs favorably against both the methods evaluated on the same backbone and existing transparent-object approaches, showing that the correction is not specific to the training distribution of ClearPose.

5.3 QUALITATIVE RESULTS

Figure 4 compares the absolute depth errors on challenging ClearPose scenes. The frozen VGGT exhibits large errors over substantial portions of the object interiors, indicating that the failure is not confined to object boundaries. LoRA reduces these errors considerably but leaves pronounced

Table 2: **Quantitative results on the ClearGrasp dataset.** All methods are evaluated on transparent-object regions. Italicized results are reported from SeeClear (Wang et al., 2026a), and † indicates methods fine-tuned on ClearGrasp. Dash (—) indicate metrics not available in the results reported by SeeClear.

Method	RMSE ↓	AbsRel ↓	MAE ↓	$\delta_{1.01}$ ↑	$\delta_{1.03}$ ↑	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑
VGGT	46.4	0.0747	39.8	7.95	25.50	42.65	72.73	98.66
VGGT + Ours	**18.8**	**0.0286**	**14.9**	25.49	65.94	**83.73**	**96.36**	99.89
VGGT + SeeClear	20.2	0.0306	16.5	**26.23**	**66.47**	82.62	94.52	**99.91**
DA3	23.9	0.0381	20.0	19.65	51.34	71.42	94.42	**99.91**
DA3 + SeeClear	21.8	0.0338	18.1	22.68	55.23	76.95	96.13	99.90
Depth4ToM	*24.8*	*0.0400*	*20.9*	—	—	*70.48*	*92.56*	—
MODEST	*39.3*	*0.0620*	*33.1*	—	—	*54.01*	*79.60*	—
D4RD	*40.0*	*0.0690*	*36.2*	—	—	*42.70*	*78.25*	—
DKT †	*22.4*	*0.0350*	*18.6*	—	—	*81.29*	*94.04*	—

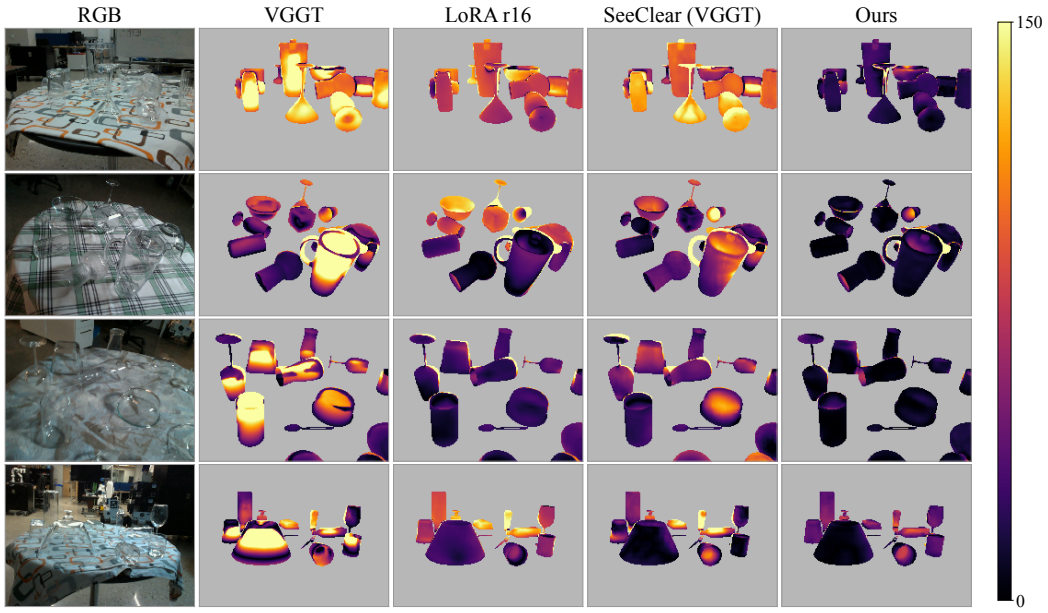


Figure 4: **Qualitative comparison on the ClearPose dataset.** Absolute depth errors are visualized within the ground-truth transparent-object masks, with gray indicating regions outside the masks. SeeClear (VGGT) denotes our implementation on the VGGT. Our method consistently reduces large errors across objects with diverse shapes, scales, and background configurations.

residuals in geometrically challenging regions, while SeeClear similarly improves some regions but retains large localized errors, particularly in scenes with cluttered backgrounds and large objects. In contrast, our method consistently suppresses large interior errors across occlusion, background, object-scale, and planar configurations, demonstrating the effectiveness of explicitly targeting structured error patterns.

5.4 Transfer across backbones and datasets

Table 3 evaluates the proposed method across three frozen backbones and three datasets. On ClearPose, our method improves all three backbones across every reported metric despite substantial differences in architecture and zero-shot performance. These improvements largely transfer to ClearGrasp and TRansPose without additional training. Most backbone and dataset combinations improve across the reported metrics, while DA3 on ClearGrasp is the only setting that does not improve on any of the reported metrics.

Table 3: **Transfer across backbones and datasets.** Our method is trained on ClearPose for each of three frozen backbones and evaluated on ClearPose, TRansPose, and ClearGrasp. TRansPose and ClearGrasp are evaluated without additional training. Bold indicates the better result between zero-shot and our method for each backbone and metric. The alignment and evaluation protocols are described in Sec. 5.1, with full results provided in Appendix F.1.

		ClearPose			TRansPose			ClearGrasp		
Backbone	Method	RMSE↓	AbsRel↓	$\delta_{1.01}$ ↑	RMSE↓	AbsRel↓	$\delta_{1.01}$ ↑	RMSE↓	AbsRel↓	$\delta_{1.01}$ ↑
VGGT	zero-shot	155.4	0.1053	7.21	76.5	0.1206	9.77	46.4	0.0747	7.95
	Ours	**40.6**	**0.0289**	**36.42**	**50.3**	**0.0790**	**12.45**	**18.8**	**0.0286**	**25.49**
π^3	zero-shot	151.6	0.1022	6.99	64.7	0.0940	**11.15**	39.7	0.0617	10.50
	Ours	**44.9**	**0.0307**	**33.47**	**57.7**	**0.0924**	10.61	**17.1**	**0.0253**	**32.28**
DA3	zero-shot	167.5	0.0938	9.27	87.2	0.1290	8.35	**23.9**	**0.0381**	**19.65**
	Ours	**39.9**	**0.0290**	**34.80**	**55.4**	**0.0840**	**10.27**	25.5	0.0394	15.29

Table 4: **Ablation study on the ClearPose dataset.** We evaluate the individual and joint contributions of BG-LoRA and CIC using the VGGT. LoRA r12 provides a parameter-matched comparison for evaluating the effect of BG-LoRA beyond additional trainable parameters. ✓indicate the modules included in each configuration.

Method	BG-LoRA	CIC	RMSE ↓	AbsRel ↓	MAE ↓	$\delta_{1.01}$ ↑	$\delta_{1.03}$ ↑	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑
VGGT			155.4	0.1053	86.7	7.21	22.41	38.97	70.18	93.25
LoRA r8			76.0	0.0446	38.0	25.89	61.72	77.61	90.13	97.81
LoRA r12			77.9	0.0510	43.3	23.06	56.99	73.56	87.70	96.79
LoRA r8	✓		60.1	0.0368	31.5	30.86	68.13	81.62	91.85	98.29
LoRA r12		✓	46.6	0.0322	27.2	35.02	71.03	83.22	93.15	98.73
Ours	✓	✓	**40.6**	**0.0289**	**24.7**	**36.42**	**73.28**	**85.37**	**94.35**	**98.92**

5.5 Ablation

Table 4 evaluates the contributions of BG-LoRA and CIC on ClearPose using the VGGT. BG-LoRA introduces an additional rank-4 update on top of LoRA r8, resulting in a trainable parameter count comparable to that of LoRA r12. We therefore use LoRA r12 as a parameter-matched baseline to distinguish the effect of background-aware adaptation from that of simply increasing the LoRA rank. Increasing the standard LoRA rank from r8 to r12 does not improve performance, whereas adding BG-LoRA to LoRA r8 substantially improves all reported metrics. This comparison indicates that the improvement from BG-LoRA is attributable to selectively applying the additional update to regions associated with background leakage rather than merely increasing the number of trainable parameters. CIC also substantially improves the LoRA r12 baseline across all reported metrics, demonstrating the benefit of direction-specific correction beyond LoRA adaptation. The additional gain supports the proposed sequential design, in which BG-LoRA first targets background leakage and CIC subsequently corrects errors in both directions.

6 Conclusion

We investigated depth estimation failures on transparent objects in feed-forward 3D foundation models and identified two asymmetric error modes, with overestimation primarily associated with background leakage and underestimation concentrated around concave structures. Based on this analysis, we introduced a failure-aware parameter-efficient adaptation framework consisting of Background-aware LoRA and Curvature-Informed Corrector, which sequentially address background leakage and directional depth errors. The proposed framework achieves strong performance on transparent-object depth estimation benchmarks and consistently improves reconstruction across different feed-forward 3D foundation models. Our analysis also suggests that accurately estimating the magnitude of underestimation remains challenging due to the limited geometric information available for correction, providing an important direction for future work (Appendix G).

AI USE STATEMENT

In this work, we used generative AI tools to implement methods, assist with translation, support qualitative and thematic data analysis, and interpret results. We did not use generative AI tools to generate synthetic data sets, develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, or design or provide feedback on research methodology or experiments. Cleaning and reformatting datasets is not applicable to this work. We have reviewed all AI-assisted work. LLM-assisted code for implementing our methods was reviewed and tested for correctness by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All experiments use publicly available datasets and backbones. Section 5.1 states the data splits, baselines and metrics, and Appendix D.2 gives the alignment protocol, scene lists, frame counts, windowing and mask handling for ClearPose, ClearGrasp and TRansPose. The two modules are specified in full: Appendix B covers the background-affinity surfaces, the gate with its fixed anchors and the attachment of the low-rank paths, and Appendix C the geometric features, the sign classifier with its fitted coefficients and the correction paths. Appendix D.1 lists every training hyperparameter by stage, the loss definitions and the hardware used. Appendix E documents the protocol behind each number quoted from prior work and our reproduction of it, and Appendix F reports all metrics and per-scene results. Source code, trained adapter weights and the sign-classifier coefficients will be released upon acceptance.

REFERENCES

Aviral Agrawal, Ritaban Roy, Bardienus Pieter Duisterhof, Keerthan Bhat Hekkadka, Hongyi Chen, and Jeffrey Ichnowski. Clear-splatting: Learning residual gaussian splats for transparent object manipulation. In *RoboNerF: 1st Workshop On Neural Fields In Robotics at ICRA 2024*, 2024.

Paul Albert, Frederic Zhang, Hemanth Saratchandran, Cristian Rodriguez-Opazo, Anton Hengel, and Ehsan Abbasnejad. Randlora: Full rank parameter-efficient fine-tuning of large models. In *International Conference on Learning Representations*, volume 2025, pp. 22258–22270, 2025a.

Paul Albert, Frederic Z Zhang, Hemanth Saratchandran, Anton van den Hengel, and Ehsan Abbasnejad. Towards higher effective rank in parameter-efficient fine-tuning using khatri-rao product. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1292–1302. IEEE, 2025b.

Kang Chen, Shaochen Wang, Beihao Xia, Dongxu Li, Zhen Kan, and Bin Li. Tode-trans: Transparent object depth estimation with transformer. *arXiv preprint arXiv:2209.08455*, 2022a.

Xiaotong Chen, Huijie Zhang, Zeren Yu, Anthony Opipari, and Odest Chadwicke Jenkins. Clearpose: Large-scale transparent object dataset and benchmark. In *European conference on computer vision*, pp. 381–396. Springer, 2022b.

Alex Costanzino, Pierluigi Zama Ramirez, Matteo Poggi, Fabio Tosi, Stefano Mattoccia, and Luigi Di Stefano. Learning depth estimation for transparent and mirror surfaces. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9210–9221. IEEE, 2023.

Qiyu Dai, Jiyao Zhang, Qiwei Li, Tianhao Wu, Hao Dong, Ziyuan Liu, Ping Tan, and He Wang. Domain randomization-enhanced depth simulation and restoration for perceiving and grasping specular and transparent objects. In *European Conference on Computer Vision*, pp. 374–391. Springer, 2022.

Qiyu Dai, Yan Zhu, Yiran Geng, Ciyu Ruan, Jiazhao Zhang, and He Wang. Graspnerf: Multiview-based 6-dof grasp detection for transparent and specular objects using generalizable nerf. In *2023 IEEE international conference on robotics and automation (ICRA)*, pp. 1757–1763. IEEE, 2023.

Hongjie Fang, Hao-Shu Fang, Sheng Xu, and Cewu Lu. Transcg: A large-scale real-world dataset for transparent object depth completion and a grasping baseline. *IEEE Robotics and Automation Letters*, 7(3):7383–7390, 2022.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Jiaqi Jiang, Guanqun Cao, Thanh-Toan Do, and Shan Luo. A4t: Hierarchical affordance detection for transparent objects depth reconstruction and manipulation. *IEEE Robotics and Automation Letters*, 7(4):9826–9833, 2022.

Jiaqi Jiang, Guanqun Cao, Jiankang Deng, Thanh-Toan Do, and Shan Luo. Robotic perception of transparent objects: A review. *IEEE Transactions on Artificial Intelligence*, 5(6):2547–2567, 2023.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

Jeongyun Kim, Myung-Hwan Jeon, Sangwoo Jung, Wooseong Yang, Minwoo Jung, Jaeho Shin, and Ayoung Kim. Transpose: Large-scale multispectral dataset for transparent object. *The International Journal of Robotics Research*, 43(6):731–738, 2024.

Jeongyun Kim, Seunghoon Jeong, Giseop Kim, Myung-Hwan Jeon, Eunji Jun, and Ayoung Kim. 2d gaussian splatting-based sparse-view transparent object depth reconstruction via physics simulation for scene update. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 27927–27936. IEEE, 2025.

Michael S Langer and Heinrich H Bülthoff. A prior for global convexity in local shape-from-shading. *Perception*, 30(4):403–410, 2001.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European conference on computer vision*, pp. 71–91. Springer, 2024.

Zongcheng Li, Xiaoxiao Long, Yusen Wang, Tuo Cao, Wenping Wang, Fei Luo, and Chunxia Xiao. Neto: Neural reconstruction of transparent objects with self-occlusion aware refraction-tracing. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 18501–18511. IEEE, 2023.

Haotong Lin, Sili Chen, Junhao Liew, Donny Y Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025.

Baoxia Liu and James T Todd. Perceptual biases in the interpretation of 3d shape from shading. *Vision research*, 44(18):2135–2145, 2004.

Jiangyuan Liu, Hongxuan Ma, Yuxin Guo, Yuhao Zhao, Chi Zhang, Wei Sui, and Wei Zou. Monocular depth estimation and segmentation for transparent object with iterative semantic and geometric fusion. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11162–11168. IEEE, 2025.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Soumava Paul, Prakhar Kaushik, and Alan Yuille. Can these views be one scene? evaluating multiview 3d consistency when 3d foundation models hallucinate. *arXiv preprint arXiv:2605.18754*, 2026.

Shreeyak Sajjan, Matthew Moore, Mike Pan, Ganesh Nagaraja, Johnny Lee, Andy Zeng, and Shuran Song. Clear grasp: 3d shape estimation of transparent objects for manipulation. In *2020 IEEE international conference on robotics and automation (ICRA)*, pp. 3634–3642. IEEE, 2020.

Chongjie Si, Zhiyi Shi, Xuehui Wang, Yichen Xiao, Xiaokang Yang, and Wei Shen. Generalized tensor-based parameter-efficient fine-tuning via lie group transformations. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 197–207. IEEE, 2025.

Fabio Tosi, Pierluigi Zama Ramirez, and Matteo Poggi. Diffusion models for monocular depth estimation: Overcoming challenging conditions. In *European Conference on Computer Vision*, pp. 236–257. Springer, 2024.

Anuj Verma, Laksh Nanwani, Sourav Garg, K Madhava Krishna, et al. Silica: Repurposing diffusion priors for joint glass segmentation and depth estimation. *arXiv preprint arXiv:2607.24249*, 2026.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306. IEEE, 2025.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20697–20709. IEEE, 2024.

Xiaoying Wang, Yumeng He, Jingkai Shi, Jiayin Lu, Yin Yang, Ying Jiang, and Chenfanfu Jiang. Seeclear: Reliable transparent object depth estimation via generative opacification. In *European Conference on Computer Vision*, pp. 75–93. Springer, 2026a.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *International Conference on Learning Representations*, volume 2026, pp. 10481–10497, 2026b.

Shaocong Xu, Songlin Wei, Qizhe Wei, Zheng Geng, Hong Li, Licheng Shen, Qianpu Sun, Shu Han, Bin Ma, Bohan Li, et al. Diffusion knows transparency: Repurposing video diffusion for transparent object depth and normal estimation. *arXiv preprint arXiv:2512.23705*, 2025.

Xiaohao Xu, Feng Xue, Xiang Li, Haowei Li, Shusheng Yang, Tianyi Zhang, Matthew Johnson-Roberson, and Xiaonan Huang. One scene, two depths: Probing geometric ambiguity in monocular foundation models. *arXiv preprint arXiv:2606.29600*, 2026.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.

Langxu Zhao, Zuan Gu, and Tianhan Gao. Ghost: Geometry-guided hallucination of opaque surface textures. *arXiv preprint arXiv:2607.11118*, 2026.

Luyang Zhu, Arsalan Mousavian, Yu Xiang, Hammad Mazhar, Jozef van Eenbergen, Shoubhik Debnath, and Dieter Fox. Rgb-d local implicit function for depth completion of transparent objects. *arXiv preprint arXiv:2104.00622*, 2021.

A NOTATION

Table 5: **Symbols.** Numeric values are given with each module in Appendices B, C and D.

Symbol	Description
Notations	
I	Input images
M	Object masks
D	Ground-truth depth
$\hat{D}_0$	Depth predicted by the frozen backbone with LoRA adaptation
$\hat{D}_1$	Depth predicted after Background-aware LoRA
$\hat{D}$	Corrected depth, after the Curvature-Informed Corrector
$e(p)$	Signed error $\hat{D}(p) - D(p)$; positive is overestimation
Regions	
Ω	Mask of one transparent-object instance
Γ	Background band surrounding Ω (Eq. 8)
$\Omega_\emptyset$	Pixels belonging to no object instance
$\mathcal{O}$	Set of object pixels in a frame
Background affinity	
$S_{\mathrm{obj}}, S_{\mathrm{bg}}$	Object-side and background reference surfaces
$A_{\mathrm{BG}}(p)$	Background affinity (Eq. 1). Positive means the prediction is closer to S_{bg}
Error direction	
$\kappa_s(p)$	Local curvature at scale s (Eq. 5)
$\sigma(p)$	Direction score produced by the logistic classifier
τ_{sign}	Threshold on σ separating the two directions
Hyperparameters	
d_c	Bottleneck width of a correction path
τ_{trim}	Fraction of largest residuals discarded by the Background-aware LoRA objective
α	Weight of the gradient matching term (Eq. 4)

B BACKGROUND-AWARE LORA

B.1 BACKGROUND AFFINITY

Reference regions. For each transparent-object instance with mask Ω, the object-side reference is fitted over Ω itself and the background reference over a band

$$\Gamma = \Big((\Omega \oplus B_{r_{\mathrm{out}}}) \setminus (\Omega \oplus B_{r_{\mathrm{in}}}) \Big) \cap \Omega_\emptyset, \tag{8}$$

where $\oplus B_r$ denotes dilation by an elliptical structuring element of radius r and $\Omega_\emptyset$ the pixels carrying no instance label. We use $r_{\mathrm{out}} = 32$ px and $r_{\mathrm{in}} = 2$ px. The outer radius keeps the band close enough to the object that a planar model is adequate. The inner radius removes the strip along the object boundary, where predicted depth interpolates between the object and the surface behind it. The intersection with $\Omega_\emptyset$ prevents a neighboring instance from entering the band.

S_{obj} and S_{bg} Both references are affine in the pixel coordinates,

$$S(u, v) = c_0 + c_1 \tilde{u} + c_2 \tilde{v}, \qquad \tilde{u} = \frac{u - u_c}{s_a}, \qquad \tilde{v} = \frac{v - v_c}{s_a}, \tag{9}$$

where (u_c, v_c) is the center of the bounding box of Ω and s_a its longer side. Normalizing by the bounding box keeps the design matrix well conditioned regardless of where the object appears and how large it is, and both surfaces use the same frame, so the two fits are directly comparable.

Fitting. The coefficients are obtained by ridge-regularized least squares against the zero-shot prediction,

$$c_{\mathrm{obj}} = \arg\min_{c} \sum_{p\in\Omega} \big(\hat{D}_0(p) - S_c(p)\big)^2 + \lambda\|c\|^2, \qquad c_{\mathrm{bg}} = \arg\min_{c} \sum_{p\in\Gamma} \big(\hat{D}_0(p) - S_c(p)\big)^2 + \lambda\|c\|^2, \tag{10}$$

with $\lambda = 10^{-6}$, solved through the normal equations and falling back to a least-squares solve if the system is singular. When a region contributes more than 12,000 valid pixels, that many are drawn at random without replacement with a fixed seed, which bounds the cost on large objects. Both surfaces are then evaluated over the whole image and read at the object pixels, giving A_{BG} through the equation. 1.

Undefined affinity. A fit is attempted only when the band carries at least 80 valid pixels and the object at least 120. Instances that fail either test, or whose fit does not return finite coefficients, are skipped. The affinity is left undefined over that instance and no background-aware update is applied there. In practice this affects objects cut off by the image border and objects almost entirely occluded.

B.2 m_{BG}

Ramp. The affinity is mapped to a gate in $[0, 1]$ by a clipped linear ramp,

$$m_{\mathrm{BG}}(p) = \mathrm{clip}\left(\frac{A_{\mathrm{BG}}(p) - q_{20}}{q_{80} - q_{20}},\ 0,\ 1\right), \tag{11}$$

where q_{20} and q_{80} are percentiles of A_{BG}. m_{BG} saturates at the clearly non-leaking and clearly leaking ends of the distribution and varies linearly between them. Pixels with undefined affinity are assigned a value of 0, restricting the gated path to objects for which both surfaces can be fitted.

Anchors. The percentiles are taken over all object pixels of the ClearPose training split that carry a defined affinity, pooled across frames rather than computed per frame (Table 6). They differ between backbones because each backbone's prediction produces its own affinity distribution. By construction, on the training split 20% of the pixels in that population receive 0, 20% of 1, and the remaining 60% a value in between.

Table 6: Ramp anchors, computed once on the ClearPose training split.

Backbone	q_{20}	q_{80}
VGGT	-0.6085	-0.0030
π^3	-1.0292	-0.0041
DA3	-0.8945	-0.0157

No recomputation at evaluation. The anchors are reused unchanged for the ClearPose validation and test splits, and for ClearGrasp and TRansPose as well since neither dataset is trained on. They are never recomputed on an evaluation split, so no statistic of the evaluation data enters the gate. The saturated fractions on an evaluation split therefore need not be 20% and 20%.

Resampling. The m_{BG} is defined per pixel and is brought to the resolution of each adapted layer. For layers operating on tokens it is bilinearly pooled to the patch grid, and the prefix tokens, which correspond to no image region, receive 0. For the convolutional layers of the depth head it is bilinearly resampled to the resolution of the corresponding feature map.

B.3 Low-rank paths and attachment

Paths. Each adapted layer receives two low-rank updates, one applied everywhere and one modulated by the m_{BG}. Each applies α/r scaling and dropout to its input, with $(r, \alpha) = (8, 16)$ for the former and $(4, 8)$ for the latter, and dropout 0.05 on aggregator layers and 0 on head layers. The down-projections are Kaiming-uniform initialized and the up-projections zero-initialized, so the adapter is the identity at the start of training. The two paths are trained in sequence rather than jointly (Appendix D.1).

Attachment points. On VGGT the paths attach to 96 attention projections (the fused query, key, value and the output projection of each of the 24 frame blocks and 24 global blocks) and to 29 convolutions of the DPT head, 125 layers in total. On π^3 they attach to 72 attention projections in the decoder blocks, 10 in the first five point-decoder blocks, and the point-head projection, 83 in total. On DA3, they attach to 80 attention projections across the 40 backbone blocks and 29 convolutions of the main DualDPT branch, 109 in total. Both paths of a layer share the same target modules.

Parameter counts. On VGGT the first path trains 3,072,256 parameters and the gated path 1,536,128, 4,608,384 in total, or 0.49% of the frozen backbone. The corresponding totals are 3,042,192 on π^3 and 5,542,272 on DA3. Neither the affinity nor the gate contributes parameters.

C Curvature-Informed Corrector

C.1 Curvature and geometric features

Scale normalization. All features are computed on the prediction $\hat{D}_1$ of the preceding stage, divided by its median over the valid object pixels. This removes the one degree of freedom (the overall scale) that a relative prediction leaves undetermined, without using ground truth. A shift is not removed, since neither the curvature nor the gradient features depend on it.

Curvature. Local curvature at scale s is the deviation of a pixel from its neighborhood, Eq. 5, evaluated at $s \in \{3, 5, 15, 31, 45, 61\}$ px. With w the instance mask, the box mean is restricted to pixels of the same instance:

$$B_s[p] = \frac{\mathrm{box}_s(p \cdot w)}{\mathrm{box}_s(w)}, \tag{12}$$

so that the window never averages object depth against the background behind it. Computing the same feature over an unrestricted window collapses its discriminative power, from a mean absolute standardized difference of 0.292 between the two error directions to 0.06, because near an object boundary the background depth enters the window and dominates the curvature.

Normalized curvature. Each curvature is also divided by a robust scale estimated within the instance,

$$\tilde{\kappa}_s = \frac{\kappa_s}{\max((q_{75} - q_{25})/1.349,\ 10^{-6})}, \tag{13}$$

where the quartiles are taken over the object pixels of that instance. The unnormalized feature carries the absolute magnitude of the curvature and the normalized one carries how curved a pixel is relative to the rest of its own object.

Feature set. The classifier reads 19 features per pixel: seven basic features, six curvatures and six normalized curvatures (Table 7). All are computed from the prediction and the instance mask alone.

Table 7: Features of the sign classifier. None reads ground-truth depth, so the same features are available at training and at inference.

Feature	Definition	Count
`pred_depth`	Normalized depth p	1
`depth_grad`	Sobel gradient magnitude of p	1
`depth_lapl_abs`	Absolute Laplacian of p, kernel size 3	1
`local_depth_std`	Standard deviation of p over a 9×9 window	1
`dist_to_obj_edge_raw`	Euclidean distance transform of the instance mask	1
`dist_to_obj_edge_norm`	The above, divided by its per-instance maximum	1
`obj_area`	Number of pixels in the instance	1
`curv_`s	Masked curvature, $s \in \{3, 5, 15, 31, 45, 61\}$	6
`curvz_`s	The above, divided by the per-instance robust scale	6
Total		19

C.2 SIGN CLASSIFIER

Fitting. The classifier is a logistic regression, fitted by iteratively reweighted least squares with an L2 penalty of 1.0. Features are standardized with the mean and standard deviation of the training split, and non-finite entries are imputed with the training median of that feature. It is supervised by the sign of the ground-truth error, which is available at training time. The fit uses the training split only, and the validation and test splits are loaded for evaluation alone.

Threshold. A pixel is assigned to the overestimation path when its score reaches $\tau_{\text{sign}} = 0.45$ and to the underestimation path otherwise, so the two assignments are exclusive and every object pixel is corrected by exactly one path. The threshold is applied per pixel and only then pooled to tokens, which keeps the two directions from mixing inside one token. A sweep over τ_{sign} is reported in Appendix F.4.

What the classifier uses. Table 8 gives the fitted coefficients. The large-scale curvatures dominate: the three coarsest scales, 31, 45 and 61 px, carry the largest weights, while the two finest contribute almost nothing. Distinguishing the two error directions is therefore a property of the shape of the object as a whole rather than of its fine detail, consistent with the convex–concave asymmetry of Sec. 3. Their signs alternate across scales, so the classifier is effectively reading differences between adjacent scales rather than any single one. Finally, two features are inactive (the absolute Laplacian and the local standard deviation, with coefficients below 0.02) so the effective feature set is smaller than 19.

Table 8: Fitted coefficients on standardized features.

Feature	Coef.	Feature	Coef.
curv_61	−0.339	obj_area	+0.114
curvz_61	−0.317	curvz_31	+0.096
pred_depth	+0.315	curvz_15	+0.071
curv_45	+0.297	curv_5	+0.058
curv_31	−0.288	curvz_3	−0.043
dist_to_obj_edge_norm	+0.166	depth_grad	+0.034
dist_to_obj_edge_raw	−0.153	curv_3	−0.025
curvz_45	+0.145	curvz_5	−0.019
curv_15	−0.117	local_depth_std	−0.016
		depth_lapl_abs	+0.001
		intercept	+0.642

Accuracy. On the ClearPose test split the classifier reaches an AUC of 0.581 over all object pixels and 0.826 over the 10% of object pixels with the largest error. The sign of a small error is close to unpredictable from local geometry, and the classifier does not predict it; what it recovers is the direction of the large errors, which are the ones the corrector is meant to remove. We report both numbers because the first alone would understate what the gate contributes and the second alone would overstate how well the direction of an arbitrary pixel is known.

C.3 CORRECTION PATHS AND ATTACHMENT

Block. The two paths C_{over} and C_{under} are independent copies of the same block, with no weight sharing. Each block is

$$\begin{aligned} &\text{LayerNorm}(d) \\ &\to \text{Linear}(d \to d_c) \\ &\to \text{4-head local attention } (7 \times 7 \text{ window}) \\ &\to \text{Linear}(d_c \to d_c) \\ &\to \text{Linear}(d_c \to d), \end{aligned}$$

with $d_c = 140$ (136 for DA3) and the output projection zero-initialized, so that the corrector is inactive at the start of training. Neighbors outside the patch grid are masked out of the attention,

and the block returns only the residual added to the head tokens (Sec. 4.2). The sweep over d_c is reported in Appendix F.4.

Attachment points. The correction is applied to the tokens that enter the depth head, through a forward pre-hook that replaces the patch-token slice of the aggregated tokens at the layers the head consumes and leaves the prefix tokens untouched. Table 9 lists the taps for each backbone.

Table 9: Attachment of the correction paths. On VGGT the taps are layers 4, 11, 17 and 23.

Backbone	Hook point	Taps	d	Patch grid
VGGT	`depth_head` pre-hook	4	2048	28×37
π^3	`point_decoder` pre-hook	1	2048	27×37
DA3	DualDPT hook	4	3072	28×37

Tap scales. The same two blocks are applied at every tap; only the scales $\lambda_{\text{over}}, \lambda_{\text{under}}$ are tap-specific, one entry per tap, initialized to 0.1. The four values learned on VGGT are $\lambda_{\text{over}} = [0.119, 0.127, 0.090, 0.143]$ and $\lambda_{\text{under}} = [0.142, 0.079, 0.081, 0.109]$.

Parameter counts. The two paths and their scales train 1,317,378 parameters on VGGT, 1,317,372 on π^3 and 1,838,938 on DA3. The sign classifier is fitted separately and is not counted here. It contributes 20 coefficients, which are fixed during adaptation.

C.4 Oracle sign classifier

The italic row of Table 15 replaces the classifier score with the sign of the ground-truth error, leaving every other component unchanged. It is not a deployable configuration: the sign of the error is exactly what the classifier exists to estimate, and it is unavailable at inference. We report it as an upper bound on what the routing mechanism can achieve given a perfect direction signal, which separates the quality of the signal from the quality of the correction. The ground-truth sign is stored alongside the classifier scores in the same cache and is read only when this configuration is requested; every other row of every table is computed from the classifier score.

D Training and Data

D.1 Training procedure

Pipeline. Adaptation proceeds in two stages, with two precomputation steps between them; Stage 1 is trained in two steps. The frozen backbone is first run over every split to produce $\hat{D}_0$, from which the background affinity, the gate and the loss weight are computed and cached; the ramp anchors are taken from the training split alone (Appendix B.2). Stage 1 then trains the path applied everywhere, followed by the gated path, which reads the cached gate. The resulting model is run over every split to produce $\hat{D}_1$, from which the geometric features are computed and the sign classifier is fitted on the training split (Appendix C.2); its scores are cached. Stage 2 trains the two correction paths against those scores. Each step freezes everything trained before it, and the backbone stays frozen throughout: of its 941.8 M parameters, 5.9 M are trained.

Optimization. All stages use AdamW with $\beta = (0.9, 0.999)$ and no weight decay, a linear warmup over 8% of steps followed by cosine decay to zero, gradient clipping at 1.0, bf16 autocast, gradient checkpointing and a single seed (42) throughout. No data augmentation is used. Table 10 gives the schedule. The loss terms it names are defined in Appendix D.2.

The smaller Adam ϵ in the CIC stage is needed because the adapter gradients are small enough that the default value dominates $\sqrt{v}$.

Checkpoint selection. Checkpoints are selected on the validation split, by affine object AbsRel for the gated path and affine object RMSE for CIC.

Table 10: Training schedule on VGGT. Where the two BG-LoRA steps differ, values are given as base path / gated path. Both steps optimize the same objective; the gated path additionally weights each retained pixel by the leakage mask m_{BG}. π^3 and DA3 follow the same recipe except that π^3 runs in fp32, both train CIC for 8 epochs rather than 5, and DA3 trains the base path for 10 epochs.

	Stage 1: BG-LoRA (base / gated)	Stage 2: CIC
Learning rate (aggregator / head)	1e-4 / 2e-4	1e-3
Epochs	5	5
Effective batch (windows)	4 / 3	3
Steps per epoch	82 / 109	109
Adam ϵ	1e-8	1e-12
Loss space	disparity	depth
Trim ratio τ_{trim}	0.2	0
Gradient matching weight α	0.5	0

Hardware and inference. Training takes about five hours in total on three NVIDIA TITAN RTX GPUs. Inference requires two backbone passes, as the BG-LoRA is defined on the frozen prediction.

D.2 Loss functions

The two module families are trained in different spaces: BG-LoRA in disparity and CIC in depth. Both objectives share an alignment, and the terms that Eq. 3 and Eq. 4 name are defined below. Throughout this section $\hat{D}$ and D denote the prediction and the ground truth in the stage's loss space, disparity for BG-LoRA, depth for CIC.

Alignment. For each frame, a scale and a shift are obtained in closed form by least squares,

$$(a, b) = \arg\min_{a,b} \sum_{p \in V} \big(a\,\hat{D}(p) + b - D(p)\big)^2, \tag{14}$$

where V is the set of all valid pixels of the frame, defined by $D(p) \geq 10^{-3}$. The fit uses the whole frame so that the objective is optimized under the same alignment the evaluation applies.

Data term. With the aligned prediction $\hat{D}_a = a\,\hat{D} + b$, the per-pixel error of Eq. 3 is $\ell(p) = |\hat{D}_a(p) - D(p)|$. Trimming is applied per frame: the pixels whose error ranks in the top $\tau_{\mathrm{trim}} = 0.2$ fraction are discarded, and are removed from the denominator as well as the numerator. Both BG-LoRA steps optimize the identical objective; the base path is trained with $w_{\mathrm{BG}} \equiv 1$, as are the PEFT baselines of Table 1, and only the gated path uses the weight of Eq. 3.

Gradient matching. $\mathcal{L}_{\mathrm{grad}}$ penalizes the spatial derivative of the residual $r = \hat{D}_a - D$ at four scales. At scale k the residual and its mask m are subsampled with stride 2^k, giving strides 1, 2, 4 and 8, and the term is the mean absolute first forward difference in both image directions, taken over the pairs of neighboring pixels that are both valid,

$$\mathcal{L}_{\mathrm{grad}} = \frac{1}{4} \sum_{k=0}^{3} \frac{\sum \big(|\Delta_x r_k| + |\Delta_y r_k|\big)}{\sum m_k}, \tag{15}$$

weighted by $\alpha = 0.5$.

CIC. The corrector uses the same alignment and sets $\tau_{\mathrm{trim}} = 0$ and $\alpha = 0$, leaving the unweighted L1 of Eq. 7 over $\mathcal{O}$, the intersection of the object mask and the valid ground-truth pixels. Training it with the BG-LoRA objective instead gives 52.3 mm against 40.6 (Table 15).

E REIMPLEMENTATION OF SEECLEAR AND MODEST

E.1 SEECLEAR PROTOCOL

SeeClear evaluates ClearGrasp with a per-frame affine fit over all valid pixels in the model's native parameterization, scored inside the transparent-object mask, which is our default protocol. Table 11 gives its two DA3 rows against our reproduction.

Table 11: SeeClear's DA3 rows (Tab. 1, ClearGrasp real, ToM) and our reproduction, on the five metrics both papers report. SeeClear reports $\delta_{1.025}$ in place of our $\delta_{1.01}$ and $\delta_{1.03}$, and does not report $\delta_{1.25}$.

	RMSE	AbsRel	MAE	$\delta_{1.05}$	$\delta_{1.10}$
DA3, published	24.03	0.0380	19.95	75.82	92.24
DA3, ours	23.87	0.0381	20.04	71.42	94.42
SeeClear + DA3, published	21.45	0.0330	17.94	78.13	95.63
SeeClear + DA3, ours	21.85	0.0338	18.13	76.95	96.13

E.2 MODEST PROTOCOL

MODEST predicts metric depth, so its row in Table 1 is scored without alignment. Its ground truth, mask and valid pixels are those of every other row. MODEST's own evaluation also restricts the ground truth to $(0.3, 3.0]$ m; on our test split no object pixel lies outside that range, so the restriction changes nothing.

The publicly released ClearPose checkpoint gives RMSE 101.0 mm, MAE 70.0 mm and AbsRel 0.0744 on our test split. MODEST reports 123 mm, 81 mm and 0.087 on a split that the release does not specify, so the two evaluations are not on the same frames; the release does not specify its training split either.

On ClearGrasp we could not reproduce the published number. The released ClearPose weights give a ToM RMSE of 173 mm unaligned and 119 mm aligned, and the released Syn-TODD weights 410 mm and 61 mm, against the 39.31 mm that SeeClear quotes. The same evaluation path reproduces SeeClear's DA3 rows, so Table 2 quotes MODEST's ClearGrasp row from SeeClear.

F ADDITIONAL RESULTS

F.1 FULL CROSS-MODEL AND CROSS-DATASET TABLES

Table 3 reports RMSE, AbsRel and $\delta_{1.01}$ for three backbones on three datasets. Tables 12–14 give all eight metrics under the protocol of Sec. 5.1; ClearGrasp and TRansPose are evaluated without retraining. Bold marks the better of zero-shot and Ours per backbone and metric.

Table 12: ClearPose test split, all metrics.

Backbone	Method	RMSE ↓	AbsRel ↓	MAE ↓	$\delta_{1.01}$ ↑	$\delta_{1.03}$ ↑	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑
VGGT	zero-shot	155.4	0.1053	86.7	7.21	22.41	38.97	70.18	93.25
	Ours	**40.6**	**0.0289**	**24.7**	**36.42**	**73.28**	**85.37**	**94.35**	**98.92**
π^3	zero-shot	151.6	0.1022	85.5	6.99	22.12	39.54	69.60	92.70
	Ours	**44.9**	**0.0307**	**26.0**	**33.47**	**71.46**	**84.82**	**94.52**	**98.97**
DA3	zero-shot	167.5	0.0938	80.7	9.27	27.97	44.69	73.69	94.78
	Ours	**39.9**	**0.0290**	**24.7**	**34.80**	**72.64**	**85.03**	**94.51**	**99.02**

F.2 FULL ABLATION TABLES

Table 15 gives all eight metrics for every configuration of Table 4. Raising the LoRA rank from r8 to r12 does not improve the frozen backbone, whereas BG-LoRA at the same budget reduces

Table 13: TRansPose, all metrics.

Backbone	Method	RMSE ↓	AbsRel ↓	MAE ↓	$\delta_{1.01}$ ↑	$\delta_{1.03}$ ↑	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑
VGGT	zero-shot	76.5	0.1206	56.3	9.77	26.73	40.50	63.05	87.20
	Ours	**50.3**	**0.0790**	**38.6**	**12.45**	**33.32**	**48.61**	**71.44**	**92.02**
π^3	zero-shot	64.7	0.0940	45.2	**11.15**	**30.78**	**45.97**	**69.28**	**91.56**
	Ours	**57.7**	**0.0924**	**44.9**	10.61	29.08	43.81	67.07	89.99
DA3	zero-shot	87.2	0.1290	61.1	8.35	23.33	35.79	57.48	82.79
	Ours	**55.4**	**0.0840**	**42.8**	**10.27**	**28.87**	**44.08**	**68.19**	**92.43**

Table 14: ClearGrasp real test, all metrics.

Backbone	Method	RMSE ↓	AbsRel ↓	MAE ↓	$\delta_{1.01}$ ↑	$\delta_{1.03}$ ↑	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑
VGGT	zero-shot	46.4	0.0747	39.8	7.95	25.50	42.65	72.73	98.66
	Ours	**18.8**	**0.0286**	**14.9**	**25.49**	**65.94**	**83.73**	**96.36**	**99.89**
π^3	zero-shot	39.7	0.0617	33.1	10.50	33.03	54.13	83.18	98.55
	Ours	**17.1**	**0.0253**	**13.4**	**32.28**	**75.42**	**88.90**	**97.36**	**99.00**
DA3	zero-shot	**23.9**	**0.0381**	**20.0**	**19.65**	**51.34**	**71.42**	**94.42**	**99.91**
	Ours	25.5	0.0394	21.8	15.29	46.19	70.97	93.65	99.65

RMSE from 77.9 to 60.1 mm; without the leakage selection it returns to 76.5 mm, the level of plain LoRA. CIC improves either base, from 77.9 to 46.6 mm on LoRA r12 and from 60.1 to 40.6 mm on BG-LoRA, and at an equal budget the full chain remains 6.0 mm below LoRA r12 + CIC. Within CIC, training with the LoRA objective gives 52.3 mm, the outlier-targeted objective 41.5 mm and the sign split 40.6 mm; the oracle sign classifier reaches 34.4 mm and bounds what a better direction estimate could recover.

F.3 Error direction across the adaptation stages

Table 15 reports how much error each component removes. This section reports which direction it removes.

Figure 5 plots the signed error over ClearPose test pixels inside the object mask, weighting each pixel by its squared error so that the area of a row is proportional to that configuration's mean squared error. Table 16 gives the same quantity as numbers: the directional RMSE over pixels with $e > 0$ and $e < 0$, the fraction of pixels in each direction, and the share of the summed squared error that each direction contributes.

Overestimation accounts for 98.1% of the zero-shot squared error. BG-LoRA reduces its RMSE by 57.5% and CIC by a further 34.5%; underestimation falls by 28.0% and then 7.4%. After both modules the two directions contribute comparably, 58.2% against 41.8%, and even the oracle sign classifier leaves underestimation at 40.2 mm against 69.6 mm zero-shot.

F.4 Sweeps

Configuration. The d_c and τ_{sign} sweeps below were run in an earlier configuration of CIC that scaled the correction by a per-patch instability estimate and used asymmetric thresholds, $\theta_+ = 0.7$ for the overestimation path and $\theta_- = 0.45$ for the underestimation path. The deployed model uses neither, so its object RMSE of 40.6 mm is not directly comparable with the rows of Tables 17 and 18; the sweeps are reported for the trends they show.

Bottleneck width. Table 17 varies d_c and the number of paths. The deployed width of 140 was chosen so that two paths match the parameter count of a single 256-wide path. Doubling the width of the two-path model, from 140 to 256, improves RMSE by 0.1 mm at twice the parameters.

Sign threshold. Table 18 varies the two thresholds at inference on a model trained once with $\theta_+ = 0.7$, $\theta_- = 0.45$. The deployed threshold, $\tau_{\text{sign}} = 0.45$ for both paths, was not chosen from

Table 15: **Ablation.** Indented rows are variants of the configuration above them, with the named element removed. LoRA r12 matches the parameter count of BG-LoRA, and LoRA r12 + CIC that of Ours. *Italic*: non-deployable oracle, excluded from the bold marking.

Method	RMSE ↓	AbsRel ↓	MAE ↓	$\delta_{1.01}$ ↑	$\delta_{1.03}$ ↑	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑
VGGT	155.4	0.1053	86.7	7.21	22.41	38.97	70.18	93.25
LoRA r8	76.0	0.0446	38.0	25.89	61.72	77.61	90.13	97.81
LoRA r12	77.9	0.0510	43.3	23.06	56.99	73.56	87.70	96.79
LoRA r12 + CIC	46.6	0.0322	27.2	35.02	71.03	83.22	93.15	98.73
LoRA r8 + BG-LoRA	60.1	0.0368	31.5	30.86	68.13	81.62	91.85	98.29
w/o selecting a fixed fraction	76.5	0.0438	37.3	26.65	62.69	78.23	90.45	97.91
Ours	**40.6**	**0.0289**	**24.7**	**36.42**	**73.28**	**85.37**	**94.35**	**98.92**
w/o sign split	41.5	0.0301	25.7	33.77	71.82	84.85	94.14	98.81
w/o sign split & outlier loss	52.3	0.0358	30.6	30.20	67.79	81.76	92.08	98.46
w/ oracle sign gate	*34.4*	*0.0223*	*19.1*	*49.59*	*81.73*	*89.88*	*95.99*	*99.15*

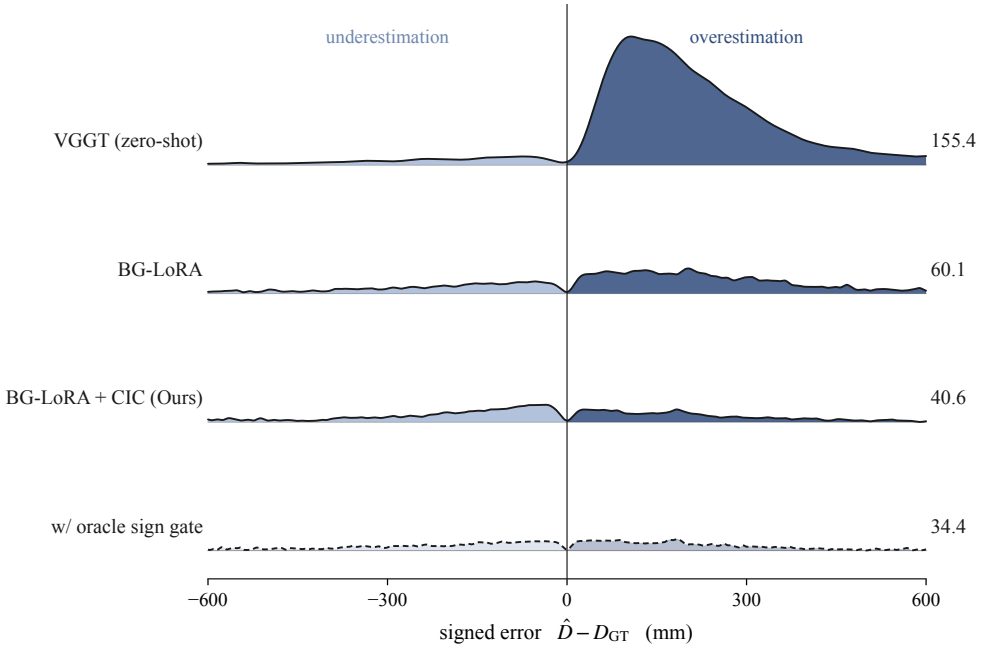


Figure 5: **Where each module acts.** Signed depth error on ClearPose test pixels inside the object mask, weighted by squared error so that a row's area is proportional to its mean squared error. Rows share a vertical scale; errors beyond ± 400 mm are clipped. Right: object RMSE (mm); the oracle row is not deployable.

this sweep. It was set so that both paths receive enough training samples: on the training split, the pixels above 0.7 are 26.3% of the object with a precision of 74.6%, and the pixels below 0.45 are 7.7% with a precision of 51.3%; tightening the lower threshold to 0.3 leaves 1.1% of pixels for the underestimation path, too few to train it.

Trim ratio. Table 19 retrains the BG-LoRA stage with $\tau_{\mathrm{trim}} = 0.3$, all other settings unchanged, and evaluates its output before CIC. To separate the tail from the body, the object pixels of each frame are split once by the zero-shot error: the 10% with the largest error form the tail, the rest the body, and the same split is used for both rows. Raising the trim ratio worsens both, and the tail by far more: the residuals that are cut from the loss are the ones the tail consists of. In an earlier, smaller configuration (a single rank-4 path in the depth head, one stage), lowering the ratio from 0.2 to 0.1, 0.05 and 0 also worsened RMSE monotonically, from 100.3 to 107.2 mm.

Table 16: Error by direction on ClearPose test pixels inside the object mask. RMSE in mm; share is the fraction of the summed squared error contributed by that direction. Bold: lowest RMSE among deployable rows.

Configuration	RMSE over	RMSE under	frac. over	frac. under	share over
VGGT	209.8	69.6	0.848	0.152	98.1%
BG-LoRA	89.2	50.1	0.609	0.391	83.1%
BG-LoRA + CIC (Ours)	**58.4**	**46.4**	0.467	0.533	58.2%
w/ oracle sign gate	*48.0*	*40.2*	*0.529*	*0.471*	*61.6%*

Table 17: CIC bottleneck width, ClearPose test. Bold: best.

Paths	d_c	Parameters	RMSE
single	256	1,318,149	41.5
two	140	1,317,378	41.3
two	256	2,636,298	**41.2**
single	448	2,646,213	41.7

Rank. LoRA rank was not swept. r8 is the base adaptation, r12 the single-path control of Table 4, chosen to match the parameter count of BG-LoRA, and r16 the baseline of Table 1, chosen to match the parameter count of Ours.

F.5 Comparison with GHOST

GHOST adapts VGGT to transparent objects on ClearGrasp and evaluates under a protocol of its own: images are resized to 256×256, the affine alignment is fitted in depth on the ground-truth object mask rather than on the full frame, and metrics are computed inside that mask. Fitting the alignment on the object makes every number lower than under our protocol, so the rows of Table 20 are not comparable with the other tables. We re-score our predictions under GHOST's protocol and place them against the two rows GHOST reports.

Our zero-shot row reproduces GHOST's on AbsRel and the five thresholds to within 2.9%; our RMSE is 14.3% lower. Over the same baseline, our adapter, trained on ClearPose only, reduces RMSE by 23.9%, and GHOST, fine-tuned on ClearGrasp, by 4.8%.

F.6 Additional qualitative results

Figures 6–8 follow the format of Fig. 4: absolute error inside the object mask after the per-frame alignment of the tables, on a fixed scale of 0–150 mm. Figure 8 shows the two ClearPose test scenes with the highest object error for Ours, at the frame where that error is largest.

G Recovering the Magnitude of Underestimation

For overestimation, the background surface behind the object indicates both where the prediction leaked and how far it should move. This appendix asks whether underestimation has an analogous source of magnitude. Throughout, $\hat{D}$ is the BG-LoRA prediction with all parameters frozen, errors are signed after the per-frame affine alignment of Sec. 5.1, and the split is ours. Table 23 summarizes the candidates; the subsections below give the details.

G.1 Headroom

On the test split, underestimated pixels make up 39.1% of the object but only 19.8% of its summed squared error. Correcting all of them exactly would reduce the pixel-pooled object RMSE of BG-LoRA by 10.5%, against 55.5% for correcting all overestimated pixels. The difference comes not from a thinner tail (the 99th percentile of underestimation is 79% of that of overestimation) but from errors that are smaller per pixel throughout.

Table 18: Sign thresholds, ClearPose test. Bold: best; the oracle row is excluded.

θ_+	θ_-	RMSE
0.0	0.45	42.4
0.6	0.45	**41.2**
0.7	0.45	41.3
0.8	0.45	42.2
0.9	0.45	43.8
0.7	0.35	41.6
0.7	0.30	41.8
0.7	0.20	42.2
oracle	*oracle*	*39.4*

Table 19: Trim ratio of the BG-LoRA stage, ClearPose test, output before CIC. Tail and body: RMSE over the 10% of object pixels with the largest zero-shot error and over the remaining 90%. Bold: best.

τ_{trim}	RMSE	AbsRel	MAE	$\delta_{1.01}$	$\delta_{1.03}$	$\delta_{1.05}$	$\delta_{1.10}$	$\delta_{1.25}$	tail	body
0.2	**60.1**	**0.0368**	**31.5**	**30.86**	**68.13**	**81.62**	**91.85**	**98.29**	**111.1**	**48.6**
0.3	77.8	0.0446	38.2	25.73	61.68	77.63	90.19	97.84	169.9	53.0

G.2 Magnitude by direction

Table 24 reports how much of the residual $r = D - \hat{D}$ each quantity built from the reference surfaces of Appendix B.1 explains, separately for overestimated and underestimated pixels.

The best GT-free quantity, the signed distance from the prediction to a plane fitted to the prediction itself, explains 46.1% of the variance on overestimated pixels but 8.0% on underestimated ones. It captures displacement away from the object plane, which describes overestimation but not a concavity that the plane already spans. The background surface and the affinity explain nothing on underestimated pixels. Even the ground-truth object surface, whose gap approximates $-r$ by construction, falls from 0.929 to 0.771 on underestimated pixels: these errors lie in local structure, such as the interior of an opening, that a single surface per object does not describe.

G.3 A rear surface inside the object

For an open container, the far wall seen through the opening could serve as the reference that the background provides for leakage. On 739 instances from three test scenes (every third frame, instances of at least 500 pixels), it fails on three counts. The rear tail in the ground-truth depth is more common where overestimation dominates (45.6%) than where underestimation does (36.9%). Its distance to the prediction does not match the required correction (median 46 mm against 12.4 mm, Spearman 0.03). And underestimated pixels lie at the middle of their instance's depth range (median relative depth 0.50), not near its rear. Among five candidate reference depths on the same scenes, only the median ground-truth depth of the instance matches the required scale (median ratio 0.91, Spearman 0.50); its counterpart computed from the prediction has a Spearman correlation of 0.03.

G.4 The backbone representation

The predicted depth is itself the product of the fill-in, but the features entering the depth head might still encode the missing depth. A ridge probe from these features to $\max(D - \hat{D}, 0)$, fitted on the training split, reaches an R^2 between -0.16 and -0.13 on the underestimated pixels of the validation split. A second probe tested whether filled-in depth depends less on the input image, using the gradient of the prediction with respect to the image. It explains 2.2% of the variance of the required correction and separates the two error signs at chance level (AUC 0.498).

Table 20: ClearGrasp real test under the GHOST protocol. *Italic*: quoted from GHOST (Tab. 2). Δ: RMSE reduction of each adapted row over its own zero-shot row. Bold: best value, quoted rows included. †: fine-tuned on this dataset.

Method	RMSE ↓	Δ	AbsRel ↓	MAE ↓	$\delta_{1.01}$ ↑	$\delta_{1.03}$ ↑	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑
VGGT	18.0		0.0272	14.2	33.56	68.06	83.32	96.06	99.86
VGGT + Ours	**13.7**	**−23.9%**	**0.0199**	**10.3**	42.69	**78.87**	**90.87**	**98.45**	**99.91**
VGGT	*21.0*		*0.028*	*15.0*	*33.4*	*70.1*	*84.6*	*95.5*	*99.6*
VGGT + GHOST †	*20.0*	*−4.8%*	*0.022*	*12.0*	***42.7***	*78.7*	*90.4*	*97.7*	*99.6*

Table 21: Object RMSE (mm) per ClearPose test scene, VGGT. Δ: Ours against zero-shot. Bold: best per scene.

Scene	Frames	Zero-shot	LoRA r16	BG-LoRA	Ours	Δ
set1/scene2	132	117.6	48.3	37.0	**24.6**	−79.0%
set1/scene4	102	180.2	87.2	55.8	**32.4**	−82.0%
set2/scene5	78	173.8	90.0	69.3	**50.6**	−70.9%
set3/scene4	72	145.9	82.6	70.1	**59.2**	−59.4%
set4/scene2	72	254.5	80.4	60.1	**33.0**	−87.0%
set4/scene3	72	192.5	49.5	45.9	**27.5**	−85.7%
set5/scene2	90	178.4	114.1	90.8	**51.9**	−70.9%
set5/scene4	96	117.5	52.7	47.8	**26.5**	−77.4%
set6/scene4	90	75.9	50.6	42.4	**34.0**	−55.2%
set7/scene2	120	189.4	114.4	86.4	**52.1**	−72.5%
set7/scene6	84	127.2	55.2	45.5	**34.4**	−72.9%
set8/scene4	18	**88.6**	113.7	114.5	128.0	+44.5%
set8/scene5	18	210.8	97.2	91.8	**88.3**	−58.1%
set9/scene11	12	95.8	55.6	47.7	**45.1**	−53.0%
set9/scene12	12	140.2	54.2	41.9	**33.1**	−76.4%

G.5 THE CONVEX HULL OF THE OBJECT

The free space between an object and its convex hull locates concavities directly and predicts the sign in the tail: among the 10% of test pixels with the largest error after BG-LoRA and CIC, 91.8% of those more than 50 mm inside the hull are underestimated. The relation weakens over all object pixels (64–68%) and vanishes below about 20 mm. It does not provide the magnitude: a convolutional head trained to regress the gap from the image and the prediction reaches an R^2 of -0.06 on the validation split, below a constant predictor.

Every candidate either requires the ground truth or carries little information about the size of an underestimation error. CIC therefore uses only its sign.

Table 22: Object RMSE (mm) on the ClearPose official test split by condition, VGGT. Δ: Ours against zero-shot. Bold: best per condition.

Condition	Frames	Zero-shot	LoRA r8	BG-LoRA	Ours	Δ
Heavy occlusion	744	151.4	80.1	68.1	**52.9**	−65.1%
New background	414	126.6	63.0	53.0	**35.3**	−72.1%
Opaque distractor	54	135.6	62.1	54.4	**45.2**	−66.7%
Translucent cover	48	**109.0**	119.8	121.7	116.9	+7.2%
Non-planar	42	157.8	76.4	67.3	**62.6**	−60.3%
Filled liquid	42	153.7	67.2	49.6	**38.7**	−74.8%

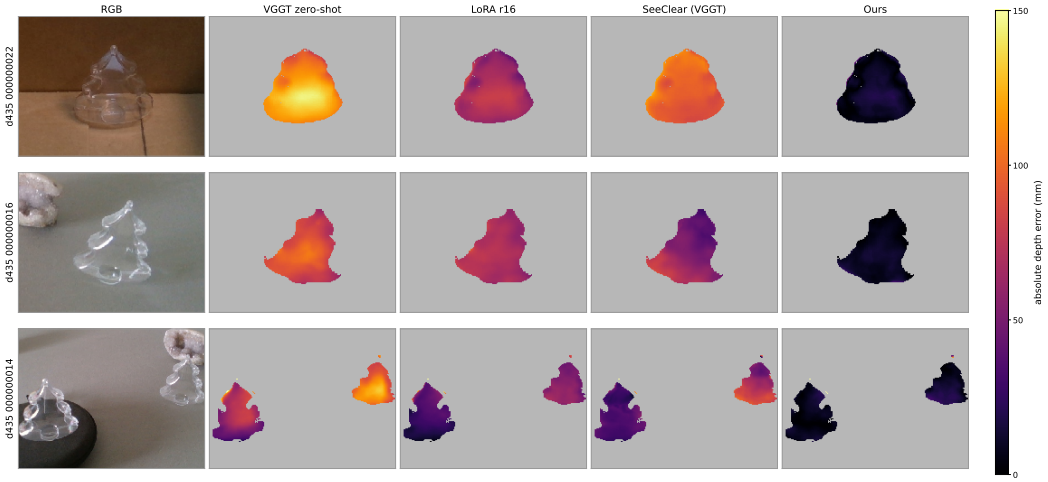


Figure 6: ClearGrasp real test.

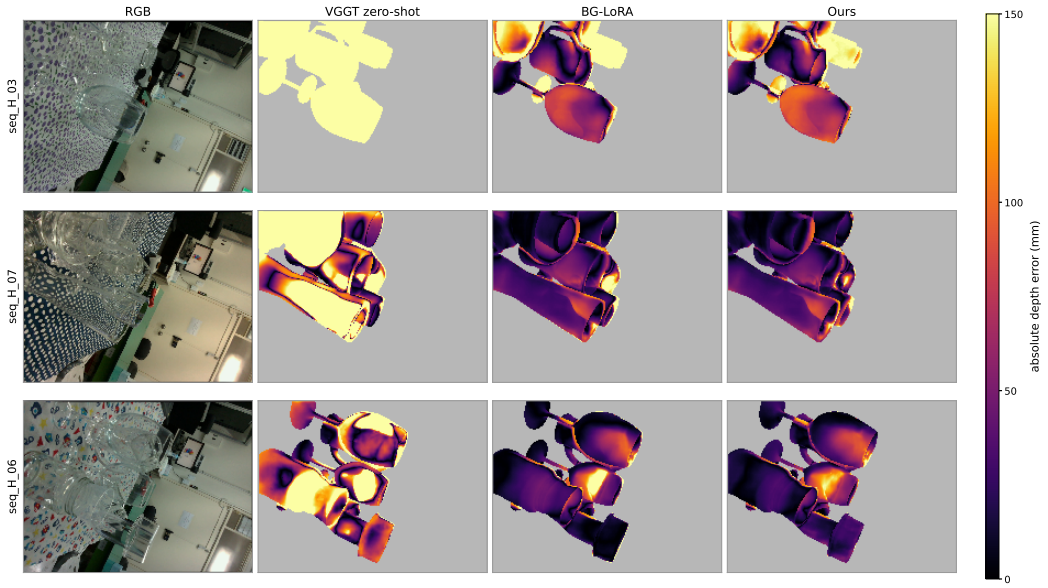


Figure 7: TRansPose. LoRA r16 and SeeClear were not run on this dataset.

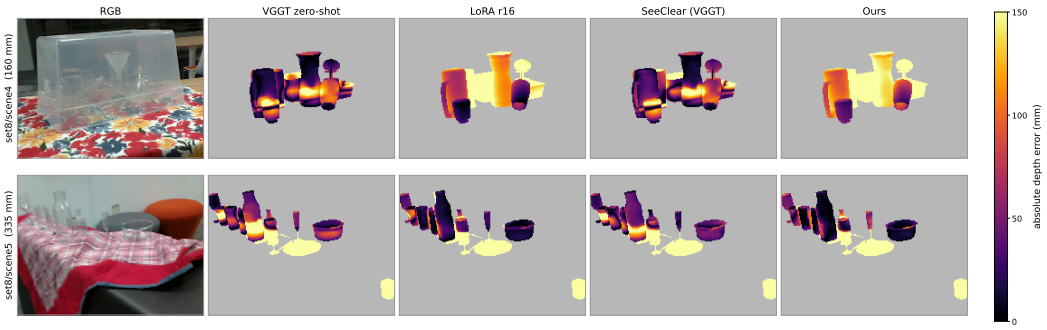


Figure 8: Failure cases on ClearPose test: set8/scene4 (top) and set8/scene5 (bottom).

Table 23: Candidate sources of magnitude for underestimation. R^2 is for the required correction on underestimated pixels unless stated otherwise. Only the ground-truth median depth of each instance matches the required scale (Appendix G.3).

Candidate	Idea	Result	
Object surface fitted to $\hat{D}$	displacement from the object plane	R^2 0.461 over, 0.080 under	G.2
Background surface, A_{BG}	the reference used for leakage	R^2 0.000 under	G.2
Rear surface inside the object	far wall seen through an opening	scale mismatch (Spearman 0.03); underestimated pixels at mid-depth	G.3
Features entering the depth head	missing depth still encoded before the head	R^2 -0.16 to -0.13	G.4
Input sensitivity	filled-in depth ignores the image	R^2 0.022; sign AUC 0.498	G.4
Convex hull	free space locates concavities	sign in the tail (91.8%); magnitude R^2 -0.06	G.5

Table 24: R^2 of a univariate linear fit to the residual after BG-LoRA. ClearPose test split, all object pixels (3,000 sampled per frame). The italic row uses the ground truth and serves as a reference. Bold: best GT-free value per column.

Quantity	GT-free	All	Over	Under
Signed gap to the object surface fitted to $\hat{D}$	✓	0.345	**0.461**	**0.080**
Signed gap to the background surface	✓	0.007	0.005	0.000
A_{BG} as deployed	✓	0.002	0.001	0.000
Signed gap to the object surface fitted to D		*0.910*	*0.929*	*0.771*