MoGe-3: Fine-Detail Monocular Geometry Estimation with Self-Guided Sparse Volumetric Refinement

Lingyu Kong[1,3*†] **Ruicheng Li**[1,3*†] **Ruicheng Wang**[2,3*†] **Sicheng Xu**[3]
Chengtang Yao[3] **Jianfeng Xiang**[1,3†] **Jiaolong Yang**[3‡]
[1]Tsinghua University [2]USTC [3]Microsoft Research

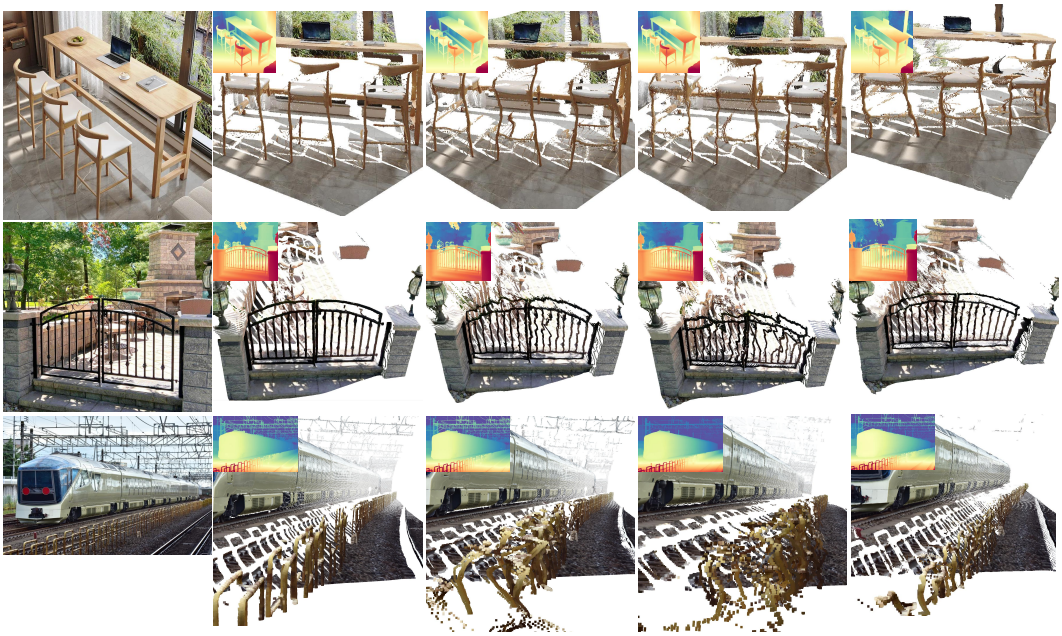

Figure 1: Our method produces geometrically undistorted and detail-preserving metric scale point maps from a single image. Compared to prior approaches [45, 22, 59], it better preserves fine structures and recovers more accurate local geometry, leading to improved 3D fidelity in challenging cases.

Abstract

Monocular geometry estimation has recently achieved impressive performance across diverse scenes. However, state-of-the-art models still face notable distortion in local 3D structure, especially in fine details, like thin structures and small objects. We attribute this limitation to an architectural mismatch: most current models decode 3D geometry within a 2D parameterization, where feature interactions are governed by image-plane proximity rather than true 3D spatial relationships. This inadvertently mixes features from geometrically distant surfaces, resulting in over-smoothed geometry particularly around thin or elongated structure. In this paper, we propose MoGe-3, a fine-detail monocular geometry estimation model with Self-Guided Sparse 3D Refinement (SSR) that lifts monocular geometry modeling from 2D image space to 3D space for high-fidelity metric-scale point

*These authors contributed equally to this work.
†Work done during internship at Microsoft Research
‡Corresponding author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

maps. MoGe-3 lifts the coarse point map from a foundation base model onto a sparse voxel shell and refines it via SSR. The SSR employs sparse convolutions that aggregate features based on 3D spatial locality, avoiding feature mixing across depth discontinuities. Extensive experiments on diverse datasets demonstrate that MoGe-3 significantly outperforms existing approaches in recovering fine detailed 3D geometry across both quantitative metrics and qualitative visualizations. Project page: `https://microsoft.github.io/MoGe-3`.

1 Introduction

Recovering the 3D geometry from a single RGB image has been a long-standing problem in computer vision, with broad applications in novel-view synthesis, AR/VR, and robotic manipulation. Building on advances in modern foundation backbones [26, 36], recent monocular geometry estimation methods [18, 30, 3, 58, 13, 27, 54, 55, 16, 4, 44, 45] have significantly improved zero-shot depth and point-map prediction. Despite this progress, we observe a systematic artifact: while current models produce plausible global layouts, they often struggle to faithfully reconstruct fine-scale geometry, particularly for thin or elongated structures, resulting in severe distortions such as twisted fences or smeared poles, as shown in Fig. 1. Notably, such artifacts are often inconspicuous in 2D depth visualizations but become evident under viewpoint changes, exposing underlying 3D geometric distortions.

We hypothesize that this issue is due to a fundamental architectural mismatch: *most current monocular reconstruction models decode 3D geometry within a 2D parameterization*. Feed-forward methods rely on 2D convolutional decoders [18, 30, 4, 44, 45, 13, 55, 59], diffusion-based approaches employ VAE decoding [16, 9], and recent methods adopt MLP heads over patch latents or 2D UV queries [52, 59]. In all cases, feature interactions are governed by image-plane proximity rather than true 3D geometric relationships. This introduces a local smoothness prior that inadvertently mixes features from geometrically distant surfaces across depth discontinuities and over-smooths geometry, particularly around thin or elongated structures. Consequently, local structural integrity breaks down when 2D proximity diverges from true 3D relationships.

Our solution to this problem is to *lift monocular geometry modeling from the 2D image plane to 3D space*. While retaining a strong 2D foundation model (e.g., DINOv2 [26]) for appearance understanding, our method performs geometry modeling under a 3D inductive bias, where feature interactions are defined by spatial neighborhoods in 3D point space. Through this design, image-adjacent pixels across depth discontinuities are decoupled, while image-disconnected pixels belonging to the same surface are associated via 3D proximity. This new paradigm addresses the fundamental causes of local structure distortions in current monocular geometry estimation models.

MoGe-3 consists of two main components: a foundational geometry estimator as the base model and a **S**elf-guided **S**parse 3D **R**efiner (SSR). Since a 3D point map occupies a thin manifold within a sparse volume, traditional implementation of a 3D-aware decoder at full image resolution is computationally inefficient. We therefore operate directly on the occupied scene manifold using 3D sparse convolutions. Specifically, we initialize the base geometry estimator from MoGe-2 [45] to generate an initial point map. We then perform iterative refinement using SSR in a self-guided manner, where the sparse 3D U-Net structure at each step is determined by the previous prediction. Before being processed by the U-Net, the point map is voxelized in a log-depth parameterized coordinate space for scale-consistent discretization. 2D features from the geometry estimator are also injected into the 3D U-Net bottlenecks. This sparse formulation operates only on occupied regions, making computation linear in point occupancy, and enables 3D-aware refinement without sacrificing the capacity of the 2D backbone.

We train our model on a large mixture of pixel-accurate synthetic data and diverse real-world datasets. Across standard zero-shot benchmarks, our default ViT-L model achieves state-of-the-art performance, consistently outperforming existing approaches in both local and global geometric accuracy. We further train a scaled ViT-G variant to show that SSR remains effective on top of a stronger foundation backbone. Beyond quantitative improvements, MoGe-3 produces clearly superior qualitative results, with more accurate fine-scale geometry, cleaner and more consistent geometric structures, and significantly reduced distortions in regions of complex geometry like thin and elongated structures,

where prior methods often fail. This bridges the gap between visually plausible depth and faithful 3D reconstruction.

2 Related Works

Monocular depth & geometry estimation. Monocular geometry estimation aims to recover depth maps or point clouds from a single image. Early works such as MiDaS [18] and DPT [30] established large-scale training and transformer-based decoding for relative depth estimation. Subsequent approaches extended this paradigm toward metric and generalizable geometry estimation, including ZoeDepth [3], Metric3D / Metric3D-v2 [58, 13], Depth Anything / V2 [54, 55], and UniDepth [27]. The MoGe family [44, 45] further reformulates the task as affine-invariant point-map regression.

Despite their strong performance, these approaches still suffer from noticeable geometric fidelity distortions. To address this problem, Depth Pro [4] turns to high-resolution image input and multi-scale vision encoder architecture. Marigold [16] and GeoWizard [9] introduce powerful generative priors for monocular geometry through VAE-based decoding. Pixel-Perfect Depth [52] further improves edge fidelity by directly performing diffusion generation in the pixel space, avoiding VAE-compression-induced artifacts. InfiniDepth [59], instead, represents depth as a neural implicit field and adopts 2D UV queries for finer-grained depth estimation.

Although these methods significantly improve reconstruction quality, they still rely on decoding 3D geometry within a 2D parameterization. As a result, feature interactions are primarily determined by image-plane proximity rather than actual 3D spatial relationships. This mismatch can limit geometric accuracy when neighboring pixels in the image are distant in 3D space, or vice versa.

Sparse 3D Aggregation. Sparse 3D aggregation focuses on explicit 3D structure and aggregates features according to 3D spatial locality. Prior methods mainly use sparse 3D convolutional networks to operate directly in voxel space while avoiding empty-space overhead, achieving strong performance in different downstream tasks, such as scene understanding [11, 6, 53, 38] and 3D generation [51]. Some methods also use sparse 3D aggregation to refine the noisy point cloud inputs [46, 32], significantly improving reconstruction fidelity and overall quality. To the best of our knowledge, we are the first to lift monocular geometry modeling from the 2D image plane to 3D space and aggregate features via sparse 3D convolutions, mitigating erroneous feature mixing across depth discontinuities.

3 Method

As illustrated in Figure 2, MoGe-3 consists of two main components: a base model initialized from MoGe-2 [45] that produces an initial point map prediction, and a Self-Guided Sparse 3D Refiner (SSR) that further enhances geometric fidelity in 3D space. The base model also extracts a dense feature map $\boldsymbol{F}$ from a DINOv2 encoder [26], which we reuse in our refinement stage.

3.1 Base Model

Given an input image, our base model extracts a dense feature map $\boldsymbol{F}$ and produces a dense per-pixel 3D point map $\boldsymbol{P}$ together with auxiliary predictions including a validity mask, surface normals, and a global metric scale, as shown in Figure 2(a). Same as [45], $\boldsymbol{P}$ is supervised only up to an unknown *affine transform along the optical axis*, i.e. the loss is invariant under

$$(X, Y, Z) \;\mapsto\; \big(sX,\; sY,\; sZ + t\big), \qquad s > 0,\; t \in \mathbb{R}. \tag{1}$$

The *metric point map* is then obtained by adding the recovered optical-axis shift, back-projecting with recovered intrinsics, and applying the predicted metric scale.

3.2 SSR: Self-Guided Sparse 3D Refiner

The SSR module takes the initial point map $\boldsymbol{P}$ as input and iteratively refines its geometry. To normalize the point cloud with different range of (X_{ij}, Y_{ij}) into a uniform space and focus on depth refinement, we employ a factorized representation $\boldsymbol{q}_{ij}$ for each point $\boldsymbol{p}_{ij}$ to :

$$\boldsymbol{q}_{ij} = (u_{ij}, v_{ij}, \zeta_{ij}) = \left(\frac{X_{ij}}{Z_{ij}}, \frac{Y_{ij}}{Z_{ij}}, \log Z_{ij} \right), \tag{2}$$

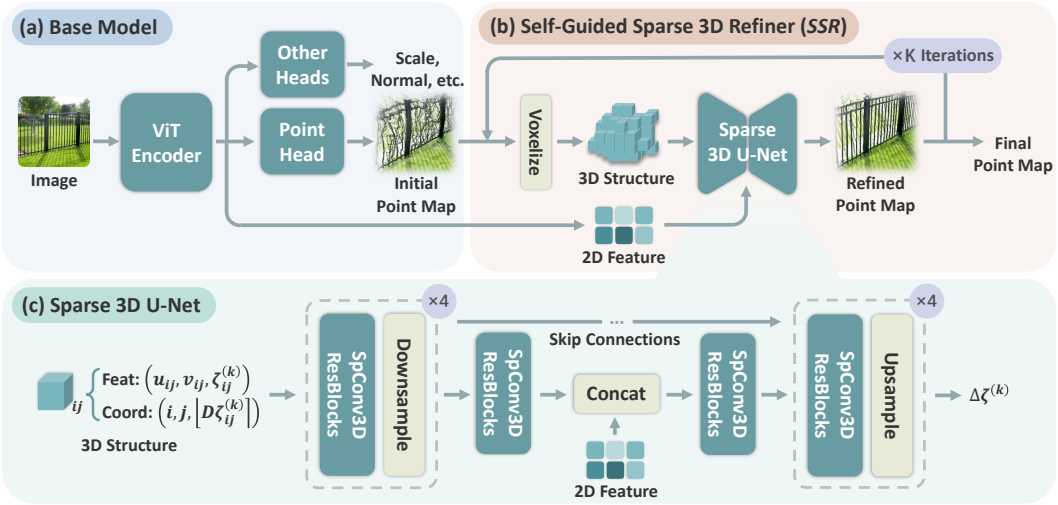

Figure 2: **Method overview.** (a) The **Base model** provides an initial point map and 2D features. (b) The **SSR module** iteratively refines this geometry: current estimates are discretized into sparse voxel shells that guide 3D inference, which then updates the point map for the subsequent cycle. (c) The **Sparse 3D U-Net** employs multi-scale sparse 3D convolutions, injecting 2D features from base model, and predicts log-depth residuals.

where i and j are the row and column indices of the corresponding pixel. In this formulation, the image-axis coordinates (u, v) remain fixed throughout the process, allowing the refiner to concentrate on predicting residuals for the log-depth ζ, starting from the initial $\zeta^{(0)}$.

As illustrated in Figure 2(b), each iteration consists of three steps: (1) Self-guided voxelization 3.2.1, where the current point map is discretized into a sparse 3D voxel shell; (2) Sparse 3D U-Net 3.2.2, which performs multi-scale feature extraction and 2D-3D feature integration; and (3) Iterative update 3.2.3, which applies the predicted residuals to refine the geometry.

3.2.1 Self-Guided Voxelization

To lift monocular geometry modeling from the 2D image plane to 3D space, we first transform the point map into a sparse voxel set, where each pixel is corresponding to exactly one voxel. In this construction, the image axes (i, j) provide a natural discretization, while the depth axis is quantized by scaling the log-depth $\zeta_{ij}^{(k)} = \log Z_{ij}^{(k)}$ with a specific resolution D. The sparse voxel set at iteration k is defined as:

$$\mathcal{V}^{(k)} = \left\{ \boldsymbol{c}_{ij}^{(k)} \right\}, \qquad \boldsymbol{c}_{ij}^{(k)} = \left(i,\, j,\, \left\lfloor D \cdot \zeta_{ij}^{(k)} \right\rceil \right) \tag{3}$$

where $\lfloor \cdot \rceil$ denotes rounding to the nearest integer. Each occupied voxel is associated with a factorized point representation $\boldsymbol{q}_{ij}^{(k)}$, which serves as a 3-channel feature input.

This construction yields two key properties: (1) *Projection-aligned & scale-adaptive shell:* $\mathcal{V}^{(k)}$ is coherent with the image resolution and depth relative precision, forming a thin manifold with exactly HW occupied voxels. The log-depth parameterization maintains a fixed relative depth resolution across metric scales, preserving the overall density distribution and locality across metric scales. (2) *Geometry-coherent receptive field:* Two pixels (voxels) are 3D-adjacent in $\mathcal{V}^{(k)}$ if and only if they are image-adjacent and their quantized log-depth indices satisfy $|\lfloor D\zeta_{ij}^{(k)} \rceil - \lfloor D\zeta_{i'j'}^{(k)} \rceil| \leq 1$. This naturally decouples features across depth discontinuities, as pixels from different surfaces land in separate voxels, preventing the "bleeding" and "twisted" artifacts inherent in 2D-based feature mixing.

3.2.2 Sparse 3D U-Net

We build a sparse 3D U-Net to process the lifted monocular geometry in 3D space. As illustrated in Figure 2(c), our sparse 3D U-Net first encodes the sparse voxel set $\mathcal{V}^{(k)}$ with multiple sparse 3D convolution blocks, each followed by a downsampling operation. Then, at the lowest resolution level, the encoded voxels are concatenated with 2D features $\boldsymbol{F}$ from base model. The concatenated results are then used to decode a *scalar log-depth residual* $\Delta\zeta_{ij}^{(k)}$ for point map refinement.

Dynamic-sparsity downsample & upsample. Different from the vanilla 2D U-Net, which performs downsampling and upsampling uniformly in image space, our method enables geometry-dependent sparse structures across resolution levels through sparse voxel representations. At each level l, the sparse voxel set is defined as $\mathcal{V}^{(k),l} = \left\{ \left\lfloor \frac{1}{2^l} \boldsymbol{c}_{ij} \right\rfloor \right\}$. While the input level ($l = 0$) contains exactly HW voxels, the number and distribution of voxels at coarser levels ($l \geq 1$) vary with the underlying scene geometry. In particular, structures that collapse to the same image-space location can still remain separated along the depth dimension, leading to multiple occupied voxels around depth discontinuities and naturally reducing foreground-background feature mixing during multi-scale processing.

2D feature injection. Since voxel features only encode geometric coordinates and lack high-level scene understanding, we further inject the features $\boldsymbol{F}$ from base model into the U-Net bottleneck for efficient fusion at reduced spatial resolution. At this level, 2D tokens are mapped to bottleneck voxels based on their image-axis coordinates. Voxels sharing an image location but differing in depth duplicate the same 2D feature, allowing 3D convolutions to read from rich appearance priors.

Output layer initialization. The U-Net predicts a *scalar log-depth residual* $\Delta\zeta_{ij}^{(k)}$ corresponding to each input voxel. In order to stabilize the optimization at the start of training, we zero-initialize the output layer of sparse 3D U-Net, allowing identity mapping.

3.2.3 Iterative update

The predicted log-depth residuals $\Delta\zeta^{(k)}$ are added to the current estimate to update the factorized representation:

$$\boldsymbol{q}_{ij}^{(k+1)} = \left(u_{ij}, v_{ij}, \zeta_{ij}^{(k)} + \Delta\zeta_{ij}^{(k)} \right). \tag{4}$$

The updated $\boldsymbol{q}^{(k+1)}$ defines the voxel shell for the subsequent iteration $k+1$. This self-guiding loop progressively refines the spatial locality of the 3D convolutions, allowing the refiner to operate on increasingly precise surface boundaries.

After K iterations, the refined affine point map $\boldsymbol{p}^{(K)}$ is recovered via:

$$\boldsymbol{p}_{ij}^{(K)} = e^{\zeta_{ij}^{(K)}} \cdot (u_{ij}, v_{ij}, 1). \tag{5}$$

3.3 Losses and Training

We optimize a composite objective that generally follows the formulation in [45]. For each SSR iteration $k \in \{0, \ldots, K\}$, we define a geometric objective $\mathcal{L}_{\text{geo}}^{(k)}$ that combines a global affine-invariant loss $\mathcal{L}_{\text{global}}^{(k)}$, a multi-scale local loss $\mathcal{L}_{\text{local}}^{(k)}$, and an edge-angle loss $\mathcal{L}_{\text{edge}}^{(k)}$:

$$\mathcal{L}_{\text{geo}}^{(k)} = \lambda_{\text{g}} \mathcal{L}_{\text{global}}^{(k)} + \lambda_{\ell} \mathcal{L}_{\text{local}}^{(k)} + \lambda_{\text{e}} \mathcal{L}_{\text{edge}}^{(k)}. \tag{6}$$

In addition to these per-iteration terms, we supervise the auxiliary outputs inherited from the base model, including mask, surface normal, and global scale predictions using their respective loss terms $\mathcal{L}_{\text{mask}}$, $\mathcal{L}_{\text{nml}}$, and $\mathcal{L}_{\text{scale}}$. Their detailed descriptions can be found in Appendix A.1. The full loss objective is defined as:

$$\mathcal{L} = \sum_{k=0}^{K} \mathcal{L}_{\text{geo}}^{(k)} + \lambda_{\text{m}} \mathcal{L}_{\text{mask}} + \lambda_{\text{n}} \mathcal{L}_{\text{nml}} + \lambda_{\text{s}} \mathcal{L}_{\text{scale}}. \tag{7}$$

The initial prediction ($K = 0$) is directly supervised by the full geometric objective $\mathcal{L}_{\text{geo}}^{(0)}$. At every refinement iteration, we detach the input point map, preventing subsequent refinement losses from backpropagating through preceding point predictions.

The model is trained in two stages. **Stage 1** (*warm-up*) detaches gradients from the base model features to the refiner. Combined with the zero-initialization of the output layer, this stage encourages the refiner to strictly learn residual corrections starting from a known-good initialization. **Stage 2** (*joint fine-tuning*) enables end-to-end training of the entire pipeline. We empirically find this two-stage training strategy essential, as it prevents large early-stage gradients of refiner from corrupting the

Table 1: Zero-shot evaluation and comparison with state-of-the-art methods. Metrics are averaged over multiple benchmarks (see text for details). Results are visualized using a color gradient from green (best) to red (worst); - indicates non-applicable. MoGe-2* denotes MoGe-2 retrained on the same data mixture as ours. Unless otherwise specified, our method uses the default ViT-L backbone; the ViT-G row reports our scaled-up SSR model. See Appendix C for per-dataset results.

Method	Local				Boundary	Global					
	Depth		Point		F1↑	Relative Depth		Relative Point		Metric Depth	
	$\text{Rel}^{\text{d}}\downarrow$	$\delta^{\text{d}}_{0.01}\uparrow$	$\text{Rel}^{\text{p}}\downarrow$	$\delta^{\text{p}}_{0.01}\uparrow$		$\text{Rel}^{\text{d}}\downarrow$	$\delta^{\text{d}}_{1}\uparrow$	$\text{Rel}^{\text{p}}\downarrow$	$\delta^{\text{p}}_{1}\uparrow$	$\text{Rel}^{\text{d}}\downarrow$	$\delta^{\text{d}}_{1}\uparrow$
Depth Pro	4.15	46.9	3.01	45.2	16.3	8.10	91.3	11.1	87.6	29.0	54.0
UniDepth V2	5.60	45.3	4.60	42.6	14.5	6.98	92.8	10.2	88.6	23.7	71.2
Depth Anything 3	10.5	41.8	7.39	44.7	6.61	7.89	91.4	10.6	87.5	17.4	70.1
UniK3D	5.52	46.2	4.37	45.0	15.1	6.95	92.8	9.63	89.6	17.9	77.8
Pixel-Perfect Depth	5.24	39.8	4.39	39.1	12.7	7.17	92.5	20.3	69.8	-	-
InfiniDepth	5.82	43.2	4.29	43.7	19.3	8.15	91.1	11.6	87.6	-	-
MoGe-2	3.81	47.1	3.19	46.6	15.6	5.98	94.1	8.73	90.4	15.6	77.3
MoGe-2*	3.60	48.8	2.94	50.3	15.3	5.54	94.6	8.00	91.9	15.0	80.6
Ours (ViT-L)	3.76	52.5	2.79	55.9	16.0	5.52	94.7	7.91	92.0	15.0	82.7
Ours (ViT-G)	3.66	54.1	2.80	56.8	17.3	4.80	95.6	7.43	92.4	15.8	82.7

representation of well-pretrained base model. We further partition the training data across components: the refiner receives gradients only from synthetic samples, whose pixel-accurate ground truth is essential for learning sharp and geometrically correct residual corrections, while the base model is supervised on the full mixture of synthetic and real-world data to retain in-the-wild generalization. Detailed hyper-parameters are provided in Appendix A.

4 Experiments

4.1 Implementation Details

We train our model on a large-scale mixture of synthetic and real-world datasets; see Appendix A for the training data details. The model is trained using AdamW with a staged training schedule. The full training details, including loss weights, learning rates, and iteration settings, are provided in Appendix A. ***All our code, model, training and evaluation scripts will be made publicly available.***

Unless otherwise specified, our default model is the full two-stage-trained model with a DINOv2 ViT-L backbone and $K = 3$ refinement iterations at inference. We also train a scaled-up variant with a DINOv2 ViT-G backbone.

4.2 Evaluation Setting

Evaluation datasets. We report zero-shot evaluation results on 9 benchmarks that span synthetic and real, indoor and outdoor, dense and sparse-LiDAR ground truth: Synth4K [59], Spring [23], NYUv2 [24], KITTI [40], ETH3D [35], DIODE [41], Sintel [5], iBims-1 [17], and HAMMER [15].

Evaluation metrics. We report three types of evaluation metrics: (i) Global metrics evaluate full-image geometry under three settings: affine-invariant point maps (*Relative Point*), affine-invariant depth (*Relative Depth*), and metric depth (*Metric Depth*). For each setting, we report relative error (Rel) and threshold accuracy δ, using thresholds of 0.25 for points and 1.25 for depth. (ii) Local (fine-detail) metrics measure accuracy in regions with thin structures and complex geometry. We automatically extract region-level masks that select small segments containing fine-detail geometry structures (see Appendix B.1 for details). For each region, we perform per-segment alignment and compute Rel and δ metrics, using stricter thresholds of 0.01 for point maps and 1.01 for depth maps. The reported numbers are the average over all selected regions. (iii) Boundary F1 (radius $r{=}1$) evaluates depth discontinuity sharpness by comparing predicted and ground-truth boundary maps.

Baselines. We compare against seven recent state-of-the-art methods: Depth Pro [4], UniDepth V2 [29], Depth Anything 3 [22], UniK3D [28], Pixel-Perfect Depth [52], InfiniDepth [59], and MoGe-2 [45]. To control for differences in training data, we additionally train a standalone MoGe-2 on the same data mixture as our method, denoted by MoGe-2*. Pixel-Perfect Depth and InfiniDepth predict relative depth and do not estimate camera intrinsics. We therefore omit metric-depth evaluation for these methods and reconstruct 3D point maps using camera intrinsics estimated by MoGe-2 on the same image for point-based metrics. All other baselines use their own predicted intrinsics.

4.3 Quantitative Evaluation

Table 1 reports zero-shot evaluation results averaged across the evaluation benchmarks. Global relative depth and point metrics are averaged over all nine test datasets, while metric depth is averaged over the seven applicable datasets, excluding Spring and Sintel. Local metrics are averaged over Spring and Synth4K, as region-level mask extraction requires dense ground truth with fine-detail structures. Boundary F1 is averaged over Spring, iBims-1, Sintel, and HAMMER. These four datasets are representative in that they include both synthetic (Spring, Sintel) and real-world (iBims-1, HAMMER) sources, and contain abundant high-fidelity depth edges for boundary evaluation.

Global metrics. Our method achieves the best global performance across all three evaluation settings. Our ViT-L model matches or outperforms MoGe-2* on all global metrics, demonstrating that the improvements extend beyond local geometric details under the same training data mixture. The ViT-G variant further improves relative depth and point estimation.

Local (fine-detail) metrics. The advantage of our method is most pronounced on local fine-detail metrics, where we *substantially outperform all baselines under the strict threshold accuracy* $\delta_{0.01}$. This confirms that the 3D sparse refinement effectively resolves thin structures and complex geometry that 2D approaches struggle with. With the same training data, our ViT-L model improves local depth and point threshold accuracy over MoGe-2* by 3.7 and 5.6 percentage points, respectively, supporting the benefit of SSR beyond the larger data mixture.

Boundary F1. InfiniDepth [59] achieves the best boundary sharpness by leveraging explicit $2\times$ higher-resolution query processing to enhance image-space edge fidelity. Among all model variants in Table 1, our ViT-G and ViT-L models rank second and fourth. The default ViT-L model is comparable to Depth Pro [4], which operates at 1536^2 native resolution, nearly $3.4\times$ as many pixels as our 840^2.

It should be noted that boundary sharpness is a 2D, pixel-level metric that mainly reflects image-space depth edge fidelity and *does not fully capture 3D geometric quality*. As shown in Fig. 3, our method produces 3D point maps with much better structural fidelity. This is because the 3D refiner sharpens depth discontinuities by resolving local geometry in 3D space, where occluding and occluded surfaces are explicitly separated into distinct voxels rather than being blended across adjacent pixels.

Scaling to a larger backbone. To examine whether SSR remains effective with stronger 2D foundation models, we train a scaled-up variant where SSR is applied on top of a MoGe-2-style base model with a DINOv2 ViT-G backbone. As shown in Table 1, the ViT-G model further improves over our ViT-L setting across most metric categories, indicating that SSR is complementary to backbone scaling and can effectively leverage the richer representations provided by larger encoders.

Inference efficiency. Table 2 reports inference latency. With $K{=}0$, our model has identical latency to MoGe-2 since they share the same base architecture. With three SSR iterations, the ViT-L model runs at 168 ms per frame, remaining competitive with prior detail-enhancement methods while achieving better accuracy across most metrics. The scaled-up ViT-G variant runs at 234 ms, still competitive in speed and further improving quality. Detailed inference latency and peak GPU memory usage are reported in Appendix C.2.

Table 2: Runtime comparison on a single A100 GPU with FP16.

Method	Resolution	Latency (ms)
MoGe-2 (ViT-L)	700^2	39
InfiniDepth	512×672	133
Depth Pro	1536^2	142
Depth Anything 3	630×840	213
Pixel-Perfect Depth	512×672	275
Ours (ViT-L, 3-step)	700^2	168
Ours (ViT-G, 3-step)	700^2	234

4.4 Qualitative Evaluation

Fig. 3 presents qualitative comparisons of point maps and disparity maps against previous methods. For methods that do not predict camera intrinsics, we use the intrinsics predicted by MoGe-2 [45] to lift their depth predictions into 3D point maps. Our method produces sharp disparity boundaries in image space while maintaining faithful 3D geometry, with accurate reconstruction of complex regions with thin structures. In contrast, although prior methods often appear sharp in 2D depth maps, their reconstructed 3D point maps exhibit noticeable structural distortions. To the best of our knowledge, we are the first monocular geometry estimation method that produces visually plausible depth while also maintaining high-fidelity 3D geometry.

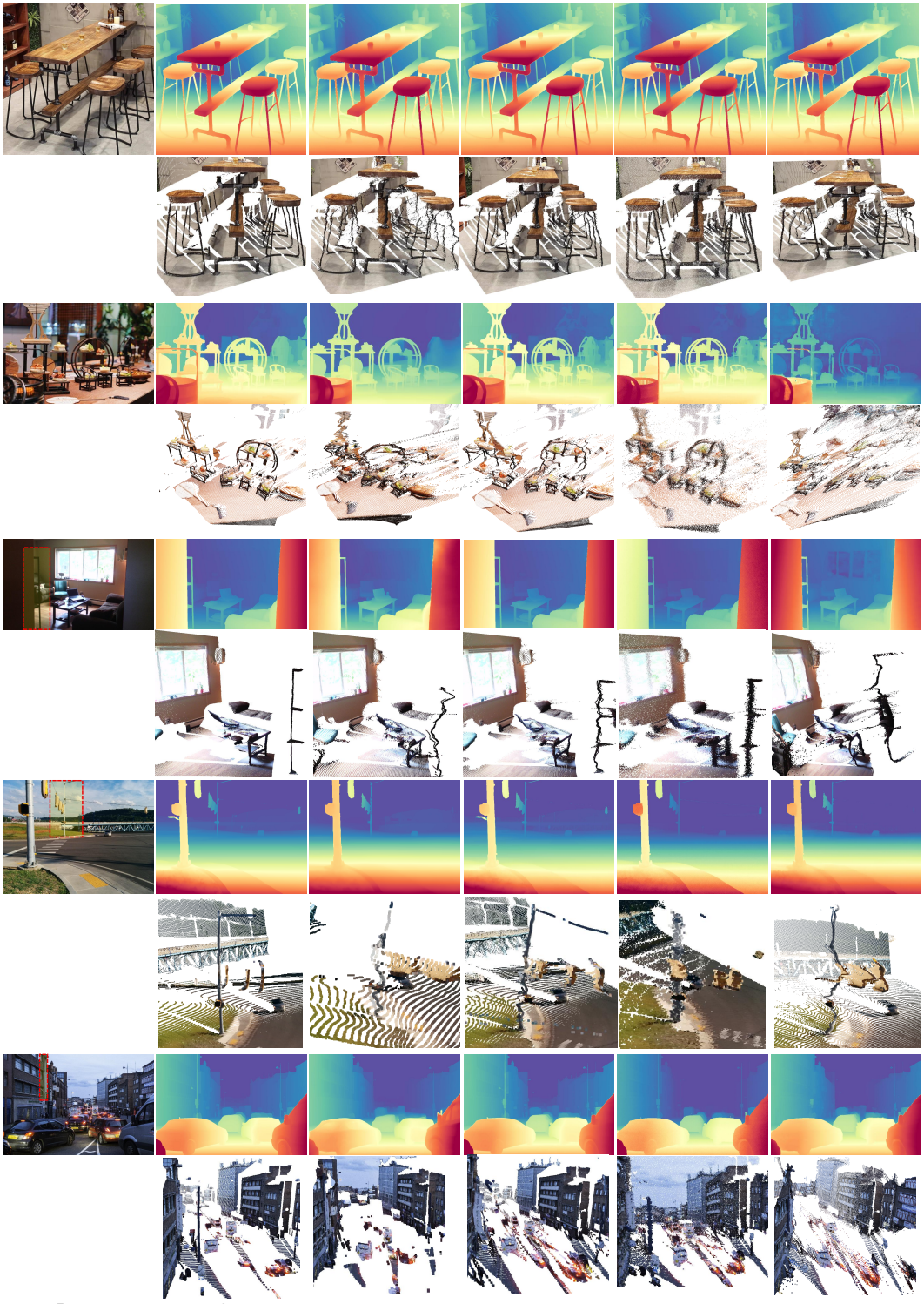

Figure 3: Qualitative comparison of point maps and disparity maps on unseen test images. Since Pixel-Perfect Depth (PPD) [52] and InfiniDepth [59] do not predict camera intrinsics, we use the intrinsics predicted by MoGe-2 [45] to lift their depth predictions into 3D. Our method produces noticeably higher-fidelity 3D geometry with reduced distortions, particularly in complex regions and thin structures. *More visual comparisons can be found in Appendix D.*

4.5 Ablation studies

For the ablation studies on voxel resolution D, visual feature injection, and the 2D convolution variant, we use a refiner-only setting where the base model is frozen and only the SSR module is trained, allowing the effect of the refiner to be evaluated independently. All models are trained and evaluated with $K = 3$ refinement iterations.

Effect of voxel resolution D. The voxel resolution D controls the spatial granularity of the 3D sparse convolutions: larger D yields finer voxels, but also increases memory usage and sparse occupancy. As shown in Table 3, increasing the voxel resolution from D=100 to D=200 improves local metrics while largely preserving global accuracy, indicating that finer voxelization better captures fine geometric structures. However, pushing the resolution further degrades both global and local performance due to increasingly sparse occupancy, which limits the effectiveness of the 3D sparse convolutions. We therefore adopt D=200 in the full model as the best trade-off between geometric precision, occupancy density, and efficiency.

Effect of visual feature injection. To examine whether SSR benefits from high-level visual representations, we construct an RGB-input variant. This model removes DINO feature injection at the bottleneck. Instead, each voxel is initialized by concatenating the RGB value of its corresponding pixel to the voxel features. The refiner therefore still observes raw color and local appearance discontinuities, but no longer has direct access to the semantic and global-context information encoded by DINO features. As shown in Table 3, the RGB-input variant improves local threshold accuracy over the frozen base model, but still remains clearly behind the default DINO-feature-injected refiner. Since this variant retains self-guided voxelization and sparse 3D convolutions, its improvement may come from both RGB appearance cues and the 3D refinement structure. These results indicate that SSR can improve geometry without DINO features to some extent, but the semantic and contextual information provided by visual features is crucial for resolving fine structures and disambiguating local 3D geometry.

3D vs. 2D refinement. To evaluate the benefit of the introduced 3D inductive bias, we replace the sparse 3D U-Net with a parameter-matched 2D U-Net operating on the dense image grid while predicting the same log-depth residual $\Delta\zeta^{(k)}$. The only difference is whether refinement is performed on the dense 2D image grid or the self-guided sparse 3D voxel set $\mathcal{V}^{(k)}$. The 2D variant in Table 3 shows that simply increasing model capacity in 2D does not provide the gains achieved by 3D refinement. Despite being parameter-matched to the sparse 3D U-Net, the 2D variant yields only marginal changes on global metrics and mixed results on local geometry: local point accuracy improves, while local depth threshold accuracy and boundary F1 slightly decrease relative to the base model. This highlights a key limitation of 2D refinement: neighboring pixels in the image may correspond to geometrically distant surfaces in 3D, especially near occlusion boundaries. In contrast, sparse 3D convolutions separate such regions into distinct voxels, reducing cross-boundary mixing and enabling substantially sharper local geometry.

Effect of refinement iterations. We study the refinement behavior by varying the number of iterations at inference time, using $K \in \{0, 1, \ldots, 5, 7\}$ with the default jointly trained ViT-L model. Figure 4 shows how global δ_1^p, local $\delta_{0.01}^p$, and boundary F1 evolve with K. At K=0 (*no refinement*), the jointly trained base model already outperforms MoGe-2 baseline. As K increases, local metrics improve substantially after the first step and continue to improve gradually before saturating around K=5, while global metrics remain nearly unchanged. This confirms that SSR selectively sharpens thin structures without disturbing overall geometry. Boundary F1 saturates even earlier, with most gains within the first iteration. Although the refiner is trained with only K=3 iterations, performance

Table 3: Ablation on voxel resolution D, visual feature injection, and the 2D convolution variant for refinement. Results are averaged across evaluation datasets. All variants use a frozen MoGe-2 (ViT-L) base model and are evaluated with $K = 3$ iterations. $\star$ indicates the default setting in this refiner-only ablation.

Method	Local				Boundary	Global					
	Depth		Point		F1↑	Relative Depth		Relative Point		Metric Depth	
	$\text{Rel}^{\text{d}}\downarrow$	$\delta^{\text{d}}_{0.01}\uparrow$	$\text{Rel}^{\text{p}}\downarrow$	$\delta^{\text{p}}_{0.01}\uparrow$		$\text{Rel}^{\text{d}}\downarrow$	$\delta^{\text{d}}_{1}\uparrow$	$\text{Rel}^{\text{p}}\downarrow$	$\delta^{\text{p}}_{1}\uparrow$	$\text{Rel}^{\text{d}}\downarrow$	$\delta^{\text{d}}_{1}\uparrow$
Base model	3.81	47.1	3.19	46.6	15.6	5.98	94.1	8.73	90.4	15.6	77.3
2D Conv.	3.79	46.8	3.01	48.6	15.3	5.95	94.1	8.40	90.9	16.0	76.5
RGB Input	3.89	48.0	3.12	49.3	14.9	6.00	94.0	8.63	90.6	16.4	76.3
$D = 100$	3.76	49.9	2.85	52.9	15.7	5.95	94.2	8.28	91.4	16.0	76.8
$D = 200$ $\star$	3.66	51.4	2.76	54.5	15.6	5.86	94.3	8.33	91.2	16.2	76.1
$D = 300$	3.56	48.8	2.90	51.4	15.1	6.02	94.1	8.33	91.5	15.8	77.4
$D = 400$	3.99	50.1	2.96	52.7	14.8	6.38	93.1	8.74	90.3	16.0	76.7

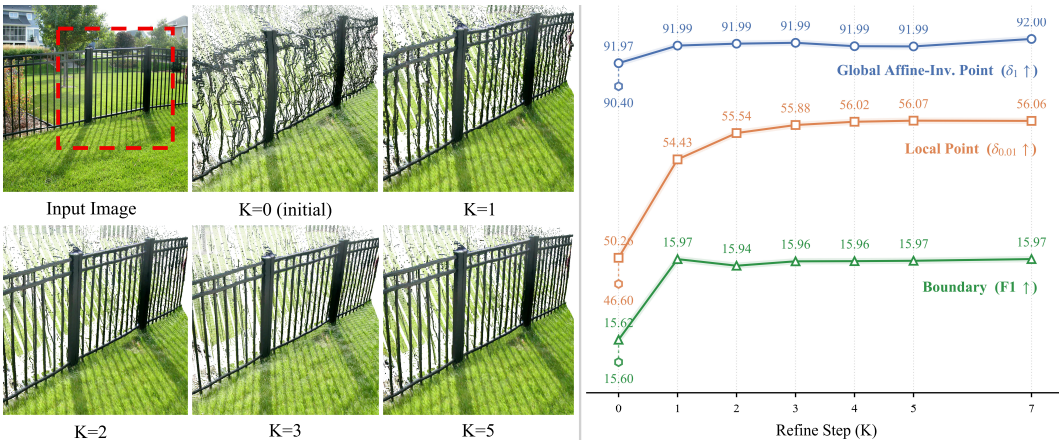

Figure 4: Effect of refinement iterations. Left: point map visualizations demonstrate progressively sharper reconstruction of thin structures. Right: quantitative metrics including global δ_1^p, local $\delta_{0.01}^p$, and boundary F1 remain stable or improve with increasing refinement iterations. The model is trained with $K{=}3$. Larger K values evaluate generalization beyond the training setting. The hexagon marker at $K{=}0$ denotes the performance of the base model MoGe-2.

does not degrade when applying more iterations at test time. Running up to $K{=}7$ yields slightly improved or stable metrics across all categories, indicating that the learned residual updates generalize well beyond the training horizon. This robustness suggests that each refinement step learns a stable, convergent correction rather than overfitting to a fixed number of iterations. The qualitative example in Figure 4 illustrates refinement behavior on thin fence railings, where SSR progressively separates occluding and background structures, recovering correct local geometry.

5 Conclusion

In this paper, we point out a potential architectural mismatch in existing monocular geometry estimation methods, where inherently 3D geometry is decoded within a 2D parameterization, leading to structural distortions and over-smoothed reconstructions around fine details and depth discontinuities. To address this problem, we propose a new paradigm that lifts geometry modeling from 2D image space to 3D space through a Self-Guided Sparse 3D Refiner (SSR), which performs geometry refinement directly on sparse 3D scene manifolds using 3D-aware feature aggregation. Extensive experiments demonstrate that MoGe-3 significantly improves local geometric fidelity while preserving globally plausible geometry. We believe our work significantly advances fine-detail monocular geometry estimation by presenting the first monocular approach that produces point maps both visually plausible in 2D depth maps and faithful in 3D geometry.

Limitations. SSR constructs its sparse neighborhoods from the current geometry estimate, making refinement sensitive to structural errors in the initial point map. When the initialization contains incorrect surface connections, severe depth-ordering errors, or missing thin structures, SSR may not fully recover the correct geometry. In addition, our regression-based formulation can leave residual "fly-points" near ambiguous object boundaries and does not always produce perfectly sharp depth discontinuities. Integrating generative formulations such as [52] may help address these boundary ambiguities.

Acknowledgments and Disclosure of Funding

This research received no external funding.

References

[1] Baidu Apollo. Apollo synthetic dataset, 2019. URL `https://developer.apollo.auto/synthetic.html`. Accessed: 2025-03-06.

[2] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, and Elad Shulman. ARKitscenes - a diverse real-world dataset for 3d indoor scene understanding using mobile RGB-d data. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. URL https://openreview.net/forum?id=tjZjv_qh_CE.

[3] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. Zoedepth: Zero-shot transfer by combining relative and metric depth, 2023. URL https://arxiv.org/abs/2302.12288.

[4] Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R. Richter, and Vladlen Koltun. Depth pro: Sharp monocular metric depth in less than a second, 2025. URL https://arxiv.org/abs/2410.02073.

[5] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, October 2012.

[6] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks, 2019. URL https://arxiv.org/abs/1904.08755.

[7] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects, 2022. URL https://arxiv.org/abs/2212.08051.

[8] Michael Fonder and Marc Van Droogenbroeck. Mid-air: A multi-modal dataset for extremely low altitude drone flights. In *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, June 2019.

[9] Xiao Fu, Wei Yin, Mu Hu, Kaixuan Wang, Yuexin Ma, Ping Tan, Shaojie Shen, Dahua Lin, and Xiaoxiao Long. Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image, 2024. URL https://arxiv.org/abs/2403.12013.

[10] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis, and Peter Schuberth. A2D2: Audi Autonomous Driving Dataset. 2020. URL https://www.a2d2.audi.

[11] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks, 2017. URL https://arxiv.org/abs/1711.10275.

[12] Jose L. Gómez, Manuel Silva, Antonio Seoane, Agnés Borràs, Mario Noriega, German Ros, Jose A. Iglesias-Guitian, and Antonio M. López. All for one, and one for all: Urbansyn dataset, the third musketeer of synthetic driving scenes. *Neurocomputing*, 637:130038, 2025. ISSN 0925-2312. doi: 10.1016/j.neucom.2025.130038. URL http://dx.doi.org/10.1016/j.neucom.2025.130038.

[13] Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Xiaoxiao Long, Hao Chen, Kaixuan Wang, Gang Yu, Chunhua Shen, and Shaojie Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10579–10596, December 2024. ISSN 1939-3539. doi: 10.1109/tpami.2024.3444912. URL http://dx.doi.org/10.1109/TPAMI.2024.3444912.

[14] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[15] HyunJun Jung, Patrick Ruhkamp, Guangyao Zhai, Nikolas Brasch, Yitong Li, Yannick Verdie, Jifei Song, Yiren Zhou, Anil Armagan, Slobodan Ilic, et al. On the importance of accurate geometry data for dense 3d vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 780–791, 2023.

[16] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation, 2024. URL https://arxiv.org/abs/2312.02145.

[17] Tobias Koch, Lukas Liebel, Marco Körner, and Friedrich Fraundorfer. Comparison of monocular depth estimation methods using geometrically relevant metrics on the ibims-1 dataset. *Computer Vision and Image Understanding (CVIU)*, 191:102877, 2020. doi: 10.1016/j.cviu.2019.102877.

[18] Katrin Lasinger, René Ranftl, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *CoRR*, abs/1907.01341, 2019. URL http://arxiv.org/abs/1907.01341.

[19] Hoang-An Le, Partha Das, Thomas Mensink, Sezer Karaoglu, and Theo Gevers. EDEN: Multimodal Synthetic Dataset of Enclosed garDEN Scenes. In *Proceedings of the IEEE/CVF Winter Conference of Applications on Computer Vision (WACV)*, 2021.

[20] Yixuan Li, Lihan Jiang, Linning Xu, Yuanbo Xiangli, Zhenzhi Wang, Dahua Lin, and Bo Dai. Matrixcity: A large-scale city dataset for city-scale neural rendering and beyond, 2023. URL https://arxiv.org/abs/2309.16553.

[21] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos, 2018. URL https://arxiv.org/abs/1804.00607.

[22] 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, 2025. URL https://arxiv.org/abs/2511.10647.

[23] Lukas Mehl, Jenny Schmalfuss, Azin Jahedi, Yaroslava Nalivayko, and Andrés Bruhn. Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[24] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[25] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3d ken burns effect from a single image, 2019. URL https://arxiv.org/abs/1909.05483.

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

[27] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. Unidepth: Universal monocular metric depth estimation, 2024. URL https://arxiv.org/abs/2403.18913.

[28] Luigi Piccinelli, Christos Sakaridis, Mattia Segu, Yung-Hsu Yang, Siyuan Li, Wim Abbeloos, and Luc Van Gool. Unik3d: Universal camera monocular 3d estimation, 2025. URL https://arxiv.org/abs/2503.16591.

[29] Luigi Piccinelli, Christos Sakaridis, Yung-Hsu Yang, Mattia Segu, Siyuan Li, Wim Abbeloos, and Luc Van Gool. Unidepthv2: Universal monocular metric depth estimation made simpler. *arXiv preprint arXiv:2502.20110*, 2025.

[30] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *CoRR*, abs/2103.13413, 2021. URL https://arxiv.org/abs/2103.13413.

[31] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos, 2024. URL https://arxiv.org/abs/2408.00714.

[32] Alexander Rich, Noah Stier, Pradeep Sen, and Tobias Höllerer. 3dvnet: Multi-view depth prediction and volumetric refinement. In *2021 International Conference on 3D Vision (3DV)*, pages 700–709. IEEE, 2021.

[33] Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding, 2021. URL https://arxiv.org/abs/2011.02523.

[34] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[35] Thomas Schöps, Torsten Sattler, and Marc Pollefeys. BAD SLAM: Bundle adjusted direct RGB-D SLAM. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[36] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3, 2025. URL https://arxiv.org/abs/2508.10104.

[37] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[38] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution, 2020. URL https://arxiv.org/abs/2007.16100.

[39] Fabio Tosi, Yiyi Liao, Carolin Schmitt, and Andreas Geiger. Smd-nets: Stereo mixture density networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[40] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *International Conference on 3D Vision (3DV)*, 2017.

[41] Igor Vasiljevic, Nick Kolkin, Shanyi Zhang, Ruotian Luo, Haochen Wang, Falcon Z. Dai, Andrea F. Daniele, Mohammadreza Mostajabi, Steven Basart, Matthew R. Walter, and Gregory Shakhnarovich. DIODE: A Dense Indoor and Outdoor DEpth Dataset. *CoRR*, abs/1908.00463, 2019. URL http://arxiv.org/abs/1908.00463.

[42] Kaixuan Wang and Shaojie Shen. Flow-motion and depth network for monocular stereo and beyond. *CoRR*, abs/1909.05452, 2019. URL http://arxiv.org/abs/1909.05452.

[43] Qiang Wang, Shizhen Zheng, Qingsong Yan, Fei Deng, Kaiyong Zhao, and Xiaowen Chu. Irs: A large naturalistic indoor robotics stereo dataset to train deep models for disparity and surface normal estimation, 2021. URL https://arxiv.org/abs/1912.09678.

[44] Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision, 2025. URL https://arxiv.org/abs/2410.19115.

[45] Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details, 2025. URL https://arxiv.org/abs/2507.02546.

[46] Tao Wang, Jing Wu, Ze Ji, and Yu-Kun Lai. Sparse convolutional networks for surface reconstruction from noisy point clouds. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3212–3221, 2024.

[47] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam, 2020. URL https://arxiv.org/abs/2003.14338.

[48] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021)*, 2021.

[49] Magnus Wrenninge and Jonas Unger. Synscapes: A photorealistic synthetic dataset for street scene parsing. *CoRR*, abs/1810.08705, 2018. URL http://arxiv.org/abs/1810.08705.

[50] Siyang Wu, Jack Nugent, Willow Yang, and Jia Deng. Toward a better understanding of monocular depth evaluation, 2025. URL https://arxiv.org/abs/2510.19814.

[51] Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, et al. Native and compact structured latents for 3d generation. *arXiv preprint arXiv:2512.14692*, 2025.

[52] Gangwei Xu, Haotong Lin, Hongcheng Luo, Xianqi Wang, Jingfeng Yao, Lianghui Zhu, Yuechuan Pu, Cheng Chi, Haiyang Sun, Bing Wang, Guang Chen, Hangjun Ye, Sida Peng, and Xin Yang. Pixel-perfect depth with semantics-prompted diffusion transformers, 2025. URL https://arxiv.org/abs/2510.07316.

[53] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10), 2018. ISSN 1424-8220. doi: 10.3390/s18103337. URL https://www.mdpi.com/1424-8220/18/10/3337.

[54] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data, 2024. URL https://arxiv.org/abs/2401.10891.

[55] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2, 2024. URL https://arxiv.org/abs/2406.09414.

[56] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks, 2020. URL https://arxiv.org/abs/1911.10127.

[57] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023.

[58] Wei Yin, Chi Zhang, Hao Chen, Zhipeng Cai, Gang Yu, Kaixuan Wang, Xiaozhi Chen, and Chunhua Shen. Metric3d: Towards zero-shot metric 3d prediction from a single image, 2023. URL https://arxiv.org/abs/2307.10984.

[59] Hao Yu, Haotong Lin, Jiawei Wang, Jiaxin Li, Yida Wang, Xueyang Zhang, Yue Wang, Xiaowei Zhou, Ruizhen Hu, and Sida Peng. Infinidepth: Arbitrary-resolution and fine-grained depth estimation with neural implicit fields, 2026. URL https://arxiv.org/abs/2601.03252.

[60] Amir Zamir, Alexander Sax, William Shen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning, 2018. URL https://arxiv.org/abs/1804.08328.

[61] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3d: A large photo-realistic dataset for structured 3d modeling. In *European Conference on Computer Vision (ECCV)*, 2020.

[62] Yang Zhou, Yifan Wang, Jianjun Zhou, Wenzheng Chang, Haoyu Guo, Zizun Li, Kaijing Ma, Xinyue Li, Yating Wang, Haoyi Zhu, Mingyu Liu, Dingning Liu, Jiange Yang, Zhoujie Fu, Junyi Chen, Chunhua Shen, Jiangmiao Pang, Kaipeng Zhang, and Tong He. Omniworld: A multi-domain and multi-modal dataset for 4d world modeling, 2025. URL https://arxiv.org/abs/2509.12201.

A Training Details

A.1 Detailed Loss Design

This appendix expands the main-text losses of Sec. 3.3 with the exact solver formulations, weight clipping, and hyper-parameter values used in our implementation.

Global affine-invariant loss. Following [44], supervision is invariant under the optical-axis affine transform of Sec. 3.1. We solve the optimal (s, t_z) alignment in closed form on a low-resolution copy of the prediction and apply a depth-relative ℓ_1 residual,

$$\mathcal{L}^{(k)}_{\text{global}} \;=\; \Big\langle\, w_{ij}\, \Big\| s^{(k)} \boldsymbol{p}^{(k)}_{ij} + t^{(k)}_z \boldsymbol{e}_z - \widehat{\boldsymbol{p}}_{ij} \Big\|_1 \Big\rangle_{(i,j)\in\mathcal{M}}, \qquad w_{ij} \propto 1/\widehat{Z}_{ij}, \tag{8}$$

where $\langle\cdot\rangle_{\mathcal{M}}$ denotes the mask-weighted mean over valid pixels, and $(s^{(k)}, t^{(k)}_z)$, which is the closed-form weighted-median solution of

$$\big(s^{(k)},\, t^{(k)}_z\big) = \arg\min_{s>0,\, t_z} \sum_{ij} w_{ij} \big\| s\, \boldsymbol{p}^{(k)}_{ij} + t_z\, \boldsymbol{e}_z - \widehat{\boldsymbol{p}}_{ij} \big\|_1. \tag{9}$$

Radial-partition local loss. To penalize local geometric distortions that global alignment might overlook, we adopt the local loss from MoGe with two modifications:

(1) **Stochastic multi-scale radial partitioning**: At each loss evaluation, with equal probability we choose either the ground-truth point map or the current globally aligned prediction as the reference point map for partitioning:

$$\boldsymbol{r}^{(k)}_{ij} = \begin{cases} \widehat{\boldsymbol{p}}_{ij}, & \text{w.p. } 0.5, \\ \widetilde{\boldsymbol{p}}^{(k)}_{ij}, & \text{w.p. } 0.5, \end{cases} \qquad \widetilde{\boldsymbol{p}}^{(k)}_{ij} = s^{(k)} \boldsymbol{p}^{(k)}_{ij} + t^{(k)}_z \boldsymbol{e}_z. \tag{10}$$

We then group pixels into 3D local regions $\mathcal{G}^{(k)}_{\alpha,g}$ by applying the radial partition function to $\boldsymbol{r}^{(k)}_{ij}$, which quantizes both the projection direction and the log-distance in a scale-invariant manner:

$$\mathcal{G}^{(k)}_{\alpha,g} = \Big\{ (i,j) \Big| \boldsymbol{\pi}^{(k)}_{\alpha}(i,j) = g \Big\}, \qquad \boldsymbol{\pi}^{(k)}_{\alpha}(i,j) = \left\lfloor \alpha \cdot \left(\frac{\boldsymbol{r}^{(k)}_{ij}}{\|\boldsymbol{r}^{(k)}_{ij}\|_\infty}, \log \|\boldsymbol{r}^{(k)}_{ij}\|_2 \right) \right\rfloor, \tag{11}$$

where α controls partition granularity and g identifies a local partition under such granularity. We use $\alpha \in \{4, 16, 64\}$ to provide multi-scale supervision. In practice, we apply a random rotation and scale before the partition function for perturbation to the partition boundaries. The stochastic source selection encourages the partitions to respect both ground-truth geometry and the model's current foreground–background separation, reducing over-smoothing across local depth discontinuities.

(2) **Scale-shared local alignment**: For each local region $\mathcal{G}^{(k)}_{\alpha,g}$, we optimize a 3-DoF shift $\boldsymbol{t}^{(k)}_{\alpha,g}$ while sharing the global scale factor $s^{(k)}$. This design is motivated by the observation that local scale-shift alignments typically yield scale factors that deviate minimally from the global one. By fixing the scale, we simplify the objective to:

$$\mathcal{L}^{(k)}_{\text{local}} = \sum_{\alpha\in\{4,16,64\}} \Big\langle\, w_{ij}\, \Big\| s^{(k)} \boldsymbol{p}^{(k)}_{ij} + \boldsymbol{t}^{(k)}_{\alpha,g} - \widehat{\boldsymbol{p}}_{ij} \Big\|_1 \Big\rangle_{(i,j)\in\mathcal{M}}, \qquad g = \boldsymbol{\pi}^{(k)}_{\alpha}(i,j) \tag{12}$$

where $\boldsymbol{t}^{(k)}_{\alpha,g}$ minimizes the alignment objective within its local region $\mathcal{G}^{(k)}_{\alpha,g}$:

$$\boldsymbol{t}^{(k)}_{\alpha,g} = \arg\min_{\boldsymbol{t}} \sum_{(i,j)\in\mathcal{G}^{(k)}_{\alpha,g}} w_{ij} \big\| s^{(k)} \boldsymbol{p}^{(k)}_{ij} + \boldsymbol{t} - \widehat{\boldsymbol{p}}_{ij} \big\|_1, \tag{13}$$

Compared to the original formulation, this approach improves the efficiency to linear computational time without compromising the effectiveness of local geometric supervision.

Edge-angle loss. The Edge-angle loss can be written as:

$$\mathcal{L}^{(k)}_{\text{edge}} = \frac{1}{\min(H,W)|\mathcal{D}_d|} \sum_{(i,j)\in\mathcal{D}_d} \Big(\angle \Big(\partial_x \boldsymbol{p}^{(k)}_{ij}, \partial_x \widehat{\boldsymbol{p}}_{ij} \Big) + \angle \Big(\partial_y \boldsymbol{p}^{(k)}_{ij}, \partial_y \widehat{\boldsymbol{p}}_{ij} \Big) \Big), \tag{14}$$

where $\mathcal{D}_d$ are the pixels whose two d-neighbors are both valid.

Auxiliary head losses. The losses for the auxiliary heads are defined as:

$$\mathcal{L}_{\text{mask}} = \text{BCE}(\boldsymbol{M}, \widehat{\boldsymbol{M}}), \quad \mathcal{L}_{\text{nml}} = \left\langle \angle(\boldsymbol{N}, \widehat{\boldsymbol{N}})^2 \right\rangle, \quad \mathcal{L}_{\text{scale}} = \mathbb{I}(\hat{s} > 0)\,(\log s - \log \hat{s})^2. \tag{15}$$

A.2 Detailed Training Setup

We train our model using loss weights $(\lambda_{\text{g}}, \lambda_{\ell}, \lambda_{\text{e}}, \lambda_{\text{m}}, \lambda_{\text{n}}, \lambda_{\text{s}}) = (1.0, 1.0, 1.0, 0.1, 0.1, 0.1)$. The local term λ_{ℓ} is implemented as a multi-scale radial-partition loss evaluated at scales $\alpha \in \{4, 16, 64\}$, with equal weight 1.0 at each scale. Optimization is performed with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay 10^{-2}), together with gradient clipping at $\|g\|_2 \leq 1.0$. We adopt a staged learning-rate schedule with warm-up and exponential decay. The backbone is frozen for the first $1{,}000$ steps, then linearly warmed up until step $2{,}000$. After warm-up, the backbone and 2D heads decay every $25{,}000$ steps, while the refiner decays every $10{,}000$ steps. Peak learning rates are 2×10^{-4} for the refiner, 1×10^{-4} for the 2D heads, and 5×10^{-6} for the DINO backbone. Training uses a global batch size of 96 on $16 \times \text{A}100$ GPUs with mixed precision. The DINO encoder operates in `bfloat16`, while all other modules use `float32`. For stability, the refiner is detached from the backbone for the first $5{,}000$ steps and then jointly optimized with gradient flow through the shared visual features.

A.3 Training Data

We train the full model on a large, heterogeneous mixture of synthetic datasets with pixel-perfect geometry, including 3D Ken Burns [25], ApolloSynthetic [1], EDEN [19], GTA-SfM [42], Hypersim [33], IRS [43], MatrixCity [20], MidAir [8], MVS-Synth [14], Objaverse [7], OmniWorld [62], Structured3D [61], Synscapes [49], Synthia [34], TartanAir [47], UnrealStereo4K [39], and UrbanSyn [12], and real / sensor-reconstructed datasets including A2D2 [10], ARKitScenes [2], Argoverse2 [48], BlendedMVS [56], MegaDepth [21], ScanNet++ [57], Taskonomy [60], and Waymo [37] that supply real-world appearance and sensor statistics.

B Details of Local Metric Evaluation

B.1 Local Fine-Grained Mask Construction

We design an automatic pipeline to identify fine-grained regions in depth maps, such as thin structures, depth discontinuities, and delicate geometric details. The pipeline consists of two stages: coarse mask detection and segment-aware expansion.

Coarse mask construction. We combine two complementary detectors: (1) *Multi-scale Laplacian residual detector.* We first compute the residual $r_s(p) = |d(p) - \tilde{d}_s(p)|$ between the disparity and its downsampled-then-upsampled version under different scales $s \in \{8, 16, 32\}$. A pixel is classified as belonging to a fine-grained region when $r_s(p) > 3.0 \cdot \hat{\sigma}_s$, where $\hat{\sigma}_s$ is the robust standard deviation based on Median Absolute Deviation (MAD). (2) *Morphological top-hat / bottom-hat.* For structuring element sizes $k \in \{3, 5, 9, 17\}$, the white top-hat $d - \gamma(d)$ detects thin *near* structures (e.g., poles), and the black top-hat $\phi(d) - d$ detects thin *far* gaps (e.g., inter-foliage holes). Each residual is thresholded at 3.0 robust standard deviations.

The final coarse mask is obtained by taking the union over both detectors across all scales. The robust scale $\hat{\sigma}$ is estimated using the Median Absolute Deviation (MAD): $\hat{\sigma} = 1.4826 \cdot \text{median}(|r - \text{median}(r)|)$, with a fallback to the above-median tail when the MAD is zero.

Segment-aware expansion. We employ SAM2 [31] to partition the RGB image into distinct regions. For each segment S_i, we compute its support density with respect to the coarse structure mask $\mathcal{S}$ as

$$\rho_i = \frac{|S_i \cap \mathcal{S}|}{|S_i|}. \tag{16}$$

A segment is retained only if it satisfies the following criteria:

$$\rho_i \geq 0.3, \qquad |S_i \cap \mathcal{S}| \geq 5, \qquad |S_i| \leq 0.05HW, \tag{17}$$

thereby preserving only small-to-medium regions with sufficient structural evidence. The final mask is obtained by taking the union of all selected segments and intersecting it with the valid depth region.

B.2 Evaluation Protocol

Given the automatically generated fine-grained masks (Sec. B.1), we design a per-segment local evaluation protocol that measures how well a model reconstructs delicate geometric structures, independent of global alignment errors.

Per-segment shift with global scale. We first fit a single global scale s^* on all valid pixels using weighted least-squares:

$$s^* = \arg\min_s \sum_{p \in \text{valid}} w_p \left\| s \cdot \hat{z}_p - z_p^{\text{gt}} \right\|^2, \tag{18}$$

where $w_p = 1/z_p^{\text{gt}}$ for depth and $w_p = 1/\|\boldsymbol{x}_p^{\text{gt}}\|$ for 3D points. Then, for each SAM2 segment S_i (with $|S_i| \geq 10$ pixels) within the fine-grained mask, we fit a per-segment shift t_i while keeping the global scale fixed:

$$t_i = \arg\min_t \sum_{p \in S_i} w_p \left\| s^* \cdot \hat{z}_p + t - z_p^{\text{gt}} \right\|^2. \tag{19}$$

For 3D point maps, the shift $\boldsymbol{t}_i \in \mathbb{R}^3$ is a per-segment translation vector. The locally-aligned prediction is $\tilde{z}_p = s^* \hat{z}_p + t_i$ for depth, or $\tilde{\boldsymbol{x}}_p = s^* \hat{\boldsymbol{x}}_p + \boldsymbol{t}_i$ for points. We adopt this formulation because it avoids both the scale-collapse degeneracy of per-segment affine alignment and the sensitivity to field-of-view mismatch inherent in scale-only alignment, while remaining applicable to both depth maps and point maps in a unified manner.

Metrics and Aggregation. On the locally-aligned prediction within each segment, we compute AbsRel and $\delta_{0.01}$. Final results are reported as the average across all segments.

C Additional Quantitative Results

Evaluation results on individual datasets We report detailed per-dataset results for global metrics, local metrics, and boundary accuracy in Table C.1, Table C.2 and Table C.3, respectively.

C.1 Evaluation with SAWA-H and RelNormal

To complement the region-based local metrics, we evaluate SAWA-H and RelNormal [50] on all nine benchmarks. RelNormal measures discrepancies in the relative orientations of surface normals, while SAWA-H combines metrics with weights chosen to align their sensitivity with human judgments. Lower values indicate better performance for both metrics. Tables C.4 and C.5 report the per-dataset results, average scores, and average ranks across real-world and synthetic benchmarks.

For our ViT-L model, refinement with $K = 3$ reduces average SAWA-H from 1.048 to 1.019 and average RelNormal from 0.315 to 0.304 compared with $K = 0$. Scaling to the ViT-G backbone further reduces these average scores to 0.968 and 0.293, respectively, and achieves the best average rank under both metrics. These results provide complementary evidence that the improvements extend beyond the selected fine-detail regions used in our local evaluation.

C.2 Inference Efficiency

Profiling protocol. We report inference latency and peak GPU memory in Table C.6 at each method's native resolution. All measurements are obtained on a single NVIDIA A100 GPU with a forward batch size of 1, and every reported latency value is the average over 200 sampled inputs. For both Pixel-Perfect Depth and InfiniDepth, we do not employ the prompt-based metric-depth branch, and profile only the relative-depth prediction stage. For all other models, we profile the complete forward pass that produces both the point map and the depth map.

Inference precision. The FP16 setting follows each method's implementation: MoGe-2, Depth Pro, and our method cast model weights to FP16, whereas Depth Anything 3, InfiniDepth, and Pixel-Perfect Depth retain FP32 weights and use automatic mixed precision (AMP), following their released code. Thus, the FP16 columns reflect different precision policies across methods; for the AMP-based implementations, peak memory can exceed that of FP32 inference.

Dependence on scene structure. Our sparse 3D U-Net uses submanifold sparse convolutions to process only occupied voxels without densifying the grid. At a fixed input resolution, the input contains HW occupied voxels, while occupancy at coarser levels varies with scene geometry. Computation and activation memory therefore depend on active voxel counts across scales. Empirically, inference latency varies by approximately 5–8% across scenes of different complexity, indicating a modest dependence on scene structure.

D Additional Qualitative Results

We provide additional qualitative results on diverse scenes in Figure D.1 and D.2, highlighting fine-grained structures, depth discontinuities, and geometric details.

E Examples of Local Fine-Grained Mask

Figure E.3 illustrates several examples of our generated local fine-grained masks from the Synth4K dataset.

Method	NYUv2		KITTI		ETH3D		DIODE		HAMMER		iBims-1		Sintel		Spring		Synth4K		Avg.		
	Rel↓	δ_1 ↑	Rel↓	δ_1 ↑	Rel↓	δ_1 ↑	Rel↓	δ_1 ↑	Rel↓	δ_1 ↑	Rel↓	δ_1 ↑	Rel↓	δ_1 ↑	Rel↓	δ_1 ↑	Rel↓	δ_1 ↑	Rel↓	δ_1 ↑	Rank↓
Affine-invariant depth map																					
Depth Pro	3.67	98.2	5.12	96.8	4.97	96.4	4.66	95.6	3.30	99.6	3.23	98.3	15.8	80.1	21.7	70.4	10.4	86.6	8.10	91.3	8.28
UniDepth V2	2.96	98.6	3.85	98.1	2.94	98.5	4.04	96.5	2.48	99.6	2.64	98.4	13.4	83.2	20.1	75.4	10.4	86.9	6.98	92.8	5.44
DA 3	3.21	98.2	5.46	96.0	4.11	96.2	4.83	94.6	2.56	99.1	2.87	98.2	17.9	76.4	19.6	75.7	10.4	88.3	7.89	91.4	8.11
UniK3D	2.81	98.6	3.81	98.1	2.98	98.6	3.57	97.0	2.47	99.6	2.54	98.4	13.6	83.2	20.0	75.8	10.8	86.2	6.95	92.8	5.00
PPD	3.29	98.4	4.66	97.3	3.27	98.6	3.95	96.6	2.99	99.2	2.94	98.6	14.1	82.0	19.1	75.1	10.2	86.8	7.17	92.5	6.83
InfiniDepth	3.97	97.7	5.13	96.2	4.62	96.5	4.94	95.2	3.77	99.1	3.13	98.5	14.4	81.8	23.7	67.2	9.75	87.8	8.15	91.1	8.61
MoGe-2	2.88	98.6	3.74	98.1	2.80	99.1	3.12	97.6	2.72	99.5	2.37	98.9	13.4	82.3	15.5	81.0	7.27	91.3	5.98	94.1	3.56
Ours ViT-L Step 0	2.90	98.6	3.70	98.2	2.76	99.0	3.33	97.1	2.72	99.4	2.45	98.8	12.4	83.6	13.3	84.6	6.55	92.6	5.56	94.7	3.44
Ours ViT-L Step 3	2.87	98.6	3.68	98.2	2.71	99.0	3.34	97.0	2.68	99.5	2.42	98.8	12.3	83.8	13.2	84.8	6.49	92.7	5.52	94.7	2.78
Ours ViT-G Step 3	2.62	98.7	3.42	98.4	2.03	99.5	2.65	98.0	2.34	99.6	2.08	98.9	10.9	86.9	12.1	85.8	5.03	94.9	4.80	95.6	1.00
Affine-invariant point map																					
Depth Pro	4.36	97.9	9.15	90.7	7.73	94.1	6.28	93.6	5.31	98.8	4.34	97.4	19.6	74.5	25.0	66.0	17.8	75.4	11.1	87.6	7.67
UniDepth V2	3.66	98.4	4.75	98.0	4.37	98.3	7.45	92.4	3.45	99.4	4.05	98.1	18.0	76.5	25.1	66.9	20.8	69.4	10.2	88.6	5.83
DA 3	3.72	97.9	10.8	86.6	5.37	95.0	6.53	92.2	3.95	98.5	3.90	97.6	21.7	70.8	21.7	72.3	17.5	76.2	10.6	87.5	6.94
UniK3D	3.48	98.3	4.56	98.1	3.86	98.7	5.90	94.6	3.43	99.5	3.37	98.3	18.0	77.2	23.5	69.8	20.6	71.8	9.63	89.6	4.11
PPD	19.4	70.5	13.0	85.1	20.5	67.7	14.5	83.9	19.9	67.9	18.6	70.3	22.9	65.8	23.7	67.4	30.6	49.1	20.3	69.8	9.61
InfiniDepth	4.46	97.5	6.25	95.9	7.37	94.7	6.61	92.7	8.70	98.4	4.38	98.1	19.4	75.4	28.8	58.5	18.1	77.4	11.6	87.6	7.61
MoGe-2	3.31	98.4	6.38	96.5	3.92	98.6	4.99	95.2	4.14	99.3	3.71	98.5	17.2	77.0	23.6	65.3	11.4	85.1	8.73	90.4	4.17
Ours ViT-L Step 0	3.37	98.4	6.71	96.4	3.85	98.7	5.23	94.9	4.04	99.2	3.64	98.5	16.0	78.7	17.7	77.2	11.0	85.7	7.94	92.0	3.28
Ours ViT-L Step 3	3.35	98.5	6.67	96.5	3.78	98.7	5.23	94.8	3.98	99.2	3.62	98.5	15.9	78.8	17.6	77.4	11.1	85.6	7.91	92.0	2.72
Ours ViT-G Step 3	2.96	98.5	6.77	95.8	3.23	99.1	4.70	95.6	3.82	99.6	3.35	98.5	14.6	81.9	18.6	73.9	8.84	88.8	7.43	92.4	2.00
Metric depth map																					
Depth Pro	10.7	91.9	23.5	38.3	38.5	32.8	31.9	37.7	39.1	63.0	15.9	81.5	-	-	-	-	43.8	32.8	29.0	54.0	7.57
UniDepth V2	10.6	92.8	8.58	95.4	20.7	69.5	43.0	51.7	38.2	46.8	9.53	93.2	-	-	-	-	35.0	49.2	23.7	71.2	5.50
DA 3	7.74	94.4	29.9	20.6	15.0	79.7	19.0	64.3	19.1	74.7	10.8	86.7	-	-	-	-	20.3	70.4	17.4	70.1	4.79
UniK3D	9.88	92.8	9.54	95.8	14.9	83.7	18.1	73.0	30.5	58.3	9.35	93.6	-	-	-	-	33.1	47.4	17.9	77.8	4.21
PPD	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
InfiniDepth	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
MoGe-2	6.90	96.7	17.6	64.7	9.99	88.8	16.7	67.9	26.0	65.3	14.6	80.5	-	-	-	-	17.2	77.4	15.6	77.3	3.86
Ours ViT-L Step 0	7.10	96.4	16.4	71.5	12.1	87.9	16.0	73.4	23.3	72.7	12.7	84.2	-	-	-	-	15.7	80.7	14.7	81.0	3.14
Ours ViT-L Step 3	8.43	95.4	12.6	85.1	14.8	85.3	15.2	77.3	23.5	74.2	11.7	88.3	-	-	-	-	18.5	73.5	15.0	82.7	3.14
Ours ViT-G Step 3	10.5	94.1	10.7	85.9	14.8	82.8	16.9	74.3	32.1	69.9	11.0	87.9	-	-	-	-	14.7	83.8	15.8	82.7	3.64
Metric point map																					
Depth Pro	6.13	97.3	11.1	85.3	21.2	64.9	13.5	81.8	14.5	86.0	6.89	96.9	-	-	-	-	42.8	26.3	16.6	76.9	7.29
UniDepth V2	4.83	98.0	5.88	97.5	9.46	95.0	17.0	80.8	15.0	83.9	5.23	97.9	-	-	-	-	24.1	65.3	11.6	88.4	4.93
DA 3	4.89	97.5	12.6	80.4	8.83	94.9	9.00	89.7	10.8	90.1	5.25	97.2	-	-	-	-	23.0	63.9	10.6	87.7	5.21
UniK3D	4.64	97.9	5.86	97.6	7.35	97.5	8.48	93.4	14.0	85.5	4.58	98.0	-	-	-	-	22.4	70.9	9.61	91.5	3.07
PPD	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
InfiniDepth	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
MoGe-2	4.21	98.3	7.32	94.5	6.46	98.0	7.40	93.3	12.9	88.2	5.92	97.2	-	-	-	-	16.3	74.7	8.65	92.0	3.14
Ours ViT-L Step 0	4.32	98.4	7.20	95.2	7.92	95.4	7.80	92.4	13.7	88.2	5.51	97.2	-	-	-	-	15.7	76.2	8.88	91.8	3.21
Ours ViT-L Step 3	4.86	98.2	6.79	96.5	9.52	93.8	8.21	92.7	15.7	86.3	5.66	97.5	-	-	-	-	16.5	76.3	9.61	91.6	4.29
Ours ViT-G Step 3	6.08	97.9	7.12	96.5	9.38	93.7	8.78	93.5	20.5	80.0	5.80	98.6	-	-	-	-	15.3	77.9	10.4	91.2	4.43
Scale-invariant depth map																					
Depth Pro	4.42	97.6	5.47	96.2	7.54	94.1	7.06	91.9	4.36	98.9	4.13	97.4	23.9	68.8	25.1	63.8	12.9	83.2	10.5	88.0	9.22
UniDepth V2	3.65	98.4	4.24	98.0	3.23	98.9	5.91	94.1	3.48	99.1	3.45	98.1	23.1	65.3	25.0	65.0	11.1	86.9	9.23	89.3	5.72
DA 3	3.74	97.7	5.95	95.3	5.17	95.5	6.61	91.6	3.52	98.5	3.54	97.5	22.2	69.2	23.4	67.9	11.3	87.0	9.50	88.9	7.83
UniK3D	3.38	98.4	4.06	98.1	3.14	98.6	5.29	94.6	3.20	99.5	3.42	98.1	21.4	67.9	24.0	67.0	11.9	86.2	8.86	89.8	4.44
PPD	3.86	98.0	5.30	97.0	3.98	98.1	6.04	93.8	4.08	99.1	3.64	98.2	20.4	71.2	21.7	70.1	10.4	87.9	8.81	90.4	6.11
InfiniDepth	4.59	97.4	5.41	96.0	5.34	97.0	6.93	92.0	5.24	98.6	3.99	97.9	20.9	69.4	33.4	56.4	10.5	87.5	10.7	88.0	8.39
MoGe-2	3.40	98.2	4.08	98.0	3.47	98.7	5.33	94.3	3.88	99.4	3.26	98.4	19.8	71.3	19.4	73.4	8.22	89.6	7.86	91.3	3.56
Ours ViT-L Step 0	3.43	98.3	3.98	98.1	3.57	98.6	5.60	94.2	4.28	99.1	3.32	98.4	19.8	72.4	17.8	74.4	7.27	91.6	7.67	91.7	3.50
Ours ViT-L Step 3	3.41	98.3	3.98	98.1	3.53	98.6	5.62	94.2	4.42	99.1	3.30	98.4	19.8	72.5	17.8	74.4	7.17	91.7	7.67	91.7	3.33
Ours ViT-G Step 3	3.03	98.4	3.66	98.3	2.57	99.2	4.87	95.5	3.96	99.3	2.70	98.9	18.6	72.6	16.1	76.4	5.49	94.4	6.78	92.6	1.33
Scale-invariant point map																					
Depth Pro	5.04	97.7	10.6	95.1	11.2	92.0	8.53	91.6	6.82	98.7	5.84	97.1	26.9	64.0	28.1	60.5	24.2	57.6	14.1	83.8	7.44
UniDepth V2	5.60	98.3	5.41	98.0	6.58	97.2	12.0	93.4	4.19	99.2	5.56	98.1	27.2	56.3	32.0	46.0	26.3	53.4	13.9	82.2	6.50
DA 3	4.25	97.7	13.3	91.8	6.70	94.9	8.49	90.3	6.44	95.6	5.24	97.4	25.8	63.2	27.5	58.6	21.2	65.5	13.2	83.9	7.11
UniK3D	4.85	98.2	5.01	98.1	5.59	98.4	9.19	94.0	4.87	99.5	4.43	98.2	25.7	58.8	28.7	56.3	26.0	58.8	12.7	84.5	4.61
PPD	4.33	98.0	8.65	96.5	5.79	97.9	8.80	92.8	9.54	97.4	5.78	98.2	23.6	66.9	32.3	40.6	17.2	73.0	12.9	84.6	6.44
InfiniDepth	5.02	97.4	8.79	95.8	8.12	96.5	9.65	90.9	11.7	96.0	6.06	97.7	24.9	64.7	41.4	36.7	20.4	70.4	15.1	82.9	8.28
MoGe-2	3.88	98.2	8.07	97.6	5.62	98.2	8.16	93.6	9.35	97.9	5.48	98.4	23.1	66.5	30.1	44.0	15.0	74.9	12.1	85.5	4.44
Ours ViT-L Step 0	3.92	98.3	8.35	97.8	5.56	98.5	8.28	93.7	9.43	97.6	5.26	98.4	22.7	68.6	24.1	59.0	14.3	75.5	11.3	87.5	3.11
Ours ViT-L Step 3	3.90	98.3	8.34	97.8	5.45	98.6	8.31	93.7	9.42	97.6	5.28	98.4	22.7	68.5	24.2	58.8	14.4	75.7	11.3	87.5	3.06
Ours ViT-G Step 3	3.50	98.4	8.34	97.9	5.83	95.0	7.91	94.9	9.99	98.2	5.04	98.9	22.2	65.1	25.9	52.3	11.2	79.2	11.1	86.7	3.22

Table C.1: Comparison on global metrics across datasets and metric groups. DA3 refers to Depth Anything 3, and PPD refers to Pixel-Perfect Depth.

Table C.2: Comparison on local metrics across datasets and metric groups.

Method	Local fine-grained point map							Local fine-grained depth map						
	Spring		Synth4K		Avg.			Spring		Synth4K		Avg.		
	Rel↓	$\delta_{0.01}$ ↑	Rel↓	$\delta_{0.01}$ ↑	Rel↓	$\delta_{0.01}$ ↑	Rank↓	Rel↓	$\delta_{0.01}$ ↑	Rel↓	$\delta_{0.01}$ ↑	Rel↓	$\delta_{0.01}$ ↑	Rank↓
Depth Pro	2.12	56.1	3.90	34.3	3.01	45.2	4.50	3.72	52.0	4.57	41.7	4.15	46.9	5.50
UniDepth V2	2.68	51.7	6.52	33.4	4.60	42.6	8.00	4.41	48.4	6.78	42.1	5.60	45.3	7.25
DA 3	6.93	51.3	7.85	38.2	7.39	44.7	8.25	11.4	43.7	9.62	39.9	10.5	41.8	9.25
UniK3D	2.40	55.5	6.35	34.5	4.37	45.0	6.50	4.25	50.1	6.79	42.3	5.52	46.2	6.50
PPD	3.35	42.8	5.43	35.4	4.39	39.1	8.00	4.61	43.3	5.88	36.3	5.24	39.8	8.50
InfiniDepth	3.82	47.6	4.77	39.8	4.29	43.7	7.25	6.57	42.7	5.07	43.7	5.82	43.2	7.50
MoGe-2	2.53	49.5	3.85	43.7	3.19	46.6	5.50	3.57	49.9	4.04	44.2	3.81	47.1	4.25
Ours ViT-L Step 0	2.25	56.1	3.61	44.4	2.93	50.3	3.25	3.43	53.0	3.78	44.8	3.60	48.9	2.25
Ours ViT-L Step 3	2.12	60.6	3.46	51.2	2.79	55.9	1.50	3.65	56.4	3.87	48.7	3.76	52.5	2.50
Ours ViT-G Step 3	2.24	57.5	3.36	56.0	2.80	56.8	1.75	3.39	56.8	3.92	51.3	3.66	54.1	1.50

Table C.3: Boundary F1 score (radius=1) on iBims-1, HAMMER, Sintel, and Spring.

Method	iBims-1	HAMMER	Sintel	Spring	Avg. F1↑	Avg. Rk.↓
Depth Pro	14.5	5.07	34.8	10.9	16.3	4.75
UniDepth V2	11.3	4.66	35.9	6.26	14.5	7.00
DA 3	8.31	5.36	9.14	3.61	6.61	8.75
UniK3D	12.8	4.65	35.9	6.97	15.1	6.50
PPD	13.8	7.80	20.7	8.61	12.7	5.25
InfiniDepth	17.9	7.50	39.8	11.8	19.3	1.75
MoGe-2	17.6	4.54	32.0	8.22	15.6	7.00
Ours ViT-L Step 0	17.8	4.71	32.4	7.54	15.6	6.00
Ours ViT-L Step 3	18.3	6.08	32.5	6.95	16.0	4.75
Ours ViT-G Step 3	18.8	5.92	35.8	8.70	17.3	3.00

Table C.4: SAWA-H results across nine benchmarks. Avg. and Rank denote the mean score and mean per-dataset rank, respectively; lower is better for all columns.

Method	NYUv2	KITTI	ETH3D	DIODE	HAMMER	iBims-1	Sintel	Spring	Synth4K	Avg.↓	Rank↓
Depth Pro	1.302	1.277	0.895	0.931	0.868	0.883	1.318	1.909	1.176	1.173	7.78
UniDepth V2	1.208	1.217	0.782	0.841	0.804	0.860	1.257	1.761	1.103	1.093	4.83
DA3	1.234	1.267	1.006	0.828	0.936	0.953	1.554	1.845	1.561	1.243	7.33
UniK3D	1.200	1.213	0.767	0.842	0.806	0.844	1.300	1.770	1.091	1.093	4.50
PPD	1.807	1.547	2.370	2.341	2.132	1.787	2.374	2.687	2.858	2.212	10.00
InfiniDepth	1.291	1.270	0.869	0.952	0.946	0.905	1.417	2.092	1.091	1.204	7.94
MoGe-2	1.238	1.217	0.753	0.710	0.810	0.779	1.383	1.696	1.088	1.075	4.28
Ours ViT-L Step 0	1.240	1.215	0.755	0.716	0.807	0.787	1.234	1.644	1.031	1.048	3.83
Ours ViT-L Step 3	1.259	1.222	0.726	0.664	0.798	0.777	1.155	1.612	0.961	1.019	3.00
Ours ViT-G Step 3	1.240	1.204	0.670	0.607	0.773	0.741	1.060	1.573	0.841	0.968	1.50

Table C.5: RelNormal results across nine benchmarks. Avg. and Rank denote the mean score and mean per-dataset rank, respectively; lower is better for all columns.

Method	NYUv2	KITTI	ETH3D	DIODE	HAMMER	iBims-1	Sintel	Spring	Synth4K	Avg.↓	Rank↓
Depth Pro	0.420	0.442	0.206	0.227	0.212	0.246	0.377	0.571	0.378	0.342	6.17
UniDepth V2	0.399	0.439	0.212	0.227	0.214	0.252	0.363	0.573	0.361	0.338	5.61
DA3	0.406	0.433	0.311	0.205	0.278	0.283	0.424	0.605	0.566	0.390	6.89
UniK3D	0.402	0.438	0.207	0.236	0.216	0.249	0.381	0.578	0.356	0.340	5.89
PPD	0.695	0.589	1.006	0.987	0.874	0.719	0.895	1.054	1.238	0.895	10.00
InfiniDepth	0.416	0.444	0.230	0.255	0.227	0.267	0.421	0.613	0.360	0.359	7.67
MoGe-2	0.418	0.435	0.184	0.160	0.208	0.226	0.375	0.544	0.342	0.321	3.50
Ours ViT-L Step 0	0.418	0.436	0.188	0.157	0.211	0.229	0.340	0.542	0.317	0.315	3.67
Ours ViT-L Step 3	0.430	0.440	0.178	0.132	0.208	0.226	0.305	0.530	0.286	0.304	3.28
Ours ViT-G Step 3	0.430	0.436	0.171	0.124	0.209	0.220	0.280	0.521	0.249	0.293	2.33

Table C.6: Inference latency and peak GPU memory measured on a single NVIDIA A100 GPU for single-frame inference at native resolutions.

Method	Native Resolution	Latency (ms)		Peak memory (GB)	
		FP16	FP32	FP16	FP32
MoGe-2 (ViT-L)	484^2	29	82	0.98	2.32
	700^2	39	157	1.23	2.69
	840^2	55	238	1.42	2.98
Depth Anything 3	378×504	152	305	8.41	6.78
	630×840	213	737	8.77	7.37
Depth Pro	1536^2	142	910	3.81	8.06
InfiniDepth	512×672	133	343	1.88	1.48
	768×1024	258	724	1.98	1.68
Pixel-Perfect Depth	512×672	275	1054	4.27	3.42
	768×1024	598	3123	4.50	3.75
Ours (ViT-L) 3-step	484^2	116	197	1.56	2.74
	700^2	168	345	1.99	3.40
	840^2	214	491	2.33	3.91

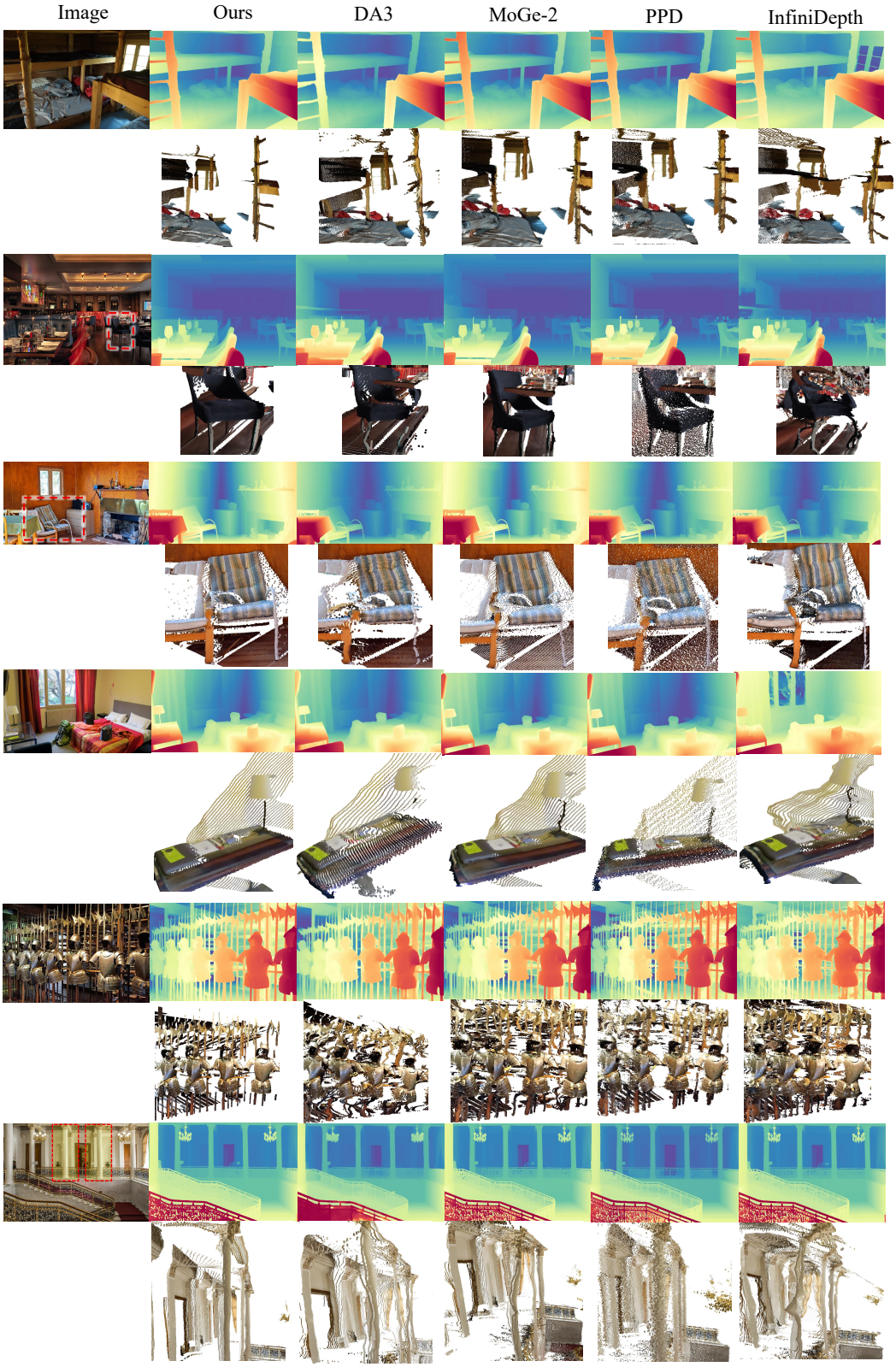

Figure D.1: **Additional qualitative results and comparisons.** Our method preserves fine structures and sharp boundaries across diverse scenes.

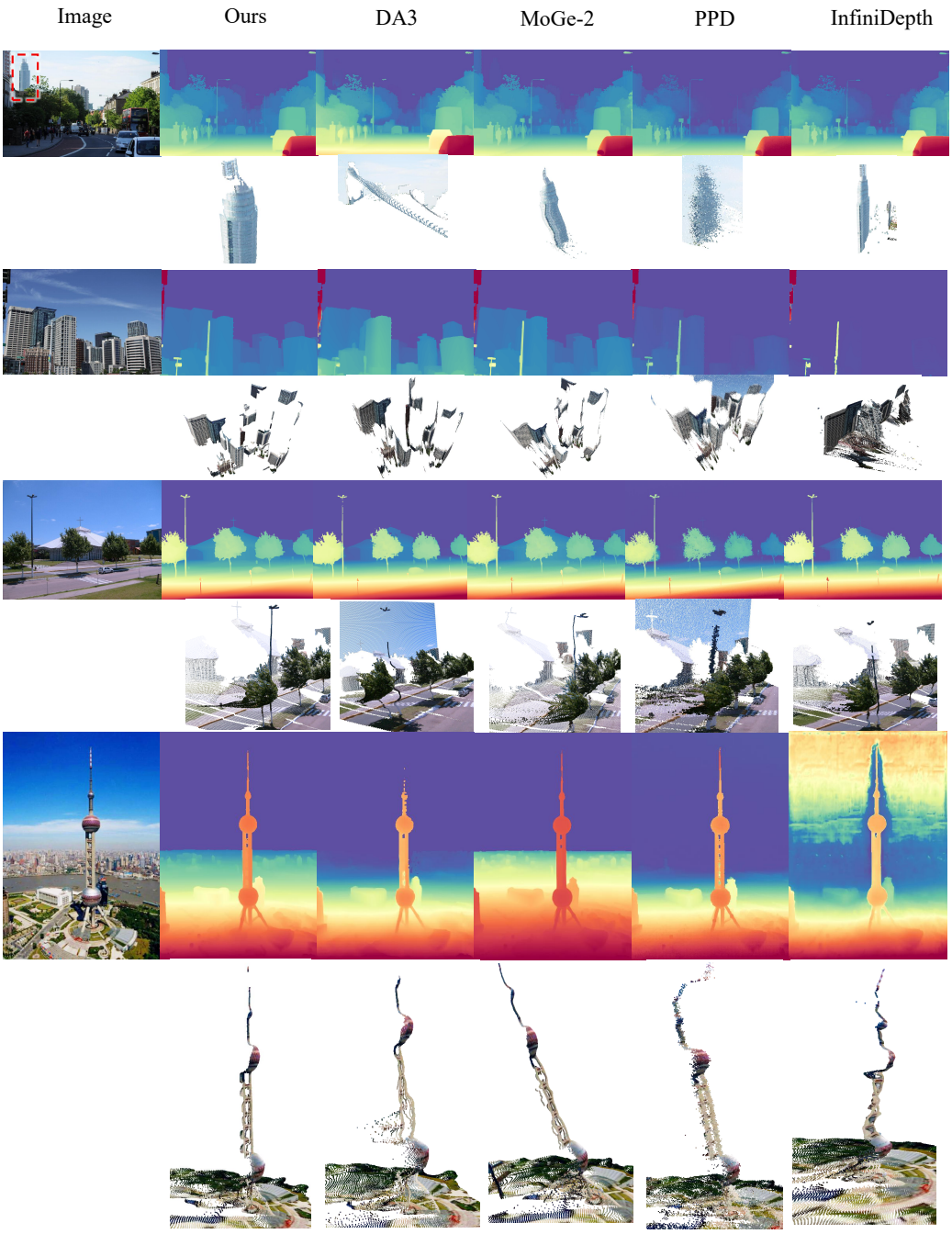

Figure D.2: **Additional qualitative results.** More examples demonstrating high-fidelity geometric reconstruction and robustness to challenging regions.

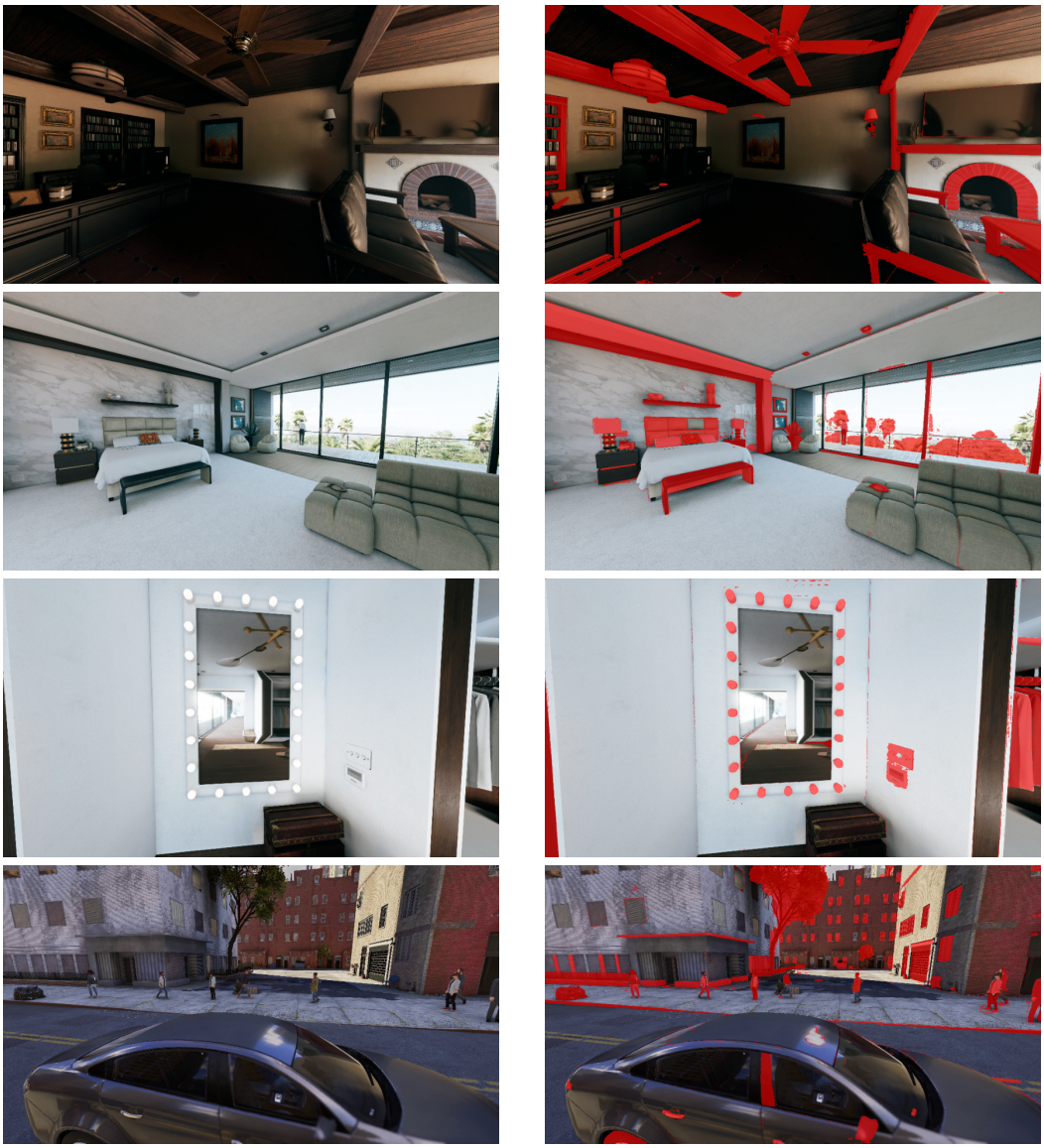

Figure E.3: **Local fine-grained masks.** For each example, we show the input image (left) and the corresponding local fine-grained mask overlaid in red (right). These masks highlight thin structures and high-frequency regions used for evaluation.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The main contributions are clearly stated and well supported by the results, with assumptions and scope appropriately reflected in the introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: The limitations of the work have been discussed in the conclusion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A].

Justification: The paper does not include theoretical results or proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: Code and models of this paper have been released at `https://github.com/microsoft/MoGe` and the results are reproducible.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes].

Justification: The code and data for this work have been released at `https://github.com/microsoft/MoGe`.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes].

   Justification: Necessary details are revealed in the main paper and the supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes].

   Justification: The experiments are benchmark evaluations with fixed test sets and deterministic metrics, so statistical variation is not applicable.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes].

   Justification: The paper specifies the type of compute used and the approximate runtime per experiment. Additional information on compute requirements is provided in the implementation or appendix section.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: The research complies with the NeurIPS Code of Ethics. All experiments use publicly available datasets with appropriate licenses. No human subjects, sensitive data, or foreseeable misuse risks are involved.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A].

    Justification: Our work focuses on monocular depth and 3D geometry estimation as a fundamental perception task. It does not involve decision-making, human-centric prediction, or system-level actions, and is not directly deployed in real-world decision pipelines. Therefore, we do not identify specific positive or negative societal impacts.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A].

    Justification: The paper does not pose such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes].

    Justification: All datasets and codebases used in the paper are properly cited with their official papers and licenses. Only publicly released assets with clear terms of use were used. Licenses (e.g., CC BY-NC, GPL, MIT) are respected and referenced in the paper or supplementary material.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes].

    Justification: The paper includes a complete description of the new asset, including its structure, intended use, and limitations. The full anonymized version of the asset will be made available in a public repository upon paper acceptance.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A].

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A].

    Justification: The paper does not involve research with human subjects and therefore does not require IRB approval.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A].

    Justification: LLMs were not used as part of the core methodology or experiments. Any LLM usage was limited to writing assistance and did not impact the scientific content.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.