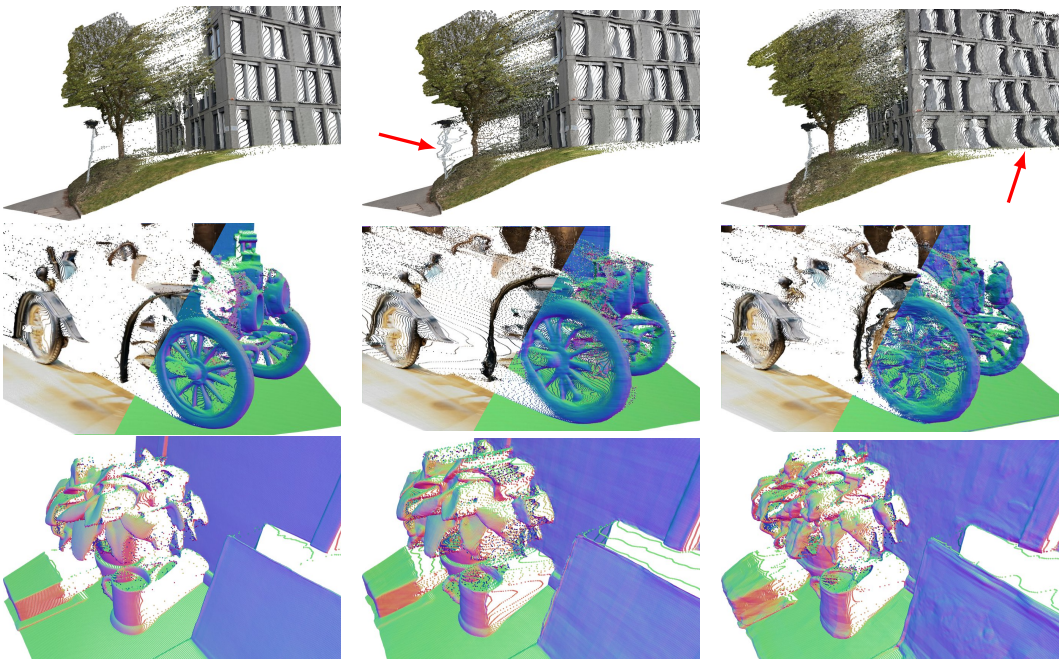
SurGe: Improved Surface Geometry in Point Maps

Karim Knaebel[1] **Gonzalo Martin Garcia**[1] **Christian Schmidt**[1] **Ilya Fradlin**[1]
Lucas Nunes[1] **Daan de Geus**[2] **Bastian Leibe**[1]

[1]RWTH Aachen University [2]Eindhoven University of Technology
https://vision.rwth-aachen.de/surge

Abstract

Recent feedforward 3D reconstruction methods predict point maps and estimate global 3D geometry remarkably well. However, their predictions still exhibit inaccurate local surface geometry, which is clearly visible qualitatively but only weakly reflected in common metrics. To make these errors more explicit in evaluation, we introduce a point map normal metric that evaluates the local surface orientation induced by neighboring 3D predictions. To reduce these errors, we propose two complementary components: a point gradient matching loss that supervises depth-normalized 3D finite differences, and a Neighborhood Attention Decoder (NAD) that progressively upsamples features and uses Neighborhood Attention for local feature mixing. Across eight zero-shot monocular geometry benchmarks, our model, SurGe, achieves the best average rank for global point map AbsRel and consistently improves local point map and point map normal evaluations.



(a) **SurGe** (b) MoGe-2 [58] (c) InfiniDepth [72]

Figure 1: **Qualitative state-of-the-art comparison.** SurGe predicts noticeably cleaner point maps.

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

1 Introduction

Monocular geometry estimation seeks to recover dense 3D scene structure from a single image. Recent feedforward models [37, 57, 58] predict a point map, assigning each pixel a 3D point. This representation was initially proposed for two-view geometry by DUSt3R [59]; it was then brought to monocular geometry by MoGe [57], and was extended to the many-view setting by VGGT [53]. Since then, the area has developed rapidly, with a growing number of follow-up models building on the same general paradigm [24, 28, 61]. While these models perform strongly overall and generalize well to in-the-wild images, their predictions still contain visible local 3D artifacts despite having plausible coarse scene geometry (Fig. 1). In particular, small deviations in the relative positions of nearby predicted points can distort local shape and orientation. We refer to the relative 3D structure of nearby points as *local surface geometry* and focus on improving it in this work.

This weakness is especially visible on thin structures. Thin foreground elements such as streetlamps, street signs, chair legs, or faucets often emerge bent or oscillatory in 3D, and the distortion typically becomes more severe as the structure gets thinner and the background is farther away. We find that this is not a 2D edge-localization problem but a 3D shape problem: the point prediction can look plausible when viewed as a depth map while neighboring 3D points form inconsistent surface patches.

We hypothesize that errors in local surface geometry have received limited attention because standard point map metrics evaluate local surface geometry only indirectly. They measure whether predicted points have accurate 3D positions on average after global alignment. However, neighboring points can fail to form a coherent surface even when the individual point errors are small on average. Surface ripples, blockiness, and distortions can arise from relatively small point displacements, so their contribution to an average point-position error can be weak compared to ordinary placement errors. The local point map metric of Wang et al. [57] makes this evaluation more fine-grained by aligning segmented instances separately, but after alignment it still averages pointwise residuals. We therefore complement pointwise metrics with a point map normal metric computed from neighboring point differences, which directly evaluates the local surface orientation induced by the predicted point map. In a normal-based metric, artifacts are measured through the changes they induce in local surface orientation, rather than through the magnitude of the underlying point-position residuals; Fig. 2 illustrates this with low- and high-frequency perturbations of the ground-truth surface.

Equipped with a metric that better tracks local structure, we revisit the loss formulation of recent models. Like the global point map metrics, commonly used global point losses [57, 59] measure average point positioning error and are dominated by global geometry. As a result, they require additional losses to enforce local consistency and to suppress oscillations and surface irregularities. Common surface losses include terms on normals estimated from point maps and edge angles [57, 58] as well as several gradient matching variants [27, 40]. However, none of these existing losses provides sufficiently strong surface supervision for point maps. Interestingly, preliminary experiments showed that the log-depth gradient matching loss originally proposed for monocular depth [27] *does* improve local surface geometry reliably, but tends to harm global geometry as it is not compatible with the point map formulation. In this work, we adapt this loss to enable its direct application to point maps, while preserving its pairwise scale invariance.

We find that this improved supervision helps, but does not remove the main failure mode. Therefore, we also investigate the architectural design of the decoder. Recent point map models typically pair a ViT backbone with a convolutional decoder such as MoGe's ConvStack [57] or a DPT head [39]. We find that even with improved surface losses, these decoders still struggle to recover the 3D shape of thin structures in front of distant or complex backgrounds. We hypothesize that convolution-based decoders struggle here because these regions require reconstructing a high-frequency, high-amplitude signal in image space, which is difficult to represent in fixed convolutional kernels. Scaling the convolutional decoder helps, but quickly reaches diminishing returns. A ViT decoder such as in π^3 [61] avoids fixed filters, but it processes all features at a single low-resolution scale determined by the patch size, which leads to patch-aligned artifacts.

Our solution is to keep the progressive multiscale decoding structure of convolution-based decoders, while replacing convolutions with blocks based on Neighborhood Attention [18]. This yields a decoder that can selectively aggregate local evidence unlike convolutional decoders, without incurring the cost of full-resolution self-attention and without producing patch-aligned artifacts as a pure ViT decoder. In practice, it produces more faithful thin structures and more stable local surface geometry than

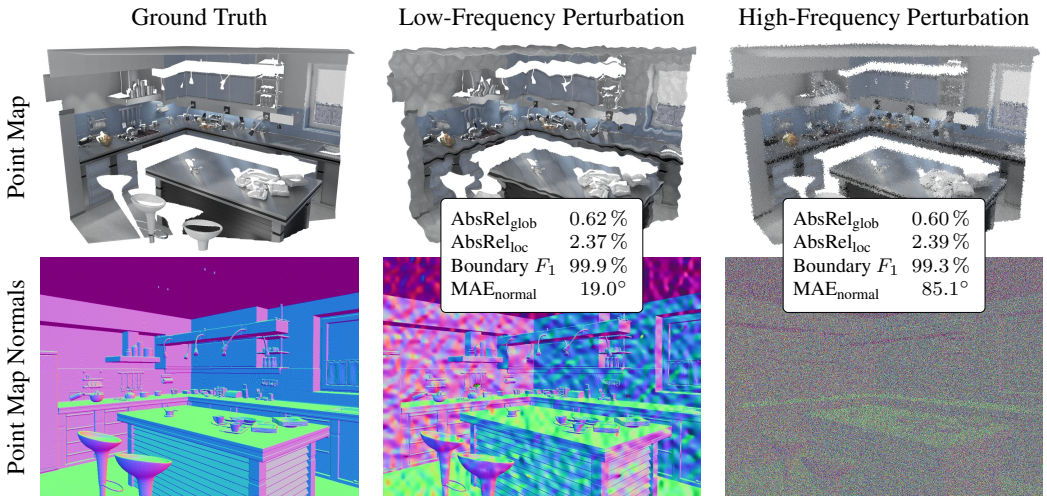


Figure 2: **Pointwise metrics only weakly capture local surface geometry.** We add low- and high-frequency perturbations to the same ground-truth point map. AbsRel$_{glob}$ and AbsRel$_{loc}$ average pointwise position errors, giving nearly identical scores even though the high-frequency perturbation yields much less coherent local surface geometry. Boundary F_1 [5] remains nearly perfect for both perturbations: it measures depth-discontinuity localization and sharpness rather than surface geometry away from those discontinuities. MAE$_{normal}$ instead compares point map normals induced by neighboring point differences, and therefore reflects this degradation.

convolutional decoders. Combined with our point gradient matching loss, the resulting model sets a new state of the art in the local point map and point map normal evaluations, while also achieving the best results in the global point map evaluation on several common zero-shot benchmarks.

Our contributions are as follows:

1. An evaluation metric based on point map normals that better reflects local surface quality than standard point map metrics (MAE$_{normal}$, Eq. (4)).
2. A scale-invariant point gradient matching loss, inspired by log-depth gradient matching [27], for supervising local surface structure in point maps ($\mathcal{L}_{\text{pgm}}$, Eq. (3)).
3. A decoder based on Neighborhood Attention for dense point map prediction that improves thin structures and local surface geometry (NAD, Sec. 3.1).

2 Related Work

2.1 Feedforward Geometry Estimation

Geometry estimation is a foundational task for many applications, including robotics, autonomous driving, and virtual reality. Many state-of-the-art architectures combine a plain ViT initialized from DINOv2 [35] with a convolutional decoder. This decoder is often implemented as a DPT head [5, 39, 69, 70], a ConvStack [57, 58], or a similar convolutional multi-resolution architecture [37].

Notable exceptions to this are InfiniDepth [72], which uses a query-based decoder that decodes depth for individual continuous image coordinates, and Pixel-Perfect Depth (PPD) [67], which uses a decoder inspired by the DiT architecture [36]. InfiniDepth and PPD predict affine-invariant depth in log space instead of point maps. Unprojecting affine-invariant log-depth to 3D geometry requires a reference point map; both models use MoGe-2 to estimate this reference point map [67, 72]. While InfiniDepth and PPD aim to improve fine-grained geometry estimation, we observe blocky surface artifacts for InfiniDepth and severe surface noise for PPD. We show a quantitative comparison in Sec. 4 and qualitative examples in Sec. G.

Current multi-view methods also achieve impressive results in the monocular setting. Their architecture follows a similar scheme as monocular models, combining a ViT encoder with an inter-frame

feature fusion module and a per-frame decoder. DUSt3R [59] uses a ViT pretrained with cross-view completion [63, 64] and a DPT head for point map predictions. It predicts point maps from two input views in a shared coordinate system. Follow-up works employ global attention over all frames, enabling faster processing of large image sets [24, 28, 53, 61, 68]. Similar to the previous two-view and monocular methods, these methods often use a DPT head to regress one point map per input image [24, 28, 53, 68]. A notable exception is π^3 [61], which combines a plain Transformer [52] with a final pixel shuffle [44] for point predictions.

The impressive performance of these methods illustrates the tremendous progress in geometry estimation thanks to strong backbones, large-scale training, and improved supervision. However, we still observe poor predictions for local surface geometry, especially around thin structures (see Fig. 1). Compared to previous methods, our Neighborhood Attention Decoder (NAD) visibly improves in this regard. Unlike DPT heads and ConvStack decoders, it is not restricted to fixed convolutional kernels, and unlike ViT decoders, it is not plagued by patching artifacts.

2.2 Evaluation of Local Surface Geometry

Point map metrics used in recent work remain pointwise after alignment: global affine-invariant AbsRel evaluates all valid pixels, while the local point map metric of Wang et al. [57] evaluates instance-mask regions after separate local alignment. This improves sensitivity to errors inside individual object masks, but still does not measure whether neighboring predictions form a coherent surface. Figure 2 illustrates this limitation and motivates the point map normal metric $\text{MAE}_{\text{normal}}$ in Eq. (4). It applies mean angular error, a standard metric in surface-normal benchmarks [2], to normals induced by predicted and ground-truth point maps. Other standard summaries, median angular error and angular accuracy below 11.25° (Tabs. I and J), support the same overall conclusions as $\text{MAE}_{\text{normal}}$ (Tab. 2), so we focus on mean angular error in the main comparisons. RelNormal [66] assesses curvature distortions by comparing relative angles between sampled normal pairs across scales, rather than per-pixel orientations. Its results likewise support our overall conclusions (Tab. K).

2.3 Point Map Supervision

DUSt3R [59] popularized the point map representation for 3D geometry estimation and introduced the corresponding point map loss as direct regression of aligned or normalized 3D points. Many follow-up works adapt this point map loss with different output parameterizations [24, 28, 53]. The residuals, however, remain pointwise and do not directly constrain whether neighboring predictions form a coherent local surface. Prior methods therefore add surface losses: point normals and edge angle losses supervise local orientation but do not constrain displacement magnitude [28, 57, 58]; other depth and point map methods compare finite differences after one shared alignment or normalization of the whole prediction [40, 53], but this does not reward local coherence in globally misplaced regions. Log-depth gradient matching compares finite differences of log depth, yielding a pairwise scale-invariant residual [27]. In Sec. 3.2, we adapt this pairwise idea to vector-valued point maps.

2.4 Neighborhood Attention

In Neighborhood Attention (NA) [18], each query only attends to its local neighborhood. It can be viewed as a way to introduce inductive bias into attention or to speed up computations due to sparsity [17–20]. To introduce inductive bias, NA can be used to implement hierarchical feature encoders [17, 18]. As a sparse attention variant, it has been used to replace full attention in isometric Transformer architectures such as video diffusion models [20].

Dilated NA has been used with encoder-decoder architectures as an efficient drop-in for regular attention, for example, in medical imaging [10, 42] and image restoration [29]. NAF [7] proposes a decoder that utilizes NA to cross-attend between high-resolution image features of a small CNN and a low-resolution feature map of a vision foundation model in order to compute high-resolution features. Compared to these works, our Neighborhood Attention Decoder (NAD, Sec. 3.1) neither cross-attends between resolutions nor uses dilated NA; instead, it uses plain NA to make attention-based mixing practical in a multi-stage dense geometry decoder, where high-resolution stages make global self-attention prohibitive. To our knowledge, we are the first to demonstrate the effectiveness of NA as the core mixing mechanism in this setting.

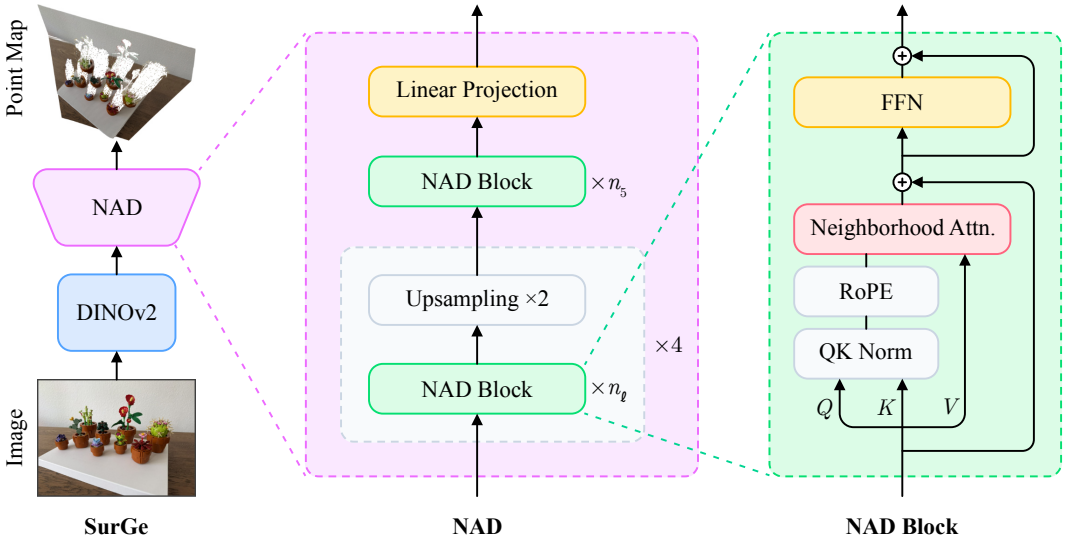


Figure 3: **SurGe architecture overview.** SurGe combines a DINOv2 [35] encoder with our Neighborhood Attention Decoder (NAD). NAD upsamples encoder features through a sequence of stages $\ell \in \{1, \ldots, 5\}$, each built from n_ℓ NAD blocks. Compared to standard ViT [11] blocks, NAD blocks replace global self-attention with Neighborhood Attention [18], use window-matched RoPE [46] and only QK normalization [8] instead of pre-attention and pre-feedforward LayerNorm [1].

3 Method

Our model, SurGe, predicts a dense point map $\hat{\mathbf{P}} \in \mathbb{R}^{H \times W \times 3}$, mapping every pixel of a single image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$ to its corresponding 3D point. It combines a DINOv2 [35]-initialized ViT [11] encoder with a Neighborhood Attention Decoder (NAD, Sec. 3.1). We train it with global and local point map losses and our proposed point gradient matching loss ($\mathcal{L}_{\text{pgm}}$, Eq. (3)).

3.1 Architecture

Figure 3 visualizes the architecture of SurGe. Following recent feedforward 3D reconstruction methods [53, 57], we use a ViT-Large backbone [11] initialized from DINOv2 [35]. Given $\mathbf{I}$, the encoder produces a low-resolution token grid $\mathbf{Z} \in \mathbb{R}^{\frac{H}{16} \times \frac{W}{16} \times C}$. The decoder must then process and upsample this grid to produce per-pixel point predictions $\hat{\mathbf{P}}$. Recent dense prediction decoders do so with a progressive multi-resolution design [37, 39, 57], which we keep. The remaining design choice is how to mix features inside each stage of this decoder. Existing methods struggle with images containing thin structures, where foreground and background evidence are placed in the same small image neighborhood. We hypothesize that feature-dependent attention can separate this evidence more easily than spatially shared convolutional kernels. Therefore, SurGe adopts attention in its decoder. In practice, we use Neighborhood Attention rather than self-attention to obtain content-dependent local mixing at linear cost. Notably, as this progressive decoder processes features at multiple resolutions, it prevents patch-aligned artifacts, unlike single-resolution ViT decoders.

We refer to the resulting decoder as Neighborhood Attention Decoder (NAD). Starting from the encoder token grid $\mathbf{Z}$, NAD consists of five stages that produce the predicted point map $\hat{\mathbf{P}}$, with stages 1 through 4 upsampling by a factor of 2×. Following Wang et al. [57], the final output projection predicts (ξ, η, ρ) and maps it to points as $(\xi e^\rho, \eta e^\rho, e^\rho)$. Each stage ℓ operates on feature map $\mathbf{X}_\ell \in \mathbb{R}^{\frac{H}{2^{5-\ell}} \times \frac{W}{2^{5-\ell}} \times C_\ell}$ and contains n_ℓ NAD blocks. For SurGe, $C_1, \ldots, C_5 = (1024, 512, 256, 128, 64)$ and $n_1 = \cdots = n_5 = 3$. For $\ell < 5$, the NAD blocks are followed by an upsampling module that doubles the spatial resolution and halves the channel dimension. We implement this module as a transposed 2×2 convolution with stride 2, followed by a 3×3 convolution [34].

Each NAD block is a Transformer-style [52] residual block with a Neighborhood Attention layer and a pointwise FFN. It uses window-matched RoPE [46] on queries and keys, and omits the usual

pre-attention and pre-FFN LayerNorm [1] layers. Across all stages, the attention window is $k = 9$ and the head dimension is $d_h = 64$; at stage ℓ, the FFN hidden dimension is $4C_\ell$. Removing LayerNorm empirically improves accuracy; we hypothesize that dense point regression benefits when activations can retain magnitude cues related to scene geometry. To stabilize attention without pre-attention normalization, we use QK normalization [8, 74]. Additional details on stage-wise positional embeddings, the RoPE temperature, and normalization choices are given in Sec. B.

3.2 Loss

Following Wang et al. [57], we use the global affine-invariant point map loss $\mathcal{L}_{\text{glob}}$ with ROE [57] alignment, as well as local patch losses $\mathcal{L}_{\text{loc},4}$, $\mathcal{L}_{\text{loc},16}$, and $\mathcal{L}_{\text{loc},64}$ with diameters $\frac{1}{4}$, $\frac{1}{16}$, and $\frac{1}{64}$ of the image diagonal. These losses supervise point positions under global and local affine-invariant alignments, but their residuals are still pointwise after alignment. They therefore only weakly constrain whether neighboring pixels form a locally consistent 3D surface. We address this by adding a surface loss on neighboring 3D point differences.

Surface supervision. Existing surface losses make different trade-offs. Losses on normals estimated from point maps and on edge angles [57, 58] compare local surface orientation, but discard the magnitude of the underlying 3D displacements. Some prior depth and point map methods compare finite differences after one shared alignment or normalization of the whole prediction [28, 40, 53]. This asks each local pair to match its target finite difference under the same aligned frame, even when that neighborhood is locally coherent but misplaced relative to the whole scene. For local surface supervision, we instead want the comparison to be defined by the neighboring pair itself.

Li and Snavely [27] define such a pairwise comparison for a ground-truth depth map $\mathbf{D}$ and predicted depth map $\hat{\mathbf{D}}$ through the log-depth gradient matching loss:

$$\mathcal{L}_{\text{gm}}(\hat{\mathbf{D}}, \mathbf{D}) = \frac{1}{2|\mathcal{V}_x|} \sum_{(i,j)\in\mathcal{V}_x} \left|\Delta_x \log \hat{D}_{ij} - \Delta_x \log D_{ij}\right| + \frac{1}{2|\mathcal{V}_y|} \sum_{(i,j)\in\mathcal{V}_y} \left|\Delta_y \log \hat{D}_{ij} - \Delta_y \log D_{ij}\right|, \quad (1)$$

where Δ_x, Δ_y are forward finite differences, and $\mathcal{V}_x, \mathcal{V}_y$ contain annotated pixels whose horizontal or vertical forward neighbor is also annotated. The logarithm makes each residual depend on a neighboring pair's depth ratio rather than a global prediction scale, but this construction is tied to positive scalar depth. Point maps are vector-valued and coordinate-wise logarithms or ratios are not geometrically meaningful. Restricting the loss to the z coordinate, as in MapAnything [24], preserves this locality for depth, but its corrections are not expressed as 3D point differences, making them poorly aligned with the point map losses.

We therefore extend the pairwise comparison to *point maps* by matching depth-normalized 3D finite differences:

$$\widetilde{\Delta}_x \mathbf{Q}_{ij} = \frac{\Delta_x \mathbf{Q}_{ij}}{\min([\mathbf{Q}_{ij}]_z, [\mathbf{Q}_{i,j+1}]_z)}, \quad (2)$$

where $\mathbf{Q}$ is a point map, $[\mathbf{Q}_{ij}]_z$ denotes the z coordinate of $\mathbf{Q}_{ij}$, and $\widetilde{\Delta}_y$ is defined analogously for vertical pairs. We normalize by z since our decoder represents predicted points as $(\xi e^\rho, \eta e^\rho, e^\rho)$, where $z = e^\rho$ is the scalar scale factor shared by all three coordinates. For a neighboring pair, we use the nearer endpoint depth as a conservative local scale.

Our point gradient matching loss averages Euclidean distances between corresponding predicted and ground-truth normalized finite differences over the same valid-pair sets:

$$\mathcal{L}_{\text{pgm}}(\hat{\mathbf{P}}, \mathbf{P}) = \frac{1}{2|\mathcal{V}_x|} \sum_{(i,j)\in\mathcal{V}_x} \left\|\widetilde{\Delta}_x \hat{\mathbf{P}}_{ij} - \widetilde{\Delta}_x \mathbf{P}_{ij}\right\|_2 + \frac{1}{2|\mathcal{V}_y|} \sum_{(i,j)\in\mathcal{V}_y} \left\|\widetilde{\Delta}_y \hat{\mathbf{P}}_{ij} - \widetilde{\Delta}_y \mathbf{P}_{ij}\right\|_2. \quad (3)$$

A Python-like version is given in Pseudocode A.

Our proposed $\mathcal{L}_{\text{pgm}}$ combines the following useful properties: the residual is computed on full 3D displacements, so it supervises both local direction and magnitude, while the depth normalization makes the comparison scale-invariant for each neighboring pair.

In practice, we evaluate $\mathcal{L}_{\text{pgm}}$ only on pairs whose two endpoints are annotated and omit pairs near occlusion boundaries. At such boundaries, adjacent pixels often belong to different surfaces and the

exact ground-truth transition depends on dataset-specific edge conventions and resampling. Masking these pairs removes largely irreducible residuals and slightly improves training.

We choose the relative weight of $\mathcal{L}_{\text{pgm}}$ empirically, finding that a coefficient of 10 gives a good balance with the global point loss. Our full dense-label objective is:

$$\mathcal{L} = \mathcal{L}_{\text{glob}} + \mathcal{L}_{\text{loc},4} + \mathcal{L}_{\text{loc},16} + \mathcal{L}_{\text{loc},64} + 10\mathcal{L}_{\text{pgm}}.$$

For noisier or sparse annotations, we omit high-frequency terms according to label quality; see Sec. C.

4 Experiments

4.1 Experimental Setup

Training. Unless otherwise stated, we use the DINOv2 [35]-Large encoder and NAD described in Sec. 3.1. We train on a balanced mix of twenty synthetic and real datasets covering outdoor, indoor, in-the-wild, driving, and object-centric domains [3, 9, 14, 16, 21, 22, 26, 27, 33, 38, 41, 47, 48, 55, 56, 60, 65, 71, 73, 76, 78]. The dataset weights are listed in Sec. D. Following common practice [5, 57], we adapt supervision to label quality: for synthetic labels we use the full objective in Sec. 3, for SfM labels we omit $\mathcal{L}_{\text{loc},64}$ and $\mathcal{L}_{\text{pgm}}$, and for LiDAR labels we keep only $\mathcal{L}_{\text{glob}}$ and $\mathcal{L}_{\text{loc},4}$. We optimize with AdamW [30] for 120K steps at total batch size 128, using peak learning rates 3×10^{-4} for the decoder and 3×10^{-5} for the backbone, and a reciprocal square root schedule [74] with 1K warmup steps and a 10% cooldown. The first 80% of training uses a low-resolution budget of 1024 encoder tokens with a fixed image area and sampled target aspect ratios. The final 20% uses higher resolutions with encoder-token budgets between 1024 and 2802, and the final 10% samples only synthetic data. More details are given in Sec. C. Every ablation follows the final model's full training protocol and changes only the component under study. This full-scale protocol is deliberate: several effects observed in smaller proxy runs did not persist at the final scale. A full run takes roughly 12 hours on 16 GH200 GPUs for the low-resolution phase and another 5 hours on 32 GH200 GPUs for the high-resolution phase.

Evaluation. We evaluate zero-shot on eight common monocular geometry benchmarks: NYUv2 [45], KITTI [50], ETH3D [43], iBims-1 [25], GSO [12], Sintel [6], DDAD [15], and DIODE [51]. Following MoGe [57], we report ROE-aligned affine-invariant point map AbsRel metrics in global and instance-wise local forms. Let $\mathcal{V}$ be the set of all valid pixels and let $\hat{\mathbf{P}}^{\star}_{ij} = s\hat{\mathbf{P}}_{ij} + \mathbf{t}$ denote aligned predictions, where s is a shared scale factor and $\mathbf{t}$ is a 3D translation. The global absolute relative error is defined as

$$\text{AbsRel}_{\text{glob}} = \frac{1}{|\mathcal{V}|} \sum_{(i,j)\in\mathcal{V}} \frac{\left\|\hat{\mathbf{P}}^{\star}_{ij} - \mathbf{P}_{ij}\right\|_2}{\|\mathbf{P}_{ij}\|_2}.$$

For $\text{AbsRel}_{\text{loc}}$, we evaluate each instance $r \in \mathcal{R}$ with segmentation region Ω_r over $\mathcal{V}_r = \mathcal{V} \cap \Omega_r$. We estimate a separate local scale-and-translation alignment $\hat{\mathbf{P}}^{\star,r}_{ij} = s_r\hat{\mathbf{P}}_{ij} + \mathbf{t}_r$, and normalize residuals by the ground-truth instance diameter $d_r = \left\|\max_{(i,j)\in\mathcal{V}_r} \mathbf{P}_{ij} - \min_{(i,j)\in\mathcal{V}_r} \mathbf{P}_{ij}\right\|_\infty$, where max and min are taken component-wise:

$$\text{AbsRel}_{\text{loc}} = \frac{1}{|\mathcal{R}|} \sum_{r\in\mathcal{R}} \frac{1}{|\mathcal{V}_r|} \sum_{(i,j)\in\mathcal{V}_r} \frac{\left\|\hat{\mathbf{P}}^{\star,r}_{ij} - \mathbf{P}_{ij}\right\|_2}{d_r}.$$

For our proposed point map normal mean angular error ($\text{MAE}_{\text{normal}}$), we first form four local normals around each pixel from cross products of adjacent point map differences. The valid local normals are averaged, giving normal maps $\mathbf{N}_{ij}$ and $\hat{\mathbf{N}}_{ij}$ for the ground truth and prediction. We report

$$\text{MAE}_{\text{normal}} = \frac{1}{|\mathcal{V}_N|} \sum_{(i,j)\in\mathcal{V}_N} \angle(\hat{\mathbf{N}}_{ij}, \mathbf{N}_{ij}) \tag{4}$$

in degrees, where $\mathcal{V}_N$ contains pixels whose annotated neighborhood defines at least one valid local normal. We report $\text{AbsRel}_{\text{loc}}$ on benchmarks with instance masks, and $\text{MAE}_{\text{normal}}$ where dense ground truth supports reliable normal estimation. Lower is better for all metrics.

Table 1: **SotA comparison for AbsRel$_{loc}$ (%).**

Method	ETH3D	iBims-1	Sintel	DDAD	DIODE
DUSt3R [59]	6.31	5.43	11.79	9.23	7.31
Depth Pro [5]	4.77	4.11	10.54	8.09	6.80
MapAnything [24]	5.80	5.16	12.93	9.10	6.92
VGGT [53]	3.89	5.09	9.62	8.11	5.90
InfiniDepth [72]*	4.65	4.29	10.56	8.57	6.77
VGGT-Ω [54]†	4.12	5.23	11.49	7.83	7.30
Depth Any. 3 [28]	5.03	4.18	11.10	8.89	6.14
PPD [67]*	4.09	4.11	11.22	8.48	6.13
UniDepthV2 [37]	4.00	4.02	9.35	8.32	6.15
π^3 [61]	4.10	4.52	10.58	7.24	5.76
MoGe-2 [58]	3.27	3.61	8.13	6.57	5.09
MoGe [57]	3.16	3.79	8.45	6.70	4.77
SurGe	**2.66**	**3.33**	**7.66**	**6.29**	**4.63**

Table 2: **SotA comparison for MAE$_{normal}$ (°).**

Method	ETH3D	iBims-1	GSO	Sintel	DIODE
DUSt3R [59]	26.2	24.0	28.7	43.6	21.1
Depth Pro [5]	24.0	20.3	18.0	33.8	21.9
MapAnything [24]	25.2	25.1	18.2	40.3	22.7
VGGT [53]	20.1	21.6	10.6	29.7	14.7
InfiniDepth [72]*	24.6	21.9	19.3	36.4	22.4
VGGT-Ω [54]†	22.6	23.0	16.1	33.6	20.4
Depth Any. 3 [28]	22.2	20.7	14.7	33.6	17.5
PPD [67]*	60.4	38.2	26.6	53.8	49.6
UniDepthV2 [37]	24.2	20.7	19.0	31.2	21.3
π^3 [61]	22.3	21.1	13.3	34.0	18.0
MoGe-2 [58]	21.0	18.4	11.5	27.9	15.2
MoGe [57]	19.5	17.6	11.0	25.2	12.4
SurGe	**18.3**	**16.5**	**10.5**	**24.5**	**12.0**

Table 3: **State-of-the-art comparison for AbsRel$_{glob}$ (%).** Avg. rank is computed over the eight datasets. * Uses MoGe-2 for alignment and unprojection. † September 2026 checkpoint.

Method	NYUv2	KITTI	ETH3D	iBims-1	GSO	Sintel	DDAD	DIODE	Avg. rank
DUSt3R [59]	4.42	12.81	7.35	5.06	4.56	30.33	19.62	9.04	12.38
Depth Pro [5]	4.36	9.16	7.73	4.34	3.16	19.57	14.44	6.29	9.75
MapAnything [24]	4.58	8.72	5.23	4.49	1.98	20.88	14.85	6.43	9.75
VGGT [53]	3.88	8.91	4.59	4.83	**1.04**	21.64	17.34	6.65	9.12
InfiniDepth [72]*	4.51	6.23	7.12	4.43	3.03	19.39	14.24	6.62	9.12
VGGT-Ω [54]†	5.60	7.09	4.20	5.31	2.23	18.92	12.82	6.34	8.38
Depth Anything 3 [28]	3.78	6.19	5.47	3.75	1.70	21.50	14.52	6.44	7.62
PPD [67]*	3.81	6.89	4.55	4.06	1.48	17.96	12.84	5.86	6.25
UniDepthV2 [37]	3.66	4.75	4.38	4.05	2.91	17.93	11.96	7.45	6.12
π^3 [61]	3.41	8.16	4.91	3.93	1.49	17.68	11.88	5.24	5.38
MoGe-2 [58]	3.33	6.47	3.89	3.65	1.16	17.45	10.14	5.13	3.25
MoGe [57]	3.69	**4.67**	3.54	3.34	1.14	**16.87**	10.44	**4.44**	2.25
SurGe	**3.31**	4.80	**3.51**	**3.31**	1.11	17.13	**9.05**	4.88	**1.62**

4.2 Comparison to the State of the Art

We compare SurGe against recent feedforward 3D reconstruction methods. The clearest gains appear in the local evaluations, AbsRel$_{loc}$ and MAE$_{normal}$. These evaluations are complementary: AbsRel$_{loc}$ measures pointwise point map accuracy after instance-level alignment, whereas MAE$_{normal}$ evaluates the local surface orientation induced by neighboring point predictions. In Tab. 1, SurGe gives the lowest AbsRel$_{loc}$ on every evaluated dataset. The same pattern holds for MAE$_{normal}$ in Tab. 2, indicating that SurGe's point maps form more accurate local surfaces rather than only lower pointwise error. These gains span real-world and synthetic benchmarks, covering indoor and outdoor scenes as well as isolated objects.

Table 3 shows that global point map accuracy is already highly competitive among the strongest recent models, especially MoGe, MoGe-2 [58], and π^3 [61]. Within this tighter regime, SurGe remains consistently strong: it ranks first on four of the eight datasets and obtains the best average rank overall. Thus, gains in the local evaluations do not come at the expense of global scene geometry.

The qualitative examples in Fig. 1 show the same trend. MoGe-2 and InfiniDepth [72] recover plausible coarse shape but produce irregular local surfaces, visible in both the rendered 3D geometry and the point map normals. In contrast, SurGe preserves thin structures and produces smoother surfaces with sharper geometric detail. Together, the quantitative and qualitative results show that SurGe improves local surface quality while preserving state-of-the-art global point map accuracy.

Table 4: **Decoder ablation for AbsRel$_{\text{loc}}$ (%).**

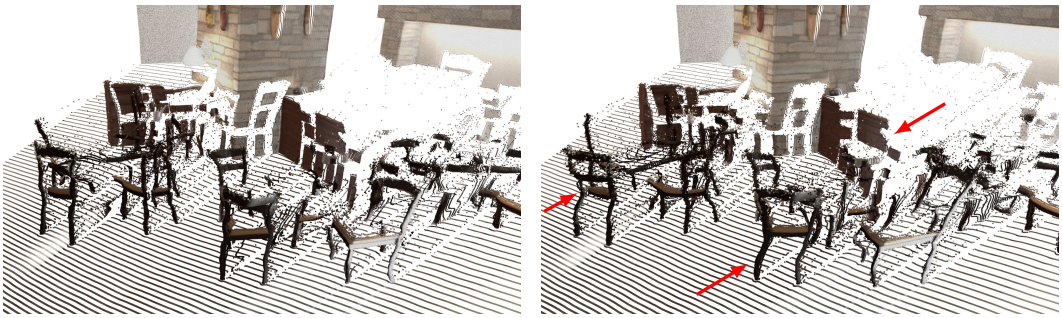
Decoder	ETH3D	iBims-1	Sintel	DDAD	DIODE
ViT	4.12	4.13	9.29	7.76	6.25
DPT head	3.25	3.58	8.42	6.62	5.06
ConvStack	3.04	3.49	8.11	6.55	4.93
ConvStack-L	2.94	3.44	8.15	6.43	4.82
NAD (ours)	**2.66**	**3.33**	**7.66**	**6.29**	**4.63**

Table 5: **Decoder ablation for MAE$_{\text{normal}}$ (°).**

Decoder	ETH3D	iBims-1	GSO	Sintel	DIODE
ViT	23.6	21.5	17.0	32.3	18.7
DPT head	20.3	18.5	11.6	28.3	14.9
ConvStack	19.5	17.3	11.1	26.9	13.3
ConvStack-L	18.8	16.9	10.9	25.4	12.5
NAD (ours)	**18.3**	**16.5**	**10.5**	**24.5**	**12.0**

Table 6: **Decoder ablation for AbsRel$_{\text{glob}}$ (%).** Avg. rank is computed over the eight datasets.

Decoder	NYUv2	KITTI	ETH3D	iBims-1	GSO	Sintel	DDAD	DIODE	Avg. rank
ViT	3.90	5.65	3.86	3.66	1.32	17.88	10.12	**4.80**	4.12
DPT head	3.44	5.08	**3.51**	3.37	1.14	18.18	9.42	5.02	3.12
ConvStack	3.54	5.15	3.53	3.37	1.17	18.07	9.40	4.82	3.25
ConvStack-L	3.49	4.99	3.65	3.43	1.17	17.93	9.25	4.93	3.19
NAD (ours)	**3.31**	**4.80**	**3.51**	**3.31**	**1.11**	**17.13**	**9.05**	4.88	**1.31**



(a) **NAD (ours)** (b) ConvStack-L

Figure 4: **Qualitative decoder ablation.** Our NAD produces less warped geometry than a convolutional decoder, visible in the chair legs and the wall to the right.

4.3 Ablation Study

Decoder design. We compare SurGe's Neighborhood Attention Decoder (NAD) against a DPT head following VGGT [53], MoGe's ConvStack [57], a larger "ConvStack-L" baseline that matches our stage layout, widths, and depth while keeping convolutional residual blocks, and a ViT [11] decoder following π^3 [61]. To isolate the effect of the decoder architecture, we attach each decoder to the same DINOv2 ViT-Large backbone and use the same point map output parameterization and loss, including $\mathcal{L}_{\text{pgm}}$. Tables 4 to 6 show that increasing convolutional decoder capacity helps: ConvStack-L moderately improves over ConvStack on most datasets in both local and global evaluations. However, our NAD performs even better, achieving the lowest error on every dataset in the AbsRel$_{\text{loc}}$ and MAE$_{\text{normal}}$ evaluations, and on seven of eight datasets for global point map accuracy. The ViT decoder is competitive on some datasets for the global evaluation, but has the highest error on every dataset in the local evaluations; patch-level artifacts are one visible failure mode.

Figure 4 shows a qualitative comparison between SurGe and the ConvStack-L configuration. Compared to SurGe, the model with the ConvStack-L decoder slightly warps thin structures such as the chair legs and misplaces parts of the wall on the right side.

Surface loss. We next ablate the surface loss while keeping the NAD and the pointwise global and local losses fixed. We compare supervision from point map normals [57] ($\mathcal{L}_{\text{normal}}$), log-depth gradient matching on the z coordinate [24, 27] ($\mathcal{L}_{\text{gm}}$), and our point gradient matching loss ($\mathcal{L}_{\text{pgm}}$). As shown in Tabs. 7 and 8, both gradient-based losses improve the local evaluations compared to $\mathcal{L}_{\text{normal}}$, and our proposed $\mathcal{L}_{\text{pgm}}$ improves over $\mathcal{L}_{\text{gm}}$ in most cases. However, in the global evaluation shown in Tab. 9, $\mathcal{L}_{\text{gm}}$ generally performs worse than $\mathcal{L}_{\text{normal}}$, while $\mathcal{L}_{\text{pgm}}$ noticeably outperforms

Table 7: **Surface loss abl. for AbsRel$_{\text{loc}}$ (%).**

Surface loss	ETH3D	iBims-1	Sintel	DDAD	DIODE
None	2.81	3.40	7.97	6.32	4.66
$\mathcal{L}_{\text{normal}}$	2.87	3.42	7.97	6.30	4.63
$\mathcal{L}_{\text{gm}}$	2.78	3.35	7.86	**6.23**	**4.59**
$\mathcal{L}_{\text{pgm}}$ **(ours)**	**2.66**	**3.33**	**7.66**	6.29	4.63

Table 8: **Surface loss abl. for MAE$_{\text{normal}}$ (°).**

Surface loss	ETH3D	iBims-1	GSO	Sintel	DIODE
None	19.8	17.5	11.1	27.4	13.5
$\mathcal{L}_{\text{normal}}$	19.1	17.0	10.9	25.8	12.6
$\mathcal{L}_{\text{gm}}$	18.4	**16.5**	10.8	24.8	**11.8**
$\mathcal{L}_{\text{pgm}}$ **(ours)**	**18.3**	**16.5**	**10.5**	**24.5**	12.0

Table 9: **Surface loss ablation for AbsRel$_{\text{glob}}$ (%).** All rows use the same global and local point losses. Avg. rank is computed over the eight datasets.

Surface loss	NYUv2	KITTI	ETH3D	iBims-1	GSO	Sintel	DDAD	DIODE	Avg. rank
None	3.46	5.10	3.58	3.39	**1.11**	17.31	9.17	5.16	3.12
$\mathcal{L}_{\text{normal}}$	3.43	5.01	**3.51**	3.40	**1.11**	18.02	9.09	5.14	2.69
$\mathcal{L}_{\text{gm}}$	3.44	5.00	3.72	3.46	1.17	17.78	9.06	4.96	3.00
$\mathcal{L}_{\text{pgm}}$ **(ours)**	**3.31**	**4.80**	**3.51**	**3.31**	**1.11**	**17.13**	**9.05**	**4.88**	**1.19**

both, which we attribute to the fact that the $\mathcal{L}_{\text{pgm}}$ signal remains in the same space as the global loss (*i.e.*, 3D point displacements). Overall, $\mathcal{L}_{\text{pgm}}$ provides a practical replacement for prior surface losses, improving local surface quality without sacrificing global point map accuracy.

5 Conclusion

We presented SurGe, a monocular point map model designed to improve local surface geometry rather than only average point accuracy. SurGe improves the local point map and point map normal performance across eight zero-shot benchmarks while preserving strong global geometry; ablations show that both the NAD head and $\mathcal{L}_{\text{pgm}}$ contribute to these gains. Our ablations further suggest that current decoder designs are a limiting factor in predicting high-quality local surface geometry. Since the NAD head increases computational cost relative to smaller convolutional decoders, more efficient variants are a useful direction for future work. Complex scenes with many simultaneous high-frequency geometry changes can still exhibit local surface artifacts; further failure cases are discussed in Sec. H. Beyond the model, our point map normal metric is a step toward improved geometric evaluation: it exposes surface artifacts that are only weakly captured by pointwise metrics. We hope that this will make local surface quality a more explicit target for future geometry models.

Acknowledgements. This work has received funding from the Jupiter AI Factory (JAIF), jointly funded by the European High Performance Computing Joint Undertaking (JU), the German Federal Ministry of Research, Technology and Space (BMFTR), and the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW) under grant agreement No. `101250682`. The authors gratefully acknowledge the Gauss Centre for Supercomputing e.V. (`www.gauss-centre.eu`) for providing computing time through the John von Neumann Institute for Computing (NIC) on the GCS Supercomputers JUWELS and JUPITER at Jülich Supercomputing Centre (JSC).

References

[1] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] G. Bae and A. J. Davison. Rethinking inductive biases for surface normal estimation. In *CVPR*, 2024.

[3] G. Baruch, Z. Chen, A. Dehghan, T. Dimry, Y. Feigin, P. Fu, T. Gebauer, B. Joffe, D. Kurz, A. Schwartz, and E. Shulman. ARKitScenes - a diverse real-world dataset for 3D indoor scene understanding using mobile RGB-D data. In *NeurIPS Datasets and Benchmarks Track*, 2021.

[4] S. Bian, C. Xu, and J. Gao. Modeling depth ambiguity: A mixture-density representation for flying-point-free depth estimation. *arXiv preprint arXiv:2606.02552*, 2026.

[5] A. Bochkovskii, A. Delaunoy, H. Germain, M. Santos, Y. Zhou, S. R. Richter, and V. Koltun. Depth Pro: Sharp monocular metric depth in less than a second. In *ICLR*, 2025.

[6] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012.

[7] L. Chambon, P. Couairon, E. Zablocki, A. Boulch, N. Thome, and M. Cord. NAF: Zero-shot feature upsampling via neighborhood attention filtering. In *CVPR*, 2026.

[8] M. Dehghani, J. Djolonga, B. Mustafa, P. Padlewski, J. Heek, J. Gilmer, A. P. Steiner, M. Caron, R. Geirhos, I. Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *ICML*, 2023.

[9] M. Deitke, D. Schwenk, J. Salvador, L. Weihs, O. Michel, E. VanderBilt, L. Schmidt, K. Ehsani, A. Kembhavi, and A. Farhadi. Objaverse: A universe of annotated 3D objects. In *CVPR*, 2023.

[10] Z. Ding, Y. Zhang, C. Zhu, G. Zhang, X. Li, N. Jiang, Y. Que, Y. Peng, and X. Guan. CAT-Unet: An enhanced U-Net architecture with coordinate attention and skip-neighborhood attention transformer for medical image segmentation. *Information Sciences*, 2024.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[12] L. Downs, A. Francis, N. Koenig, B. Kinman, R. Hickman, K. Reymann, T. B. McHugh, and V. Vanhoucke. Google Scanned Objects: A high-quality dataset of 3D scanned household items. In *ICRA*, 2022.

[13] X. Fang, J. Gao, Z. Wang, Z. Chen, X. Ren, J. Lyu, Q. Ren, Z. Yang, X. Yang, Y. Yan, and C. Lyu. Dens3R: A foundation model for 3D geometry prediction. In *ICLR*, 2026.

[14] M. Fonder and M. V. Droogenbroeck. Mid-Air: A multi-modal dataset for extremely low altitude drone flights. In *CVPR Workshops*, 2019.

[15] V. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon. 3D packing for self-supervised monocular depth estimation. In *CVPR*, 2020.

[16] J. L. Gómez, M. Silva, A. Seoane, A. Borràs, M. Noriega, G. Ros, J. A. Iglesias-Guitian, and A. M. López. All for one, and one for all: UrbanSyn dataset, the third Musketeer of synthetic driving scenes. *Neurocomputing*, 2025.

[17] A. Hassani and H. Shi. Dilated neighborhood attention transformer. *arXiv preprint arXiv:2209.15001*, 2022.

[18] A. Hassani, S. Walton, J. Li, S. Li, and H. Shi. Neighborhood attention transformer. In *CVPR*, 2023.

[19] A. Hassani, W.-M. Hwu, and H. Shi. Faster neighborhood attention: Reducing the $O(n^2)$ cost of self attention at the threadblock level. In *NeurIPS*, 2024.

[20] A. Hassani, F. Zhou, A. Kane, J. Huang, C.-Y. Chen, M. Shi, S. Walton, M. Hoehnerbach, V. Thakkar, M. Isaev, et al. Generalized neighborhood attention: Multi-dimensional sparse attention at the speed of light. In *CVPR Findings*, 2026.

[21] D. Hernandez-Juarez, L. Schneider, A. Espinosa, D. Vazquez, A. M. Lopez, U. Franke, M. Pollefeys, and J. C. Moure. Slanted stixels: Representing San Francisco's steepest streets. In *BMVC*, 2017.

[22] P.-H. Huang, K. Matzen, J. Kopf, N. Ahuja, and J.-B. Huang. DeepMVS: Learning multi-view stereopsis. In *CVPR*, 2018.

[23] H. Jung, P. Ruhkamp, G. Zhai, N. Brasch, Y. Li, Y. Verdie, J. Song, Y. Zhou, A. Armagan, S. Ilic, A. Leonardis, N. Navab, and B. Busam. On the importance of accurate geometry data for dense 3D vision tasks. In *CVPR*, 2023.

[24] N. Keetha, N. Müller, J. Schönberger, L. Porzi, Y. Zhang, T. Fischer, A. Knapitsch, D. Zauss, E. Weber, N. Antunes, J. Luiten, M. Lopez-Antequera, S. R. Bulò, C. Richardt, D. Ramanan, S. Scherer, and P. Kontschieder. MapAnything: Universal feed-forward metric 3D reconstruction. In *3DV*, 2026.

[25] T. Koch, L. Liebel, F. Fraundorfer, and M. Körner. Evaluation of CNN-based single-image depth estimation methods. In *ECCV Workshops*, 2018.

[26] Y. Li, L. Jiang, L. Xu, Y. Xiangli, Z. Wang, D. Lin, and B. Dai. MatrixCity: A large-scale city dataset for city-scale neural rendering and beyond. In *ICCV*, 2023.

[27] Z. Li and N. Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018.

[28] H. Lin, S. Chen, J. H. Liew, D. Y. Chen, Z. Li, G. Shi, J. Feng, and B. Kang. Depth Anything 3: recovering the visual space from any views. In *ICLR*, 2026.

[29] H. Liu, B. Li, C. Liu, and M. Lu. DiNAT-IR: Exploring dilated neighborhood attention for high-quality image restoration. *International Robotics & Automation Journal*, 2025.

[30] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[31] G. Martin Garcia, K. Knaebel, C. Schmidt, D. de Geus, A. Hermans, and B. Leibe. Fine-tuning image-conditional diffusion models is easier than you think. In *WACV*, 2025.

[32] L. Mehl, J. Schmalfuss, A. Jahedi, Y. Nalivayko, and A. Bruhn. Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo. In *CVPR*, 2023.

[33] S. Niklaus, L. Mai, J. Yang, and F. Liu. 3D Ken Burns effect from a single image. *ACM TOG*, 2019.

[34] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.

[35] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. DINOv2: Learning robust visual features without supervision. *TMLR*, 2024.

[36] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *ICCV*, 2023.

[37] L. Piccinelli, C. Sakaridis, Y.-H. Yang, M. Segu, S. Li, W. Abbeloos, and L. V. Gool. UniDepthV2: Universal monocular metric depth estimation made simpler. *IEEE TPAMI*, 2026.

[38] L. Qiu, G. Chen, X. Gu, Q. Zuo, M. Xu, Y. Wu, W. Yuan, Z. Dong, L. Bo, and X. Han. RichDreamer: A generalizable normal-depth diffusion model for detail richness in text-to-3D. In *CVPR*, 2024.

[39] R. Ranftl, A. Bochkovskiy, and V. Koltun. Vision transformers for dense prediction. In *ICCV*, 2021.

[40] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE TPAMI*, 2022.

[41] M. Roberts, J. Ramapuram, A. Ranjan, A. Kumar, M. A. Bautista, N. Paczan, R. Webb, and J. M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *ICCV*, 2021.

[42] D. Saadati, O. N. Manzari, and S. Mirzakuchaki. Dilated-UNet: A fast and accurate medical image segmentation approach using a dilated transformer and U-Net architecture. *arXiv preprint arXiv:2304.11450*, 2023.

[43] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *CVPR*, 2017.

[44] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016.

[45] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012.

[46] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2024.

[47] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov. Scalability in perception for autonomous driving: Waymo Open Dataset. In *CVPR*, 2020.

[48] F. Tosi, Y. Liao, C. Schmitt, and A. Geiger. SMD-Nets: Stereo mixture density networks. In *CVPR*, 2021.

[49] H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, and H. Jégou. Going deeper with image transformers. In *ICCV*, 2021.

[50] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger. Sparsity invariant CNNs. In *3DV*, 2017.

[51] I. Vasiljevic, N. I. Kolkin, S. Zhang, R. Luo, H. Wang, F. Z. Dai, A. F. Daniele, M. Mostajabi, S. Basart, M. R. Walter, and G. Shakhnarovich. DIODE: A dense indoor and outdoor DEpth dataset. *arXiv preprint arXiv:1908.00463*, 2019.

[52] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[53] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny. VGGT: Visual geometry grounded transformer. In *CVPR*, 2025.

[54] J. Wang, M. Chen, S. Zhang, N. Karaev, J. Schönberger, P. Labatut, P. Bojanowski, D. Novotny, A. Vedaldi, and C. Rupprecht. VGGT-Ω. In *CVPR*, 2026.

[55] K. Wang and S. Shen. Flow-motion and depth network for monocular stereo and beyond. *IEEE Robotics and Automation Letters*, 2020.

[56] Q. Wang, S. Zheng, Q. Yan, F. Deng, K. Zhao, and X. Chu. IRS: A large naturalistic indoor robotics stereo dataset to train deep models for disparity and surface normal estimation. In *ICME*, 2021.

[57] R. Wang, S. Xu, C. Dai, J. Xiang, Y. Deng, X. Tong, and J. Yang. MoGe: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In *CVPR*, 2025.

[58] R. Wang, S. Xu, Y. Dong, Y. Deng, J. Xiang, Z. Lv, G. Sun, X. Tong, and J. Yang. MoGe-2: Accurate monocular geometry with metric scale and sharp details. In *NeurIPS*, 2025.

[59] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. DUSt3R: Geometric 3D vision made easy. In *CVPR*, 2024.

[60] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer. TartanAir: A dataset to push the limits of visual SLAM. In *IROS*, 2020.

[61] Y. Wang, J. Zhou, H. Zhu, W. Chang, Y. Zhou, Z. Li, J. Chen, J. Pang, C. Shen, and T. He. π^3: Permutation-equivariant visual geometry learning. In *ICLR*, 2026.

[62] Waymo. Waymo Open Dataset: An autonomous driving dataset, 2019. URL https://www.waymo.com/open.

[63] P. Weinzaepfel, V. Leroy, T. Lucas, R. Brégier, Y. Cabon, V. Arora, L. Antsfeld, B. Chidlovskii, G. Csurka, and J. Revaud. CroCo: Self-supervised pre-training for 3D vision tasks by cross-view completion. In *NeurIPS*, 2022.

[64] P. Weinzaepfel, T. Lucas, V. Leroy, Y. Cabon, V. Arora, R. Brégier, G. Csurka, L. Antsfeld, B. Chidlovskii, and J. Revaud. CroCo v2: Improved cross-view completion pre-training for stereo matching and optical flow. In *ICCV*, 2023.

[65] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, D. Ramanan, P. Carr, and J. Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *NeurIPS Datasets and Benchmarks Track*, 2021.

[66] S. Wu, J. Nugent, W. Yang, and J. Deng. Toward a better understanding of monocular depth evaluation. *arXiv preprint arXiv:2510.19814*, 2025.

[67] G. Xu, H. Lin, H. Luo, X. Wang, J. Yao, L. Zhu, Y. Pu, C. Chi, H. Sun, B. Wang, et al. Pixel-Perfect Depth with semantics-prompted diffusion transformers. In *NeurIPS*, 2025.

[68] J. Yang, A. Sax, K. J. Liang, M. Henaff, H. Tang, A. Cao, J. Chai, F. Meier, and M. Feiszli. Fast3R: Towards 3D reconstruction of 1000+ images in one forward pass. In *CVPR*, 2025.

[69] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao. Depth Anything: Unleashing the power of large-scale unlabeled data. In *CVPR*, 2024.

[70] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao. Depth Anything V2. In *NeurIPS*, 2024.

[71] Y. Yao, Z. Luo, S. Li, J. Zhang, Y. Ren, L. Zhou, T. Fang, and L. Quan. BlendedMVS: A large-scale dataset for generalized multi-view stereo networks. In *CVPR*, 2020.

[72] H. Yu, H. Lin, J. Wang, J. Li, Y. Wang, X. Zhang, Y. Wang, X. Zhou, R. Hu, and S. Peng. InfiniDepth: Arbitrary-resolution and fine-grained depth estimation with neural implicit fields. In *CVPR*, 2026.

[73] A. R. Zamir, A. Sax, W. B. Shen, L. Guibas, J. Malik, and S. Savarese. Taskonomy: Disentangling task transfer learning. In *CVPR*, 2018.

[74] X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer. Scaling vision transformers. In *CVPR*, 2022.

[75] B. Zhang and R. Sennrich. Root mean square layer normalization. In *NeurIPS*, 2019.

[76] J. Zheng, J. Zhang, J. Li, R. Tang, S. Gao, and Z. Zhou. Structured3D: A large photo-realistic dataset for structured 3D modeling. In *ECCV*, 2020.

[77] J. Zhu, X. Chen, K. He, Y. LeCun, and Z. Liu. Transformers without normalization. In *CVPR*, 2025.

[78] J. Zolfaghari Bengar, A. Gonzalez-Garcia, G. Villalonga, B. Raducanu, H. H. Aghdam, M. Mozerov, A. M. Lopez, and J. van de Weijer. Temporal coherence for active learning in videos. In *ICCV Workshops*, 2019.

Appendix

A Details on the Point Gradient Matching Loss

Pseudocode A gives a Python-like specification of $\mathcal{L}_{\text{pgm}}$ in Eq. (3). The loss optimizes the orientation and magnitude of local point gradients by matching 3D displacements; normalization by z makes the loss scale-invariant. We compute $\mathcal{L}_{\text{pgm}}$ only on point pairs where all involved points are valid.

Pseudocode A: **Point gradient matching loss** $\mathcal{L}_{\text{pgm}}$

```
def point_gradient_loss(pred, gt, valid):
  # pred, gt: [B, H, W, 3]
  # valid: [B, H, W]

  def pair_loss(pred0, pred1, gt0, gt1, valid0, valid1):
      valid_pair = valid0 & valid1

      pred_grad = (pred1 - pred0) / minimum(pred0[..., 2:], pred1[..., 2:])
      gt_grad = (gt1 - gt0) / minimum(gt0[..., 2:], gt1[..., 2:])

      loss = (pred_grad - gt_grad).norm(dim=-1)
      return (loss * valid_pair).sum() / valid_pair.sum()

  loss_dx = pair_loss(
      pred[:, :, :-1, :], pred[:, :, 1:, :],
      gt[:, :, :-1, :],   gt[:, :, 1:, :],
      valid[:, :, :-1],   valid[:, :, 1:],
  )
  loss_dy = pair_loss(
      pred[:, :-1, :, :], pred[:, 1:, :, :],
      gt[:, :-1, :, :],   gt[:, 1:, :, :],
      valid[:, :-1, :],   valid[:, 1:, :],
  )
  return (loss_dx + loss_dy) / 2
```

B Architectural Details

Window-matched RoPE. In addition to the embedded UV coordinates added at the beginning of each decoder stage, we apply RoPE [46] to the queries and keys in each NAD block. The UV grid provides absolute coordinates, while RoPE gives each attention head a consistent representation of relative offsets inside local windows, making local attention patterns easier to reuse across neighborhoods. We adapt the RoPE base frequency to the kernel size where attention is computed. For a Neighborhood Attention window size k, we set the base frequency $\tau = {}^{k}/_{\pi}$. For head dimension d_h, axial 2D RoPE uses $M = {}^{d_h}/_{4}$ geometrically spaced frequency bands per spatial axis, with frequencies $\omega_m = \tau^{-m/M}$ for $m = 0, \ldots, M-1$. With this choice, the lowest-frequency band changes by π radians, *i.e.*, half a rotation, over one neighborhood width, providing meaningful phase variation within each local window. In preliminary experiments, RoPE with this window-matched base frequency improves accuracy over using only the UV embedding.

Normalization in NAD blocks. We remove the usual normalization layers before the attention and FFN blocks. Wang et al. [58] remove normalization layers from the ConvStack without observing instabilities or reduced accuracy. In our attention decoder, removing LayerNorm [1] *improves* accuracy, but occasionally leads to irrecoverable training instabilities. We tried to replace LayerNorm with several alternative normalization approaches (RMSNorm [75], DyT [77], and LayerScale [49]), and found that each of them recovers training stability, but none of them improves accuracy over LayerNorm. We therefore suspect that per-token normalization can interfere with the unbounded regression task of 3D geometry estimation. To stabilize training without compromising task performance, we keep the norm-free block structure and apply QK normalization [8, 74] in the NAD blocks.

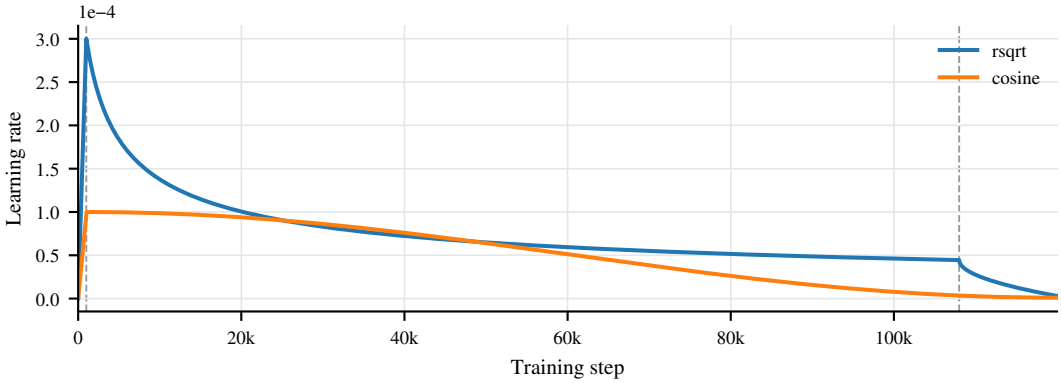


Figure A: **Learning rate schedule.** We use a reciprocal square root learning rate schedule [74] (rsqrt) with a timescale of 2400, a linear warmup of 1000 steps, and a 10% cooldown phase with schedule $1 - \sqrt{p} + \mathrm{lr}_{\min}$, where $p \in (0, 1]$ is the progress of the cooldown phase and $\mathrm{lr}_{\min}$ is the peak learning rate multiplied by 10^{-2}. Early training is stable enough to support a much larger peak learning rate than we can use with standard cosine decay. The rsqrt schedule uses this large initial rate to make rapid progress, then quickly decays to a moderate rate, whereas a cosine schedule at the same peak becomes unstable later in training. In our setting, rsqrt tends to outperform cosine decay even when the total number of training steps is known in advance, while still allowing training to continue indefinitely before cooldown.

Stage-wise UV embedding. Following Wang et al. [57], we add a learned positional embedding to the features at the beginning of each stage in the NAD head. This embedding is produced by linearly projecting a UV grid at the stage resolution, with coordinates normalized by the image diagonal. It gives the decoder aspect-ratio-aware absolute position information for the current feature map.

C Training Details

All main models and ablations are trained for 120 000 optimizer steps with total batch size 128. We use AdamW [30] with peak learning rate 3×10^{-4} for the decoder and 3×10^{-5} for the DINOv2 [35] backbone, gradient clipping at 1.0, and the reciprocal square root schedule shown in Fig. A. The decoder learning rate is linearly warmed up for 1000 steps, during which the backbone remains frozen. After unfreezing, the backbone learning rate is also warmed up for 1000 steps, ending at 0.1 times the decoder schedule evaluated at step 2000; after this point, it remains phase-aligned with the decoder schedule at the same 0.1 scale. Training uses bf16 mixed precision for the backbone, while the point decoder, output remapping, and losses are evaluated in fp32.

For the first 80% of training, we sample images at a fixed area of 512^2 pixels with target aspect ratios in $[0.5, 2.0]$, and use a low-resolution encoder-token budget of 1024. The final 20% of steps sample image areas uniformly from $[512^2, 960^2]$ pixels and encoder-token budgets uniformly from $\{1024, \ldots, 2802\}$. The cooldown phase in the final 10% samples only high-quality synthetic data: all synthetic training datasets except Ken Burns [33] and G-Objaverse [9, 38].

Following common practice [5, 57], we adapt supervision to label quality by selecting loss terms by label type:

$$\begin{aligned} \mathcal{L}_{\text{synthetic}} &= \mathcal{L}_{\text{glob}} + \mathcal{L}_{\text{loc},4} + \mathcal{L}_{\text{loc},16} + \mathcal{L}_{\text{loc},64} + 10\mathcal{L}_{\text{pgm}}, \\ \mathcal{L}_{\text{sfm}} &= \mathcal{L}_{\text{glob}} + \mathcal{L}_{\text{loc},4} + \mathcal{L}_{\text{loc},16}, \\ \mathcal{L}_{\text{lidar}} &= \mathcal{L}_{\text{glob}} + \mathcal{L}_{\text{loc},4}. \end{aligned}$$

D Training Data

We train on a subset of the datasets used by MoGe-2 [58]; the full list is provided in Tab. A. Licenses and usage terms are listed in Tab. L.

Table A: **List of training datasets.** The mix and weighting are similar to MoGe-2 [58]. Sampling probabilities are proportional to the listed weights.

Name	Domain	Frames	Type	Weight
Argoverse2 [65]	Outdoor/Driving	2.2M	LiDAR	7.4
ARKitScenes [3]	Indoor	449K	SfM	8.6
BlendedMVS [71]	In-the-wild	112K	SfM	12.0
G-Objaverse [9, 38]	Object	894K	Synthetic	4.8
GTA-SfM [55]	Outdoor/In-the-wild	18K	Synthetic	2.8
Hypersim [41]	Indoor	67K	Synthetic	5.0
IRS [56]	Indoor	48K	Synthetic	5.6
Ken Burns [33]	In-the-wild	304K	Synthetic	1.6
MatrixCity [26]	Outdoor/Driving	392K	Synthetic	1.3
MegaDepth [27]	Outdoor/In-the-wild	103K	SfM	5.6
MidAir [14]	Outdoor/In-the-wild	391K	Synthetic	4.0
MVS-Synth [22]	Outdoor/Driving	12K	Synthetic	1.2
Structured3D [76]	Indoor	71K	Synthetic	4.8
Synthia-AL [78]	Outdoor/Driving	142K	Synthetic	1.1
Synthia-SF [21]	Outdoor/Driving	2K	Synthetic	0.1
TartanAir [60]	In-the-wild	586K	Synthetic	5.0
Taskonomy [73]	Indoor	4.5M	SfM	14.1
UnrealStereo4K [48]	In-the-wild	14K	Synthetic	1.6
UrbanSyn [16]	Outdoor/Driving	8K	Synthetic	2.1
Waymo [47, 62]	Outdoor/Driving	1M	LiDAR	6.4

E Decoder Runtime Tradeoff

Table B compares the inference latency and peak memory of NAD and ConvStack-L, both as standalone decoders and within the full model. NAD's decoder latency is $2.24\times$ that of ConvStack-L. In the full model, the shared encoder accounts for part of the inference time, reducing the latency ratio to $1.34\times$ with DINOv2-giant and $1.49\times$ with DINOv2-Large. Peak memory increases modestly with NAD. NAD therefore trades inference speed for more accurate local surface geometry.

Table B: **Runtime comparison with ConvStack-L.** Benchmarks run on an NVIDIA GH200 GPU with PyTorch 2.15.0, `torch.compile` with static shapes, and `tf32` enabled. NAD uses a Triton implementation of neighborhood attention. Encoders use `bf16` autocast and decoder residual branches use `fp16` autocast; residual accumulation, input/output projections, and upsampling remain in `fp32`. We report median inference latency over 50 timed iterations after 10 warmup iterations, with CUDA synchronization, at 512×512 resolution and batch size 1. Peak memory is measured with PyTorch `max_memory_allocated`.

Encoder	Decoder	Latency (ms) ↓	Relative latency (×) ↓	Peak memory (GiB) ↓
DINOv2-giant	ConvStack-L	13.12	1.00	4.87
	NAD (ours)	17.55	1.34	5.03
DINOv2-Large	ConvStack-L	9.23	1.00	1.72
	NAD (ours)	13.73	1.49	1.88
None	ConvStack-L	3.07	1.00	0.56
	NAD (ours)	6.88	2.24	0.72

F Additional Quantitative Results

For direct comparison with depth estimation methods, Tabs. C and D report affine-invariant depth AbsRel and δ_1 on the same eight benchmarks as the main tables. Table E reports affine-invariant point map accuracy $\delta_{0.05}$, complementing $\text{AbsRel}_{\text{glob}}$ in Tab. 3. Depth δ_1 measures the fraction of valid pixels with $\max(\hat{D}^\star/D, D/\hat{D}^\star) < 1.25$ after depth alignment. Point map δ_τ instead measures the fraction satisfying $\|\hat{\mathbf{P}}^\star - \mathbf{P}\|_2 < \tau \min(\|\hat{\mathbf{P}}^\star\|_2, \|\mathbf{P}\|_2)$ after point map alignment. Method order and footnote markers in the full comparisons follow Tab. 3.

Table F reports depth boundary F_1 [5] on iBims-1, Sintel, HAMMER [23], and Spring [32]. Because boundary F_1 is highly sensitive to output resolution, Tab. G also compares MoGe, MoGe-2, and SurGe at a matched resolution of 2551 encoder tokens. Resampling can smooth or shift depth discontinuities, changing boundary F_1 even when the underlying scene geometry is similar. MoGe-2 likewise reports higher boundary F_1 scores at higher output resolutions [58]. At the matched resolution, SurGe achieves the highest boundary F_1 on all four datasets.

Finally, Tab. H reports mean angular error for three direct normal estimators alongside SurGe to put the magnitude of our errors in context. The results are not directly comparable because the direct estimators use dataset normal annotations as ground truth, while SurGe uses normals derived from ground-truth point maps ($\text{MAE}_{\text{normal}}$).

Table C: **SotA comparison for affine-invariant depth AbsRel (%, ↓).**

Method	NYUv2	KITTI	ETH3D	iBims-1	GSO	Sintel	DDAD	DIODE
DUSt3R [59]	3.72	7.28	5.08	3.94	3.30	25.44	16.85	6.68
Depth Pro [5]	3.67	5.12	4.97	3.23	1.46	15.82	12.61	4.66
MapAnything [24]	3.67	4.85	4.09	3.37	1.44	16.58	13.04	4.46
VGGT [53]	3.03	5.36	3.43	3.37	**0.81**	18.15	13.90	5.07
InfiniDepth [72]*	3.98	5.09	4.65	3.14	1.85	14.27	12.16	4.90
VGGT-Ω [54]†	3.45	4.73	2.82	3.26	1.20	15.05	10.67	4.27
Depth Anything 3 [28]	3.28	5.24	3.94	2.81	1.33	17.28	12.23	4.78
PPD [67]*	4.29	10.29	6.75	4.43	1.32	16.20	16.78	6.12
UniDepthV2 [37]	2.96	3.86	2.95	2.64	1.37	13.33	10.48	4.05
π^3 [61]	2.95	4.29	2.70	2.83	1.08	14.04	9.76	3.63
MoGe-2 [58]	2.89	**3.75**	2.80	2.36	0.94	13.37	8.26	**3.14**
MoGe [57]	2.91	3.92	2.69	2.53	0.94	**13.03**	8.40	3.16
SurGe	**2.85**	4.00	**2.64**	**2.34**	0.87	13.41	**7.77**	3.60

Table D: **SotA comparison for affine-invariant depth δ_1 (%, ↑).**

Method	NYUv2	KITTI	ETH3D	iBims-1	GSO	Sintel	DDAD	DIODE
DUSt3R [59]	97.86	91.57	96.24	96.57	99.23	64.05	76.35	92.52
Depth Pro [5]	98.20	96.76	96.43	98.33	99.99	80.12	84.08	95.60
MapAnything [24]	98.20	97.01	97.54	97.82	99.95	78.72	84.77	95.93
VGGT [53]	98.40	95.06	97.81	97.18	99.99	76.09	81.23	94.63
InfiniDepth [72]*	97.77	96.31	96.42	98.48	99.95	81.88	86.15	95.28
VGGT-Ω [54]†	98.17	96.81	98.81	97.64	99.95	80.65	86.92	95.83
Depth Anything 3 [28]	98.14	96.28	96.62	98.09	99.98	76.95	84.81	94.46
PPD [67]*	97.97	85.04	93.14	97.02	99.99	77.92	74.76	92.76
UniDepthV2 [37]	**98.64**	98.08	98.53	98.36	99.99	83.18	90.89	96.49
π^3 [61]	98.48	97.79	98.82	98.11	99.98	81.30	89.49	96.80
MoGe-2 [58]	98.60	**98.10**	99.05	**98.75**	99.99	82.53	**92.49**	97.44
MoGe [57]	98.64	98.03	**99.17**	98.43	99.99	83.23	92.20	**97.47**
SurGe	98.63	97.88	99.04	98.71	**99.99**	**83.47**	92.27	97.02

Table E: **SotA comparison for affine-invariant point map $\delta_{0.05}$ (%, ↑).**

Method	NYUv2	KITTI	ETH3D	iBims-1	GSO	Sintel	DDAD	DIODE
DUSt3R [59]	75.14	22.72	59.12	72.67	69.30	21.64	22.91	64.16
Depth Pro [5]	75.08	37.79	57.56	75.01	79.75	26.88	32.97	67.80
MapAnything [24]	73.55	38.57	67.94	75.55	92.71	26.86	31.67	69.49
VGGT [53]	81.02	44.47	72.39	72.66	**98.06**	29.85	22.80	73.11
InfiniDepth [72]*	75.81	62.53	56.09	74.63	85.51	16.06	28.41	68.41
VGGT-Ω [54]†	60.97	51.84	75.83	66.72	94.16	32.42	34.52	69.87
Depth Anything 3 [28]	81.48	62.77	74.20	80.30	93.22	28.31	29.09	73.17
PPD [67]*	80.46	52.02	74.51	77.33	96.02	31.48	38.68	71.80
UniDepthV2 [37]	82.62	**73.61**	76.73	77.87	81.90	27.75	44.90	61.98
π^3 [61]	83.51	47.20	68.92	79.65	94.70	30.45	35.33	75.15
MoGe-2 [58]	84.20	53.69	76.68	80.47	97.82	33.34	43.74	74.00
MoGe [57]	81.68	72.74	79.63	**83.95**	97.61	32.16	38.35	**78.24**
SurGe	**84.51**	70.56	**80.12**	83.08	97.78	**34.16**	**48.05**	77.66

Table F: **SotA comparison for depth boundary F_1 (%, ↑).**

Method	iBims-1	Sintel	HAMMER	Spring
DUSt3R [59]	3.38	2.75	0.23	0.20
Depth Pro [5]	14.30	41.34	5.36	11.05
MapAnything [24]	5.50	6.85	1.14	4.80
VGGT [53]	3.49	13.23	0.63	4.04
InfiniDepth [72]*	18.04	**42.54**	**7.76**	**12.60**
VGGT-Ω [54]†	4.55	15.01	0.12	5.09
Depth Anything 3 [28]	9.55	15.94	1.69	5.25
PPD [67]*	8.58	17.12	2.24	7.38
UniDepthV2 [37]	11.15	39.68	4.41	7.08
π^3 [61]	6.07	18.58	0.66	5.09
MoGe-2 [58]	17.89	35.23	5.42	8.63
MoGe [57]	14.91	24.95	5.15	7.89
SurGe	**18.85**	32.87	5.44	8.40

Table G: **Depth boundary F_1 at matched resolution (%, ↑).** All models use MoGe's resolution of 2551 encoder tokens, the highest common supported setting.

Method	iBims-1	Sintel	HAMMER	Spring
MoGe-2 [58]	17.19	32.08	4.47	7.31
MoGe [57]	14.91	24.95	5.15	7.89
SurGe	**18.59**	**32.29**	**5.33**	**8.15**

Table H: **Comparison with direct normal prediction.** Mean angular error (°, ↓). Results are not directly comparable: DSINE, E2E-FT, and Dens3R use dataset normal annotations as ground truth, while SurGe uses normals derived from ground-truth point maps (MAE_{normal}).

Method	Normal source (prediction / ground truth)	iBims-1	Sintel
DSINE [2]	direct / dataset	17.1	34.9
E2E-FT [31]	direct / dataset	15.8	33.5
Dens3R [13]	direct / dataset	16.0	30.7
SurGe	point map / point map	16.5	24.5

Additional normal metrics. Tables I to K complement $\text{MAE}_{\text{normal}}$ with median angular error, angular accuracy below $11.25°$, and RelNormal [66], using normals induced from point maps.

Table I: **Point map normal median angular error (°, ↓).**

Method	ETH3D	iBims-1	GSO	Sintel	DIODE
DUSt3R [59]	18.1	14.4	21.1	36.5	13.2
Depth Pro [5]	18.0	12.5	15.2	26.5	16.4
MapAnything [24]	17.5	16.6	14.6	32.7	16.2
VGGT [53]	13.7	13.5	7.8	21.6	8.4
InfiniDepth [72]*	17.9	13.6	15.6	29.8	16.1
Depth Anything 3 [28]	15.0	11.7	11.2	26.4	11.1
PPD [67]*	59.3	33.5	23.2	51.1	47.7
UniDepthV2 [37]	18.7	13.2	17.1	24.3	17.2
π^3 [61]	15.3	12.6	9.9	25.9	11.3
MoGe-2 [58]	14.4	10.4	8.8	20.9	9.7
MoGe [57]	13.2	9.6	8.3	18.2	7.4
SurGe	**12.1**	**8.6**	**7.8**	**17.8**	**6.8**

Table J: **Point map normal angular accuracy below $11.25°$ (%, ↑).**

Method	ETH3D	iBims-1	GSO	Sintel	DIODE
DUSt3R [59]	36.7	44.8	36.3	13.7	52.7
Depth Pro [5]	34.4	47.9	44.8	20.8	44.9
MapAnything [24]	36.4	37.1	46.1	15.6	41.2
VGGT [53]	44.9	45.2	**69.0**	27.3	65.9
InfiniDepth [72]*	34.8	45.3	40.4	16.1	44.0
Depth Anything 3 [28]	42.8	50.7	57.6	21.6	58.2
PPD [67]*	5.1	13.0	20.8	5.7	7.0
UniDepthV2 [37]	30.3	45.1	32.6	21.9	36.0
π^3 [61]	41.3	47.9	60.6	22.6	56.6
MoGe-2 [58]	43.7	54.2	65.4	27.6	62.7
MoGe [57]	47.0	56.9	66.9	33.1	70.1
SurGe	**50.1**	**59.6**	68.4	**34.0**	**72.1**

Table K: **RelNormal (↓).** Values are normalized to $[0, 1]$ following Wu et al. [66].

Method	ETH3D	iBims-1	GSO	Sintel	DIODE
DUSt3R [59]	0.258	0.322	0.346	0.510	0.240
Depth Pro [5]	0.206	0.246	0.195	0.376	0.227
MapAnything [24]	0.246	0.307	0.218	0.484	0.270
VGGT [53]	0.209	0.319	0.140	0.385	0.167
InfiniDepth [72]*	0.233	0.268	0.223	0.419	0.254
Depth Anything 3 [28]	0.218	0.267	0.178	0.407	0.192
PPD [67]*	0.631	0.387	0.248	0.577	0.611
UniDepthV2 [37]	0.212	0.251	0.202	0.361	0.227
π^3 [61]	0.214	0.277	0.169	0.422	0.216
MoGe-2 [58]	0.193	0.232	0.149	0.331	0.163
MoGe [57]	0.189	0.242	0.143	0.327	0.139
SurGe	**0.183**	**0.229**	**0.137**	**0.315**	**0.137**

G Additional Qualitative Examples

Figures B and C compare SurGe with recent methods on in-the-wild scenes with large depth ranges. PPD [67] and InfiniDepth [72] struggle most in this setting: both recover plausible near-field geometry but strongly compress distant scene structure. Their normal maps further show strong surface noise for PPD and visible surface artifacts for InfiniDepth. VGGT [53] is less affected but still produces broken near-field geometry or slightly compressed distant structure. MoGe [57] and MoGe-2 [58] are closest to SurGe in global layout, but their rendered point maps and normals show less coherent local surface geometry, including bending and oscillatory artifacts.

Figures D to F show additional in-the-wild predictions from SurGe.

H Failure Cases

The qualitative comparisons show substantial reductions in bending, oscillation, and surface noise, consistent with the improvements in the local metrics. However, these artifacts are not entirely eliminated: in complex scenes with many simultaneous high-frequency geometry changes, we still observe occasional local artifacts. Such scenes place evidence from several surfaces in the same local neighborhoods. The decoder ablation in Tabs. 4 and 5 and Fig. 4 suggests that NAD's content-dependent feature mixing helps separate this evidence, but fully disentangling many nearby structures remains challenging.

Floaters around occlusion boundaries are another recurring failure mode, although SurGe produces substantially fewer than the compared baselines, as visible in Fig. 1. All qualitative results shown in this work use raw predictions without floater removal or other geometric cleanup. SMD-Nets [48] and MDA [4] address this ambiguity by estimating multimodal disparity and depth distributions, respectively. These output formulations are orthogonal to NAD and could be combined with it.

A further limitation comes from the training data. Real-world datasets are necessary for strong zero-shot accuracy, but their SfM reconstructions often omit moving objects such as pedestrians or absorb them into the background. Training on these labels can suppress fine structures and cause foreground objects to bleed into the predicted background geometry. Refining real-world annotations is therefore a complementary direction for improving local geometry.

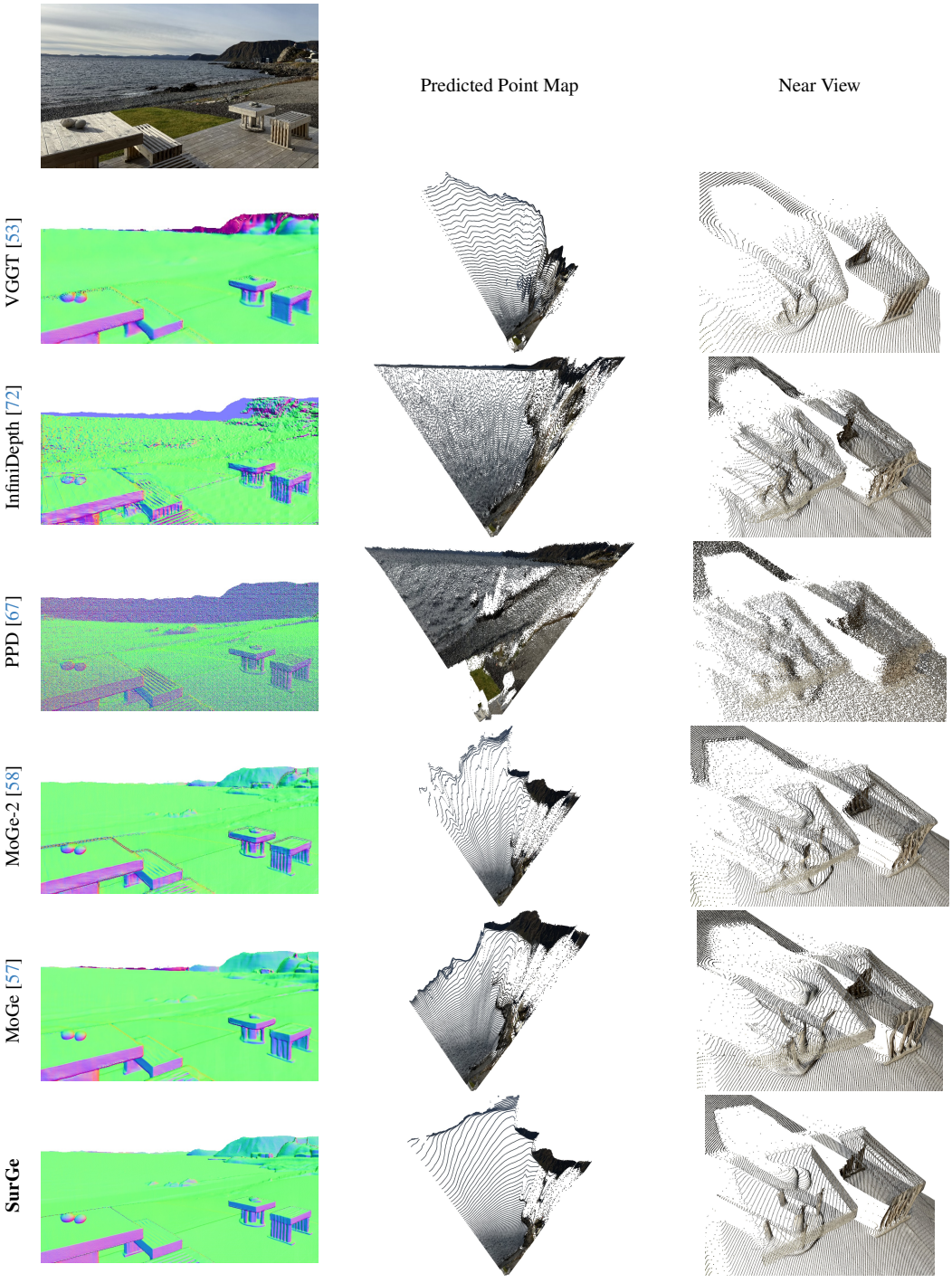


Figure B: **Qualitative comparison of state-of-the-art methods in a high-dynamic-range scene.** The top-left panel shows the input image. Each row compares one method; columns show point map normals, followed by bird's-eye and near-camera renderings of the predicted point map. VGGT, InfiniDepth, and PPD struggle with the high dynamic range and collapse distant geometry. VGGT oversmooths surfaces, InfiniDepth, MoGe-2, and MoGe introduce oscillatory or bending artifacts on the table and bench legs, and PPD produces heavy surface noise. SurGe exhibits the best local surface geometry with sharp edges and straight surfaces. Best viewed zoomed in.

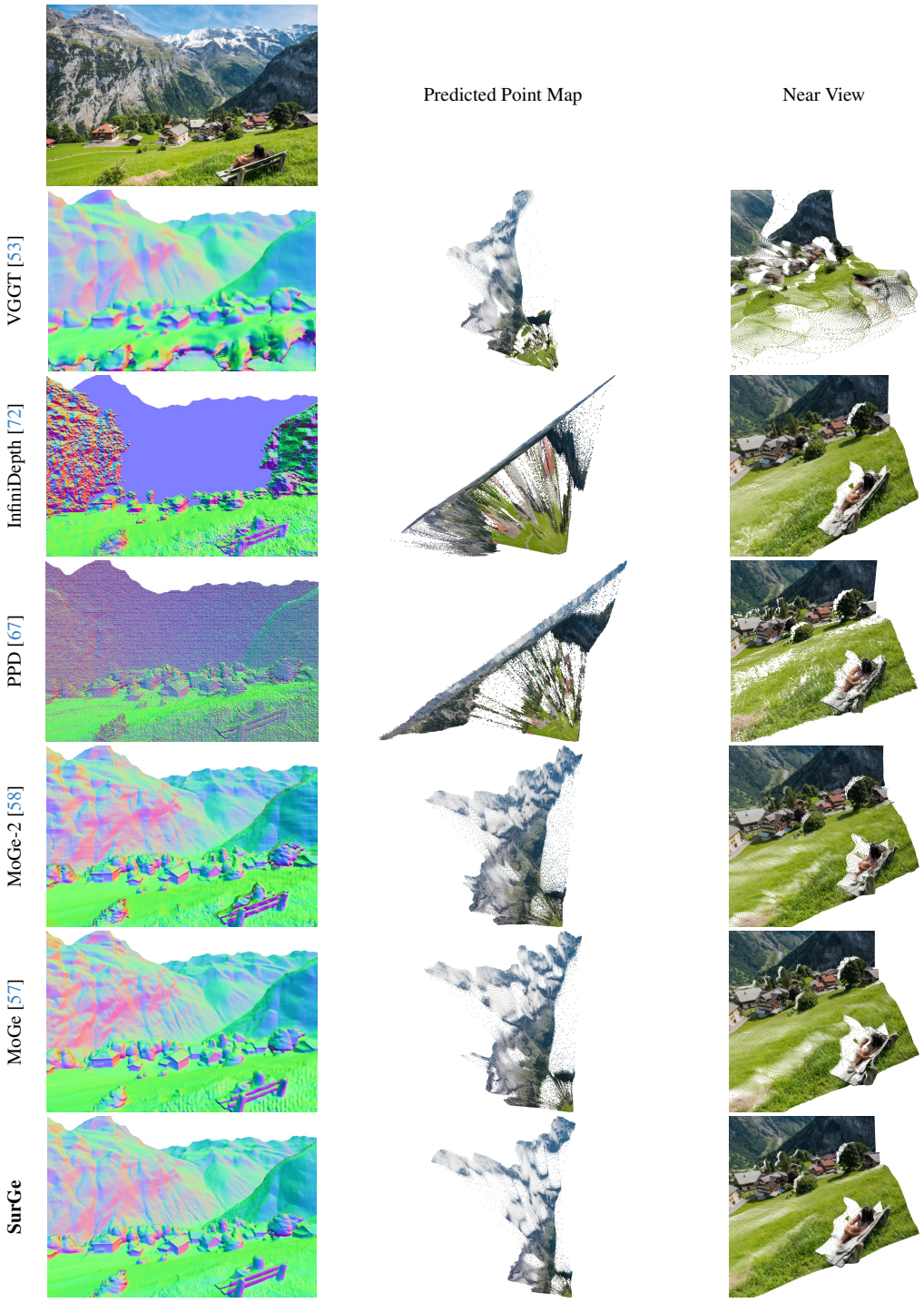


Figure C: **Qualitative comparison of state-of-the-art methods in a high-dynamic-range scene.** The top-left panel shows the input image. Each row compares one method; columns show point map normals, followed by bird's-eye and near-camera renderings of the predicted point map. InfiniDepth and PPD struggle with the high dynamic range, VGGT fails to reconstruct geometry close to the camera, and SurGe produces cleaner local surface geometry than MoGe. Best viewed zoomed in.



Figure D: **Additional qualitative results from SurGe.** Each example includes the RGB input, predicted depth, point map normals, and rendered point map.

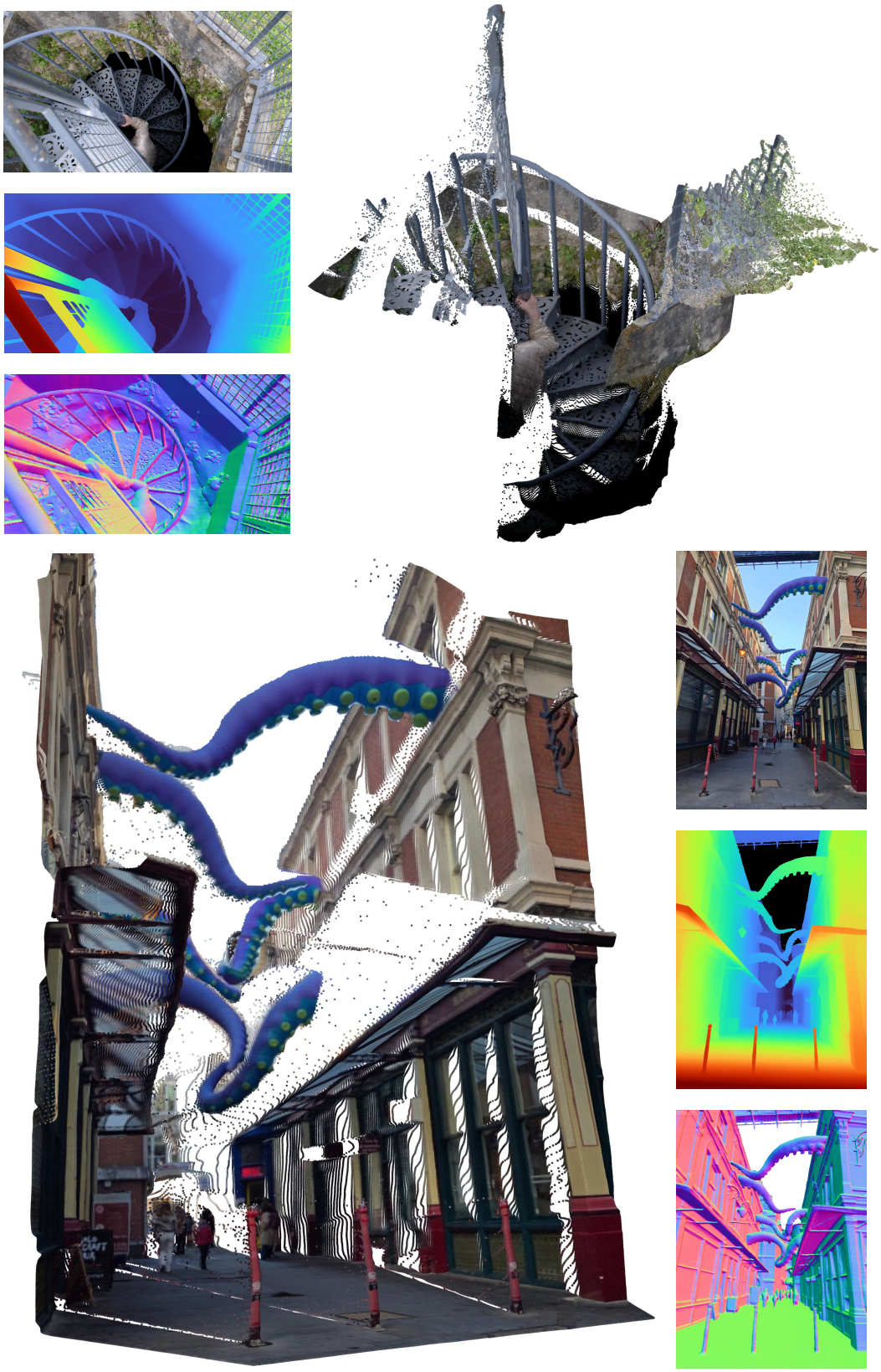


Figure E: **Additional qualitative results from SurGe.** Each example includes the RGB input, predicted depth, point map normals, and rendered point map.

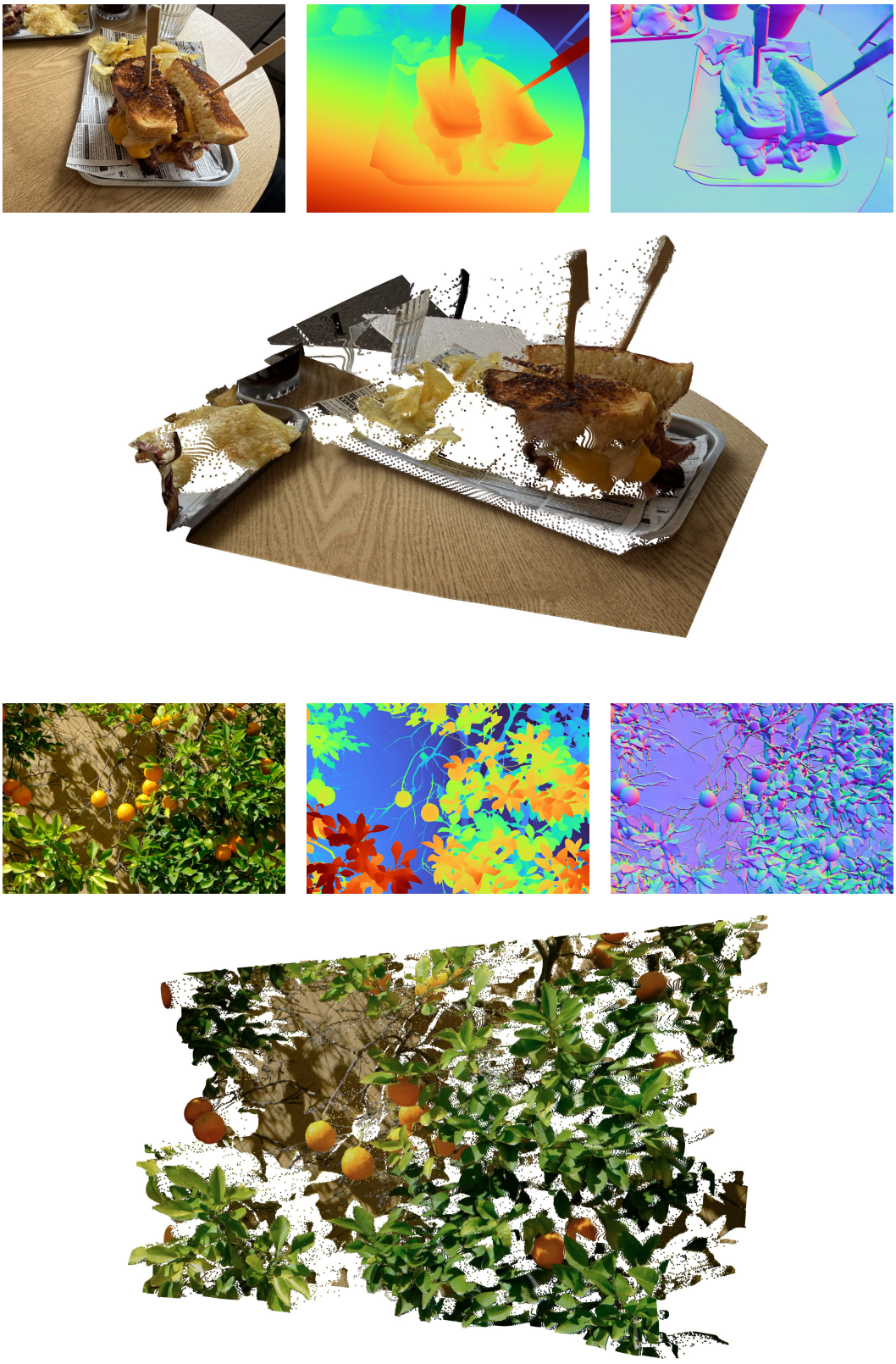


Figure F: **Additional qualitative results from SurGe.** Each example includes the RGB input, predicted depth, point map normals, and rendered point map.

I Asset Licenses

Tables L to N link to licenses and usage terms for the datasets and pretrained weights. "Unknown" means that we could not find an explicit license for the dataset or weights.

Table L: **Training dataset licenses.**

Dataset	License / terms
Argoverse2	CC BY-NC-SA 4.0
ARKitScenes	Apple terms
BlendedMVS	CC BY 4.0
G-Objaverse	MIT*
Objaverse	ODC-By v1.0*
GTA-SfM	Unknown
Hypersim	CC BY-SA 3.0
IRS	Apache 2.0†
Ken Burns	CC BY-NC-SA 4.0
MatrixCity	CC BY-NC 4.0
MegaDepth	CC BY 4.0*
MidAir	CC BY-NC-SA 4.0
MVS-Synth	Research terms
Structured3D	Research terms
Synthia-AL	CC BY-NC-SA 3.0
Synthia-SF	CC BY-NC-SA 3.0
TartanAir	CC BY 4.0
Taskonomy	Dataset EULA
UnrealStereo4K	MIT
UrbanSyn	CC BY-SA 4.0‡
Waymo	Noncommercial terms

* Individual Objaverse objects and MegaDepth source images retain their own licenses.
† Applies to the IRS repository; coverage of the separately hosted dataset is unclear.
‡ Additional restrictions apply to underlying assets.

Table M: **Evaluation dataset licenses.**

Dataset	License / terms
NYUv2	Unknown
KITTI	CC BY-NC-SA 3.0
ETH3D	CC BY-NC-SA 4.0
iBims-1	CC BY 4.0
GSO	CC BY 4.0
Sintel	Noncommercial research / education
DDAD	CC BY-NC-SA 4.0
DIODE	MIT
HAMMER	MIT
Spring	CC BY 4.0

Table N: **Pretrained weight licenses.**

Model	License / terms
DINOv2	Apache 2.0
MoGe	MIT
MoGe-2	MIT
DUSt3R	CC BY-NC-SA 4.0*
DepthPro	Apple AMLR
MapAnything	CC BY-NC 4.0
VGGT-1B	CC BY-NC 4.0
InfiniDepth	Apache 2.0
VGGTΩ	FAIR Noncommercial Research
DA3	CC BY-NC 4.0
PPD	Apache 2.0
UniDepthV2	Unknown
π^3	CC BY-NC 4.0

* Training data and base checkpoint licenses also apply.

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: Section 4 provides empirical evidence for the claims made in the abstract & 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: We mention limitations of our method in Sec. 5.

   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 contain theoretical results.

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: We describe the core model architecture in Sec. 3.1 with additional details in Sec. B, training recipe in Sec. 4.1 with additional details in Sec. C, and dataset mixture used for training in Sec. D.

   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: [No]

Justification: We release inference and evaluation code, pretrained weights, and instructions for obtaining evaluation data at https://github.com/karimknaebel/surge. Although the full training pipeline and scripts for reproducing all ablations are not released, reproduction should be possible using the details provided in Secs. C, D and 4.1.

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: We describe the core model architecture in Sec. 3.1 with additional details in Sec. B, training recipe in Sec. 4.1 with additional details in Sec. C, and dataset mixture used for training in Sec. D.

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: [No]

Justification: Error bars are not reported because it would be too computationally expensive.

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: We report GPU type, count, and training time per full run in Sec. 4.1, and inference latency and peak memory in Sec. E. We do not report aggregate compute for the full project, including preliminary experiments.

   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: We believe this research fully conforms with the NeurIPS Code of Ethics.

   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: [No]

    Justification: We believe that this research has little direct societal impact.

    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 poses no 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: We credit the datasets, models, and methods used and report identified licenses and usage terms in Sec. I. Unavailable or ambiguous license information is explicitly indicated.

    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 released code and pretrained weights include installation and usage instructions and license information at `https://github.com/karimknaebel/surge` and `https://huggingface.co/karimknaebel/surge-large`. The code is released under the MIT license and the weights under CC BY-NC 4.0; training details and limitations are described in the linked paper.

    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 nor 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 crowdsourcing nor research with human subjects.

    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: The method introduced in the paper is not related to LLMs.

    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.