======Supplementary======
Revisit Self-supervised Depth Estimation with Local Structure-from-Motion

Shengjie Zhu and Xiaoming Liu

Department of Computer Science and Engineering,
Michigan State University, East Lansing, MI, 48824
zhusheng@msu.edu, liuxm@cse.msu.edu

1 Extended Methodology

1.1 Hough Transform with 3D Scoring Function

In Sec. 4.3 and Fig. 9, with RGB-D inputs, we utilize the 3D scoring function Eq. (6). For a pixel $\mathbf{p}_i$ on frame i, denote its backprojected 3D ray on the image coordinate system of frame j as $\hat{\mathbf{l}}_i$. The ray $\hat{\mathbf{l}}_i$ is determined by two 3D points:

$$\hat{\mathbf{p}}_1 = \mathbf{R}_{i,j}\mathbf{K}^{-1}\mathbf{p}_i + \mathbf{t}_{i,j}, \quad \hat{\mathbf{p}}_2 = \mathbf{t}_{i,j}. \tag{1}$$

The depth of 3D point $\hat{\mathbf{p}}_1$ can be set to arbitrary values. We set it to 1 for convenience. 3D point $\hat{\mathbf{p}}_2$ is the frame i camera origin on frame j. Correspondingly, the backprojected 3D point $\hat{\mathbf{p}}_\pi$ is:

$$\hat{\mathbf{l}}_i^\intercal \hat{\mathbf{p}}_\pi = 0, \ \hat{\mathbf{p}}_\pi = \pi^{-1}(s_i, s_j, r_i \mid \overline{\mathbf{P}}_i, \overline{\mathbf{P}}_j, d_i). \tag{2}$$

To be an inlier of the scoring function $f^{\text{3D}}(\cdot)$, we have:

$$\|\hat{\mathbf{p}}_\pi - \hat{\mathbf{p}}_j\|_2 \leqslant \lambda^{\text{3D}}, \ \hat{\mathbf{p}}_j = \pi^{-1}(s_i, s_j, r_j \mid \overline{\mathbf{P}}_i, \overline{\mathbf{P}}_j, d_j). \tag{3}$$

Eq. (3) suggests a 3D line and sphere intersection, with the two end-points denoted as $\hat{\mathbf{p}}_\pi^{\text{st}}$ and $\hat{\mathbf{p}}_\pi^{\text{ed}}$. Compare 2D intersection condition Eq. (11) with 3D intersection condition Eq. (3), we find Eq. (3) further relates to the depth adjustment r_j on frame j. Similarly, compare Fig. 5 with Fig. 1, the circle center of 2D scoring function is independent of depth adjustments r_j. The sphere center of the 3D scoring function, however, changes *w.r.t.* depth adjustments. This introduces one additional variable in Hough Transform, *i.e.*, the depth adjustment r_j at frame j, also shown in Fig. 1. The additional dimensionality creates an excessively high time and space complexity excessively for Hough Transform. Hence, we are unable to optimize depth adjustment in 3D scoring function. One viable solution is to first execute with 2D scoring function, fix the adjustments, and then employ the 3D scoring function. We did not experiment with this strategy since we already had competitive performance. When with RGB-D inputs, we use 3D scoring function as groundtruth depth does not require any scale adjustment. The rest is similar to Sec. 3.1.2 and skipped.

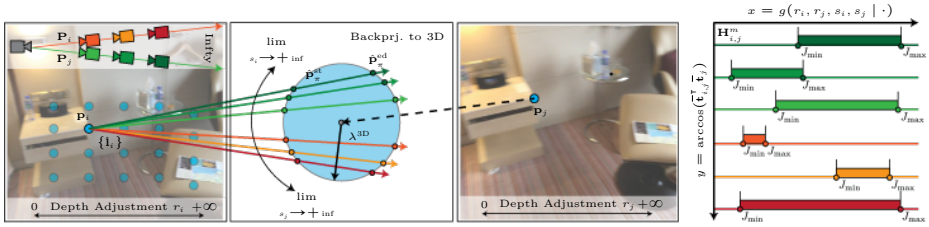


Fig. 1: Two-view Hough Transform on 3D Scoring Function. Pixels $\mathbf{p}_i$ and $\mathbf{p}_j$ are corresponded. Similar to Fig. 5, ablating pose scales map pixel $\mathbf{p}_i$ to a set of 3D rays originated from camera origin, denoted as $\{\hat{\mathbf{l}}_i\}$. To be an inlier, a backprojected 3D point $\hat{\mathbf{p}}_\pi$ has to reside within a sphere centered with $\hat{\mathbf{p}}_j$ with a radius λ^{3D}, *i.e.*, between 3D segments $\hat{\mathbf{p}}_\pi^{\text{st}}$ and $\hat{\mathbf{p}}_\pi^{\text{ed}}$. Different from Fig. 5, with fixed normalized poses, there exists **four** variables to optimize, including the additional frame j depth adjustment r_j.

1.2 Proof of Eq. (8).

From Eq. (7), the relative pose translation magnitude $s_{i,j}$ is:

$$s_{i,j}^2 = s_i^2 + s_j^2 + k_\theta \cdot s_i s_j,\ k_\theta = 2\bar{\mathbf{t}}_j^{\intercal}\mathbf{R}_j\mathbf{R}_i^{-1}\bar{\mathbf{t}}_i. \tag{4}$$

The relative pose normalized translation vector $\bar{\mathbf{t}}_{i,j}$ is:

$$\bar{\mathbf{t}}_{i,j} = -s_i/s_{i,j}\mathbf{R}_j\mathbf{R}_i^{-1}\bar{\mathbf{t}}_i + s_j/s_{i,j}\bar{\mathbf{t}}_j. \tag{5}$$

Illustrate with s_i, when $s_i \to +\infty$, we have:

$$\lim_{s_i \to +\inf} s_i/s_{i,j} = 1,\ \lim_{s_i \to +\inf} s_j/s_{i,j} = 0. \tag{6}$$

Combined, we have:

$$\lim_{s_i \to +\inf} \bar{\mathbf{t}}_{i,j} = -\mathbf{R}_j\mathbf{R}_i^{-1}\bar{\mathbf{t}}_i. \tag{7}$$

The other case of Eq. (8) follows a similar logic.

1.3 Proof of Eq. (12) and Eq. (13).

1.3.0.1 Notations. For simplicity, the relative pose $\mathbf{P}_{i,j}$ between frame i and j is defined as $\mathbf{P}_{i,j} = \left[\mathbf{R}\ s \cdot \bar{\mathbf{t}}\right]$. Denote the 2D pixel locations on frame i as $\mathbf{p}$. Its projection is set to $\mathbf{q}$, *i.e.*, the variable $\mathbf{p}_\pi$ in Sec. 3.1.2. The projection is:

$$d'\mathbf{q} = d'\begin{bmatrix} q_x & q_y & 1 \end{bmatrix}^{\intercal} = r \cdot d\mathbf{K}\mathbf{R}\mathbf{K}^{-1}\mathbf{p} + s\mathbf{K}\bar{\mathbf{t}}. \tag{8}$$

Variable r is the depth adjustment on frame i. Variable d is the depth of pixel $\mathbf{p}$. The pixel location q_x and q_y are:

$$q_x = \frac{r \cdot d\mathbf{m}_1^{\intercal}\mathbf{p} + s \cdot x}{r \cdot d\mathbf{m}_3^{\intercal}\mathbf{p} + s \cdot z},\ q_y = \frac{r \cdot d\mathbf{m}_2^{\intercal}\mathbf{p} + s \cdot y}{r \cdot d\mathbf{m}_3^{\intercal}\mathbf{p} + s \cdot z}. \tag{9}$$

The remaining variables are defined as follows:

$$\left[\mathbf{m}_1\ \mathbf{m}_2\ \mathbf{m}_3\right]^\intercal = \mathbf{K}\mathbf{R}\mathbf{K}^{-1},\ s\cdot\left[x\ y\ z\right]^\intercal = s\cdot\mathbf{K}\bar{\mathbf{t}}. \tag{10}$$

Here, we slightly abuse the notation of x and y without implying the Hough Transform locations. The mapping function $J(\cdot)$ from projection to camera scale is hence:

$$s = J(\overline{\mathbf{P}}, r\cdot d, \mathbf{q}) = r\cdot\frac{d(\mathbf{m}_1^\intercal\mathbf{p} - q_x\mathbf{m}_3^\intercal\mathbf{p})}{z\cdot q_x - x}. \tag{11}$$

There exists another analytical mapping from the axis-y direction. We skip this for simplicity.

Proof of Corollary 2. From Eq. (11), it is obvious that:

$$s = J(\overline{\mathbf{P}}, r\cdot d, \mathbf{q}) = r\cdot J(\overline{\mathbf{P}}, d, \mathbf{q}). \tag{12}$$

Proof of Corollary 1. We decompose the relative camera pose $\mathbf{P}$ from frame i and j into a rotation movement $\mathbf{P}^r$ followed by a translation movement $\mathbf{P}^t$:

$$\mathbf{P}^r = \left[\mathbf{R}\ \mathbf{0}\right],\ \mathbf{P}^t = \left[\mathbf{E}\ s\cdot\mathbf{R}\bar{\mathbf{t}}\right],\ \mathbf{P} = \mathbf{P}^t\mathbf{P}^r. \tag{13}$$

Correspondingly, we introduce an intermediate pixel $\mathbf{q}^r$ as a consequence of the pure rotation movement. That is to say, between pixels $\mathbf{p}$ and $\mathbf{q}^r$ are a pure rotation movement. And between pixels $\mathbf{q}^r$ and $\mathbf{q}$ are a pure translation movement. The pixels $\mathbf{q}^r$ and $\mathbf{q}$ have the following relationship:

$$\mathbf{q}^r = \lim_{s\to 0}\mathbf{q} = \lim_{r\cdot d\to+\infty}\mathbf{q}. \tag{14}$$

Eq. (14) is obvious. We skip its proof. We introduce it to illustrate two facts. First, from Fig. 5, the intersection point of all epipolar lines $\{\mathbf{l}_i\}$ is $\mathbf{q}^r$, *i.e.*, the projected pixel from pure rotation. Second, the pixel $\mathbf{q}^r$ and $\mathbf{q}$ formulates a pure translation movement. It simplifies our proof if we replace the pixel $\mathbf{p}$ to pixel $\mathbf{q}^r$ in Eq. (9). Due to a pure translation movement, the $\mathbf{K}\mathbf{R}\mathbf{K}^{-1}$ is an identity matrix. We have:

$$q_x = \frac{r\cdot d^r\cdot q_x^r + s\cdot x}{r\cdot d^r + s\cdot z},\ q_y = \frac{r\cdot d^r\cdot q_y^r + s\cdot y}{r\cdot d^r + s\cdot z}. \tag{15}$$

We compute the squared magnitude of the vector $\|\overrightarrow{\mathbf{q}^r\mathbf{q}}\|_2^2$:

$$\begin{aligned}\|\overrightarrow{\mathbf{q}^r\mathbf{q}}\|_2^2 &= (q_x - q_x^r)^2 + (q_y - q_y^r)^2 \\ &= \frac{(s\cdot x)^2 + (s\cdot y)^2}{(r\cdot d^r + s\cdot z)^2} = \frac{x^2 + y^2}{(r\cdot d^r/s + z)^2}.\end{aligned} \tag{16}$$

Meanwhile, we underlyingly require the pixel $\mathbf{q}$ to be visible in both frames. This requires a positive depth:

$$r\cdot d^r/s + z > 0. \tag{17}$$

Combining Eq. (16) and Eq. (17), we conclude that the vector $\|\overrightarrow{\mathbf{q}^r\mathbf{q}}\|_2^2$ monotonously increases as translation magnitude s increases. It fits the intuition that the projected pixel $\mathbf{q}$ moves further as the camera translation magnitude increases. To prove the corollary 1, we have:

$$\|\overrightarrow{\mathbf{q}^r\mathbf{q}^{\text{st}}}\|_2^2 \leqslant \|\overrightarrow{\mathbf{q}^r\mathbf{q}}\|_2^2 \leqslant \|\overrightarrow{\mathbf{q}^r\mathbf{q}^{\text{ed}}}\|_2^2. \tag{18}$$

The s monotonously increases with vector magnitude, then:

$$J(\overline{\mathbf{P}}, r \cdot d, \mathbf{q}^{\text{st}}) \leqslant J(\overline{\mathbf{P}}, r \cdot d, \mathbf{q}) \leqslant J(\overline{\mathbf{P}}, r \cdot d, \mathbf{q}^{\text{ed}}). \tag{19}$$

We apologize for the abuse of the notation. The variable $\mathbf{q}^{\text{ed}}$ refers to $\mathbf{p}_\pi$ in 3.1.2. We next address some corner cases unattended in main paper due to space issues.

- The Epipolar line and the circle do not intersect:
  We exclude Hough Transform under this condition since there are no inlier projected pixels.
- Eq. (17) does not hold:
  To be visible in both views, Eq. (17) has to hold. Hence it provides an additional bound to $J_{\min}$ and $J_{\max}$, if needed.

Compute Start and End Intersected Points. To compute $\mathbf{p}_\pi^{\text{st}}$ and $\mathbf{p}_\pi^{\text{ed}}$ in Sec. 3.1.2, we follow the analytical line-circle intersection solution.

2 Extended Experiments

2.1 Implementation Details

Sec. 3.1 Camera Pose Estimation. The proposed pose estimation algorithm runs on GPU devices. Per frame pair, we extract the top $K = 128$ normalized poses to formulate the candidate pool $\overline{\mathcal{Q}}$. We sample $M = 10,000$ points per frame pair. We exclude correspondence with a confidence lower than 0.2. We set λ^{2D} to 2 pixels and λ^{3D} to 0.025 meter. The Hough matrix $\mathbf{H}$ resolution is set to 100×200. The maximum Hough transform on axis-x, *i.e.*, the $x_{\max}$ is set to 1 meter. The BA optimization iterations are set to $T = 200$ at a learning rate of $5e^{-4}$ with an Adam optimizer.

Sec. 3.2 Triangulation. For the frustum radiance field $\mathbf{V}$, we configure its resolution as $240 \times 320 \times 128$ given a ScanNet image resolution of 480×640. We scale the depth consistent loss L_D by a factor of 0.01. This is due to the small magnitude of correspondence consistent loss L_C which is defined over normalized pixel coordinates, i.e., pixels divided by focal length. In triangulation, we optimize the Radiance Field $\mathbf{V}$ using the Adam optimizer with a learning rate of $1e^{-4}$ over $80,000$ iterations. In testing, we set the geometric verification threshold λ^c to 0.01 meter.

Sec. 3.3 Geometric Verification. The consistent threshold λ^c is set to 0.01 meters, and the minimum consistent view number n^c is set to 2 frames.

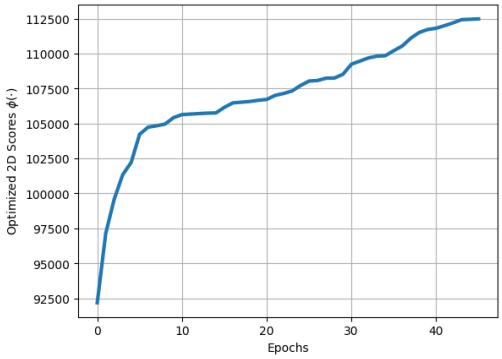
Table 1: Extended Sparse-view Pose Comparison with optimization-based and learning-based methods on ScanNet. We extend main paper Tab. 4 across 3 to 9 frames.

Frames	Method	Zero-shot	Suc. (%)	PCK-3	C3D-3	Rot.	Trans.
3	COLMAP [48]	✓	10.0	0.379	0.686	1.055	1.765
	Ours	✓	100.0	**0.750**	**0.890**	**0.210**	**0.624**
	DeepV2D [55] - ScanNet	✗	100.0	0.591	0.863	0.319	0.907
	DeepV2D [55] - NYUv2	✓		0.619	0.839	0.381	0.946
	LightedDepth [77]	✓		0.742	0.874	0.380	1.127
	DRO [22] - ScanNet	✗		0.757	0.883	0.368	0.881
	DRO [22] - KITTI	✓		0.006	0.258	2.043	3.435
	DUSt3R [61] *w.o.* Intrinsic	✓		0.409	0.770	0.356	1.355
	DUSt3R [61] *w.t.* Intrinsic	✓		0.574	0.827	0.467	1.245
	Ours	✓		**0.858**	**0.918**	**0.275**	**0.820**
5	COLMAP [48]	✓	36.7	0.584	0.863	0.577	1.296
	Ours	✓	100.0	0.727	0.904	0.422	1.062
	DeepV2D [55] - ScanNet	✗	100.0	0.526	0.805	0.945	1.496
	DeepV2D [55] - NYUv2	✓		0.530	0.771	1.041	1.568
	DeepV2D [55] - KITTI	✓		0.125	0.387	4.908	4.231
	LightedDepth [77]	✗		0.651	0.832	0.469	1.550
	DRO [22] - ScanNet	✗		0.656	0.853	0.385	1.200
	DRO [22] - KITTI	✓		0.003	0.211	3.610	5.469
	DUSt3R [61] *w.o.* Intrinsic	✓		0.364	0.705	0.487	2.074
	DUSt3R [61] *w.t.* Intrinsic	✓		0.594	0.824	0.570	1.759
	Ours	✓		**0.799**	**0.900**	**0.368**	**1.120**
7	COLMAP [48]	✓	70.6	0.571	0.852	0.596	1.521
	Ours	✓	100.0	**0.723**	**0.881**	**0.470**	**1.383**
	DeepV2D [55] - ScanNet	✗	100.0	0.395	0.667	1.719	2.673
	DeepV2D [55] - NYUv2	✓		0.434	0.674	1.830	2.740
	LightedDepth [77]	✓		0.560	0.801	0.551	1.909
	DRO [22] - ScanNet	✗		0.567	0.790	0.619	1.744
	DRO [22] - KITTI	✓		0.002	0.191	5.040	7.375
	DUSt3R [61] *w.o.* Intrinsic	✓		0.343	0.659	0.558	2.428
	DUSt3R [61] *w.t.* Intrinsic	✓		0.558	0.804	0.561	2.017
	Ours	✓		**0.739**	**0.873**	**0.458**	**1.428**
9	COLMAP [48]	✓	86.7	0.558	0.812	0.756	2.281
	Ours	✓	100.0	**0.687**	**0.852**	**0.538**	**1.675**
	DeepV2D [55] - ScanNet	✗	100.0	0.303	0.569	4.279	4.573
	DeepV2D [55] - NYUv2	✓		0.349	0.590	5.509	4.654
	LightedDepth [77]	✓		0.487	0.766	0.631	2.322
	DRO [22] - ScanNet	✗		0.483	0.710	1.313	3.093
	DRO [22] - KITTI	✓		0.002	0.174	6.397	9.142
	DUSt3R [61] *w.o.* Intrinsic	✓		0.333	0.631	0.615	2.740
	DUSt3R [61] *w.t.* Intrinsic	✓		0.568	0.801	0.584	2.278
	Ours	✓		**0.695**	**0.850**	**0.535**	**1.689**

2.2 Additional Comparisons

Prior public benchmarks are inapplicable to measuring sparse-view pose estimation performance. DeepV2D [55], DRO [22], and LightedDepth [77] only report two-view relative pose performance. DUSt3R [61] reports on classic SfM and SLAM benchmarks with significantly more view variations. Only NeRF-based [58] methods benchmark sparse-view pose performance and have been included in the main paper Tab. 5. For a comprehensive comparison, we additionally experiment on the ScanNet with other SoTA methods.

In Tab. 1, we divide the comparisons into two groups. The group above the dashed lines only evaluates the COLMAP successfully executed sequences. The



(a) Inlier score $f^{\text{2D}}(\cdot)$ / ZoeDepth [5] / PDC-Net [57]

Fig. 2: Scores *w.r.t* Optimization Epochs. The inlier scores always rise throughout the optimization. With 5 frames, our algorithm terminates on average at 23.5 epochs. [Key: Inlier Score / Monodepth Estimator / Correspondence Estimator]

group below the dashed lines includes all ScanNet test sequences. For two-view methods LightedDepth [77] and DRO [22], we repetitively apply two-view pose estimation between each support and root frame. For DUSt3R [61] with intrinsic, we set the intrinsic and disable its update in optimization. In Tab. 1, we achieve **unanimous** improvement on all frame numbers with **substantial** margin.

2.3 Additional Ablation Studies

Scores *w.r.t* Optimization Epochs. As shown in Fig. 2, inlier scores consistently rise during optimization. Despite optimizing over discrete local optimal sets, *i.e.*, predefined candidate pool $\overline{\mathcal{Q}}$, the convergence curve exhibits similar trends with smooth manifold optimization. We attribute this to a sufficiently large candidate pool size, set at $K = 128$ per frame in our experiment.

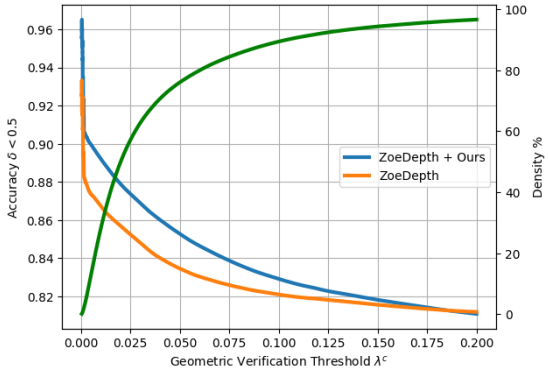
Geometric Verification Threshold. In Tab. 2, we benchmark the triangulated depth without geometric verification, *i.e.*, depth performance without filtering. On ZeroDepth [35], we outperform the supervised inputs even without geometric verification. On ZoeDepth [5] and Metric3D [68], we outperform the supervised inputs after the geometric verification, suggesting its necessity. Across all cases, geometric verification improves performance. In Fig. 3, we ablate different triangulation threshold λ^c values. From Fig. 3, our method maintains improvement at a much higher density than reported in the main paper Tab. 1.

2.4 Evaluation Metrics

Depth Related Metrics. In Tab. 1, we follow [4] for most evaluation metrics. As the accuracy metric δ_1 saturates, [43] introduces a stricter counterpart, $\delta_{0.5}$. The term $\text{SI}_{\log}$ is a scale-invariant metric [43]. As a result, the scale adjustment

Table 2: Ablation on Geometric Verification on ScanNet dataset. We additionally report the main paper Tab. 1 without applying the Sec. 3.3 Geometric Verification, *i.e.*, the radiance field intermediate depthmap with full density. See visual examples in the main paper Fig. 2 and supplementary Fig. 4.

Method	Geo. Ver.	Density	$\delta_{0.5}$	δ_1	SI_{log}	A.Rel	S.Rel	RMS	RMS_{log}
ZoeDepth [5]	✗	Dense	**0.808**	**0.937**	**9.592**	**0.074**	**0.028**	**0.211**	**0.102**
↳ Ours			0.800	0.925	11.006	0.082	0.032	0.227	0.118
ZoeDepth [5]	✓	9.1%	0.877	0.963	6.655	0.056	0.016	0.154	0.075
↳ Ours			**0.902**	**0.976**	**5.901**	**0.050**	**0.014**	**0.149**	**0.070**
ZeroDepth [35]	✗	Dense	0.557	0.771	20.124	0.166	0.118	0.427	0.211
↳ Ours			**0.595**	**0.809**	**17.744**	**0.147**	**0.094**	**0.388**	**0.190**
ZeroDepth [35]	✓	5.6%	0.641	0.834	12.860	0.124	0.086	0.337	0.152
↳ Ours			**0.686**	**0.877**	**9.463**	**0.106**	**0.067**	**0.295**	**0.133**
Metric3D [68]	✗	Dense	**0.747**	**0.905**	12.325	**0.089**	**0.043**	**0.259**	**0.129**
↳ Ours			0.746	0.883	**12.273**	0.109	0.089	0.306	0.147
Metric3D [68]	✓	2.6%	0.804	0.946	6.708	0.067	0.020	0.150	0.084
↳ Ours			**0.854**	**0.968**	**4.170**	**0.055**	**0.014**	**0.125**	**0.068**



(a) Accuracy $\delta < 0.5$ / ZoeDepth [5] / PDC-Net [57]

Fig. 3: Depth Density & Accuracy *w.r.t* Geometric Verification Threshold λ^c. Tab. 1 reports triangulated accuracy and density at $\lambda^c = 0.01$. From the plot, if set the threshold λ^c to a larger value, we maintain significant improvement while preserving a significantly larger density, *e.g.*, $\lambda^c = 0.05$. The unit is in meters. [Key: Accuracy $\delta < 0.5$ / Monodepth Estimator / Correspondence Estimator]

on support frames does not impact metric SI_{log}, as in main paper Tab. 1 row 5 and 6.

Correspondence Related Metrics. In Tab. 3, we follow [57] in evaluation metrics. The metric AEPE refers to average end-point-error, *i.e.*, the L2 norm between estimated and groundtruth correspondence. PCK-K refers to the percentage of correct correspondence if its end-point-error is smaller than a given pixel threshold K.

Pose Related Metrics. In Tab. 4 and Tab. 5, we follow [58] in using Rot. and Trans.. The metric Rot. measures the average difference in degree after alignment. Similarly, metric Trans. measures the translation magnitude after multi-

plied by $\times 100$. We additionally report PCK-K and C3D-K. The metric PCK-K compares the projected correspondence using the groundtruth depthmaps. Its accuracy is only related to the estimated camera poses. The threshold is set to 3 pixels Similarly, the metric C3D-K compares the backprojected 3D points using the groundtruth depthmap with estimated poses. The threshold is set to 5 centimeters.

2.5 Visualization

We additionally visualize the self-supervised depth estimation, consistent depth estimation, and self-supervised correspondence estimation in Fig. 4, Fig. 5, and Fig. 6, respectively. Please see the captions for detailed analysis.

Self-supervised Depth Estimation. In Fig. 4, the Radiance Field (RF) rendered dense depthmap has poor visual quality, especially around object boundaries. This is expected, as the RF essentially applies triangulation, relying on consistent multi-view depth and correspondence estimation. Yet, both monocular depthmap and correspondence estimation exhibit lower quality around borders. When the inconsistency is significant, the triangulated results become underdefined, leading to noisy artifacts.

Self-supervised Consistent Depth. When generating Fig. 5, we produce groundtruth correspondence via groundtruth pose and depthmap. Using this correspondence, we composite multi-frame depthmaps with groundtruth correspondence depthmaps from various frames and subsequently backproject them to the root frame. Each column corresponds to sampling from a different frame's depthmap, and the final image is composed over all input frames.

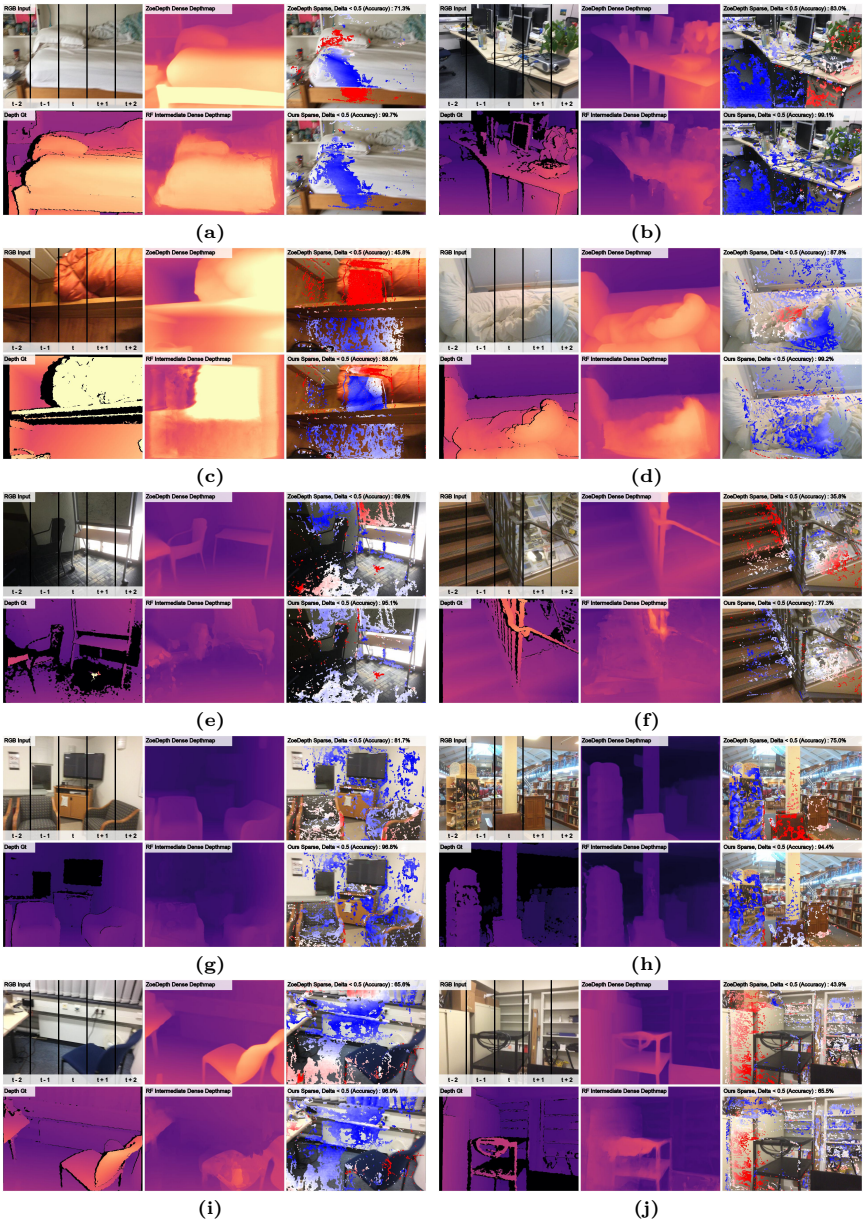


Fig. 4: Self-supervised Depth Estimation. Qualitative comparison with supervised model ZoeDepth [5]. We represent the depth quality of each pixel with the metric **A.Rel**. Blue to red scatter indicates small to large errors, ranging between 0.0 and 0.2.

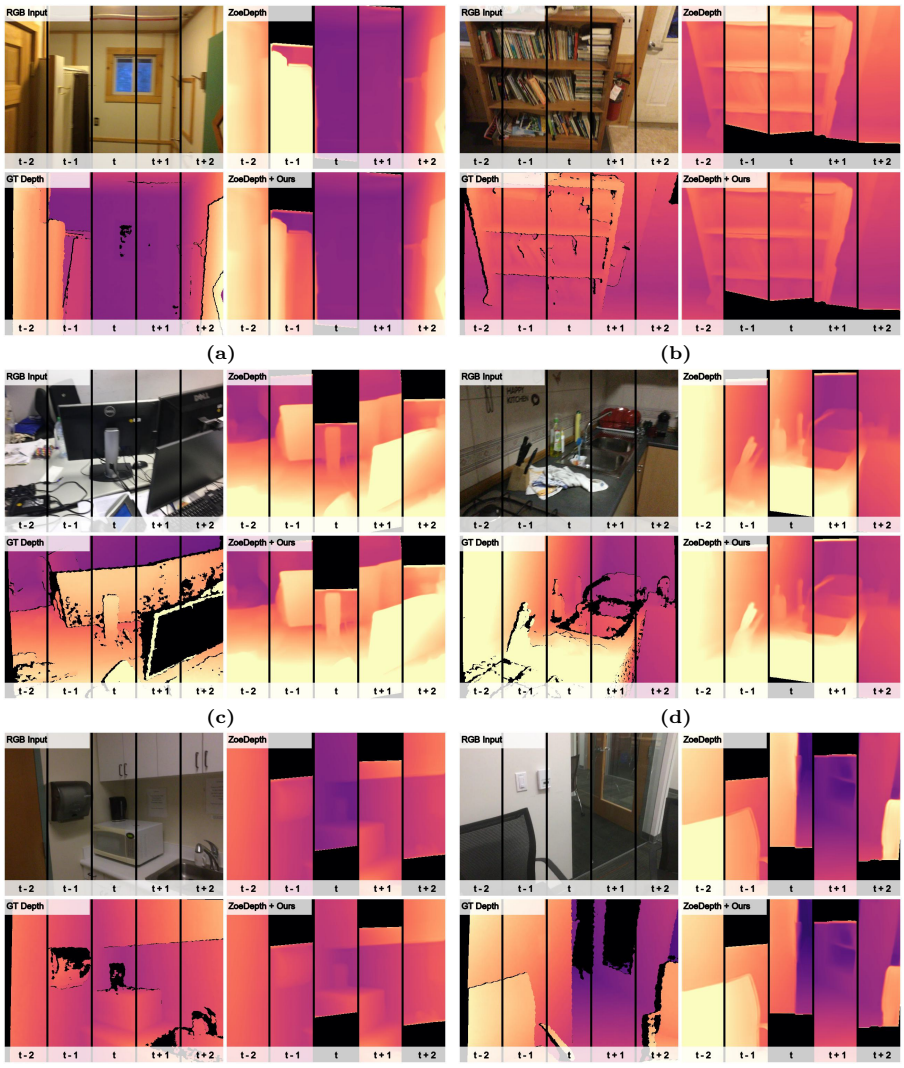


Fig. 5: Consistent Depth Estimation. Qualitative comparison with input SoTA monocular depth estimator ZoeDepth [5].

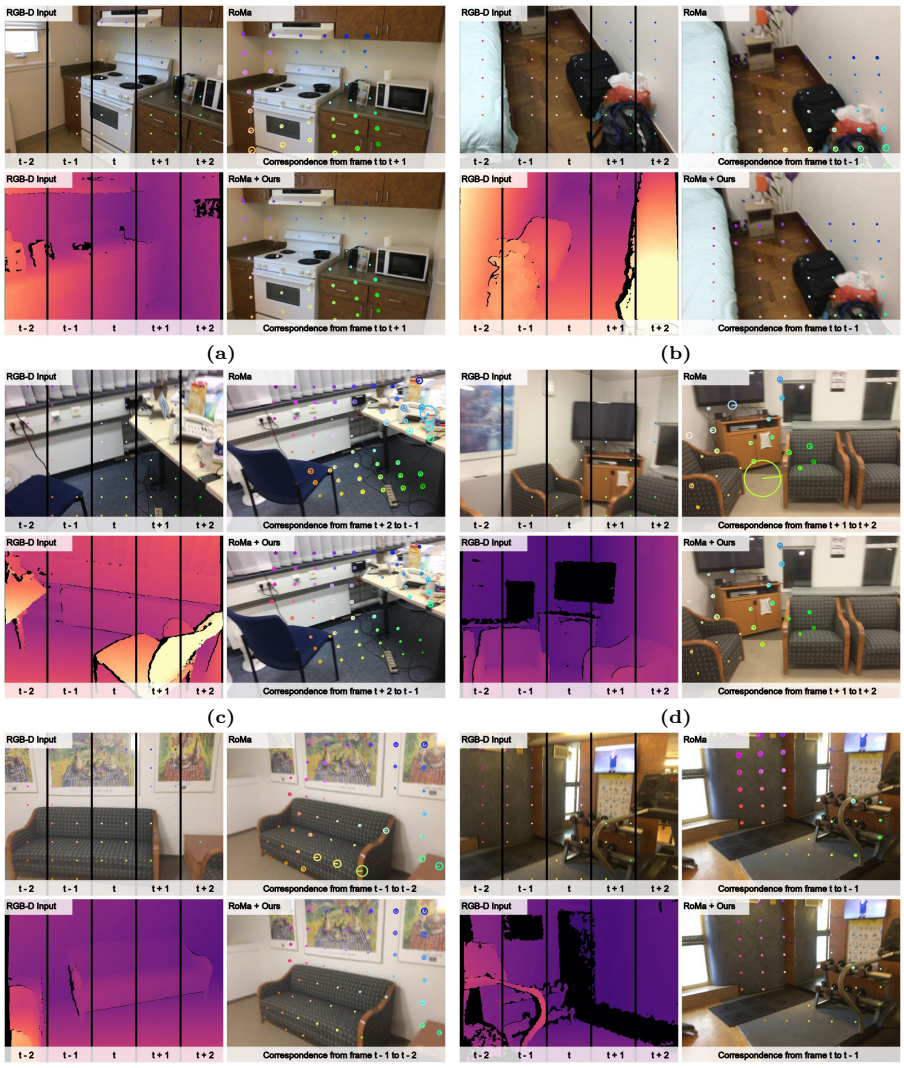


Fig. 6: Self-supervised Correspondence Estimation. Qualitative comparison with input SoTA correspondence estimator RoMa [15].