Supplementary Document for Efficient PrivacyPreserving Visual Localization Using 3D Ray Clouds

Heejoon Moon
Department of Artificial Intelligence
Hanyang University
wilko97@hanyang.ac.kr

Chunghwan Lee
Department of Electronic Engineering
Hanyang University
jhlee612@hanyang.ac.kr

Je Hyeong Hong[†]
Department of Artificial Intelligence & Electronic Engineering
Hanyang University
jhh37@hanyang.ac.kr

Abstract

This supplementary document provides (1) ablation studies, (2) concrete discussions of revealing points in the ray cloud, (3) further discussions on experimental results, and (4) additional experimental results. Code is available at https://github.com/PHANTOM0122/Ray-cloud.

1. Ablation study

1.1. Comparing minimal solvers for ray clouds

In order to verify the effectiveness of the proposed minimal solver (*p*5+1R) compared to the existing minimal solver (*p*6L) [7], we compared the performance of two minimal solvers, with the same ray cloud and presented results in Table 1. We confirm that *p*5+1R solver surpasses *p*6L in pose accuracy and inference speed. We cautiously anticipate the reason for this lies in that Stewenius' 6-point solver [8] basically does not assume concentric generalized camera rays like a ray cloud. As concentric generalized rays can induce degeneracy in generalized essential matrix estimation, it can lead to trivial solutions as mentioned in [10]. On the other hand, the *p*5+1R solver utilizes a classical relative pose estimation solver [5] thereby achieving real-time inference speed without heavily compromising pose accuracy. Also, algorithmic details for pose estimation of ray cloud using *p*5+1R solver are described in our Algo. 1.

1.2. Changing distance between two ray centers

To set the position of center points of the ray cloud, we employed K-means clustering (K=2) and regarded the center of clusters as center points. The motivation for this came from the expectation that the centers of two clusters will not be close unless the 3D points are close together. Fig. 1 illustrates the empirical results of pose accuracy according to the baseline distance between two ray centers. We measured pose estimation errors by using the distance between two points determined by K-means clustering as the reference baseline distance and adjusting the distance between the two centers by reducing or increasing with various distance ratios from the reference distance.

Dataset	Solver	Rot. err. [°] ↓		Trans. err. [m] ↓		t [ms] ↓
		mean	median	mean	median	mean
Energy Landscape	*p*6L [7]	0.457	0.264	0.016	0.008	88
	*p*5+1R(ours)	**0.262**	**0.087**	**0.010**	**0.003**	**6**
Cambridge	*p*6L [7]	0.950	0.346	0.269	0.155	147
	*p*5+1R(ours)	**0.855**	**0.222**	**0.219**	**0.107**	**12**

Table 1. Effect of minimal solvers on localization with the same ray cloud. **Bold** results indicate the best result in each metric. Our proposed minimal solver *p*5+1R shows great improvements in inference speed and accuracy compared to the *p*6L solver [7].

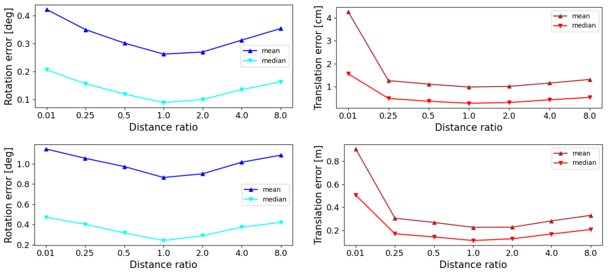


Figure 1. Effects of distance of baseline between two virtual perspective camera in the ray cloud in pose accuracy. (Top) Energy Landscape [9], (bottom) Cambridge [2] dataset.

We observe that when the baseline distance is extremely small (distance ratio = 0.01), rotation error and translation error increase compared to the results of the original base-

[†] Corresponding author

line results (distance ratio = 1.0). Particularly for translation, we confirm that, as the two center points get closer, the error increases significantly due to the increased ambiguity in the scale of the overall scene. On the other hand, as the distance increases, the pose errors in rotation and translation rise, which is thought to originate from more quasi-parallel rays. The empirical results in Fig. 1 indicate that using the centers from K-means clustering provides a "sweet spot" for ray cloud construction although estimating the optimal (trade-off) position requires further investigation.

1.3. Effect of changing voxel size in ray sampling

In [4], we expected a trade-off between localization accuracy and privacy-preserving performance in voxel-based ray clouds depending on the size of the voxel. To verify this, we estimated localization accuracy and privacy-preserving performance for different voxel sizes as illustrated in Table 2.

Dataset	Voxel size [m]	$\Delta R[°]$ mean	ΔT[cm] mean	e_g [m] median	PSNR mean	SSIM mean	MAE mean
Energy landscape	None	**0.262**	**0.99**	0.669	10.29	0.319	62.97
	0.1	0.267	1.01	0.685	10.26	0.317	63.42
	0.25	0.277	1.03	0.723	10.15	0.314	64.00
	0.5	0.297	1.09	**0.770**	**10.05**	**0.315**	**64.84**
Cambridge	None	**0.855**	**21.91**	9.614	10.41	0.261	60.98
	1.0	0.861	22.26	9.806	10.44	0.258	60.91
	5.0	0.880	23.46	**10.54**	10.23	**0.253**	62.56
	10.0	0.934	27.27	10.27	**10.01**	0.254	**64.27**

Table 2. Comparison of pose accuracy and reconstruction results of our voxel-based sampled ray cloud with different voxel sizes. Note we set ray rejection hyperparameter β=25% for both datasets, denoting a fraction of rays aligned to the baseline.

As expected, the camera localization accuracy decreases as the voxel size increases, indicating that introducing non-uniform distribution of 3D rays arising from voxel-based sampling lead to a slight decrease in localization accuracy. Regarding the privacy-preserving capability, voxel-based ray sampling has degraded the quality of point reconstruction more than lifted from random point-center pairing, demonstrating more 3D lines from closest points are excluded during neighbor line estimations and consequently yielding more false positive rays in density-based inversion.

As rejection when β=50% shows the highest reconstruction quality of scene images (Table 2 in [4]) on the Energy Landscape [9] dataset, we also conducted experiments to verify the robustness of voxel-based ray sampling against rejection (β=50%) and present overall results in Table 3. These results also confirm the enhanced privacy-preserving capability of the voxel-based approach.

Dataset	Voxel size [m]	$\Delta R[°]$ mean	ΔT[cm] mean	e_g [m] median	PSNR mean	SSIM mean	MAE mean
Energy landscape	None	**0.262**	**0.99**	0.669	10.29	0.319	62.97
	0.1	0.267	1.01	0.755	10.38	0.320	62.08
	0.25	0.277	1.03	0.783	10.31	0.318	62.82
	0.5	0.297	1.09	**0.792**	**10.11**	**0.316**	**64.36**

Table 3. Comparison of pose accuracy and reconstruction results of our voxel-based sampled ray cloud with different voxel sizes. Note we set β=50% for Energy Landscape [9].

2. Discussions on revealing points in ray cloud

In [4], we raised two reasons for trivial recovery of 3D points, i) lower significance of nearest points for estimating 3D points, and ii) bad estimation of true nearest rays lifted from nearest points. In this section, we present additional experimental results to support the underlying reasons.

2.1. Reduced significance of nearest points for estimating 3D point

Chelani *et al.* [1] used the assumption that when two lines have uniform line direction, the closest point to another line along the current line can be regarded as a good point candidate based on their empirical finding, that there is a higher probability (0.8) of the distance between the point and the candidate is less than the distance between the two 3D points. As the distance between two points decreases, accurate point estimation by point candidates becomes feasible. This means that accurate point estimates can be obtained from the nearest neighboring points, enabling accurate recovery of 3D points.

To investigate whether this assumption is also valid in ray cloud, we conducted experiments using the real scene data (*apt1 living* [9], *old hospital* [2]), randomly sampled 1000 lines from each scene representation and measured the empirical probability that the distance between the point ($\mathbf{p}_1$) and its candidate ($\hat{\mathbf{p}_1}$) is less than the distance between the two points ($\mathbf{p}_1, \mathbf{p}_2$) and present overall results in Table 4. The uniform line cloud [7] showed a high probability of about 0.8, indicating that the nearest points can lead to highly accurate point recovery by good point candidates.

In contrast, the ray cloud exhibited a probability lower than 0.8, especially showing an average probability of about 0.64 for both scenes when trivial solutions are included. This suggests that when the ray cloud contains trivial solutions, the accuracy of point recovery with nearest points is lower than that of the uniform line cloud.

On the other hand, only using rays from opposite center (described in Sec. 4.1 of [4]) to prevent trivial point recovery showed some improved probability. The removal of trivial solutions prevents point candidates from being generated at the current ray center and this might lead to an

	apt1 living [9]	*old hospital [2]*
Uniform line cloud [7]	0.785	0.783
Ray cloud (w/ trivial recovery)	0.638	0.637
Ray cloud (w.o trivial recovery)	0.728	0.683
Voxel-based ray cloud (w/ trivial recovery)	0.632	0.623
Voxel-based ray cloud (w.o trivial recovery)	0.722	0.669

Table 4. Suppose we sample two points $\mathbf{p}_1$ and $\mathbf{p}_2$ from a point cloud. Their lifted line representations are denoted as $\mathbf{l}_1$ and $\mathbf{l}_2$ respectively. Then, $\hat{\mathbf{p}}_1$ is the closest point to $\mathbf{l}_2$ along the line $\mathbf{l}_1$ (lifted from $\mathbf{p}_1$) and serves as the estimate for the location of $\mathbf{p}_1$. Above table shows the probability of this assumption being true for different scene representations across different datasets. Randomly selected 1000 lines in *apt1 living* [9] and *old hospital* scene [2] were used for experiments.

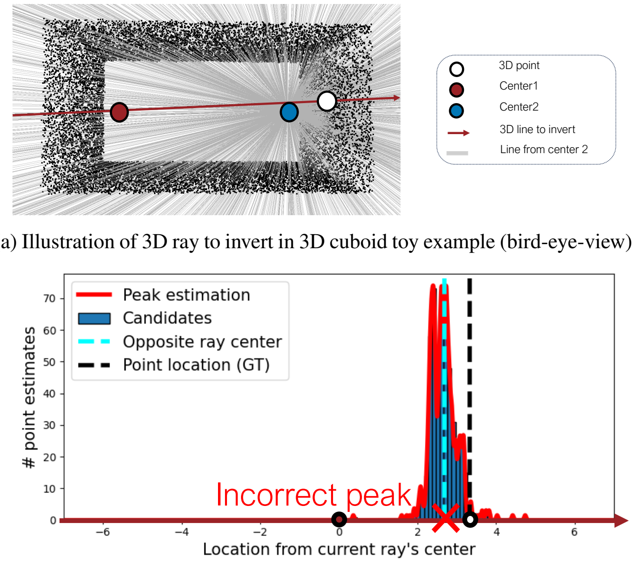
increase in the probability. However, it still showed a lower probability than the uniform line cloud and this supports the results of recovery in the oracle case (Fig. 6 in our main paper [4]) that when nearest points are known, the restored result of the ray cloud with removed trivial solutions shows more accurate recovery but is still inferior to the result of the uniform line cloud.

2.2. False positive rays

In [4], we suggested two cases where false positive rays can lead to bad estimations, namely due to i) the shrinking effect towards the ray centers and ii) rise of multiple peaks. In this subsection, we report our empirical findings on these two cases in a synthetic "room" environment.

Illustration of the shrinking effects: Fig. 2 illustrates the shrinking effect when the current ray passes close to the opposite ray center as shown in Fig. 2a. In such case, the estimated peak incorrectly identified near the opposite ray center as shown in Fig. 2b due to high density of rays around the opposite ray center. Vice versa, Fig. 3b shows that when there are rays from the opposite center near the current ray center as Fig. 3a, the generated point candidates exist around the current ray center. The above two empirical cases explicitly show the shrinking recovery of 3D points into the ray centers and the qualitative restoration results of real scene geometries are shown in Fig. 7d and Fig. 7e.

Illustration of multiple peaks in histogram of point candidates: In a room-like environment as shown in Fig. 3a, we observe the existence of multi peaks in the histogram of point candidates (Fig. 3b) potentially due to congested rays producing may close rays whose actual points are far away from the current ray, degrading the geometry inversion qualtiy. Similar phenomenon is also observed in Fig. 3d for a real scene (Fig. 3c). This shows that, unless the actual location of the points are provided in advance (*i.e.* the oracle case), it will be very challenging to estimate the actual peak among multiple peaks.



(a) Illustration of 3D ray to invert in 3D cuboid toy example (bird-eye-view)

(b) Histogram of the point candidates along the ray in (a)

Figure 2. Illustration of shrinking effect in a toy example. In (b), we projected the opposite ray center into the current ray (dotted line with sky color) to figure out the location of the closest point to the opposite ray center. (b) demonstrates that the estimated peak is close to the opposite ray center.

	apt1 living [9]	*old hospital [2]*
Uniform line cloud [7]	0.336	0.151
Ray cloud (w.o trivial solutions)	0.224	0.078
Ray cloud (w/ rejection)	0.290	0.120
Voxel-based ray cloud (w/ rejection)	0.191	0.043

Table 5. Empirical probability of rays from k-nearest points (k=100) being found in the set of K-nearest rays (K=1000) estimated with neighboring lines estimation based on line-to-line distance. Lower probability means the line-to-line neighborhood estimation does not provide a good estimate of point-wise nearest neighbors which is required for the method of Chelani et al. [1] to recover the scene geometry. For experiments, *apt1 living* [9] and *old hospital* scene [2] were used.

Quantitative analysis of effect of false positive rays: We also empirically observe that the assumption [1] in line clouds that the K-nearest lines will overlap enough with the set of lines lifted from the k-nearest points ($K>k$) is weakened for ray clouds.

In Table 5, we measured the probability of rays from k-nearest points (k=100) found on indices of rays from K-nearest points (K=1000). The uniform line cloud exhibited the highest empirical probability in both scenes, supporting the assumption in [1]. In the case of ray cloud, by estimating K-nearest rays using only rays from the opposite ray center to eliminate trivial solutions, the probability becomes lower than the uniform line cloud. However, applying the proposed rejection (beta=25%) along with using rays from the opposite ray center, the probability increased in both scenes.

This demonstrates that the rejection eliminated some false positive rays, leading to a higher inclusion of rays from the k-nearest points. However, in ray clouds generated based on a voxel grid, this probability sharply declines. This may be due to a large number of adjacent k-nearest points existing within the same voxel, causing them to be paired with the same ray center and not included in the estimation of K-nearest rays. The overall results in Table 5 indicate that the accuracy of point restoration through the estimation of K-nearest lines (rays) is lower in ray cloud than in uniform line cloud showing the existence of the false positive rays.

In summary, the reduced accuracy in geometric restoration in the ray cloud can be attributed to the lower precision in estimating rays from k-nearest points through K-nearest lines (rays) estimation. Furthermore, the probability of obtaining good point candidates from nearest points is also lower in ray cloud compared to uniform line cloud. These two reasons seem to diminish the geometric restoration accuracy of the ray cloud.

3. Further discussions on experimental results

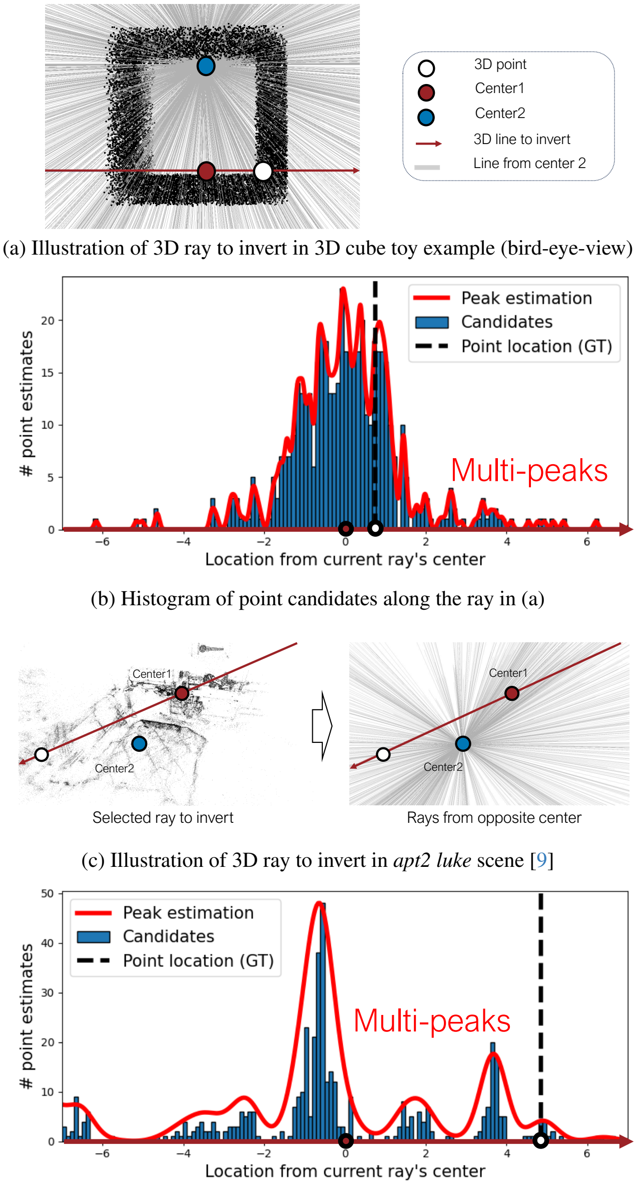
3.1. Inversion results with refined estimation

In the coarse estimation step in [1], initial points are estimated by performing neighboring line estimation based on line-to-line distance. Then there is a refinement step [1] based on computing the distances between the coarsely estimated points and lines. However, for the case of the ray cloud, we observe in Table 6 that the refine results is actually worse than the coarse estimation results reported in [4]. We anticipate this may be due to highly erroneous corase estimation leading to diverging solutions.

For a fair comparison, we also compared the results of the coarse estimation of uniform line clouds [7] and PPL [3] with the results of ray clouds (see Fig. 11). The geometric error shows that the ray cloud and voxel-based ray cloud still have higher values compared to the uniform line cloud and PPL, and the visualization results are also much more unrecognizable, demonstrating that the reconstruction of ray clouds is not more inaccurate than other scene representations simply due to the lack of refinement.

3.2. Increased SSIM for reduced ray density

In [4], we observed a slight increase in SSIM with reduced line density. However, the additional figures provided in this document suggest this quantitative difference is not highly noticeable in terms of qualitative analysis. Fig. 6 shows the qualitative and SSIM comparison of the restored images at different line densities. Fig. 6b showed the highest SSIM, restored at 10% line density, but the qualitative results of the restored scene details still remain unrecognizable. Similarly, Fig. 6i exhibited the highest SSIM, but the reconstructed image is also unrecognizable.



(a) Illustration of 3D ray to invert in 3D cube toy example (bird-eye-view)

(b) Histogram of point candidates along the ray in (a)

(c) Illustration of 3D ray to invert in *apt2 luke* scene [9]

(d) Histogram of point candidates along the ray in (c)

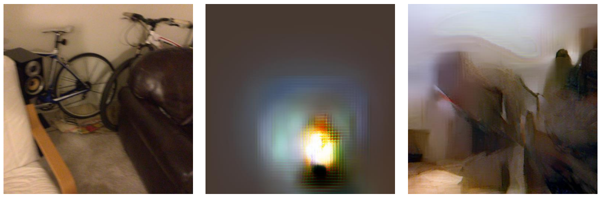
Figure 3. Illustration of current ray that needs to be inverted in (a) toy example, (c) *apt2 luke*, and (b), (d) the histogram of point candidates along the current ray obtained through the nearest line (ray) estimation.

4. Additional results

Fig. 4 shows the results of point-to-image translation using InvSfM when trivial 3D point recovery is performed. When all points are restored to the two centers, the obtained images(Fig. 4b) seem to have no meaningful content. Fig. 5 illustrates image reconstruction quality in relation to localization performance across different methods and line density with Cambridge [2] dataset, demonstrating the capability of privacy-preserving of ray cloud-based approaches, especially for dense scenes. Fig. 7 and Fig. 8 show addi-

Dataset	Rejection	e_g[m]	PSNR	SSIM	MAE
	threshold (β)	median	mean	mean	mean
Energy Landscape	None	1.027 / 1.515	9.96 / 9.99	0.330 / 0.375	66.06 / 70.41
	10%	0.761 / 1.300	10.24 / 9.75	0.323 / 0.353	63.62 / 70.57
	25%	0.669 / 0.964	10.29 / 9.78	0.319 / 0.343	62.97 / 69.30
	50%	0.746 / 0.918	10.32 / 9.72	0.320 / 0.337	62.52 / 69.87
Cambridge	None	11.47 / 18.81	10.10 / 9.70	0.273 / 0.288	64.06 / 69.48
	10%	10.29 / 15.57	10.35 / 9.57	0.267 / 0.265	61.60 / 68.94
	25%	9.614 / 15.11	10.41 / 9.59	0.261 / 0.256	60.98 / 68.28
	50%	12.98 / 15.77	9.94 / 9.42	0.242 / 0.245	64.70 / 69.88

Table 6. *Coarse/refined* estimation results of our modified inversion attack [4] on the ray cloud with different rejection thresholds (β).



(a) Ground truth

(b) Trivial solution (PSNR: 10.56)

(c) Non-trivial solution (PSNR: 12.78)

Figure 4. Revealed images of *apt2 living* scene [9] from the ray cloud without preventing trivial solutions (b) and with preventing trivial solutions [4]. Since we got only center points as a result of inversion including trivial solutions, the obtained image (b) with InvSfM [6] doesn't contain any detailed contents of the scene. In contrast, the reconstructed image (c) with our proposed method for preventing trivial solutions shows qualitative and improved PSNR compared to the non-trivial result.

tional qualitative results for the recovered point clouds and the images obtained from them. As can be seen, the recovery of the geometry of the ray cloud and voxel-based ray cloud showed degraded results than the uniform line cloud and PPL. Also, synthesized images of ray cloud and voxel-based from reconstructed point cloud are unrecognizable, demonstrating their ability to conceal the scene details.

References

[1] Kunal Chelani, Fredrik Kahl, and Torsten Sattler. How privacy-preserving are line clouds? Recovering scene details from 3D lines. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15663–15673, 2021. 2, 3, 4, 8

[2] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2938–2946, 2015. 1, 2, 3, 4, 6, 10

[3] Chunghwan Lee, Jaihoon Kim, Chanhyuk Yun, and Je Hyeong Hong. Paired-point lifting for enhanced privacy-preserving visual localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17266–17275, 2023. 4, 7, 8, 10

[4] Heejoon Moon, Chunghwan Lee, and Je Hyeong Hong. Efficient privacy-preserving visual localization using 3D ray clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2, 3, 4, 5

[5] David Nistér. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–770, 2004. 1, 11

[6] Francesco Pittaluga, Sanjeev J Koppal, Sing Bing Kang, and Sudipta N Sinha. Revealing scenes by inverting structure from motion reconstructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 145–154, 2019. 5, 8

[7] Pablo Speciale, Johannes L Schonberger, Sing Bing Kang, Sudipta N Sinha, and Marc Pollefeys. Privacy preserving image-based localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5493–5503, 2019. 1, 2, 3, 4, 7, 8, 10

[8] Henrik Stewénius, Magnus Oskarsson, Kalle Aström, and David Nistér. Solutions to minimal generalized relative pose problems, 2005. 1

[9] Julien Valentin, Angela Dai, Matthias Niessner, Pushmeet Kohli, Philip Torr, Shahram Izadi, and Cem Keskin. Learning to navigate the energy landscape. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 323–332, 2016. 1, 2, 3, 4, 5, 9, 10

[10] Enliang Zheng and Changchang Wu. Structure from motion using structure-less resection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2075–2083, 2015. 1

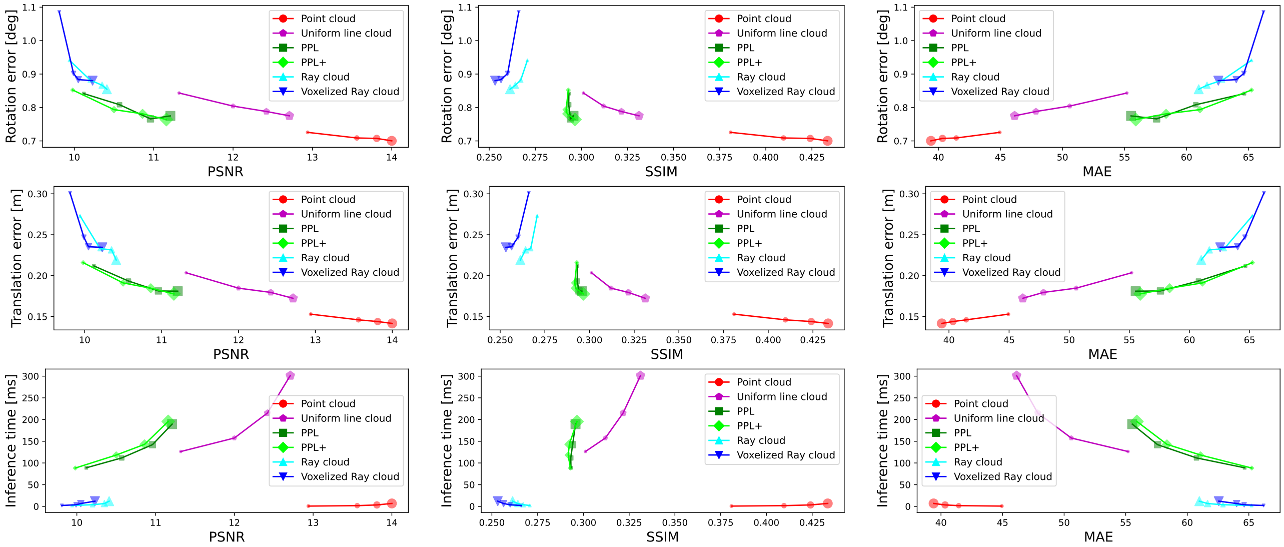
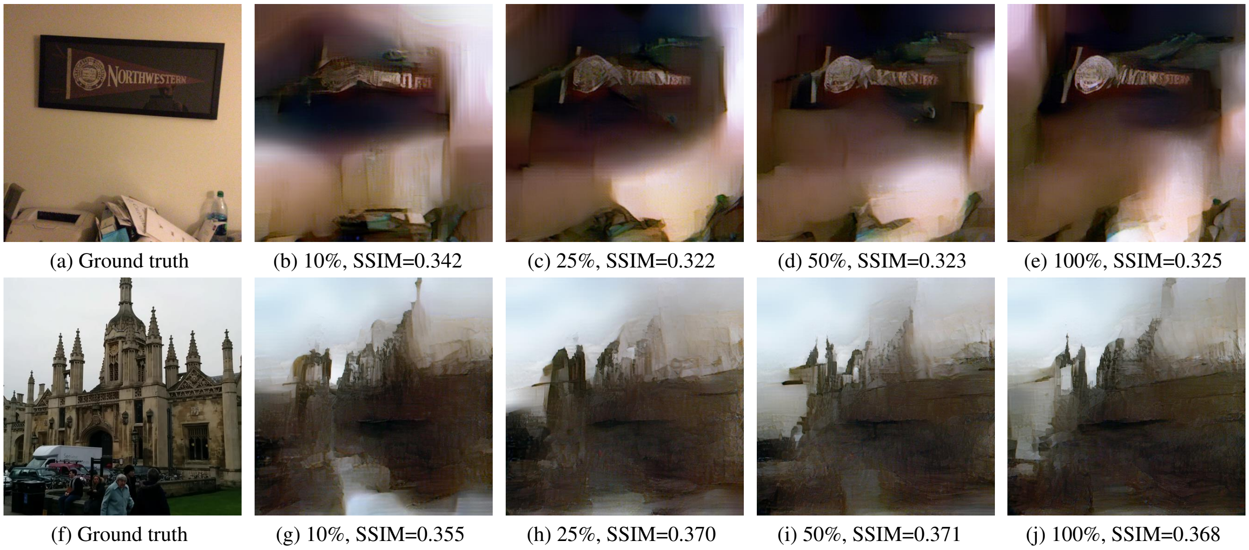


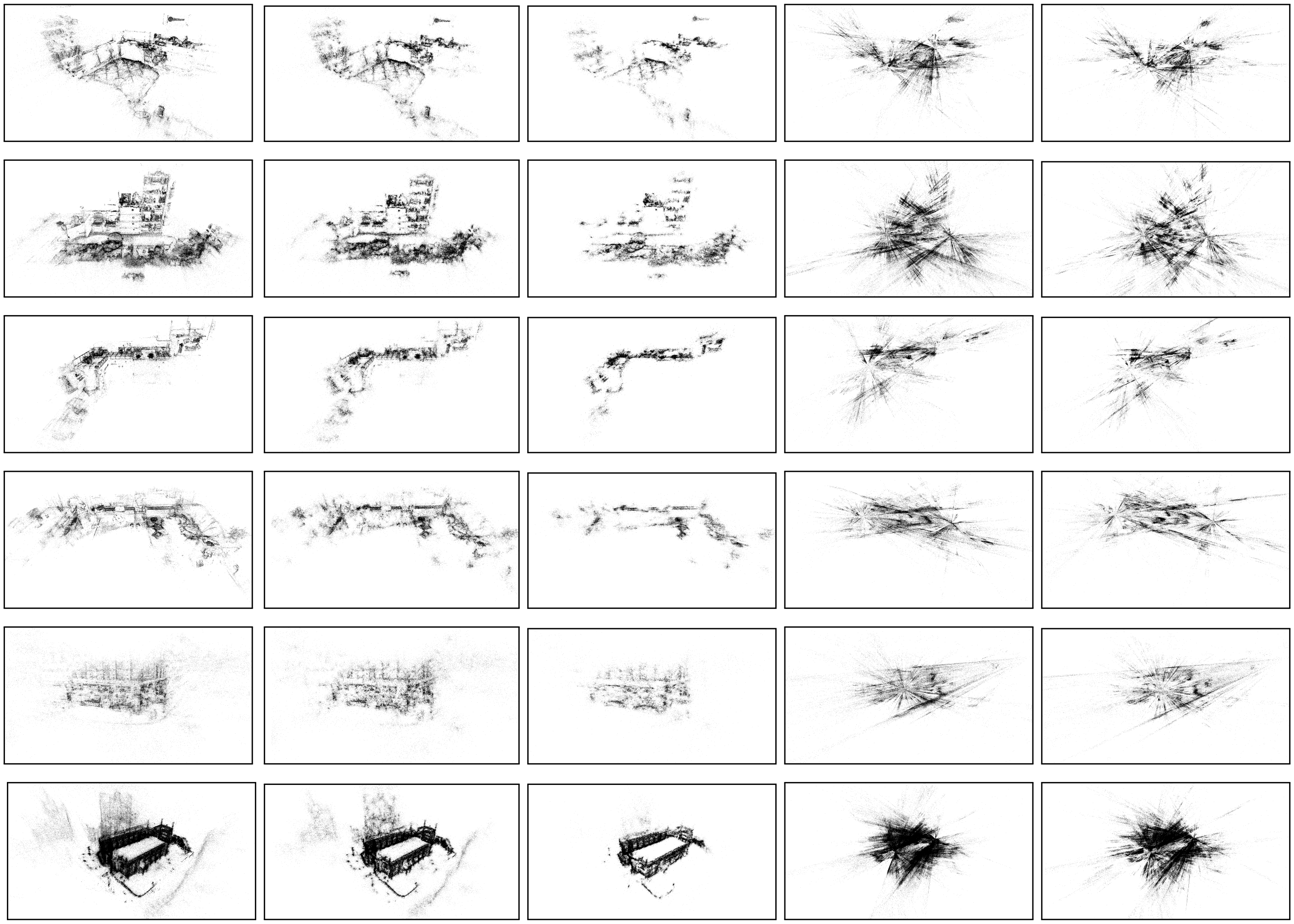
Figure 5. Averaged localization performance vs. reconstructed image quality across different levels of ray density on the Cambridge [2] dataset. The tested line densities are 100% (biggest marker), 50%, 25%, and 10% (smallest marker).



(a) Ground truth (b) 10%, SSIM=0.342 (c) 25%, SSIM=0.322 (d) 50%, SSIM=0.323 (e) 100%, SSIM=0.325

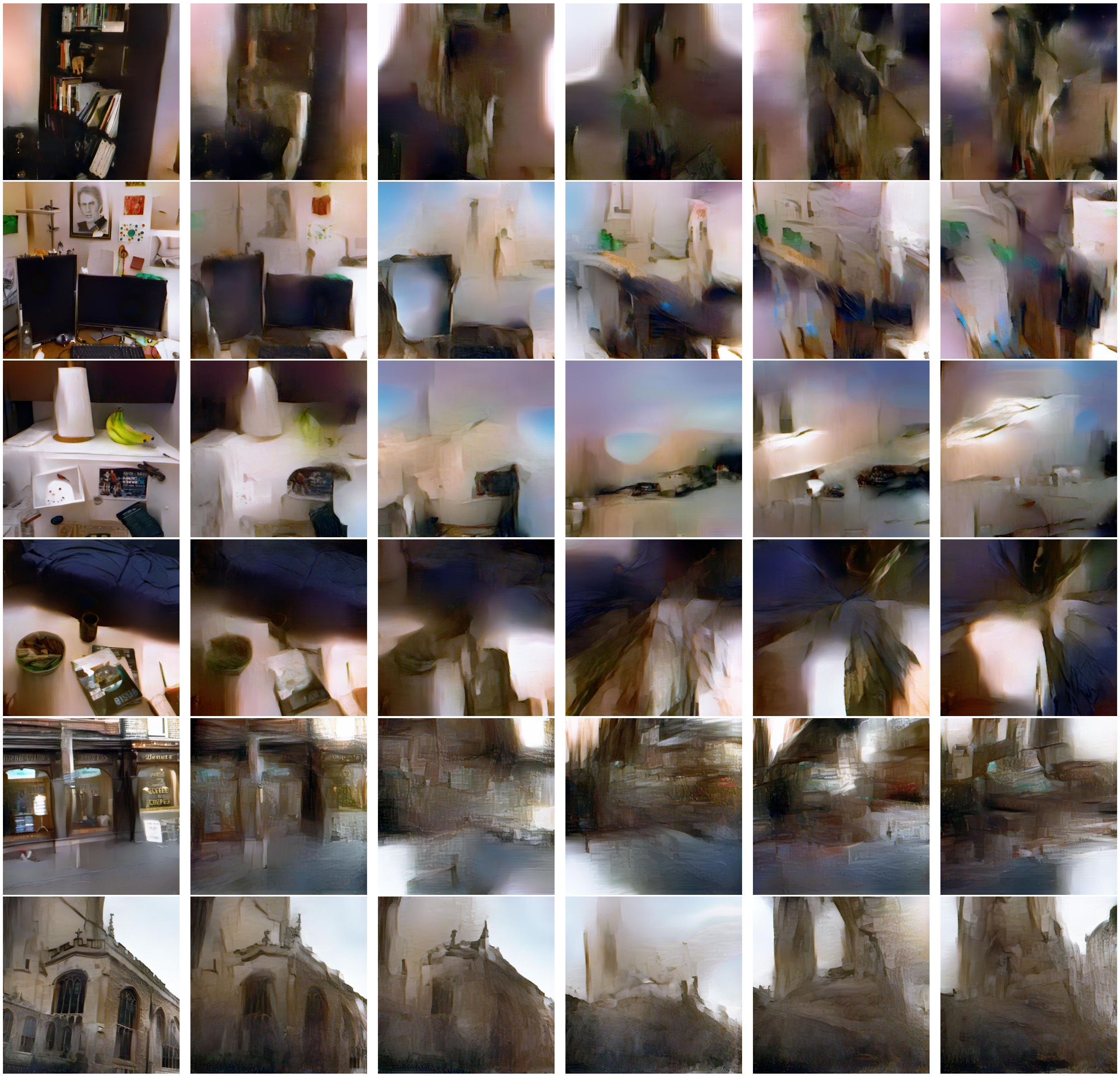
(f) Ground truth (g) 10%, SSIM=0.355 (h) 25%, SSIM=0.370 (i) 50%, SSIM=0.371 (j) 100%, SSIM=0.368

Figure 6. Comparison SSIM of revealed images from different levels of line density in the ray cloud (w.o. voxel-based ray sampling). Note, β=25% rejection was applied.



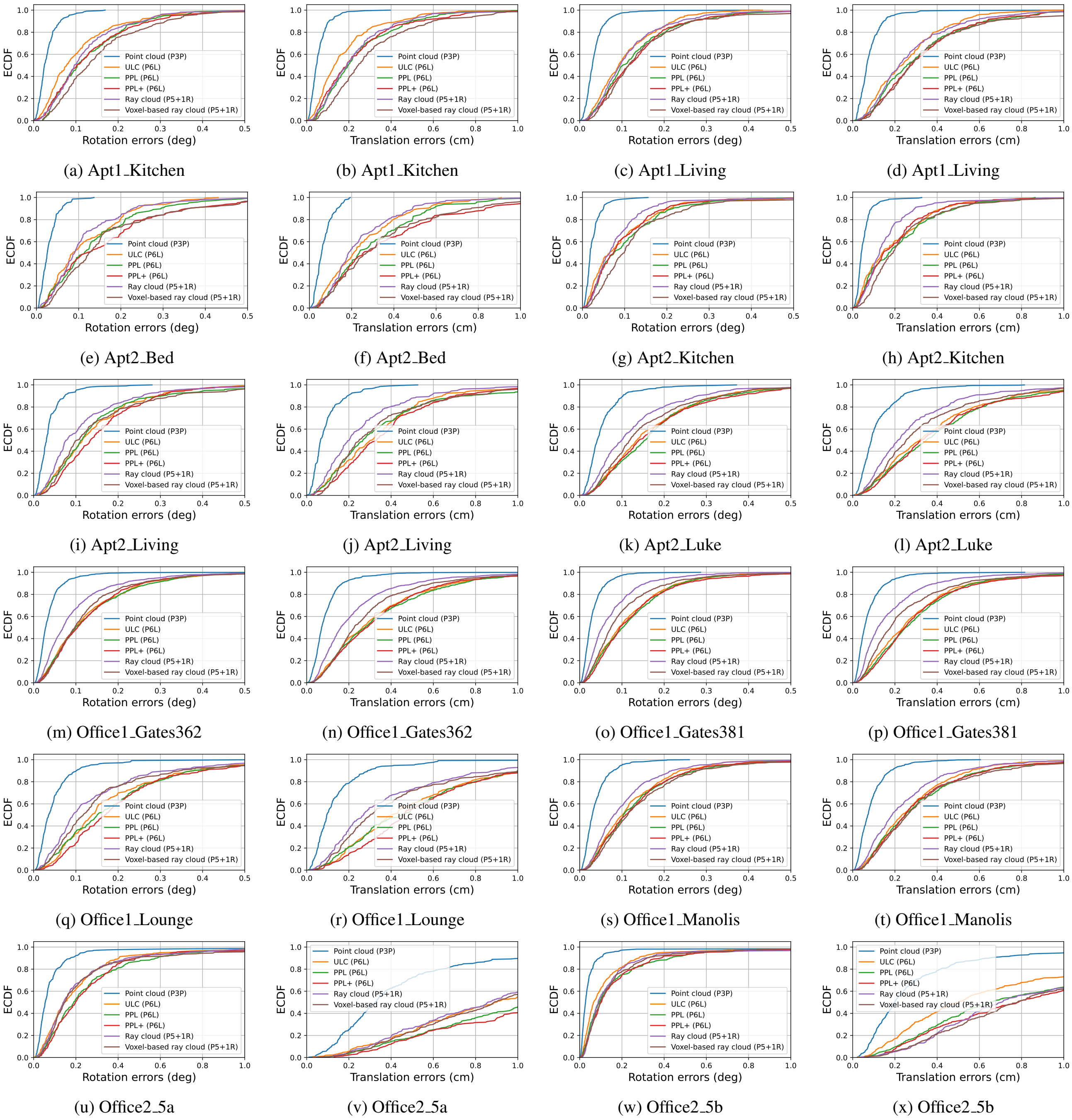
(a) Ground truth (b) Uniform line cloud [7] (c) PPL [3] (d) Ray cloud (ours) (e) Voxel-based ray cloud(ours)

Figure 7. Visualization of qualitative results of the reconstructed point cloud from (b) uniform line cloud [7], (c) Paired-Point Lifting(PPL) [3], (d) Ray cloud, and (e) Voxel-based ray cloud. For ray cloud and voxel-based ray cloud, rejection (β=25%) was applied during neighbor estimations.



(a) Point cloud (b) Uniform line cloud [1] (c) PPL [3] (d) Ray cloud (w.o rejection) (e) Ray cloud (w. rejection) (f) Voxel-based Ray cloud

Figure 8. Comparison of revealed images from ground truth point cloud, uniform line cloud [7], PPL [3], ray cloud without and with rejection (β=25%) and using InvSfM [6].



(a) Apt1_Kitchen

(b) Apt1_Kitchen

(c) Apt1_Living

(d) Apt1_Living

(e) Apt2_Bed

(f) Apt2_Bed

(g) Apt2_Kitchen

(h) Apt2_Kitchen

(i) Apt2_Living

(j) Apt2_Living

(k) Apt2_Luke

(l) Apt2_Luke

(m) Office1_Gates362

(n) Office1_Gates362

(o) Office1_Gates381

(p) Office1_Gates381

(q) Office1_Lounge

(r) Office1_Lounge

(s) Office1_Manolis

(t) Office1_Manolis

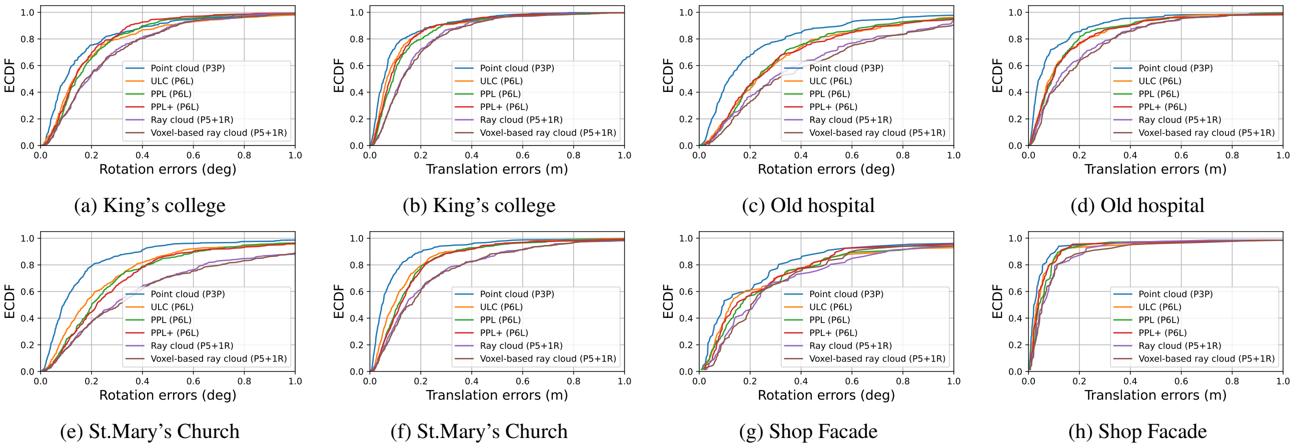
(u) Office2_5a

(v) Office2_5a

(w) Office2_5b

(x) Office2_5b

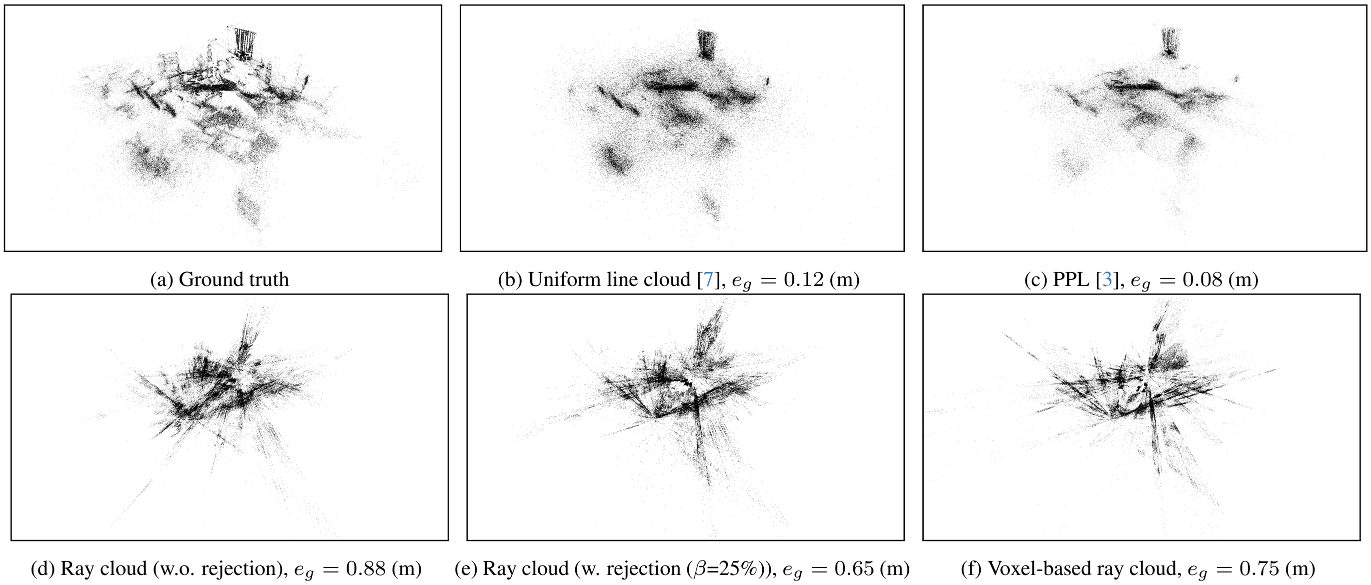
Figure 9. ECDF of rotation & translation errors of estimated camera pose by each data from [9]



(a) King's college (b) King's college (c) Old hospital (d) Old hospital

(e) St.Mary's Church (f) St.Mary's Church (g) Shop Facade (h) Shop Facade

Figure 10. ECDF of rotation & translation errors of estimated camera pose by each data from [2]



(a) Ground truth (b) Uniform line cloud [7], $e_g = 0.12$ (m) (c) PPL [3], $e_g = 0.08$ (m)

(d) Ray cloud (w.o. rejection), $e_g = 0.88$ (m) (e) Ray cloud (w. rejection (β=25%)), $e_g = 0.65$ (m) (f) Voxel-based ray cloud, $e_g = 0.75$ (m)

Figure 11. Visualization of coarse inversion results from each scene representation of *apt2 living* [9]. e_g denotes median 3D point error (m).

Algorithm 1 Algorithmic procedures of LO-RANSAC implementation for pose estimation of ray cloud.

```
Input: Correspondences $\mathbf{C}(\mathbf{p}, \mathbf{L})$ between 2D point $\mathbf{p}$ and 3D rays $\mathbf{L}$
Output: Absolute pose $\mathtt{R}_{abs}, \mathbf{t}_{abs}$
Given: Generalized rig1 ← Two virtual cameras in the ray cloud
Generalized rig2 ← Query camera
Inlier threshold (max epipolar error) $\tau \leftarrow 2.0(px)/$ Query camera focal length

▷ Calculate pose with proposed minimal $p$5+1R solver
BEST_POSE ← $\emptyset$
inlier_count$_{best}$ ← 0, inlier_count$_{LO}$ ← 0
MSAC_SCORE$_{best}$ ← inf, MSAC_SCORE$_{LO}$ ← inf
for $iter \leftarrow 0$ $to$ MAX_ITER do
  CUR_POSE = [ ]
  $j$ ← Randomly selected $j^{th}$ camera in rig1 ($j \in (1, 2)$)
  ▷ Initialize $j^{th}$ camera
  $[\mathtt{R}_j|\mathbf{t}_j] \leftarrow [\mathtt{I}| -\mathbf{c}_j]$

  ▷ Draw 5 correspondences $\mathbf{C}_{i_5}$ between $j^{th}$ virtual camera and query image
  for $\mathbf{C}_{i_5} \in \mathbf{C}$ do
    Calculate POSE$_{rel}$ from Relative 5-points algorithm [5] using $\mathbf{C}_{i_5}$ (unknown scale $\alpha$)
    for $[\mathtt{R}_{qj}|\alpha\hat{\mathbf{t}}_{qj}] \in$ POSE$_{rel}$ do
      Absolute pose of pinhole camera $[\mathtt{R}_q|\mathbf{t}_q] \leftarrow [\mathtt{R}_{qj}|\alpha\hat{\mathbf{t}}_{qj} - \mathtt{R}_{qj}\mathbf{c}_j]$
      ▷ Draw extra correspondence $\mathbf{C}_{i_1}$ between the remaining virtual camera and query image
      $(p_1, p_2)^\top \in \mathbb{R}^2$ ← Normalized 2D key point $\mathbf{p}_{i_1}$ from $\mathbf{C}_{i_1}(\mathbf{p}_{i_1}, \mathbf{L}_{i_1})$
      $\hat{\mathbf{n}}$ ← Normalize($\mathbf{L}_{i_1}$) $\in S^2$
      ▷ Rescale $\hat{\mathbf{t}}_{qj}$ by using the extra correspondence
      Calculate $(\lambda_1, \lambda_2, \alpha)$ w.r.t. the Equation: $\lambda_1[p_1, p_2, 1]^\top = \lambda_2\mathtt{R}_{qj}\hat{\mathbf{n}} + \alpha\hat{\mathbf{t}}_{qj} - \mathtt{R}_{qj}\mathbf{c}_j$
      CUR_POSE.$append$($[\mathtt{R}_{qj}|\alpha\hat{\mathbf{t}}_{qj} - \mathtt{R}_{qj}\mathbf{c}_j]$)
    end for
  end for

  ▷ Find best local model
  Index of BEST_POSE : $M \leftarrow -1$
  for $k$ ← 0 to CUR_POSE.size()-1 do
  ▷ Align two generalized rigs with model$_k$ and calculate
    $\mathbf{p}_i \in \mathbb{R}^2 \leftarrow i^{th}$ normalized keypoint in query image
    $\hat{\mathbf{n}}_i \in S^2 \leftarrow$ corresponding $i^{th}$ 3D normalized ray
    $\Omega \leftarrow$ All of the pairing between $i^{th}$ keypoints and $j^{th}$ ray center in the query image
    $\mathtt{E}_j := [\mathbf{e}_{j1}, \mathbf{e}_{j2}, \mathbf{e}_{j3}]^\top \leftarrow$ Essential matrix between $j^{th}$ virtual camera and query camera
    MSAC_SCORE$_k \leftarrow \sum_{(i,j)\in\Omega} \min\left(\frac{([\mathbf{p}_i^\top, 1]\ \mathtt{E}_j\ \hat{\mathbf{n}}_i)^2}{(\mathbf{e}_{j1}^\top\hat{\mathbf{n}}_i)^2 + (\mathbf{e}_{j2}^\top\hat{\mathbf{n}}_i)^2}, \tau^2\right)$
    inlier_count$_k \leftarrow \sum_{(i,j)\in\Omega} \mathbb{1}_{\mathtt{A}}(i, j)$, where $\mathtt{A} := \{(i, j)| \frac{([\mathbf{p}_i^\top, 1]\ \mathtt{E}_j\ \hat{\mathbf{n}}_i)^2}{(\mathbf{e}_{j1}^\top\hat{\mathbf{n}}_i)^2 + (\mathbf{e}_{j2}^\top\hat{\mathbf{n}}_i)^2} \leq \tau^2\}$

    ▷ Check to update best model
    if inlier_count$_{LO}$ ≤ inlier_count$_k$ or MSAC_SCORE$_k$ ≤ MSAC_SCORE$_{LO}$ then
      $M \leftarrow k$
      inlier_count$_{LO}$ ← inlier_count$_k$
      MSAC_SCORE$_{LO}$ ← MSAC_SCORE$_k$
      if MSAC_SCORE$_k$ < MSAC_SCORE$_{best}$ then
        inlier_count$_{best}$ ← inlier_count$_k$
        MSAC_SCORE$_{best}$ ← MSAC_SCORE$_k$
        BEST_POSE ← CUR_POSE$[k]$
      end if
    end if
  end for

  ▷ Local refinement
  BEST_POSE ← CUR_POSE$[M]$
  POSE$_{refined}$ ← Non-linear (LM) optimization from BEST_POSE
  if MSAC_SCORE$_{refined}$ < MSAC_SCORE$_{best}$ then
    inlier_count$_{best}$ ← inlier_count$_{refined}$
    MSAC_SCORE$_{bestd}$ ← MSAC_SCORE$_{refined}$
    BEST_POSE ← POSE$_{refined}$
  end if
end for

▷ Final refinement using only inliers
POSE$_{refined}$ ← Non-linear (LM) optimization from BEST_POSE
if MSAC_SCORE$_{refined}$ < MSAC_SCORE$_{best}$ then
  inlier_count$_{best}$ ← inlier_count$_{refined}$
  MSAC_SCORE$_{best}$ ← MSAC_SCORE$_{refined}$
  BEST_POSE ← POSE$_{refined}$
end if
Return $\mathtt{R}_{abs}, \mathbf{t}_{abs}$ ← BEST_POSE
```