Paired-Point Lifting for Enhanced Privacy-Preserving Visual Localization

Chunghwan Lee[1] Jaihoon Kim[2,3] Chanhyuk Yun[1] Je Hyeong Hong[1*]
[1]Dept. Electronic Engineering, Hanyang University [2]Samsung Electronics [3]KAIST

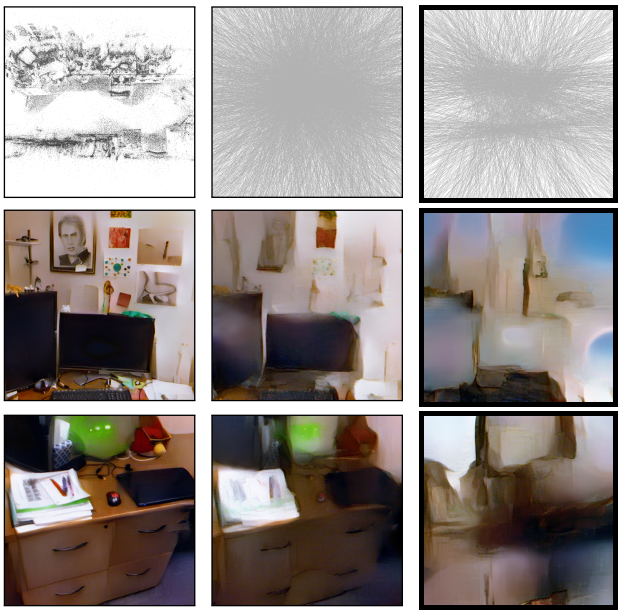
Abstract

Visual localization refers to the process of recovering camera pose from input image relative to a known scene, forming a cornerstone of numerous vision and robotics systems. While many algorithms utilize sparse 3D point cloud of the scene obtained via structure-from-motion (SfM) for localization, recent studies have raised privacy concerns by successfully revealing high-fidelity appearance of the scene from such sparse 3D representation. One prominent approach for bypassing this attack was to lift 3D points to randomly oriented 3D lines thereby hiding scene geometry, but latest work have shown such random line cloud has a critical statistical flaw that can be exploited to break through protection. In this work, we present an alternative lightweight strategy called Paired-Point Lifting (PPL) for constructing 3D line clouds. Instead of drawing one randomly oriented line per 3D point, PPL splits 3D points into pairs and joins each pair to form 3D lines. This seemingly simple strategy yields 3 benefits, i) new ambiguity in feature selection, ii) increased line cloud sparsity and iii) non-trivial distribution of 3D lines, all of which contributes to enhanced protection against privacy attacks. Extensive experimental results demonstrate the strength of PPL in concealing scene details without compromising localization accuracy, unlocking the true potential of 3D line clouds.

1. Introduction

Visual localization is the fundamental problem of estimating the camera pose from an input image with respect to a known 3D scene. It plays a crucial role in many computer vision and robotics applications, including human-computer interaction based on augmented or mixed reality (AR/MR), 3D reconstruction via structure-from-motion (SfM) [1, 36, 37, 39] and autonomous navigation systems in drones, self-driving vehicles, or robots using simultaneous localization and mapping (SLAM) [5, 22, 26].

To this date, many practical visual localization algorithms are still structure-based approaches [12], utilizing a global sparse 3D model of the scene obtained from SfM or SLAM. In such approaches, 2D-3D correspondences are formed between 2D image points and the global 3D structure by comparing feature descriptors (e.g., SIFT [21], ORB [34], or other alternatives [25, 43]), after which they are used to perform robust camera pose estimation based on geometric constraints and RANSAC [2, 8, 31]. While most research over the last decade have made significant progress in improving algorithm accuracy, scalability [19, 23, 35, 47] and efficiency [12, 20], the requirement that sparse 3D point cloud and associated feature descriptors need to be persistently accessible has largely remained unchanged.

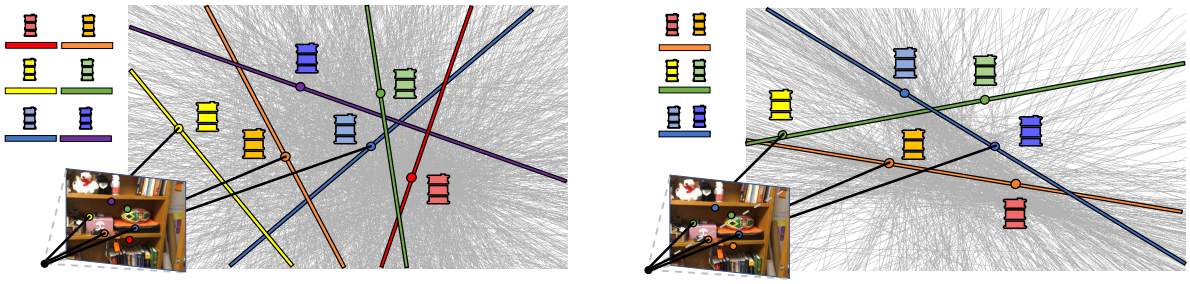


(a) Point cloud (b) Line cloud [40] (c) **PPL (ours)**

Figure 1. Visualization of different 3D scene representations and respective image reconstruction results using InvSfM [30]. Images in (b) and (c) are reconstructed by estimating the 3D point clouds from line clouds via [3]. While images revealed from original line clouds (OLC) [40] still contain basic scene details, those from the proposed method (PPL) exhibit much degraded scene quality.

Recently, it caught the research community by surprise when Pittaluga et al. [30] showed sparse 3D point clouds can retain enough scene details such as characters and textures that can be used to uncover a high-fidelity image of the scene. Since this process only requires 2D projection of 3D points and their respective feature descriptors, this raised privacy concerns related to uploading sparse 3D models in the cloud or storing them on the end-user device, both of which are commonly found settings in visual localization.

*Corresponding author



(a) Standard lifting with uniformly-distributed line directions [40]

(b) Paired-point lifting (PPL) (ours)

Figure 2. Illustration of the previous standard lifting approach (also referred to as the original line cloud (OLC)) [40] and PPL (ours) for constructing 3D line clouds from sparse 3D point clouds. In (a), each line passes through one 3D point with its direction uniformly sampled on a unit sphere, carrying one descriptor. In (b), each line passes through two 3D points selected at random, carrying two descriptors.

One of the most notable approaches for addressing above privacy attack on point clouds is geometric lifting, whereby each 3D point is transformed into a randomly-oriented 3D line passing through the respective point [40]. By sampling each line direction uniformly on the unit sphere, the approach aims to conceal the scene geometry and prevent meaningful 2D projections of the sparse point cloud.

Nevertheless, a recent work by Chelani et al. [3] showed 3D lines whose directions are uniformly drawn from a unit sphere carry an unintended statistical characteristic that the original 3D points are likely to be near the high-density regions of the closest points between lines. This property can be exploited to reveal the scene geometry (see §2), and to the best of our knowledge, the only suggested option for mitigating this attack so far is to use sparser point clouds, which comes at the cost of reduced localization accuracy.

In this work, we argue that the privacy-preserving property of 3D line clouds can be enhanced by applying a different line-construction approach. Specifically, we propose to yield 3D lines by joining random pairs of 3D points. This approach, which we call *Paired-Point Lifting (PPL)*, is motivated by the three key observations listed below:

1. **confusion over feature descriptors:** each line carries two feature descriptors, so assigning the correct descriptors to each of two 3D points requires guesswork,
2. **line cloud sparsification:** PPL results in 50% more sparse line clouds, thus degrading the accuracy of 3D scene point recovery using [3], which relies on a sufficient number of neighboring lines, and
3. **non-trivial distribution of lines:** the finding from [3] that the original 3D point is likely to be near the high-density region of closest points to other lines is plagued by the presence of extra 3D point in each line and non-uniformly distributed line directions.

We illustrate in §3 and §5 that each of the above factors adds a layer of difficulty in revealing the scene geometry and image-level details, synergistically enhancing privacy-preserving visual localization (partly shown in Fig. 1) without compromising camera localization accuracy.

Overall, our contributions can be summarized as follows:

+ a new strategy called *paired-point lifting (PPL)* for constructing 3D line clouds with each line concealing two 3D points from the respective sparse point clouds,
+ careful empirical analysis of success factors in PPL through an ablation study using synthetic and real data,
+ improving the 3D point recovery algorithm in [3] to allow handling PPL-based line clouds and locate two 3D points in each line for a fairer comparison,
+ a subsequent upgrade of PPL called PPL+ to address potential drawback of PPL with planar scenes, and
+ extensive experimental evaluation of both PPL and PPL+ against other baselines on a range of public datasets using different point recovery techniques.

2. Related work

Revealing images from feature descriptors: Initial study dates back to the work of Weinzaepfel et al. [46], which fused patches obtained through retrieval from a pool of image patches for each SIFT descriptor [21]. Vondrick et al. [45] visualized images from the histogram of gradients by applying paired dictionary learning, and Kato et al. [13] reconstructed images by computing a histogram of visual words and optimizing their spatial arrangement. Mahendran et al. [24] and Dosovitskiy et al. [6] adopted CNNs to invert feature descriptors to images. Yet, none of these approaches yields detailed photorealistic images.

The real possibility of privacy leak was first raised by Pittaluga et al. [30] who showed sparse point clouds projected to 2D with SIFT keypoints, depth and color information can reveal photorealistic images of the scene using a cascaded U-Net [32]. This work was later extended by Dangwal et al. [4] who enabled reconstruction of photorealistic images without RGB or depth information. Current literature in privacy-preserving visual localization attempts to lower the quality of images reconstructed from these approaches.

Privacy-preserving localization: Two main approaches exist for visual localization addressing the above privacy attack, namely *feature encoding* and *geometric lifting*.

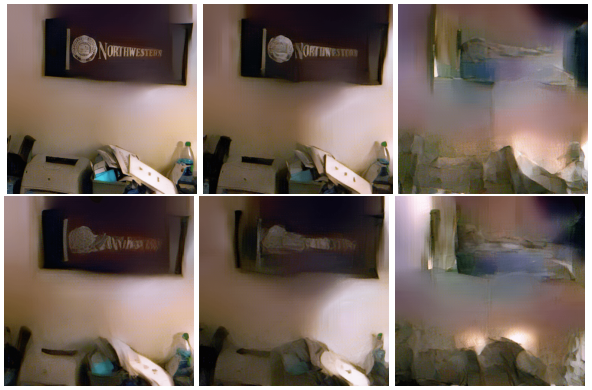


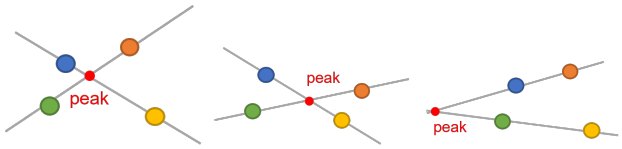
Figure 3. Simulated qualitative analysis of sources-of-errors in revealing images from a paired-point lifting (PPL)-based line cloud. Along the rows, we simulate errors in estimating 3D points from line clouds by [3] (top-no error, bottom-2% $\mathcal{N}(0,1)$ perturbations in 2D keypoints). Along the columns, we simulate the effect of feature confusion by randomly swapping SIFT descriptors between randomly selected pairs of 3D points (left-0%, middle-50% and right-100% swapped descriptors respectively). Both sources-of-errors contribute towards worse image reconstruction quality.

Feature encoding attempts to modify each point's feature descriptor to be less invertible by discarding excess scene details without hurting camera relocalization accuracy. This has the benefit that the same point-based geometry can be applied for visual localization. Dusmanu et al. [7] embedded each feature descriptor within an affine subspace, making it difficult to recover images, but this comes at the cost of lowered camera localization accuracy. Ng et al. [27] trained a MLP-based descriptor-concealing network and a U-net based scene-inversion network in an adversarial fashion to discard excess details of the scene for each descriptor, but this requires to be individually trained for each type of feature descriptors and may overfit to training datasets.

On the other hand, standard geometric lifting [40] transforms each 3D point into a randomly oriented line whose direction follows a uniform distribution on the unit sphere. Speciale et al. [40] showed it is possible to estimate camera poses from such *original line cloud (OLC)* by formulating the problem as finding the relative pose between two general cameras [42]. Successive works inspired by this approach include privacy-preserving SfM and SLAM [9, 10, 38]. While geometric lifting is straightforward to implement and can be universally applied across different types of feature descriptors, the method has recently been shown to be partially reversible by Chelani et al. [3] as illustrated below.

Revealing sparse point cloud from line cloud: Chelani et al. [3] showed aforeillustrated geometric lifting may not sufficiently conceal the scene geometry if each line direction is sampled uniformly from the unit sphere when constructing a line cloud from the sparse point cloud.

In their work, recovering the 3D point cloud $\mathcal{P} = \{\mathbf{x}_j\}$ from input line cloud $\mathcal{L} = \{\mathbf{l}_j\}$ is formulated as maximizing



(a) Initial PPL lines (b) Orange point moved (c) Different pairs

Figure 4. A 2D toy example demonstrating the effect of changes in point locations and pairing arrangements change on the estimated location of the point (peak). (b) and (c) illustrate that the PPL-based line directions are directly determined by the location of the 3D points and their pairing arrangement.

the posterior probability $P(\mathcal{P}|\mathcal{L})$, where $\mathbf{x}_j \in \mathbb{R}^3$ is the j-th 3D point in the sparse point cloud and $\mathbf{l}_j \in \mathbb{R}^6$ is the corresponding lifted line for $\mathbf{x}_j$. By applying Bayes' rule:

$$P(\mathcal{P}|\mathcal{L}) = \frac{P(\mathcal{L}|\mathcal{P})P(\mathcal{P})}{P(\mathcal{L})} \propto P(\mathcal{L}|\mathcal{P})P(\mathcal{P}) \qquad (1)$$

For standard lifting [40], the line direction is drawn uniformly from the unit sphere, so $P(\mathcal{L}|\mathcal{P}) = \Pi_{j=1}^{N} P(\mathbf{l}_j|\mathbf{x}_j)$ where $P(\mathbf{l}_j|\mathbf{x}_j)$ is constant if $\mathbf{l}_j$ passes through the point $\mathbf{x}_j$, or zero otherwise. The likelihood $P(\mathcal{L}|\mathcal{P})$ remains even for different point clouds as long as they are equal in size.

Subsequently, Chelani et al. [3] attempted to find the points maximizing the prior $P(\mathcal{P})$ under the constraints that each point $\mathbf{x}_j$ lies on the corresponding line $\mathbf{l}_j$. The method is motivated by the empirical observation that, when two arbitrary 3D points $\mathbf{x}_1$ and $\mathbf{x}_2$ are lifted to random 3D lines $\mathbf{l}_1$ and $\mathbf{l}_2$ respectively and we compute the closest points between the two lines, denoted as $\mathbf{x}_1^{\perp}$ for $\mathbf{l}_1$ and $\mathbf{x}_2^{\perp}$ for $\mathbf{l}_2$ respectively, it is around 80% likely that the distance between $\mathbf{x}_1$ and $\mathbf{x}_1^{\perp}$ is shorter than that between the original points $\mathbf{x}_1$ and $\mathbf{x}_2$. For the case of random multiple lines, this constraint is satisfied by each pair of lines, and one can subsequently construct a histogram of closest-points-to-other-lines for each line $\mathbf{l}_i$. Chelani et al. [3] proposed to estimate the 3D point $\mathbf{x}_i$ by finding the peak of this histogram.

The work showed recovery of the scene from line clouds is possible across various scenes but the results are limited to the case of uniformly distributed line directions, providing a motivation for our proposed method.

Peak finding: Peak finding is the process of estimating peaks in the probability density function (PDF). Chelani et al. [3] devised an efficient *single-peak finding (SPF)* algorithm by running iterative Kuiper's test [16] on the cumulative distribution function (CDF). One drawback of SPF is that it provides an accurate peak value only when the function is unimodal. Since our approach in §3 promotes non-unimodal and often bimodal distribution of closest-points-to-other-lines, we have devised a *Two-peak finding (TPF)* algorithm to encourage better recovery of bimodal peaks aiming towards fairer comparison (see details in §5).

(a) Ground truth point cloud (b) OLC [40](SPF) (c) PPL (SPF) (d) PPL(TPF) (e) PPL+(TPF)

Figure 5. Visualization of the point clouds recovered from different line-lifting methods and different recovery algorithms. (b) shows the point cloud recovered from the original line cloud (OLC) [40] via single-peak finding (SPF) [3], (c) and (d) are the point clouds revealed from the PPL-based line cloud via SPF [3] and two-peak finding (TPF, refer to §5) respectively, and (e) is the result from PPL+ via TPF.

3. Paired-point lifting

To overcome the shortcomings of the original line cloud (OLC) [40], we propose a different strategy for sampling lines in line clouds. Our approach named *paired-point lifting (PPL)* is straightforward to implement—randomly assign non-overlapping pairs of 3D points from the sparse point cloud and then join each pair of points to create a line cloud. A detailed algorithm chart can be found in [18].

As each line passes through two 3D points (with the exception of an ideal situation where three or more points are lined up), this results in N number of 3D lines, where $2N$ denotes the number of 3D points. As shown in Fig. 2, the major difference between standard lifting [40] and PPL is the number of 3D points concealed in each line—standard lifting conceals one 3D point per line, while PPL hides two points per line. As briefly mentioned in §1, having an extra 3D point for each line results in added ambiguity in descriptor selection, a reduced number of lines in the line clouds, and non-uniformly distributed line directions. The effect of each component is illustrated in the forthcoming sections.

Pose estimation: Before discussing the above benefits of PPL, we emphasize that the PPL-based line clouds only differ from the original line clouds (OLCs) in terms of line directions. This implies the PPL-derived line clouds are compatible with the minimal solver (P6L [40]) for OLC. Table 1 and §5 report similar degree of accuracy in camera pose estimation for both OLC and PPL-based line clouds.

3.1. Effect of additional feature descriptor per line

We start by analyzing the effect of having an extra feature descriptor for each line. We will assume two 3D points, A and B, are concealed in a single PPL-based line.

The current approach for estimating the point cloud from the line cloud [3] (see §2) is purely geometry-based, using the statistics of the closest points to other neighboring lines. Hence, even if this approach is extended to allow estimation of two points per line ([3] is limited to single peak finding), we have an ambiguity over which peak corresponds to A and which corresponds to B. Hence, there is only a 50% chance of locating the descriptors correctly for each line.

As above probability is independent for each line, the number of lines with correct assignments is a binomial random variable $X \sim \mathcal{B}(N, 0.5)$ where N is the total number of lines. This has two implications—first, the probability of correctly assigning descriptors for all lines is 0.5^N and thus quickly diminishes to 0 for large N. Second, the expected number of X is $0.5N$ with standard deviation $0.5\sqrt{N}$, so we can expect the number of correctly assigned lines to be roughly $0.5N$ for sufficiently large line clouds.

Since the quality of images revealed by InvSfM [30] depends on correct feature descriptors, above ambiguity in binary feature selection adds another layer of difficulty in revealing scenes from the sparse line cloud. This is empirically demonstrated through simulated examples in Fig. 3.

3.2. Effect of reduced line cloud density

As discussed in [3], recovering the point cloud from line cloud (reviewed in §2) requires sufficient number of lines to accurately estimate the peak of each line's histogram of closest-points-to-other-lines. It is also shown in [3], §5 and Table 1 that reducing the line cloud density from 100% to 50% for OLCs [40] results in higher 3D point estimation errors and subsequently lower image reconstruction quality.

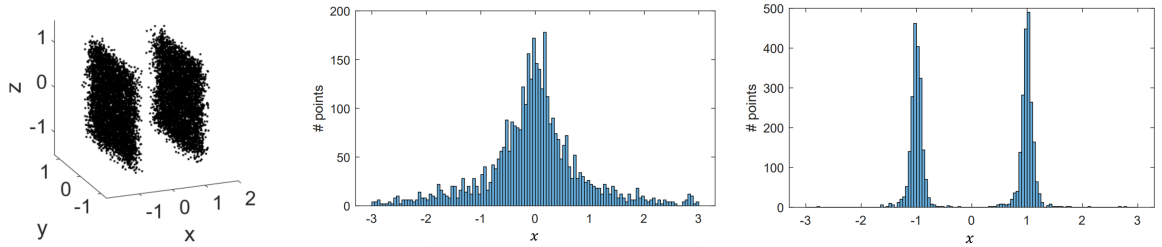
As PPL lifts two points to a line, the PPL-based line clouds are 50% sparser than the OLCs [40] (see Fig. 1), indicating they are more difficult to reveal scene geometry.

3.3. Effect of non-uniform line directions

We first illustrate that the line directions induced by PPL is no longer uniformly distributed on the unit sphere. For this purpose, we again define the set of original 3D points as $\mathcal{P} = \{\mathbf{x}_j\}$ where $\mathbf{x}_j \in \mathbb{R}^3$ is the j-th point in the point cloud. The corresponding lifted lines form a set $\mathcal{L} = \{\mathbf{l}_j\}$ where each $\mathbf{l}_j \in \mathbb{R}^6$ consists of the line direction and offset.

As shown in Fig. 4, the line directions are determined (up to a sign ambiguity) given the point locations $\mathcal{P}$ and their pairing arrangement $\mathcal{S} := \{(i, j)\}$ where (i, j) denotes point i and point j form a pair. Hence, we can write $P(\mathcal{L}|\mathcal{S}, \mathcal{P}) = \delta(\mathcal{L} - \mathcal{L}_\mathcal{S})$ where $\mathcal{L}_\mathcal{S}$ is the set of PPL-based lines created by joining pairs of points as prescribed in $\mathcal{S}$. Assuming each arrangement $\mathcal{S}$ is equally likely (i.e. same $P(\mathcal{S}|\mathcal{P}) \, \forall \, S \in \Omega$), the likelihood can be written as:

$$P(\mathcal{L}|\mathcal{P}) = \sum_{\mathcal{S}\in\Omega} P(\mathcal{L}|\mathcal{S}, \mathcal{P}) P(\mathcal{S}|\mathcal{P}) = \frac{1}{|\Omega|} \sum_{\mathcal{S}\in\Omega} \delta(L - L_\mathcal{S})$$



(a) Grouth truth point cloud (b) Histogram of closest points (cross-plane) (c) Histogram of closest points (same plane)

Figure 6. Visualization of distributions of closest-points-to-other lines formed using paired-point lifting (PPL) in two extreme cases. (a) is the ground truth 3D point cloud comprising two noisy planes, (b) shows the distribution of closest-points-to-other-lines (along a hypothetical line parallel to the x-axis for visualization) when PPL samples pairs of points from different planes and (c) shows the same distribution (along the same hypothetical line for visualization) when PPL samples pairs of points from the same plane. In practice, both effects may be co-visible, hence lines created by joining planar points should be avoided. This provides motivation for PPL+ in §4.

where Ω denotes the set of all possible pairing arrangements of the 3D points and $|\Omega|$ is the total number of possible pairing arrangements. Hence, $P(\mathcal{L}|\mathcal{P})$ is non-uniform and different from the piecewise uniform likelihood exhibited by each line in the original line cloud (OLC) [40].

Effect on peak finding: Regarding Equation (1), the single-peak finding (SPF) algorithm in [3] can be seen as approximately finding the 3D points maximizing the posterior $P(\mathcal{P}|\mathcal{L})$ by maximizing the prior $P(\mathcal{P})$ subject to each $\mathbf{x}_j$ lying on its corresponding line $\mathbf{l}_j$. This is based on the assumption that the likelihood $P(\mathcal{L}|\mathcal{P})$ is piecewise constant since the line directions are uniformly distributed.

For PPL, $P(\mathcal{P}|\mathcal{L}) \propto P(\mathcal{P})(\sum_{\mathcal{S}\in\Omega}\delta(L-L_{\mathcal{S}}))$, so we need to maximize $P(\mathcal{P})$ subject to the points lying on their respective PPL-sampled lines. However, this cannot be efficiently addressed by SPF for the following reasons:

1. due to the non-uniform distribution of line directions, the peak may arise in a location far from the actual positions of points (see Fig. 6b for an example), and
2. even if the line directions end up closely following a uniform distribution for a particular point cloud, the histogram of closest-points-to-other-lines may result in two peaks as we conceal two points per line. While we attempt to partially address this by implementing a basic two-peak finding (TPF) algorithm in §5, estimating two peaks is fundamentally a non-trivial task.

Above statements are based on our empirical observations, and we leave detailed analysis of these for future work.

3.4. Limitations

Despite the aforementioned advantages of PPL, there are two potential limitations associated with PPL.

Potential recovery of planes: Fig. 6 shows a corner case of PPL where the location of 3D points from PPL can be recovered accurately by locating two peaks along the distribution of closest-points-to-other lines if the scene contains two planes and each pair of points is sampled from the same plane. We make efforts to avoid this corner case in §4.

Inferring structure directly from PPL-based line clouds: As shown in Fig. 1, it is normally difficult to infer the scene geometry just from the PPL-based line cloud itself. For certain planar scenes (e.g., a long corridor), however, there could be concerns that the basic outline of the underlying scene geometry may be revealed. In our opinion, details of the underlying structure are still indistinguishable in most real-world planar scenes (more discussions in [18]).

4. Improving PPL by rejecting lines on planes

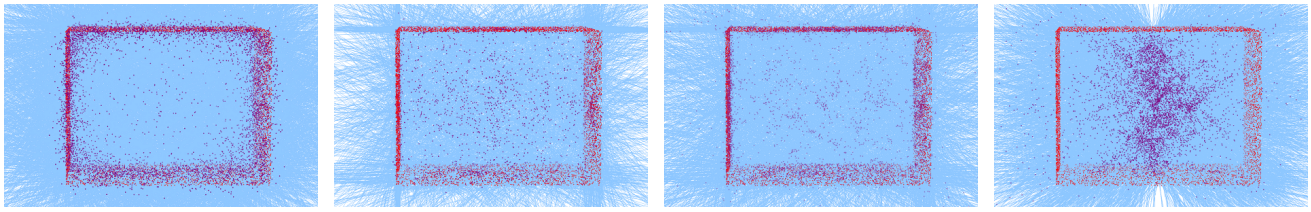
In this section, we propose a simple extension of paired-point lifting to overcome its limitation described in §3.4 regarding potential recovery of planar points.

Our extension called *PPL+* is motivated by Fig. 6c, which shows the lines lifted on planes by joining points from the same plane, can provide an undesirable clue to recover the points, as well as Fig. 6b, demonstrating that removing lines on planes further obfuscates the distribution of closest-points-to-other lines by shifting the peak to an incorrect location.

PPL+ performs rejection sampling of 3D lines to discourage them from lying on planar regions of the point cloud. In summary, PPL+ scans through each point $\mathbf{x}_j$ in the point cloud, and checks whether the $\mathbf{x}_j$'s corresponding line direction is nearly parallel to the plane or not.

For this purpose, PPL+ follows the procedures below:

1. Sample a point $\mathbf{x}_j$ from the point cloud.
2. Find the K-nearest neighboring 3D points of $\mathbf{x}_j$, $\{\mathbf{x}_{j1}, \cdots, \mathbf{x}_{jK}\}$ and compute the normalized displacement vectors $\mathtt{D} := [\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_K]$ between these points and $\mathbf{x}_j$ such that each $\mathbf{d}_k := \mathbf{x}_{jk} - \mathbf{x}_j$.
3. Compute the singular value decomposition (SVD) of $\mathtt{D}$ and choose the basis vector with the largest singular value as a basis vector of the plane $\mathbf{v}_p$.
4. Compute the pseudo-normal vectors of the plane by computing $\mathbf{v}_p \times \mathtt{D}$, and perform SVD and use the basis vector with the largest singular value to estimate the plane normal $\mathbf{v}_o$.



(a) OLC [40] (SPF [3]) (b) PPL (SPF [3]) (c) PPL (TPF) (d) PPL+ (TPF)

Figure 7. Bird's-eye view of 3D points recovered from different types of line cloud constructions using different point estimation (peak-finding) approaches. The line clouds are denoted by blue lines, the ground truth point cloud is shown as red dots and the revealed 3D points are represented as purple dots. (a) shows OLC [40] can be well recovered by SPF [3] while (b) demonstrates applying SPF [3] on PPL-based line cloud incurs geometric uncertainty in 3D points. (c) shows the PPL-based line clouds can be better inverted to point clouds via our two-peak finding algorithm (TPF) in §5, but this can be ameliorated through PPL+ (§4) by rejecting lines on planes as shown in (d).

5. Compute the cosine similarity between $\mathbf{v}_o$ and $\mathbf{l}_j$ to get an angle between two vectors, and subtract from $90°$ in order to acquire the angle between the plane and the lifted line $\mathbf{l}_j$. If the smaller angle is $< 20°$, reject the line as it is nearly on the plane.
6. Move to step 1 unless all 3D points are visited.
7. Randomly form pairs between rejected points and repeat steps 1–7 (for changed lifted lines and respective 3D points) for 1000 iterations.

In practice, $\mathbf{v}_o$ can be stored and re-used such that we just need to re-run steps 5 to 7 iteratively for newly formed pairs. More details are provided in [18]. We also provide a toy example of a room in Fig. 7 demonstrating PPL+ can intentionally mislead the estimated location of 3D points.

Limitation: We note PPL+ with over-strict threshold (e.g. $< 5°$ instead of $< 20°$) may trigger an undesirable effect of yielding degenerate quasi-parallel lifted lines that can degrade pose estimation accuracy. An appropriate hyperparameter needs to be selected for desired performance.

5. Experimental results

In this section, we present extensive qualitative and quantitative results demonstrating the effectiveness of paired-point lifting in terms of three metrics, namely quality of estimated point clouds, quality of recovered image and camera relocalization accuracy, all of which are essential metrics related to privacy-preserving visual localization.

Datasets: As described in [3], all experiments were carried out on two widely-used public datasets called Cambridge Landmarks [14, 15] and Energy Landscape [44]. We mostly followed the protocols suggested in [40]: for each scene, we constructed a sparse 3D point cloud using COLMAP [36] across all images, and discarded all 2D-to-3D correspondences for query images in order to maintain an accurate 3D map while simulating localization from an unseen image (see [18]). The *TrinityGreatCourt* and *Street* sequences contained excessive number of outliers and subsequently were excluded from comparison.

Peak-finding algorithms: We provide results of estimating points from line clouds using both the *single-peak finding (SPF)* [3] (in §2) and *two-peak finding (TPF)* algorithms. TPF is designed to handle the case of bimodal distribution of closest-points-to-other-lines frequently exhibited by PPL as shown in Fig. 6. It utilizes kernel density estimation (KDE) [28, 33], which can be used to detect multiple peaks via thresholding. TPF is introduced to demonstrate that the PPL's performance gain is not simply due to the limitation of SPF [3] not being able to pick up two peaks.

Regarding algorithmic details, we set the number of iterations for refinement to 3 for SPF as in [3]. For TPF, we excluded refinement procedures as the estimation result did not converge. The width of the Gaussian kernel estimator was set to 0.07 found through our empirical investigation. More algorithmic details can be found in [18].

5.1. Comparing quality of revealed point clouds

In order to quantify the geometric uncertainty induced by different line-sampling approaches, we compared the errors between the ground truth point cloud and the point clouds revealed from the line clouds via peak finding.

As shown in Table 1, the 3D point errors of PPL and PPL+ are larger than those of OLC (100%), indicating the combined effect of sparsity and non-uniform distribution of lines prevent accurate reconstruction of the point cloud. When comparing against OLC with 50% sparsity, our methods still produce larger point reconstruction errors, and we anticipate this is due to the non-trivial distribution of line directions making peak finding difficult.

The difficulty of inverting a PPL-based line cloud to point cloud is also qualitatively illustrated in Fig. 5 in which the point clouds reconstructed from PPL/PPL+-based line clouds are more sparse and noisy than that from the OLC.

One rather surprising result is that the point estimation errors are higher for TPF than those for SPF. While this requires further investigation, we cautiously think this is due to mismatches between the location of peaks and the ground truth 3D points rather than TPF failing to detect them. For this purpose, we compared peaks found by humans against

Dataset	Energy landscape [44]							Cambridge [15]						
Scene representation	PC	OLC		PPL		PPL+		PC	OLC		PPL		PPL+	
Minimal solver	P3P	P6L		P6L		P6L		P3P	P6L		P6L		P6L	
Original point used (%)	100	100	50	100	100	100	100	100	100	50	100	100	100	100
Line cloud used (%)	-	100	50	50	50	50	50	N/A	100	50	50	50	50	50
Peak finding algorithm used	-	SPF	SPF	SPF	TPF	SPF	TPF	N/A	SPF	SPF	SPF	TPF	SPF	TPF
Proportion of revealed points (%)	100	100	50	50	100	50	100	100	100	50	50	100	50	100
Median 3D point error (m, ↑)	-	0.046	0.055	0.056	0.224	0.056	0.227	-	0.752	0.854	0.973	2.845	0.948	2.858
Image quality (PSNR ↓)	15.70	13.30	12.84	**10.42**	11.21	10.44	11.22	14.00	12.67	12.40	**11.20**	11.43	11.22	11.40
Image quality (SSIM ↓)	0.514	0.402	0.388	0.356	0.325	0.355	**0.323**	0.433	0.331	0.320	0.296	0.283	0.297	**0.283**
Image quality (MAE ↑)	33.58	43.62	45.83	**62.40**	56.99	62.11	56.93	39.41	46.21	47.93	**55.75**	54.01	55.42	54.22
Mean rotation error (°, ↓)	0.178	0.272	0.294	0.283		0.287		0.700	0.807	0.833	0.789		0.809	
Mean translation error (m, ↓)	0.0071	0.0100	0.0108	0.0105		0.0106		0.141	0.184	0.195	0.193		0.195	
Median rotation error (°, ↓)	0.031	0.104	0.116	0.112		0.116		0.100	0.194	0.200	0.196		0.194	
Median translation error (m, ↓)	0.0010	0.0032	0.0036	0.0035		0.0037		0.042	0.072	0.078	0.085		0.089	

Table 1. Results obtained from different scene representations on two public datasets (PC: point cloud, OLC: original line cloud, PPL: paired-point lifting and PPL+: PPL with rejection sampling). ↑ denotes higher values imply better privacy-preserving behavior, and vice versa for ↓. Bold text denotes lowest image quality for each metric. Mean and median values are averaged across all views and sequences.

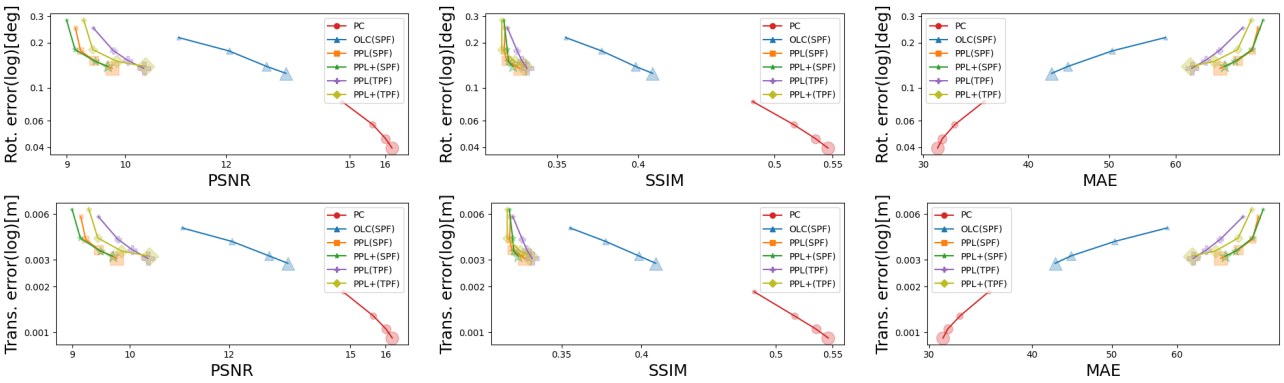


Figure 8. Pose estimation error versus image quality across different levels of point cloud sparsity for the scene 'Office1_manolis'. The size of the marker in the graph represents the proportion of the sampled points in the following order: 100% 50% 25% 10% 5% cloud density. The PPL class of approaches achieve lower PSNR and SSIM values and higher MAE values than those of OLC with comparable camera pose estimation errors, suggesting PPL/PPL+ enhance the privacy-preserving visual localization performance.

those found by TPF, and the results in the supplementary material [18] shows TPF mostly functions as expected.

5.2. Comparing quality of revealed images

We also compared the quality of revealed images from different scene representations.

For a quantitative comparison, we reconstructed images of the scenes by projecting a 3D point cloud (revealed via peak-finding for line clouds) to 2D using the ground truth camera poses and feeding this along with feature descriptors to the InvSfM network [30]. We then computed the average PSNR, SSIM and MAE values of the revealed images against the ground truth images.

Table 1 shows that the image quality of recovered images from PPL and PPL+ are much lower than those from OLC (100% and 50%), indicating improved privacy preserving performance even without the effect of reduced cloud density. This trend is maintained even when SPF is replaced by TPF, demonstrating the potential errors due to misassignment of descriptors also affect image quality.

In order to disentangle the effect of feature misassignment and geometric uncertainty (induced by non-uniform distribution of lines), we resort to the qualitative comparison of revealed images from different scene representations and different peak finding algorithms. The comparison includes an oracle setting in which the effect of feature descriptor misallocation in PPL is minimized to only visualize the effect of non-uniformly distributed line directions (mimicing 0% feature swap) by assigning descriptors based on the two points' proximity to ground truth points.

As shown in Fig. 9, image quality is disimproved by both the effect of feature swap and non-uniformly distributed line directions. Additionally, Fig. 9h and 9i demonstrate PPL+ further degrades the image reconstruction quality, enhancing the privacy preserving performance.

(a) Ground truth (b) Point cloud (c) OLC [40] (d) PPL (SP-O) (e) PPL (TP-O) (f) PPL+ (TP-O) (g) PPL(SPF [3]) (h) PPL (TPF) (i) PPL+ (TPF)

Figure 9. Images revealed from various scene representations using InvSfM [30]. To disentangle the effect of feature swapping and geometric uncertainty in PPL, we provide results for the "oracle" case (SP-O and TP-O) where the ground truth feature descriptor is assigned for each revealed point based on the point's proximity to ground truth points thereby minimizing the effect of descriptor misassignment.

5.3. Comparing camera relocalization accuracy

We also compared the camera pose estimation accuracy when using different types of scene representations.

For this purpose, we estimated the absolute pose of the query image from the constructed 3D point cloud maps. This was carried out using the P3P solver [29] for point clouds (PC) and the P6L solver [42] for line clouds (OLC, PPL and PPL+). Our code implementation is based on Poselib [17] maintained by Viktor Larsson, and we made a few adjustments to accommodate the point-to-line distance as the objective function, enforce a cheirality constraint for P6L and add bundle adjustment for line clouds.

Following [40, 41], we evaluated the localization accuracy by computing the rotational error $\Delta\mathtt{R} = \arccos\big(\mathrm{Tr}(\hat{\mathtt{R}}^\top\mathtt{R}) - 1\big)/2$ and the translation error $\boldsymbol{\Delta t} = \big\|\hat{\mathtt{R}}^\top\hat{\mathbf{t}} - \mathtt{R}^\top\mathbf{t}\big\|_2$, where $(\mathtt{R}, \mathbf{t})$ denote the ground truth absolute rotation and translation of the query image and $(\hat{\mathtt{R}}, \hat{\mathbf{t}})$ are the estimated absolute rotation and translation of the query image. In order to yield translation error in meters, we multiplied a number to all camera translations in a sequence to match the translation scale of the ground truth datasets provided in [14, 15, 44] (see [18] for details).

Table 1 shows visual localization on PPL/PPL+-based line clouds show similar pose accuracy when compared with OLC with standard lifting [40]. Specifically, the median rotation and translation errors of PPL/PPL+ are lower or at least similar to those of OLC with 50% sparsity, which have the same number of lines for pose estimation.

For a detailed comparison, we also plot pose estimation accuracy against image reconstruction quality across different scene representations and different peak finding algorithms in Fig. 8. The plots confirm PPL/PPL+ are more robust to inversion attacks than OLC without compromising camera relocalization accuracy.

6. Conclusion

This work was motivated by the major limitation of the original line cloud (OLC) that the point cloud and the respective scene details can be revealed by utilizing the statistics of the 3D points and lines. To mitigate this issue, we proposed a new lightweight strategy called paired-point lifting (PPL), which creates line clouds by joining random pairs of sparse points thereby concealing two points per line. We illustrated three main benefits of PPL, namely added uncertainty in descriptor assignment for each line, increased sparsity of the cloud representation and non-uniform distribution of line directions, all of which were experimentally demonstrated to enhance the privacy-preserving nature of the line clouds without trading off camera localization performance. We also proposed an extension of PPL (PPL+) to avoid theoretical corner cases. We put effort into comparing between different scene representations on a level playground by implementing a two-peak finding algorithm to handle PPL-based line clouds.

An interesting avenue for future work includes devising a more protective lifting approach or developing a faster relocalization algorithm for line clouds (e.g. by combining homotopy and deep learning such as in [11]) for real-time applications in need of privacy-preserving measures.

Acknowledgement This work was partly supported by the National Research Foundation of Korea(NRF) grants funded by the Korea government(MSIT) (No. 2022R1C1C1004907, No. 2022R1A5A1022977), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2020-0-01373, Artificial Intelligence Graduate School Program(Hanyang University)) and Hanyang University (HY-202100000003084).

References

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building rome in a day. In *2009 IEEE 12th International Conference on Computer Vision (ICCV)*, pages 72–79, 2009. 1

[2] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC — Differentiable RANSAC for camera localization. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2492–2500, 2017. 1

[3] 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. 1, 2, 3, 4, 5, 6, 8

[4] Deeksha Dangwal, Vincent T Lee, Hyo Jin Kim, Tianwei Shen, Meghan Cowan, Rajvi Shah, Caroline Trippel, Brandon Reagen, Timothy Sherwood, Vasileios Balntas, et al. Mitigating reverse engineering attacks on local feature descriptors. In *British Machine Vision Conference 2021, BMVC 2021*, 2021. 2

[5] Andrew J. Davison, Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 2007. 1

[6] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4829–4837, 2016. 2

[7] Mihai Dusmanu, Johannes L. Schönberger, Sudipta N. Sinha, and Marc Pollefeys. Privacy-preserving image features via adversarial affine subspace embeddings. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14262–14272, 2021. 3

[8] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981. 1

[9] Marcel Geppert, Viktor Larsson, Pablo Speciale, Johannes L Schönberger, and Marc Pollefeys. Privacy preserving structure-from-motion. In *European Conference on Computer Vision (ECCV)*, pages 333–350. Springer, 2020. 3

[10] Marcel Geppert, Viktor Larsson, Pablo Speciale, Johannes L Schonberger, and Marc Pollefeys. Privacy preserving localization and mapping from uncalibrated cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1809–1819, 2021. 3

[11] Petr Hruby, Timothy Duff, Anton Leykin, and Tomas Pajdla. Learning to solve hard minimal problems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5532–5542, 2022. 8

[12] Arnold Irschara, Christopher Zach, Jan-Michael Frahm, and Horst Bischof. From structure-from-motion point clouds to fast location recognition. In *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2599–2606, 2009. 1

[13] Hiroharu Kato and Tatsuya Harada. Image reconstruction from bag-of-visual-words. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 955–962, 2014. 2

[14] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6555–6564, 2017. 6, 8

[15] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2938–2946, 2015. 6, 7, 8

[16] Nicolaas H Kuiper. Tests concerning random points on a circle. In *Nederl. Akad. Wetensch. Proc. Ser. A*, volume 63, pages 38–47, 1960. 3

[17] Viktor Larsson. PoseLib - Minimal Solvers for Camera Pose Estimation. https://github.com/vlarsson/PoseLib, 2020. Accessed: 2022-10-30. 8

[18] Chunghwan Lee, Jaihoon Kim, Chanhyuk Yun, and Je Hyeung Hong. Supplementary document for paired-point lifting for enhanced privacy-preserving visual localization. https://github.com/Fusroda-h/ppl, 2023. Accessed: 2023-03-20. 4, 5, 6, 7, 8

[19] Yunpeng Li, Noah Snavely, Daniel P Huttenlocher, and Pascal Fua. Worldwide pose estimation using 3D point clouds. *Large-Scale Visual Geo-Localization*, pages 147–163, 2016. 1

[20] Hyon Lim, Sudipta N Sinha, Michael F Cohen, and Matthew Uyttendaele. Real-time image-based 6-DOF localization in large-scale environments. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1043–1050. IEEE, 2012. 1

[21] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004. 1, 2

[22] Simon Lynen, Torsten Sattler, Michael Bosse, Joel A Hesch, Marc Pollefeys, and Roland Siegwart. Get out of my lab: Large-scale, real-time visual-inertial localization. In *Robotics: Science and Systems*, volume 1, page 1, 2015. 1

[23] Simon Lynen, Bernhard Zeisl, Dror Aiger, Michael Bosse, Joel Hesch, Marc Pollefeys, Roland Siegwart, and Torsten Sattler. Large-scale, real-time visual-inertial localization revisited. *The International Journal of Robotics Research*, 39(9):1061–1084, 2020. 1

[24] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5188–5196, 2015. 2

[25] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017. 1

[26] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. 1

[27] Tony Ng, Hyo Jin Kim, Vincent T Lee, Daniel DeTone, Tsun-Yi Yang, Tianwei Shen, Eddy Ilg, Vassileios Balntas, Krystian Mikolajczyk, and Chris Sweeney. NinjaDesc: Content-concealing visual descriptors via adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12797–12807, 2022. 3

[28] Emanuel Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962. 6

[29] Mikael Persson and Klas Nordberg. Lambda twist: An accurate fast robust perspective three point (p3P) solver. In *Proceedings of the European conference on computer vision (ECCV)*, pages 318–332, 2018. 8

[30] 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. 1, 2, 4, 7, 8

[31] Rahul Raguram, Ondrej Chum, Marc Pollefeys, Jiri Matas, and Jan-Michael Frahm. USAC: A universal framework for random sample consensus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):2022–2038, 2012. 1

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241. Springer, 2015. 2

[33] Murray Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, pages 832–837, 1956. 6

[34] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *2011 International Conference on Computer Vision (ICCV)*, pages 2564–2571, 2011. 1

[35] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(9):1744–1756, 2016. 1

[36] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 6

[37] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 1

[38] Mikiya Shibuya, Shinya Sumikura, and Ken Sakurada. Privacy preserving visual SLAM. In *European Conference on Computer Vision (ECCV)*, pages 102–118. Springer, 2020. 3

[39] Noah Snavely, Steven M Seitz, and Richard Szeliski. Modeling the world from internet photo collections. *International Journal of Computer Vision (IJCV)*, 80:189–210, 2008. 1

[40] 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, 5, 6, 8

[41] Pablo Speciale, Johannes L Schonberger, Sudipta N Sinha, and Marc Pollefeys. Privacy preserving image queries for camera localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, pages 1486–1496, 2019. 8

[42] Henrik Stewénius, Magnus Oskarsson, Kalle Aström, and David Nistér. Solutions to minimal generalized relative pose problems, 2005. 3, 8

[43] Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. SOSNet: Second order similarity regularization for local descriptor learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11016–11025, 2019. 1

[44] 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. 6, 7, 8

[45] Carl Vondrick, Aditya Khosla, Tomasz Malisiewicz, and Antonio Torralba. HOGgles: Visualizing object detection features. In *2013 IEEE International Conference on Computer Vision (ICCV)*, pages 1–8, 2013. 2

[46] Philippe Weinzaepfel, Hervé Jégou, and Patrick Pérez. Reconstructing an image from its local descriptors. In *CVPR 2011*, pages 337–344. IEEE, 2011. 2

[47] Bernhard Zeisl, Torsten Sattler, and Marc Pollefeys. Camera pose voting for large-scale image-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2704–2712, 2015. 1