Efficient Privacy-Preserving Visual Localization Using 3D Ray Clouds

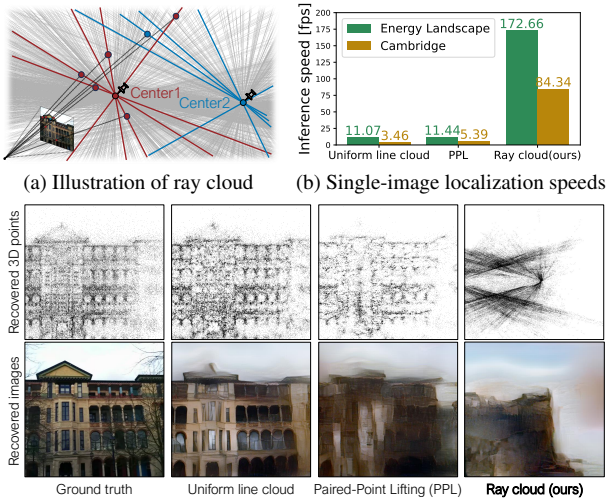
Heejoon Moon[1] Chunghwan Lee[2] Je Hyeong Hong[1,2†]
[1]Dept. Artificial Intelligence, Hanyang University [2]Dept. Electronic Engineering, Hanyang University

Abstract

The recent success in revealing scene details from sparse 3D point clouds obtained via structure-from-motion has raised significant privacy concerns in visual localization. One prominent approach for mitigating this issue is to lift 3D points to 3D lines thereby reducing the effectiveness of the scene inversion attacks, but this comes at the cost of increased algorithmic complexity for camera localization due to weaker geometric constraints induced by line clouds. To overcome this limitation, we propose a new lifting approach called "ray cloud", whereby each lifted 3D line intersects at one of two predefined locations, depicting omnidirectional rays from two cameras. This yields two benefits, i) camera localization can now be cast as relative pose estimation between the query image and the calibrated rig of two perspective cameras which can be efficiently solved using a variant of the 5-point algorithm, and ii) the ray cloud introduces erroneous estimations for the density-based inversion attack, degrading the quality of scene recovery. Moreover, we explore possible modifications of the inversion attack to better recover scenes from the ray clouds and propose a ray sampling technique to reduce the effectiveness of the modified attack. Experimental results on two public datasets show real-time localization speed as well as enhanced privacy-preserving capability over the state-of-the-art without overly sacrificing the localization accuracy.

1. Introduction

Visual localization refers to the task of estimating the camera pose w.r.t. a known 3D scene given a query image. It serves a central role for cutting-edge applications involving robotics [18], augmented reality (AR) [3], virtual reality (VR) [6] and simultaneous localization and mapping (SLAM) [11, 29, 33]. To this date, many visual localization algorithms [27, 37] still utilize sparse 3D point clouds constructed via structure-from-motion (SfM) [1, 44, 45, 48] through which the camera pose is estimated from the correspondences between 2D image features and 3D points. This is commonly solved using a perspective-n-point (PnP) solver [12, 20, 37] equipped with the random sample consensus (RANSAC) meta-algorithm [9, 15, 42] or its deep network alternative such as differentiable RANSAC [4, 5].



(a) Illustration of ray cloud (b) Single-image localization speeds

(c) Comparison of revealed scene geometries and images

Figure 1. Main characteristics of the proposed ray cloud. (a) shows 3D points lifted as 3D rays half of which intersects at center 1 and the other half intersects at center 2. Estimating the camera pose resembles relative pose estimation with respect to a calibrated rig of perspective cameras. As shown in (b), this allows real-time single-image-based localization on two public datasets [22, 55] while preventing scene-revealing attacks [7, 40] as shown in (c).

Surprisingly, it was recently shown [8, 10, 13, 30, 40, 57, 58] that a sparse 3D point cloud may contain enough information to reveal image details of the underlying scene. Most prominently, Pittaluga *et al*. [40] demonstrated a cascaded U-Net model [43] can generate high-fidelity images of the scene just from the 2D projection of sparse 3D points and their SIFT features [28]. This raised significant privacy concerns [39, 50] as anyone with access to the 3D point cloud could potentially search for sensitive details in the private scene by employing this scene inversion network.

Currently, the mainstream approach for preventing the aforementioned image inversion attack on the sparse 3D point cloud is geometric lifting [26, 46, 50], whereby each 3D point is transformed to a 3D line passing through the original point with aims to conceal the 3D scene geometry (see Fig. 2).While the earlier approach of randomly drawing the line direction (also known as the uniform line cloud) was shown to be susceptible to geometry inversion attack [7],

†Corresponding author

this was remedied by recent methods utilizing pairing of 3D points [26, 36] to increase ambiguity in the lifted scene representation. Nevertheless, single image-based localization speeds for these algorithms [26, 36] are far from achieving real-time performance, providing motivation for this work.

In this paper, we present a new geometric lifting approach called *ray cloud* with aims to address above issue of slow camera localization runtime exhibited by previous studies. As will be illustrated in Sec. 3, ray cloud is simple to construct and robust to the geometry inversion attack [7]. Most importantly, camera localization using a ray cloud can be formulated as a well-known relative pose estimation problem between the query image and a virtual calibrated rig of cameras, enabling utilization of the efficient 5-point algorithm [35] with known scale [59]. Moreover, by modifying and enhancing the geometry inversion attack [7] in Sec. 4, we explore potential hidden privacy risks associated with ray clouds, and subsequently propose a technique to enhance the privacy protection of ray clouds in Sec. 5.

The contributions of this work are summarized below:

- an efficient privacy-preserving visual localization framework based on a new lifting approach called 3D ray cloud,
- customization of the geometry inversion attack [7] to yield undesired (privacy leak) corner cases for ray clouds,
- a voxel-based ray sampling technique to prevent above privacy leaks by disrupting nearest neighbor statistic and inducing erroneous recovery of the scene points, and
- extensive experimental comparison of ray cloud against other line cloud-based approaches on two public datasets, achieving real-time single-image based localization speed for the first time while preserving privacy-preserving capability and maintaining similar localization accuracy.

2. Related work

Privacy threats from image inversion attack While several prior studies addressed the possibility of revealing images from sparse keypoints [13, 19, 30, 57, 58], it was Pittaluga *et al*. [40] who posed a real privacy threat for the first time by recovering detailed scene images from a sparse 3D point cloud using a cascaded U-Net [43]. This network called InvSfM takes in the 2D locations of the projected 3D points as well as the corresponding depths and SIFT descriptors, and outputs a synthesized image of the scene.

More recently, Song *et al*. [49] used a PointNet++ architecture to extract deep features and synthesize scene images. Dangwal *et al*. [10] devised a generic inversion model that can work with various descriptors [2, 28, 53] and proposed selective feature suppression to hide privacy-sensitive regions. Nonetheless, the pretrained InvSfM model still serves as the baseline for qualitative and quantitative analyses in privacy-preserving visual localization [7, 26, 36, 38].

Privacy-preserving scene representations After the emergence of InvSfM [40], Speciale *et al*. [50] quickly embraced a geometric lifting approach, also known as the *uniform line cloud*, where each 3D point in the original sparse point cloud is replaced by a 3D line passing through the point with its direction drawn from a uniform distribution on the unit sphere. This scene representation aims to introduce anonymity to the locations of the 3D points thereby preventing any image inversion via InvSfM. The work was later extended by Shibuya *et al*. [46] to visual SLAM with a prebuilt map provided in the form of uniform line cloud.

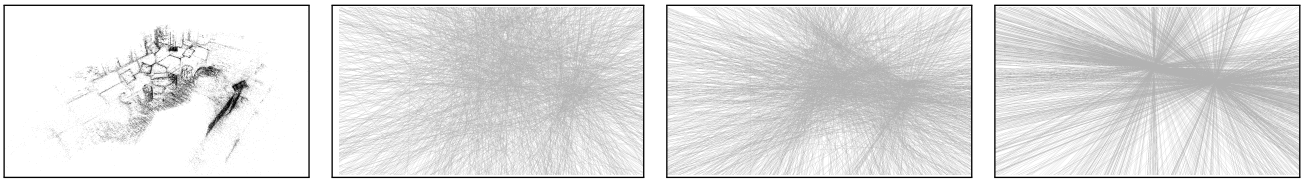
Unfortunately, above methods [46, 50] are now outdated and susceptible to the geometry inversion attack proposed by Chelani *et al*. [7], which successfully recovers the sparse 3D point cloud from a uniform line cloud (details in the next subsection). To overcome this limitation, Lee *et al*. [26] proposed to draw 3D lines through random pairs of 3D points, thereby introducing non-uniform line distributions and multiple 3D point candidates for each line both of which are empirically shown to hinder point cloud inversion. Pan *et al*. [36] proposed to permute the coordinates of random pairs of keypoints to preserve privacy while still enabling localization from multiple axis-aligned lines. Geppert *et al*. [16] presented a partial localization approach whereby three 1-dimensional partial maps are constructed from the 3D points and separately stored to enhance security. While these approaches can evade the geometry inversion attack, they are not runtime-efficient to allow real-time camera localization from single images. This issue is addressed through use of the ray cloud proposed in this work.

Other explored approaches include keypoint lifting [14], carving [34] or obfuscation [41], whereby the feature descriptors of the 3D points are transformed to hide keypoint details, but this does not conceal the point cloud geometry.

Privacy threats for line clouds Chelani *et al*. [7] presented a density-based inversion attack that can reveal the 3D point geometry from a uniform line cloud [50]. While the posterior probability of the point cloud $\mathcal{P}$ given a line cloud $\mathcal{L}$ can be expressed as $P(\mathcal{P}|\mathcal{L}) \propto P(\mathcal{P})P(\mathcal{L}|\mathcal{P})$ using Bayes' rule, it is noted that the likelihood $P(\mathcal{L}|\mathcal{P})$ is constant for uniform line clouds. Hence, maximizing the posterior amounts to maximizing the marginal prior $P(\mathcal{P})$.

For maximizing $P(\mathcal{P})$, Chelani *et al*. [7] resorted to an empirical observation that for any two 3D points $\mathbf{x}_A$ and $\mathbf{x}_B$ with their corresponding uniformly-drawn lines $\mathbf{l}_A \in Gr(1,3)$ and $\mathbf{l}_B \in Gr(1,3)$, $P(\left\|\mathbf{x}_A - \mathbf{x}_A^{\perp}\right\|_2 < \|\mathbf{x}_A - \mathbf{x}_B\|_2) \approx 0.8$, where $\mathbf{x}_A^{\perp}$ denotes the point on $\mathbf{l}_A$ closest to $\mathbf{l}_B$. This implies that if $\mathbf{x}_A$ and $\mathbf{x}_B$ are close to each other, then $\mathbf{x}_A^{\perp}$ is likely to become a good approximation of $\mathbf{x}_A$. Hence, $P(\mathcal{P})$ can be maximized if the nearest neighboring points of each 3D point are known in advance.

As it is non-trivial to find a set of true k-nearest neighboring points in line clouds, Chelani *et al*. [7] approximates this by the set of K-nearest lines $\{\mathbf{l}_K\}$ for each line $\mathbf{l}_i$, expecting some to include the true positives of the k-nearest



(a) Point cloud (b) Uniform line cloud [50] (c) Paired-Point Lifting (PPL) [26] (d) Ray cloud (ours)

Figure 2. 3D line clouds yielded from different lifting approaches on the *Office1 manolis* in the Energy Landscape [55] dataset. Note each line (ray) in the ray cloud passes through one of two center points. We show 10% of the total number of lines for visual comparison.

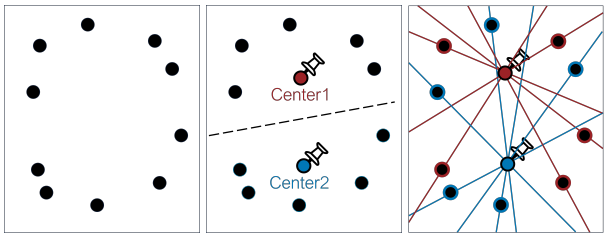
points when $K > k$ is large enough (we distinguish K for lines and k for points). Bearing this assumption, [7] computes the closest point $\mathbf{x}_k^{\perp}$ to each of the K-nearest lines for the current line $\mathbf{l}_i$, builds a histogram of closest points $\{\mathbf{x}_k^{\perp}\}$ and finds the peak of the histogram via Kuiper's test [23] to estimate the point $\mathbf{x}_i$. This attack is effective for uniform line clouds [50], allowing recovered points to be used for scene recovery with InvSfM [40]. However, it is less effective against more recent line-based approaches [26, 36] and ray clouds presented in this work.

Camera localization using line clouds As mentioned in Sec. 1, camera localization typically involves a RANSAC algorithm [15] coupled with an efficient minimal solver that can run fast over thousands of iterations [25]. For single-image based localization, a standard line cloud introduces 2D points-to-3D line constraints, which are weaker than 2D point-to-3D point constraints provided by the point clouds. Consequently, the minimal solver for line clouds (*p*6L) requires 6 correspondences between 2D points and 3D lines [50]. Since *p*6L innately uses the solver for generalized relative pose estimation [52], achieving real-time speed with *p*6L is challenging due to the large number of possible solutions (up to 64) and high algorithmic complexity [52, 56]. Shibuya *et al*. [46] achieved real-time localization with line clouds in the SLAM setup, but this requires a video stream as input and construction of a client-side point map. On the other hand, we will show it is possible to perform image-based localization in real-time using ray clouds.

3. 3D Ray cloud

We now illustrate the process of constructing a 3D ray cloud shown in Fig. 2 and estimating camera pose from ray clouds. We also discuss inversion attack [7] on ray clouds.

Motivation The concept of lifting 3D lines to intersect at a particular point has been briefly addressed without detailed investigation in [7, 47]. Chelani *et al*. [7] mentioned that, if all lines were to intersect at a single point, then the geometry inversion attack in Sec. 2 would fail. This is because the histogram of closest points (to other lines) becomes concentrated about the point of intersection, leading to a degenerate solution. Unfortunately, this representation possesses an intrinsic limitation that the camera pose can only be recovered up to scale as camera localization resembles relative pose estimation [35] between the query camera and the intersecting 3D lines. We aim to overcome this issue by utilizing two intersection points. This can be viewed as adding another camera to the scene, forming a calibrated rig of perspective cameras to resolve the scale ambiguity.



(a) Original 3D points (b) Center assignment (c) Ray cloud

Figure 3. Illustration of the procedure for selecting points of intersection and constructing a 3D ray cloud.

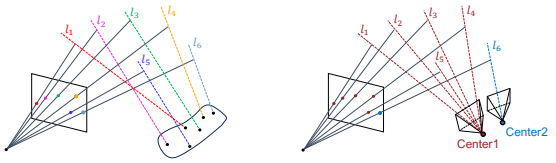
3.1. Construction procedure

Lifting 3D points to a ray cloud comprises three steps, namely setting two ray centers and assigning 3D points to one of two ray centers. We choose to set the ray centers by applying the K-means clustering algorithm on the point cloud with K=2, from which the centers of two clusters are set as the ray centers (explained below). Then, we assign half of randomly sampled points from the point cloud to center 1 ($\mathbf{c}_1 \in \mathbb{R}^3$), and the other half to center 2 ($\mathbf{c}_2 \in \mathbb{R}^3$). Note this assignment is not related to the K-means clustering. Finally, each point $\mathbf{x}_i$ is lifted to a line as $\mathbf{c}_j + \lambda(\mathbf{x}_i - \mathbf{c}_j)$ for $\lambda \in \mathbb{R}$, where $j \in [1, 2]$ is the assigned center index for point i. The final output resembles omnidirectional rays from two perspective cameras (see Fig. 3).

Effect of changing ray center locations If the ray centers are located very close to each other, then they begin to resemble a single ray center, increasing ambiguity in translation scale. On the other hand, if the ray centers are distant from each other and the scene, then the resulting rays are nearly parallel which degrades localization accuracy. We empirically observe that using the centers from K-means clustering usually avoids both extreme cases (see [32]).

3.2. Camera pose estimation

We now illustrate an efficient localization approach for ray clouds that can benefit from faster inference speed com-



(a) Standard line cloud (p6L) (b) Ray cloud (p5+1R)

Figure 4. Illustration of minimal pose estimation problems solved by different line representations. Camera localization using a uniform line cloud can be seen as the special case of generalized relative pose estimation problem [52] involving one perspective camera (query) and one generalized camera (line cloud). On the other hand, camera localization using a ray cloud can be regarded as the case of perspective relative pose estimation problem between the query image and a rig of calibrated cameras (ray cloud). The latter enjoys shorter runtime as shown in Sec. 6 partly due to stronger ray constraints reducing the number of algebraic solutions.

pared to other line-based scene representations [26, 36, 50]. This is inspired by the illustration in Fig. 4b that the ray cloud can be viewed as two virtual perspective cameras located at the ray centers $\mathbf{c}_1 \in \mathbb{R}^3$ and $\mathbf{c}_2 \in \mathbb{R}^3$. Since the rays from these cameras are omnidirectional, we can set their rotation matrices $\mathtt{R}_1, \mathtt{R}_2 \in SO(3)$ to be aligned with world coordinates, *i.e.* $[\mathtt{R}_1|\mathbf{t}_1] = [\mathtt{I}|-\mathbf{c}_1]$ and $[\mathtt{R}_2|\mathbf{t}_2] = [\mathtt{I}|-\mathbf{c}_2]$, where $\mathbf{t}_1, \mathbf{t}_2 \in \mathbb{R}^3$ are the camera translations.

Below, we show that having 5 correspondences between 2D keypoints from the query image and 3D rays from a randomly selected virtual camera and one correspondence between a 2D query keypoint and a 3D ray from a remaining virtual camera allows us to estimate the absolute camera pose using a ray cloud. We denote this adaptation of minimal 5+1 solver [59] as the perspective-(5+1)-rays (p5+1R) solver. Similar to previous studies [26, 50], we assume the camera intrinsics $\mathtt{K} \in \mathbb{R}^{3\times3}$ are known.

***p*5+1R solver** We start by estimating the relative pose between the query image and the first virtual camera using the 5-point algorithm [35]. It yields multiple candidates for the relative pose of the query image w.r.t. the first virtual camera defined as $[\mathtt{R}_{q1}|\alpha\hat{\mathbf{t}}_{q1}]$, where $\mathtt{R}_{q1} \in SO(3)$ is the relative rotation, $\hat{\mathbf{t}}_{q1} \in S^2$ is the normalized relative translation and $\alpha \in \mathbb{R}$ is an unknown scale factor. Subsequently, the absolute pose of query camera is $[\mathtt{R}_{q1}|\alpha\hat{\mathbf{t}}_{q1} - \mathtt{R}_{q1}\mathbf{c}_1]$.

In order to disambiguate α, we utilize the additional correspondence between a normalized query keypoint $\mathbf{p} := (p_1, p_2)^\top \in \mathbb{R}^2$ and a 3D ray $\mathbf{l} \in Gr(1,3)$ from the second virtual camera. We note the ray from the query camera center passing through $\mathbf{p}$ would ideally intersect $\mathbf{l}$. Expressing both rays in the query-centered coordinates yields

$$\lambda_1[p_1, p_2, 1]^\top = \lambda_2\mathtt{R}_{q1}\hat{\mathbf{n}} + \alpha\hat{\mathbf{t}}_{q1} - \mathtt{R}_{q1}\mathbf{c}_1 \tag{1}$$

where $\hat{\mathbf{n}} \in S^2$ is the line direction of $\mathbf{l}$ in world coordinates and $\lambda_1, \lambda_2 \in \mathbb{R}$ are scale parameters. Since we have 3 equations with 3 unknowns, α can be determined.

While the p5+1R solver requires 6 correspondences like p6L [26, 50], the utilized 5-point algorithm only yields up to 10 solutions [35] that is smaller than 64 solutions from the generalized solver [52] in p6L. This significantly reduces the computational complexity for determining the correct configuration. We also enforce the cheirality constraint [17] on the query image to discard infeasible solutions.

Robust estimation We equip above p5+1R solver with the LO-RANSAC [31] algorithm for robust camera localization. For selecting the best model candidate, we adopt an MLESAC [54]-type cost function based on the squared sum of epipolar distances in the query image defined as

$$\sum_{(i,j)\in\Omega} \rho\left(\frac{([\mathbf{p}_i^\top, 1]\,\mathtt{E}_j\,\hat{\mathbf{n}}_i)^2}{(\mathbf{e}_{j1}^\top\hat{\mathbf{n}}_i)^2 + (\mathbf{e}_{j2}^\top\hat{\mathbf{n}}_i)^2}\right), \tag{2}$$

where $\rho(s) := \min(s, \tau^2)$ is a robust kernel with upper bound of threshold τ^2, Ω denotes the pairing arrangement of which keypoint i matches with ray i from center j, $\mathbf{p}_i \in \mathbb{R}^2$ is the i-th normalized keypoint in the query image, $\hat{\mathbf{n}}_i \in S^2$ is ray i passing through the ray center $j \in [1, 2]$, and $\mathtt{E}_j := [\mathbf{e}_{j1}, \mathbf{e}_{j2}, \mathbf{e}_{j3}]^\top$ is the essential matrix between the virtual camera j and the query camera. Since the two virtual cameras share the same rotation and only differ by translation, $\mathtt{E}_2$ can be easily computed from $\mathtt{E}_1$. The best solution from RANSAC is refined by iteratively minimizing (2).

3.3. Geometry inversion attack on ray clouds

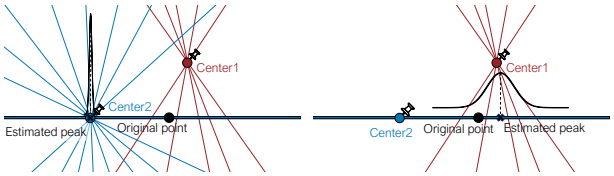
We show that applying the geometry inversion attack [7] naively to a ray cloud yields a degenerate solution. For this purpose, we make use of some prior knowledge regarding the probabilistic viewpoint and algorithmic details of the geometry inversion attack [7] reviewed in Sec. 2.

Let $\mathbf{x}_i \in \mathbb{R}^3$ be the i-th point and $\mathbf{l}_i \in Gr(1,3)$ be its lifted ray. The point cloud is defined as $\mathcal{P}{:=}\{\mathbf{x}_i\}$ and the ray cloud is defined as $\mathcal{L}{:=}\{\mathbf{l}_i\}$. Since ray centers $\mathcal{C}{:=}\{\mathbf{c}_1, \mathbf{c}_2\}$ are easily found by inspecting the intersection points, we aim to maximize the conditional posterior of $\mathcal{P}$,

$$P(\mathcal{P}|\mathcal{L}, \mathcal{C}) \propto P(\mathcal{L}|\mathcal{P}, \mathcal{C})P(\mathcal{P}|\mathcal{C}). \tag{3}$$

Note the conditional likelihood $P(\mathcal{L}|\mathcal{P}, \mathcal{C})$ is a non-zero constant as long as each ray intersects the corresponding pair of ray center and 3D point and the probability of each ray being assigned to one of the ray centers is equally likely. As long as each point $\mathbf{x}_i$ is estimated to be on its lifted ray $\mathbf{l}_i$, then maximizing (3) amounts to maximizing $P(\mathcal{P}|\mathcal{C})$.

However, since $P(\mathcal{P}|\mathcal{C})$ can depend significantly on the ray centers $\mathcal{C}$, it seems difficult to yield a point statistic similar to $P(\mathcal{P})$ in [7]. We implicitly figure out the difference between $P(\mathcal{P}|\mathcal{C})$ and the original prior $P(\mathcal{P})$ by applying the geometry inversion attack [7], which maximizes $P(\mathcal{P})$, as-is to the ray cloud. As visualized in Fig. 5a, the estimated



(a) Naive neighbor estimation [7] (b) Modified neighbor estimation

Figure 5. Illustration of peak-finding [7] results from (a) naive neighbor-lines estimation [7] and (b) modified neighbor-lines estimation. (a) indicates that the estimated peak will always be located at the center of the current ray. In contrast, our modification only samples the neighborhood of lines using the bundle of rays originating from the opposite center, eliminating the trivial solution.

point for each ray becomes the center of the constituent ray, yielding a trivial recovery. This is because, in the process of the geometric inversion, the K-nearest neighboring lines include a bundle of rays originating from the same center, which yields an erroneous peak at the ray center on the histogram of point candidates, hindering general recovery.

4. Adapted inversion attack for ray clouds

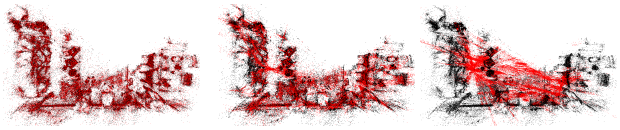
In this section, we illustrate two main reasons for trivial recovery of 3D points in Sec. 3.3, namely i) lower significance of nearest points for estimating 3D points, and ii) bad estimation of true nearest rays lifted from nearest points.

First, the assumption in [7] that the near-80% of candidates which are closest to lines lifted from k-nearest neighboring points yields accurate points does not hold for ray clouds as some of the rays from these k-nearest points will yield estimates at the current ray center. Second, the assumption in [7] that K-nearest lines will overlap enough with the set of rays lifted from the k-nearest points ($K > k$) is weakened as the K-nearest lines will be rays sharing the same center whose original points may be far away from the original point of the current lifted ray.

In Sec. 4, we attempt to address these issues by considering known ray centers in order to approximately maximize $P(\mathcal{P}|\mathcal{C})$ instead of $P(\mathcal{P})$. We also aim to address trivial solutions by incorporating the known positions of the ray centers $\mathcal{C}$ to approximate maximizing over $P(\mathcal{P}|\mathcal{C})$.

4.1. Eliminating the degenerate solution

The trivial recovery of 3D points (estimated 3D points $\mathcal{P}$ always located at the two ray centers $\mathcal{C}$) is due to rays intersecting the same ray center always ending up as the nearest neighboring (NN) lines, producing the peak density of closest points at the ray center (see Fig. 5a). To bypass this issue, we exclude the rays intersecting through the current ray when constructing the set of nearest neighboring lines for estimating the peak density, and only select the NN-lines from the other ray center. This can be viewed as incorporating the known locations of the ray centers for recovering the points (see Fig. 5b) to maximize $P(\mathcal{P}|\mathcal{C})$.



(a) ULC [50] (oracle) (b) Ray cloud (oracle) (c) Ray cloud

Figure 6. 3D point cloud recovered on different settings of nearest neighbors for the *Apt1 living* from [55] (black–original points, red–recovered points). In (b) and (c), we have only utilized neighboring rays connected to the opposite ray center to avoid degenerate solutions (in Sec. 4.1). For the oracle cases of (a) and (b), rays from the true nearest points of the original points are used to estimate the points, showing feasible recovery. In contrast, when the nearest neighboring points are approximated by the nearest neighboring rays, it leads to worse recovery as shown in (c). This implies false positive rays, which are close but whose original points are distant, significantly degrade the recovery of points.

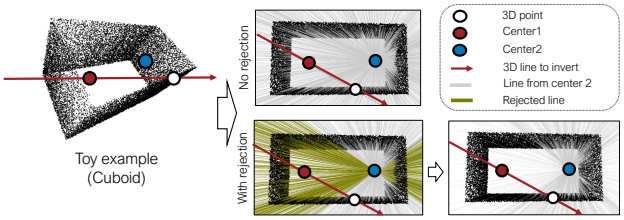
Fig. 6b shows that, in the oracle case when the ground truth nearest points of the current ray's original 3D point are known, discarding NN-lines intersecting the current ray can avoid degenerate solution and yield more feasible point cloud. At the same time, Fig. 6c shows that the actual point recovery from a ray cloud is worse. We conjecture this is due to the misalignment between the set of nearest neighboring rays and the set of nearest neighboring points for a given point and its lifted ray which is addressed in Sec. 4.2.

4.2. Rejecting rays closely aligned to the baseline

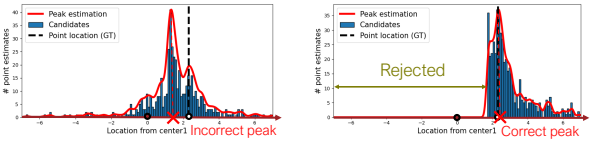
We first analyze the cause of the bad recovery in Fig. 6c even after applying the trick in Sec. 4.1. We define the rays close to the current ray but whose original points are far away from the original point of the current ray as the false positive rays. By comparing the oracle case (b) and the normal case (c) in Fig. 6, we anticipate that the false positive rays are the culprits for degraded estimation of the points.

We conjecture two cases where the false positive rays can lead to misleading estimations. First, when there are many rays from the opposite center that are in the proximity of the current ray center, the peak of the histogram may be located near the current ray center. Vice versa, when the current ray passes through the denser area around the opposite center, the peak will be located near the opposite center. These two cases partially cause the 3D points to shrink (see Fig. 6c). Second, when there are many diverse rays in the proximity of the current ray, multiple peaks can be formed from the point candidates (closest points on the ray to neighboring rays) along the current ray. This can arise when the ray centers are embedded in a room-like environment to form congested rays (see [32] for more discussions).

Out of above cases, we attempt to reduce the shrinking effect of false positive rays by incorporating known $\mathcal{C}$ and not considering any rays from the opposite center which are within the proximity of the current ray center. This can be effectively achieved by excluding a fraction (β) of rays most



(a) Visualization of rejection-sampling of rays near the baseline



(b) Peak-fiding [7] w/o rejection (c) Peak-finding [7] with rejection

Figure 7. Illustration of the effect of line rejection method. For a synthetic room comprising 20k points, in (b), peak estimation fails to estimate the correct point due to false positive rays. In (c), the rejection excludes false candidates to estimate an accurate point.

aligned with the baseline when estimating the histogram of point candidates. Fig. 7 illustrates the effect of this ray rejection which can lead to better point estimation.

We outline some limitations of this approach. First, if the ray lifted from a 3D point is within the region of rejection sampling, then the estimated point can be inaccurate due to the lack of nearby rays to support the estimation process. Second, the method introduces a hyperparameter (β) that needs to be carefully set (see results in Table 2).

5. Voxel-based ray sampling

We now present a new ray sampling technique that can neutralize the modified inversion attack from Sec. 4.

Our approach is motivated from Fig. 6 that the quality of point recovery depends on finding the true point candidates once the degenerate solution is eliminated. This time, we try to discourage the true nearest neighboring points from being included in the set of K-nearest neighboring lines when lifted to rays with aims to lower the inversion accuracy.

For the above purpose, we partition the 3D points into voxels and assign one of two ray centers to each voxel such that i) 3D points within the same voxel are lifted to rays intersecting through the assigned ray center and ii) no neighboring voxels share the same ray center (see Fig. 8). As long as the voxels are large enough, many 3D points except those near the voxel boundaries will have most of their respective k-nearest points to be inside the same voxel and subsequently share the same ray center when lifted to rays. Since the rays lifted from these true k-nearest neighboring points inside the same voxel will yield a trivial solution as illustrated in Sec. 4.1, these will be disallowed from being used for peak estimation. Hence, we essentially minimize the overlap between the set of K-nearest lines and the set of k-nearest points which is a necessary condition for successful recovery of points from lines in [7].

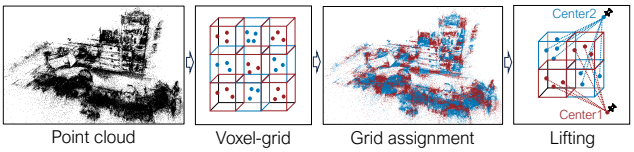


Figure 8. The procedure of voxel-based ray sampling. After partitioning the point cloud into a voxel grid, the whole grid is divided into two clusters in such a way that neighboring grids belong to different clusters. Then, all 3D points in each grid within the same cluster are paired with one center, while the other cluster is paired with the remaining center for lifting.

We adopt a 3D checkerboard pattern for assigning ray centers to voxels with aims to achieve stable camera localization accuracy by uniformly distributing rays from different ray centers across the scene. While a larger voxel size will allow more true nearest neighboring points to be discarded and provide better privacy protection, it will also cause the rays to be more non-uniform and lower the camera localization accuracy (see [32]). Currently, the voxel size needs to be manually determined for each dataset as the optimal trade-off point can vary across different scenes.

6. Experimental Results

We present comprehensive experimental results showing the effectiveness of ray cloud in terms of localization accuracy, inference speed, and privacy-preserving performance.

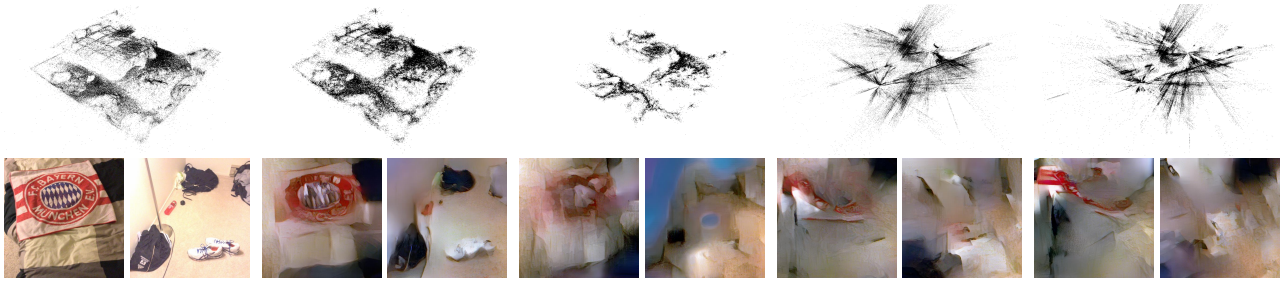
Datasets We used the same two public datasets as in [7, 26]. The Energy Landscape [55] dataset consists of 12 indoor scenes with multiple sequences of scanned RGB images. The Cambridge [21, 22] dataset comprises 6 distinct outdoor landmarks, containing multiple image frames with labeled 6-DOF camera poses for the camera re-localization. As in [26], we excluded the scenes of *Street* and *Great Court* from [22] due to the presence of large geometric outliers. We followed the same experimental protocols from [26, 50] for constructing sparse 3D point clouds using COLMAP [44] and evaluation. Separated queries not included in 3D map construction were used for evaluation.

Implementation details We used Poselib [24] to utilize the $p5$+1R solver and the repository of PPL [26] to run experiments. We report coarse estimation results of inversion in ray cloud and voxel-based ray cloud as the refine estimation [7] diverges due to incorrect initial estimation (see [32]). We used a workstation with an Intel CPU i9-13900K running at 3.0 GHz and NVIDIA RTX 4090 GPUs.

Evaluation metrics For camera localization, we estimated the absolute 6-DOF pose of the query image with given 2D-3D correspondences. We define $\mathtt{R} \in SO(3)$ and $\mathbf{t} \in \mathbb{R}^3$ as the ground truth camera rotation and translation respectively, and $\hat{\mathtt{R}} \in SO(3)$ and $\hat{\mathbf{t}} \in \mathbb{R}^3$ as the estimated rotation and translation respectively. Then, we obtained the estimated pose errors by following [26, 50, 51], where the

Dataset	Scene representation	Minimal solver	Pose error				RANSAC			Privacy preserving			
			$\Delta R\downarrow$		$\Delta \mathbf{t}\downarrow$		i	r	t $\downarrow$	$e_g\uparrow$	PSNR $\downarrow$	SSIM $\downarrow$	MAE $\uparrow$
			mean	median	mean	median	mean	mean	mean	median	mean	mean	mean
Energy Landscape	Point cloud	*p*3P	0.178	0.031	0.007	0.001	11	99.54	4	-	15.76	0.516	33.29
	Uniform line cloud [50]	*p*6L	0.272	0.098	0.010	0.003	124	81.12	90	0.045	13.45	0.403	42.83
	PPL [26]	*p*6L	0.292	0.113	0.011	0.004	123	81.51	88	0.056	10.42	0.355	62.61
	PPL+ [26]	*p*6L	0.293	0.115	0.011	0.004	125	81.30	85	0.056	10.39	0.355	62.70
	Ray cloud (ours)	*p*5+1R	**0.262**	**0.087**	**0.010**	**0.003**	146	80.03	**6**	0.669	10.29	0.319	62.97
	Voxel-based ray cloud (ours)	*p*5+1R	0.297	0.116	0.011	0.004	144	80.15	**6**	**0.770**	**10.05**	**0.315**	**64.84**
Cambridge	Point cloud	*p*3P	0.700	0.101	0.142	0.043	12	98.48	7	-	14.00	0.433	39.40
	Uniform line cloud [50]	*p*6L	0.775	**0.169**	**0.172**	**0.066**	436	71.87	289	0.745	12.71	0.331	46.08
	PPL [26]	*p*6L	0.775	0.183	0.181	0.081	264	74.49	186	0.977	11.21	0.296	55.53
	PPL+ [26]	*p*6L	**0.764**	0.183	0.178	0.081	284	74.11	190	0.948	11.16	0.297	55.91
	Ray cloud (ours)	*p*5+1R	0.855	0.222	0.219	0.107	239	74.89	**12**	9.614	10.41	0.261	60.98
	Voxel-based ray cloud (ours)	*p*5+1R	0.880	0.256	0.235	0.120	250	75.24	**12**	**10.54**	**10.23**	**0.253**	**62.56**

Table 1. Comparison of localization and privacy-preserving performance with different 3D scene representations (ΔR: rotation error [°], $\Delta \mathbf{t}$: translation error [m], **i**: number of iterations, **r**: inlier ratio (%), **t**: runtime [ms], e_g: geometric (3D point) error [m]). We applied modified inversion attack from Sec. 4 with 25% line rejection to ray cloud and voxel-based ray cloud with voxel size of 0.5m and 5.0m for Energy Landscape and Cambridge respectively. **Bold** indicate the best result in each metric among privacy-preserving representations.



(a) Ground truth (b) Uniform line cloud [50] (c) PPL [26] (d) Ray cloud (e) Voxel-based ray cloud

Figure 9. Comparison of revealed scene geometries and images for *Apt2 bed* [55]. Modified inversion attack is applied to (d) and (e).

rotational error is computed as $\Delta \mathtt{R} = \arccos \frac{\mathrm{Tr}(\hat{\mathtt{R}}^\top \mathtt{R})-1}{2}$ and the translation error computed as $\Delta \mathbf{t} = \left\| \hat{\mathtt{R}}^\top \hat{\mathbf{t}} - \mathtt{R}^\top \mathbf{t} \right\|_2$.

To compare the quality of recovered point clouds, we calculated the 3D median point errors between the original and estimated points in meters as in [26]. We also compared the quality of revealed images by feeding recovered point cloud and ground truth camera pose of query images to InvSfM [39] and calculating peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and mean absolute error (MAE) values against the queries.

6.1. Camera localization accuracy and runtime

As shown in Table 1, our ray cloud family of approaches achieved the fastest inference speed among the line-lifting methods with no significant decline in pose accuracy. Notably, these achieved real-time performance on both datasets, reaching 137 FPS for the ray cloud and 136 FPS for the voxel-based ray cloud on average. As further demonstrated in [32], this is mainly due to the introduction of the *p*5+1R solver which is only available for ray clouds.

The ray cloud showed the lowest localization error on [55] but slightly higher errors than ULC and PPL on [22]. We conjecture this is due to the increased baseline between two ray centers for the outdoor scenes. We also note that the voxel-based ray cloud does not exhibit a significant reduction in pose error compared to the standard ray cloud.

6.2. Quality of revealed scene geometry and images

The quantitative comparisons of scene reconstruction quality are also shown in Table 1. For a fair comparison, we applied our adapted inversion attack in Sec. 4 for ray cloud and voxel-based ray cloud, while we used the original geometry inversion attack [7] with 3 refinement steps for uniform line cloud (ULC), PPL and PPL+. We set β=25% for the ray rejection threshold as it shows the best point estimation in Table 2. Results with β=50% are included in [32].

Recovery of 3D point clouds The median point errors for ray cloud and voxel-based ray cloud are notably larger than those of other methods, even after applying our adjusted density-based attacks. Additionally, to validate the impact of our adapted inversion attack, we compared the geometric error with and without rejection sampling, showing that rejection sampling contributes to error reduction in Table 2. The qualitative result in Fig. 9 shows that the results from ray cloud and voxel-based ray cloud are unrecognizable.

Degraded quality of scene details Fig. 9 shows the images recovered from the ray cloud and the voxel-based ray cloud are hardly recognizable. In Table. 1, the voxel-based ray cloud yields the lowest PSNR, SSIM and MAE values with the original ray cloud as a runner-up. This indicates large point errors exhibited by ray clouds translate to lower image quality.

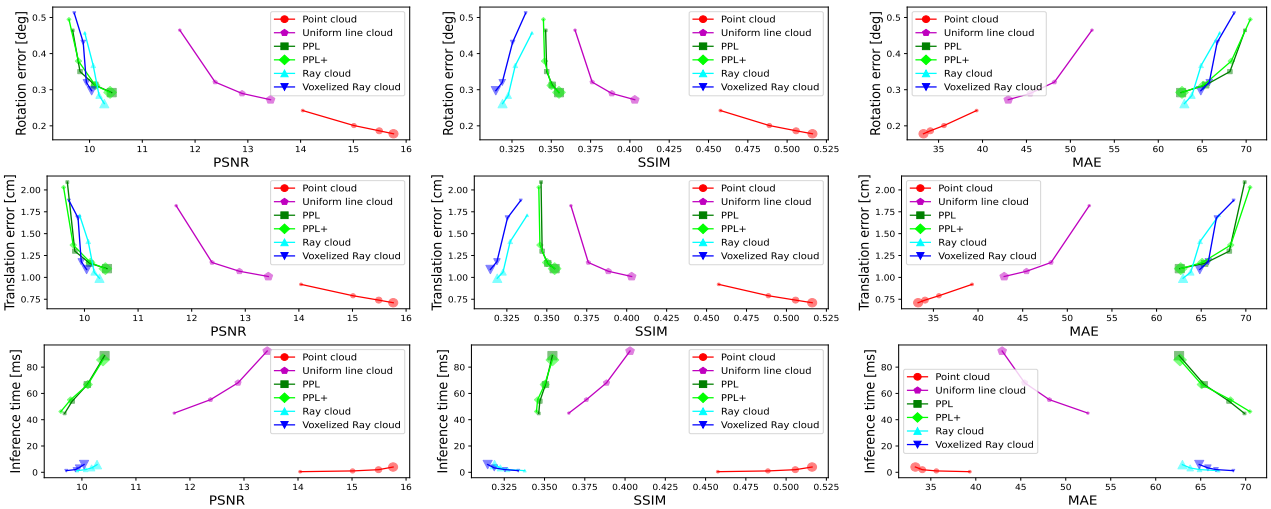


Figure 10. Average localization performance vs. reconstructed image quality across different levels of the ray density on the Energy Landscape [55] dataset. The tested line densities are 100% (biggest marker), 50%, 25% and 10% (smallest marker).

Dataset	Rejection threshold (β)	e_g[m] median	PSNR mean	SSIM mean	MAE mean
Energy Landscape	None	1.027	9.959	0.330	66.06
	10 (%)	0.761	10.24	**0.323**	63.62
	25 (%)	**0.669**	10.29	0.319	62.97
	50 (%)	0.746	**10.32**	0.320	**62.52**
Cambridge	None	11.47	10.10	0.273	64.06
	10 (%)	10.29	10.35	**0.267**	61.60
	25 (%)	**9.614**	**10.41**	0.261	**60.98**
	50 (%)	12.98	9.944	0.242	64.70

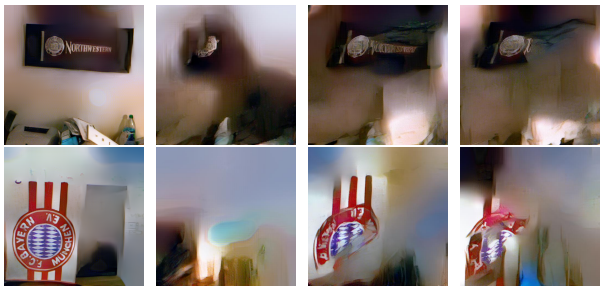
Table 2. Effect of ray rejection (see Sec. 4.2) on the quality of scene recovery. Rejecting rays close to the baseline of two ray centers overall reduces the point estimation errors. **Bold** represents the best inversion results amongst different thresholds.

Additionally, Fig. 10 comparing the pose estimation errors against image reconstruction qualities demonstrate that the ray cloud-based approaches are capable of preserving the privacy especially for dense scenes while significantly decreasing the inference runtime. In contrast, the voxel-based ray cloud shows slightly higher pose estimation errors potentially due to coagulation of rays yielding less uniform distribution of rays across the scene. The fact that the SSIM value increases for lower density ray clouds also needs further investigation (see [32] for qualitative results).

In Fig. 11, we demonstrate some corner cases revealed through our adapted inversion attack in Sec. 4. Fig. 11d shows the voxel-based ray sampling proposed in Sec. 5 can degrade the quality of scene recovery.

7. Conclusion

We presented a privacy-preserving visual localization framework based on a new scene representation called 3D ray cloud, which is robust to the known geometry inversion attack and achieves an order of magnitude faster localization than previous line-based approaches without significantly compromising the localization accuracy. This is enabled by the intrinsic geometry of the ray cloud allowing utilization of efficient widely-available perspective relative pose estimation solver. We have also explored potential improvements for the geometry-inversion attack on ray clouds and proposed a respective defense mechanism.



(a) ground truth (b) w/o rejection (c) w/ rejection (d) voxel-based

Figure 11. Illustration of point-to-image translation [40] results with point clouds from (a) ground truth, (b) recovered from ray cloud without rejection sampling, (c) recovered with rejection sampling, and (d) recovered from voxel-based ray cloud. While (c) shows improved recovery via ray rejection in Sec. 4.2, (d) shows voxel-based ray sampling lowering the reconstruction quality.

Limitation Although this work can successfully defend the known inversion attack [7], we note it may be potentially vulnerable to a new type of attack in which omnidirectional rays are projected to a cylindrical or spherical image plane which is then unfolded and directly passed through an inversion network for image reconstruction. While realizing this attack may face its own challenges such as unknown depths and mixed cheirality, we believe utilizing more ray centers will provide a defense against such attack by reducing the density of ray projections in each ray center.

Acknowledgement This work was supported by the NRF (Natioanl Research Foundation of Korea) grants funded by the Korea government (MSIT) (No. 2022R1C1C1004907). We thank anonymous reviewers for their valuable feedback.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building Rome in a day. *Communications of the ACM*, 54(10):105–112, 2011. 1

[2] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 510–517, 2012. 2

[3] Clemens Arth, Daniel Wagner, Manfred Klopschitz, Arnold Irschara, and Dieter Schmalstieg. Wide area localization on mobile phones. In *2009 8th IEEE International Symposium on Mixed and Augmented Reality*, pages 73–82. IEEE, 2009. 1

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

[5] Eric Brachmann and Carsten Rother. Learning less is more - 6D camera localization via 3D surface regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4654–4662, 2018. 1

[6] Robert Castle, Georg Klein, and David W Murray. Video-rate localization in multiple maps for wearable augmented reality. In *2008 12th IEEE International Symposium on Wearable Computers*, pages 15–22. IEEE, 2008. 1

[7] 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, 7, 8

[8] Kunal Chelani, Torsten Sattler, Fredrik Kahl, and Zuzana Kukelova. Privacy-preserving representations are not enough: Recovering scene content from camera poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13132–13141, 2023. 1

[9] Ondrej Chum, Jirí Matas, and Josef Kittler. Locally optimized ransac. In *Pattern Recognition: 25th DAGM Symposium, Magdeburg, Germany, September 10-12, 2003. Proceedings 25*, pages 236–243. Springer, 2003. 1

[10] 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 *Proceeding of the British Machine Vision Conference (BMVC)*, 2021. 1, 2

[11] 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

[12] Yaqing Ding, Jian Yang, Viktor Larsson, Carl Olsson, and Kalle Åström. Revisiting the p3p problem. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4872–4880, 2023. 1

[13] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4829–4837, 2016. 1, 2

[14] Mihai Dusmanu, Johannes L. Schönberger, Sudipta N. Sinha, and Marc Pollefeys. Privacy-preserving image features via adversarial affine subspace embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14262–14272, 2021. 2

[15] 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, 3

[16] Marcel Geppert, Viktor Larsson, Johannes L Schönberger, and Marc Pollefeys. Privacy preserving partial localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17337–17347, 2022. 2

[17] Richard Hartley. Cheirality. *International journal of computer vision*, 26(1):41–61, 1998. 4

[18] Lionel Heng, Benjamin Choi, Zhaopeng Cui, Marcel Geppert, Sixing Hu, Benson Kuan, Peidong Liu, Rang Nguyen, Ye Chuan Yeo, Andreas Geiger, et al. Project autovision: Localization and 3d scene perception for an autonomous vehicle with a multi-camera system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4695–4702, 2019. 1

[19] 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

[20] Tong Ke and Stergios I Roumeliotis. An efficient algebraic solution to the perspective-three-point problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7225–7233, 2017. 1

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

[22] 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, 6, 7

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

[24] Viktor Larsson. PoseLib - Minimal Solvers for Camera Pose Estimation. https://github.com/vlarsson/PoseLib, 2020. Accessed: 2022-10-30. 6

[25] Viktor Larsson, Kalle Astrom, and Magnus Oskarsson. Efficient solvers for minimal problems by syzygy-based reduction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 820–829, 2017. 3

[26] 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. 1, 2, 3, 4,

6, 7

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

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

[29] 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

[30] 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. 1, 2

[31] 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. 4

[32] Heejoon Moon, Chunghwan Lee, and Je Hyeong Hong. Supplementary document for efficient privacy-preserving visual localization using 3d ray clouds, 2024. 3, 5, 6, 7, 8

[33] 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

[34] 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. 2

[35] 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. 2, 3, 4

[36] Linfei Pan, Johannes Lutz Schönberger, Viktor Larsson, and Marc Pollefeys. Privacy Preserving Localization via Coordinate Permutations. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023. 2, 3, 4

[37] 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. 1

[38] Maxime Pietrantoni, Martin Humenberger, Torsten Sattler, and Gabriela Csurka. Segloc: Learning segmentation-based representations for privacy-preserving visual localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15380–15391, 2023. 2

[39] Francesco Pittaluga, Sanjeev Koppal, and Ayan Chakrabarti. Learning privacy preserving encodings through adversarial training. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 791–799. IEEE, 2019. 1, 7

[40] 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, 3, 8

[41] Francesco Pittaluga and Bingbing Zhuang. Ldp-feat: Image features with local differential privacy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17580–17590, 2023. 2

[42] 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

[43] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241. Springer, 2015. 1, 2

[44] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 6

[45] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 501–518. Springer, 2016. 1

[46] Mikiya Shibuya, Shinya Sumikura, and Ken Sakurada. Privacy preserving visual SLAM. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 102–118. Springer, 2020. 1, 2, 3

[47] Sudipta Narayan Sinha, Pablo Alejandro Speciale, Sing Bing Kang, and Marc Andre Leon Pollefeys. Device pose estimation using 3d line clouds, Mar. 30 2021. US Patent 10,964,053. 3

[48] 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

[49] Zhenbo Song, Wayne Chen, Dylan Campbell, and Hongdong Li. Deep novel view synthesis from colored 3d point clouds. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*, pages 1–17. Springer, 2020. 2

[50] 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, 7

[51] 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 (ICCV)*, pages 1486–1496, 2019. 6

[52] Henrik Stewénius, Magnus Oskarsson, Kalle Aström, and David Nistér. Solutions to minimal generalized relative pose problems, 2005. 3, 4

[53] 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. 2

[54] Philip HS Torr and Andrew Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer vision and image understanding*, 78(1):138–156, 2000. 4

[55] 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, 3, 5, 6, 7, 8

[56] Jonathan Ventura, Clemens Arth, and Vincent Lepetit. An efficient minimal solution for multi-camera motion. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 747–755, 2015. 3

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

[58] Philippe Weinzaepfel, Hervé Jégou, and Patrick Pérez. Reconstructing an image from its local descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 337–344, 2011. 1, 2

[59] 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. 2, 4