Supplementary Material for "Robust Incremental Structure-from-Motion with Hybrid Features"

Shaohui Liu[1⋆], Yidan Gao[1⋆], Tianyi Zhang[1⋆], Rémi Pautrat[2], Johannes L. Schönberger[2], Viktor Larsson[3], and Marc Pollefeys[1,2]

[1]ETH Zurich [2]Microsoft Mixed Reality & AI Lab Zurich [3]Lund University

This document provides a list of supplementary materials that accompany the main paper. The content of this supplementary material is organized as follows:

- In Section A, we provide details on the construction and maintenance of vanishing point tracks during the reconstruction process.
- In Section B, we present detailed derivations for uncertainty propagation from the 2D observations to the 3D points and lines bundle adjusted across views. In particular, we show how to use sensitivity analysis to derive the uncertainty of the optimized 3D line in its Plücker form. We further give details on how to use the propagated uncertainty in the geometric pipeline.
- In Section C, we present details on integration of point-line and VP-line associations in the hybrid bundle adjustment. We additionally discuss on the challenges on efficiency and some practical solutions on implementation.
- In Section D, we discuss how to use auxiliary vanishing point correspondences to help improve absolute pose estimation (localization/registration) by providing more combinations of minimal configurations.
- In Section E, we provide more details on implementation, datasets and experimental setup.
- Finally, in Section F, we provide some additional results to support the content of the main paper.

A Maintenance of Vanishing Point Tracks

As mentioned in the main paper, when a new image is registered, we not only triangulate and update the 3D points and lines, but also construct vanishing point (VP) tracks to model the parallelism relations among lines. In practice we find that the VP matching is a relatively easy task such that computing from the consensus of line matches generally gives very reasonable results, thus making VP a good resource to help improve the structures of the 3D line maps. A 2D VP feature is represented with a 3-dimensional homogeneous coordinate, which equals the 3D direction in the local frame left multiplied by the intrinsic matrix. Thus, the cross-view consistency check for VP only involves checking the angle between two 3D directions.

Similar to points and lines, the incremental update of the VP tracks involve all the required operations as follows:

⋆ Equal contribution

- **Continue**: *extend an existing VP track.* Given a 2D VP feature, we first test if there exists a matched VP (in the previously registered images) that is already triangulated in the map and test if the reprojected 3D direction is within 3 degrees compared to the local one from the 2D VP. If so, we add the VP into the corresponding track.
- **Create**: *triangulate a new VP.* If we are unable to assign a VP to any existing track, we try to create a new 3D VP track. Since a 3D VP is a special point feature that only encodes rotation information, it has only 2 degrees of freedom in total, making the two-view triangulation problem even more overconstrained. In fact, the 3D VP direction can be computed from only one view, and the multi-view triangulation of VPs can be easily achived by taking the average over all the computed directions. Thus, we perform a simple RANSAC loop by iterating over all directions, taking the best one with the most agreement, and computing the average of all the agreed VPs to get the final 3D direction of the newly created VP track.
- **Merge**: *merge two VP tracks into a single one.* We attempt to merge VP tracks after the triangulation of each newly registered image. Though the VP merging can be done solely on the sphere due to its limited DoFs, in our system we require shared visibility for the VP track merging to avoid wrongly chained associations from noisy tracks. Specifically, we test only on pairs of VP tracks that share at least three matches among their supports, and check if their direction is within 3 degrees. If so we merge the two together and recompute the 3D VP using all the supports by taking the averaged direction.
- **Complete**: *recollect supports.* We test reprojection agreement on the neighbors of the included supports in the matching graph to collect potentially missing supports. This is similar to the practices for points and lines as discussed in the main paper.

By iteratively performing those steps we can maintain 3D VP tracks of reasonably good quality, which not only enables richer 3D maps with structural information but also helps on both the refinement and registration.

At the step of hybrid refinement, the presence of VP can help regularizing the line maps by enforcing structural constraints. We perform the active supports caching and two-step refinement on the VP tracks as discussed in the main paper. Specifically, we only use the VP tracks with at least three active supports at pose optimization, and perform fixed-pose VP optimization and active label update afterwards. The fixed-pose VP optimization only involves a straightforward multi-view triangulation process. As previously discussed, we perform a one-point LO-RANSAC [6] by iterating over all the supports, selecting the best one with the most agreement, and taking the average over all the inliers.

B Full Derivations on Uncertainty Propagation

In this section, we provide detailed derivations on propagating uncertainty from 2D observations to the optimized 3D line. In the following parts, we first provide

some backgrounds on covariance propagation and the representation of 3D line. Then, we present the proposed formulation with the assistance of sensitivity analysis and provide details on the validity tests. Finally, we discuss how to use the acquired 3D uncertainty at refinement and registration (localization).

B.1 Background

Covariance Propagation For a standard non-linear least squares problem with the optimized variable $\boldsymbol{x}$ and the observations $\boldsymbol{y}$:

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \|f(\boldsymbol{x}) - \boldsymbol{y}\|^2, \tag{1}$$

we can directly propagate the uncertainty from the observation $\boldsymbol{y}$ to the optimal solution $\boldsymbol{x}^*$ upon the assumption that the noise of the observations $\boldsymbol{y}$ follows the distribution $\boldsymbol{y} = f(\boldsymbol{x}) + N(\boldsymbol{0}, \boldsymbol{1})$. This can be achieved as:

$$\boldsymbol{\Sigma}_{\boldsymbol{x}^*} = (J_f^T(\boldsymbol{x}^*) J_f(\boldsymbol{x}^*))^{-1}, \tag{2}$$

where J_f is the Jacobian of the function $f(\cdot)$. This corresponds to using the approximate Hessian as in [22].

For a non-uniform distribution $\boldsymbol{y} = f(\boldsymbol{x}) + N(\boldsymbol{0}, \boldsymbol{S})$, the clean formulation is provided in [1] when the residuals in the least squares problem is rescaled with $\boldsymbol{S}^{-1/2}$, which is generally the practice in factor graphs [7]. With the original formulation, one can also use the first-order approximation [25] for propagating covariance from the observation noise in the linear form:

$$\boldsymbol{\Sigma}_{\boldsymbol{x}^*} = \frac{\partial \boldsymbol{x}^*}{\partial \boldsymbol{y}} \boldsymbol{\Sigma}_y \frac{\partial \boldsymbol{x}^*}{\partial \boldsymbol{y}}^T = \frac{\partial \boldsymbol{x}^*}{\partial \boldsymbol{y}} \boldsymbol{S} \frac{\partial \boldsymbol{x}^*}{\partial \boldsymbol{y}}^T \tag{3}$$

$$\frac{\partial \boldsymbol{x}^*}{\partial \boldsymbol{y}} = J_f^\dagger(\boldsymbol{x}^*) = (J_f^T(\boldsymbol{x}^*) J_f(\boldsymbol{x}^*))^{-1} J_f^T(\boldsymbol{x}^*) \tag{4}$$

where $J_f^\dagger$ corresponds to the Moore-Penrose Inverse of J_f. In our case, the discussion above applies to the 3D point optimized with the reprojection error across multiple views, where the function f corresponds to the perspective reprojection and J_f is its Jacobian.

Note that this only applies to the case where the error uncertainty can be directly propagated from the observation uncertainty. In other words, the partial derivative of the residual over the observation is not dependent on the optimal x^*, which can be formulated in the context of non-linear least squares regression. However, the multi-view line optimization with the endpoint-to-line distance as residuals does not fall into this category, as discussed in Sec. B.2.

Representation of an Infinite 3D Line While in the final reconstruction a 3D line is represented with its two endpoints, its optimization across multiple views is generally operated on its infinite form due to inconsistent endpoint observations in 2D. This optimization is followed by endpoint unprojection to decide the spatial extent of the 3D line segment. Note that the infinite form is more crucial as it is used in both bundle adjustment and localization, while the endpoints are mainly for correct track merging, robust point-to-line association and final map visualization. Thus, to be able to correctly propagate the 3D uncertainty for a optimized 3D line across views, we need to first study the representation of a 3D infinite line in the optimization.

An infinite line has 4 degrees of freedom (DOF) and is generally represented in its Plücker coordinates [10]. In the optimization [14], the orthonormal representation [4] of the Plücker coordinates is generally used to minimally constrain the 3D infinite line.

Plücker Coordinates. A 3D line in Plücker coordinates can be represented by two vectors, namely $\boldsymbol{L} = \begin{bmatrix}\boldsymbol{d}\, \boldsymbol{m}\end{bmatrix}$, where $\boldsymbol{d}$ is the direction vector of the line and $\boldsymbol{m}$ is the vector normal to the plane containing the origin and the line. Given a 3D line segment with its two endpoints $(\boldsymbol{x}_s, \boldsymbol{x}_e)$, the Plücker coordinates of its corresponding infinite line is

$$\tilde{\boldsymbol{L}} = \begin{bmatrix}\tilde{\boldsymbol{x}}_s - \tilde{\boldsymbol{x}}_e \\ \tilde{\boldsymbol{x}}_s \times \tilde{\boldsymbol{x}}_e\end{bmatrix} = \begin{bmatrix}\tilde{\boldsymbol{d}} \\ \tilde{\boldsymbol{m}}\end{bmatrix} \tag{5}$$

where $\tilde{\boldsymbol{x}}$ denotes the homogeneous coordinates of the endpoints and $\tilde{\boldsymbol{d}}$, $\tilde{\boldsymbol{m}}$ refer to the the unnormalized vectors. Note that Plücker coordinates are a homogeneous representation, so all pairs $\begin{bmatrix}\alpha\tilde{\boldsymbol{d}}\, \alpha\tilde{\boldsymbol{m}}\end{bmatrix}$ $(\alpha \neq 0)$ represent the same infinite line.

We use here $\tilde{\boldsymbol{L}} = \begin{bmatrix}\tilde{\boldsymbol{d}}\, \tilde{\boldsymbol{m}}\end{bmatrix}$ to denote the unnormalized Plücker coordinates and $\boldsymbol{L} = \begin{bmatrix}\boldsymbol{d}\, \boldsymbol{m}\end{bmatrix}$ to denote normalized Plücker coordinates (displacement $\|\boldsymbol{d}\| = 1$), so that we have

$$\boldsymbol{L} = \begin{bmatrix}\boldsymbol{d} \\ \boldsymbol{m}\end{bmatrix} = \frac{1}{\|\tilde{\boldsymbol{d}}\|}\begin{bmatrix}\tilde{\boldsymbol{d}} \\ \tilde{\boldsymbol{m}}\end{bmatrix} \tag{6}$$

In this way, the Plücker coordinates establish a one-to-one correspondence between the 4 DoF infinite lines and points.

Orthonormal Representation. Since the Plücker coordinates are over-parameterized, we follow [14] to use their orthonormal representation during optimization, which was initially introduced in [4]. Specifically, the 4-DOF minimal representation of the Plücker coordinates is formulated as $\boldsymbol{\Phi} = [\boldsymbol{\theta}, \rho] \in \mathbb{R}^4$, which can be computed by QR decomposition.

$$\begin{aligned}\boldsymbol{L} &= \begin{bmatrix}\boldsymbol{d} | \boldsymbol{m}\end{bmatrix} \\ &= \begin{bmatrix}\frac{\boldsymbol{d}}{||\boldsymbol{d}||} & \frac{\boldsymbol{m}}{||\boldsymbol{m}||} & \frac{\boldsymbol{d}\times\boldsymbol{m}}{||\boldsymbol{d}\times\boldsymbol{m}||}\end{bmatrix}\begin{bmatrix}||\boldsymbol{d}|| & 0 \\ 0 & ||\boldsymbol{m}|| \\ 0 & 0\end{bmatrix} \\ &\propto \boldsymbol{U}\begin{bmatrix}w_1 & 0 \\ 0 & w_2 \\ 0 & 0\end{bmatrix}\end{aligned} \tag{7}$$

Then, the orthonormal representation of a line can be formulated as

$$\boldsymbol{L}(\boldsymbol{\Phi}) = (\boldsymbol{U}(\boldsymbol{\theta}), \boldsymbol{W}(\rho)) \in SO(3) \times SO(2), \tag{8}$$

where

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{u}_1 \ \boldsymbol{u}_2 \ \boldsymbol{u}_3 \end{bmatrix}, \boldsymbol{W} = \begin{bmatrix} w_1 & -w_2 \\ w_2 & w_1 \end{bmatrix} = \begin{bmatrix} \cos\rho & -\sin\rho \\ \sin\rho & \cos\rho \end{bmatrix} \tag{9}$$

So the Plücker coordinates can be represented as

$$\begin{bmatrix} \tilde{\boldsymbol{d}} \\ \tilde{\boldsymbol{m}} \end{bmatrix} = \begin{bmatrix} w_1 \boldsymbol{u}_1 \\ w_2 \boldsymbol{u}_2 \end{bmatrix} = \begin{bmatrix} \cos(\rho)\boldsymbol{u}_1 \\ \sin(\rho)\boldsymbol{u}_2 \end{bmatrix} \tag{10}$$

Note that here we use axis-angle representation at uncertainty derivation for $\boldsymbol{U}(\boldsymbol{\theta})$, such that $\boldsymbol{\theta} = \theta\boldsymbol{\eta}$, where θ is the rotation angle and $\boldsymbol{\eta}$ is the rotation axis:

$$\theta = \|\boldsymbol{\theta}\| = \arccos(\frac{\mathrm{tr}(\boldsymbol{U}) - 1}{2}) \tag{11}$$

$$\boldsymbol{\eta} = \frac{\boldsymbol{\theta}}{\|\boldsymbol{\theta}\|} = \frac{1}{2\sin\|\boldsymbol{\theta}\|} \begin{bmatrix} \boldsymbol{U}_{32} - \boldsymbol{U}_{23} \\ \boldsymbol{U}_{13} - \boldsymbol{U}_{31} \\ \boldsymbol{U}_{21} - \boldsymbol{U}_{12} \end{bmatrix} \tag{12}$$

Projection Matrix for Lines. For lines in their Plücker form, the perspective projection can be done by either constructing the Plücker matrix [10, 14] or constructing the projection matrix for lines [10]. We use the latter one for the simplicity at derivation which is formulated as:

$$\tilde{\boldsymbol{l}} = \boldsymbol{P}_l \boldsymbol{L} = \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix}, \tag{13}$$

where $\tilde{\boldsymbol{l}}$ are the unnormalized coordinates of the back-projected 2D line and $\boldsymbol{P}_l$ is the line projection matrix. The construction of $\boldsymbol{P}_l$ can be done as $\boldsymbol{P}_l = \begin{bmatrix} \boldsymbol{K}_l \ \boldsymbol{0} \end{bmatrix} \boldsymbol{H}$, where:

$$\boldsymbol{H} = \begin{bmatrix} [\boldsymbol{t}]_\times \boldsymbol{R} \ \boldsymbol{R} \\ \boldsymbol{R} \quad \boldsymbol{0} \end{bmatrix}, \boldsymbol{K}_l = \begin{bmatrix} f_v & 0 & 0 \\ 0 & f_u & 0 \\ -f_v c_u & -f_u c_v & f_u f_v \end{bmatrix}. \tag{14}$$

Here f_u, f_v denote the focal lengths, (c_u, c_v) denotes the location of the principal point, and $(\boldsymbol{R}, \boldsymbol{t})$ denote the extrinsic parameters.

B.2 Covariance Propagation for the Optimal 3D Line

Residuals: Endpoint-to-Line Distance For the refinement of a 3D infinite line across multiple views, we optimize over its minimal parameters $\boldsymbol{\Phi} = [\boldsymbol{\theta}, \rho] \in \mathbb{R}^4$ with respect to the line reprojection error, which is generally formulated as the perpendicular distance [14] from the two endpoints of the 2D line observation.

Denoting the two endpoint as $\boldsymbol{x}_s$ and $\boldsymbol{x}_e$ and the perpendicular distance function as $D(\cdot)$, the line reprojection error can be formulated as:

$$\boldsymbol{r} = \begin{bmatrix} r_s \\ r_e \end{bmatrix} = \begin{bmatrix} D(\boldsymbol{x}_s, \boldsymbol{l}(\boldsymbol{\Phi})) \\ D(\boldsymbol{x}_e, \boldsymbol{l}(\boldsymbol{\Phi})) \end{bmatrix} = \begin{bmatrix} \frac{\tilde{\boldsymbol{x}}_s^T \boldsymbol{l}}{\sqrt{l_1^2 + l_2^2}} \\ \frac{\tilde{\boldsymbol{x}}_e^T \boldsymbol{l}}{\sqrt{l_1^2 + l_2^2}} \end{bmatrix}, \tag{15}$$

where $\boldsymbol{l}(\boldsymbol{\Phi}) = \mathit{\Pi}(\boldsymbol{L}(\boldsymbol{\Phi}))$ is the reprojected line on the image with perspective projection $\mathit{\Pi}$. Then, the optimization problem across N_k different views can be written into:

$$\boldsymbol{\Phi}^* = \arg\min_{\boldsymbol{\Phi}} E = \arg\min_{\boldsymbol{\Phi}} \frac{1}{2} \sum_{k}^{N_k} \sum_{j}^{s,e} [D(\boldsymbol{x}_j^k, \mathit{\Pi}_k(\boldsymbol{L}(\boldsymbol{\Phi})]^2)). \tag{16}$$

Here E is the optimization objective and $\mathit{\Pi}_k$ is the line reprojection function for the kth view. Different from points, the residuals described in Eq. (15) cannot be written in the standard from of non-linear least squares problem as in Eq. (1). In particular, taking the derivative of the endpoint-to-line perpendicular distance $D(\cdot)$ we have:

$$\frac{\partial D(\boldsymbol{x}, \boldsymbol{l})}{\partial \boldsymbol{x}} = \frac{\boldsymbol{l}}{\sqrt{l_1^2 + l_2^2}} \frac{\partial \tilde{\boldsymbol{x}}}{\partial \boldsymbol{x}} \tag{17}$$

$$\frac{\partial D(\boldsymbol{x}, \boldsymbol{l})}{\partial \boldsymbol{l}} = \frac{1}{\sqrt{l_1^2 + l_2^2}} (\tilde{\boldsymbol{x}}^T - \tilde{\boldsymbol{x}}^T \boldsymbol{l} \begin{bmatrix} \frac{l_1}{l_1^2 + l_2^2} & \frac{l_2}{l_1^2 + l_2^2} & 0 \end{bmatrix}) \tag{18}$$

Note that in Eq. (17), the derivative of the error over the measured endpoint $\boldsymbol{x}$ depends on the reprojection $\boldsymbol{l}$, which is dependent on the optimal parameters $\boldsymbol{\Phi}^*$. This is a main difference from the point case and prevents the optimization problem to be written in the standard form as in Eq. (1). Thus, the uncertainty of the optimal 3D line is intractable with the Jacobian-based propagation in the least squares form with first-order approximation as in Eq. (4).

Second-order Sensitivity Analysis We propose to perform uncertainty propagation for the optimized 3D line using second-order sensivity analysis [9]. As mentioned in the main paper, this relies on the fact that:

$$\frac{\partial E}{\partial \boldsymbol{\Phi}} |_{\boldsymbol{\Phi} = \boldsymbol{\Phi}^*} = \boldsymbol{0} \tag{19}$$

Note that here $\boldsymbol{\Phi}^*$ is an implicit function of the input $\boldsymbol{x}_j^k$ ($j \in \{s, e\}$, $k = 0, 1, ..., N_k$). Thus, we have the following property:

$$\frac{\partial^2 E}{\partial \boldsymbol{\Phi} \partial \boldsymbol{x}_j^k} |_{\boldsymbol{\Phi} = \boldsymbol{\Phi}^*} = \boldsymbol{0}. \tag{20}$$

This can be used to derive the target Jacobian $\partial \boldsymbol{\Phi}^* / \partial \boldsymbol{x}_j^k$, which can enable uncertainty propagation with Eq. (3). Specifically, relating Eq. (16) the left-hand side of Eq. (20) becomes:

$$\begin{aligned}
\frac{1}{2}\frac{\partial^2[D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))]^2}{\partial\boldsymbol{\Phi}\partial\boldsymbol{x}_j^k} =& \frac{\partial(D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi})))}{\partial\boldsymbol{\Phi}}}{\partial\boldsymbol{x}_j^k} \\
=& \frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi})))}{\partial\boldsymbol{\Phi}}\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{x}_j^k} + D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))\frac{\partial^2 D(\boldsymbol{x}_j^k,\mathit{\Pi}_k(\boldsymbol{L}(\boldsymbol{\Phi}))}{\partial\boldsymbol{\Phi}\partial\boldsymbol{x}_j^k} \\
=& \frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi})))}{\partial\boldsymbol{\Phi}}\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{x}_j^k} + D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))\Big(\frac{\partial(\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{l}(\boldsymbol{\Phi})}\frac{\partial\boldsymbol{l}(\boldsymbol{\Phi})}{\partial\boldsymbol{\Phi}})}{\partial\boldsymbol{x}_j^k}\Big) \\
=& \frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{l}(\boldsymbol{\Phi})}\frac{\partial\boldsymbol{l}(\boldsymbol{\Phi})}{\partial\boldsymbol{\Phi}}\Big(\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{x}}\frac{\partial\boldsymbol{x}}{\partial\boldsymbol{x}_j^k} + \frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{l}(\boldsymbol{\Phi})}\frac{\partial\boldsymbol{l}(\boldsymbol{\Phi})}{\partial\boldsymbol{\Phi}}\frac{\partial\boldsymbol{\Phi}}{\partial\boldsymbol{x}_j^k}\Big) + \\
& D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))\big(\frac{\partial\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{l}(\boldsymbol{\Phi})}}{\partial\boldsymbol{x}}\frac{\partial\boldsymbol{x}}{\partial\boldsymbol{x}_j^k} + \frac{\partial\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{l}(\boldsymbol{\Phi})}}{\partial\boldsymbol{l}(\boldsymbol{\Phi})}\frac{\partial\boldsymbol{l}(\boldsymbol{\Phi})}{\partial\boldsymbol{\Phi}}\frac{\partial\boldsymbol{\Phi}}{\partial\boldsymbol{x}_j^k}\big)\frac{\partial\boldsymbol{l}(\boldsymbol{\Phi})}{\partial\boldsymbol{\Phi}} + \\
& D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{l}(\boldsymbol{\Phi})}\frac{\partial^2\boldsymbol{l}(\boldsymbol{\Phi})}{\partial\boldsymbol{\Phi}^2}\frac{\partial\boldsymbol{\Phi}}{\partial x_j^k}
\end{aligned} \tag{25}$$

$$\frac{1}{2}\sum_k^{N_k}\sum_j^{s,e}\frac{\partial^2[D(\boldsymbol{x}_j^k,\mathit{\Pi}_k(\boldsymbol{L}(\boldsymbol{\Phi})]^2))}{\partial\boldsymbol{\Phi}\partial\boldsymbol{x}_j^k}|_{\boldsymbol{\Phi}=\boldsymbol{\Phi}^*} = \boldsymbol{0} \tag{21}$$

With the denotation $\boldsymbol{l}_k(\boldsymbol{\Phi}) = \mathit{\Pi}_k(\boldsymbol{L}(\boldsymbol{\Phi}))$ we have:

$$\frac{1}{2}\frac{\partial^2[D(\boldsymbol{x}_j^k,\mathit{\Pi}_k(\boldsymbol{L}(\boldsymbol{\Phi}))]^2}{\partial\boldsymbol{\Phi}\partial\boldsymbol{x}_j^k} = \frac{1}{2}\frac{\partial^2[D(\boldsymbol{x}_j^k,\boldsymbol{l}_k(\boldsymbol{\Phi}))]^2}{\partial\boldsymbol{\Phi}\partial\boldsymbol{x}_j^k} \tag{22}$$

Since we have the following property for the first-order derivatives:

$$\begin{aligned}
\frac{\partial[D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))]^2}{\partial\boldsymbol{\Phi}} &= 2D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{\Phi}} \\
&= 2D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{l}(\boldsymbol{\Phi})}\frac{\partial\boldsymbol{l}(\boldsymbol{\Phi})}{\partial\boldsymbol{\Phi}}
\end{aligned} \tag{23}$$

$$\begin{aligned}
\frac{\partial[D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))]^2}{\partial\boldsymbol{x}_j^k} &= 2D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{x}_j^k} \\
&= 2D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))\Big(\frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{x}}\frac{\partial\boldsymbol{x}}{\partial\boldsymbol{x}_j^k} + \\
&\quad \frac{\partial D(\boldsymbol{x},\boldsymbol{l}(\boldsymbol{\Phi}))}{\partial\boldsymbol{l}(\boldsymbol{\Phi})}\frac{\partial\boldsymbol{l}(\boldsymbol{\Phi})}{\partial\boldsymbol{\Phi}}\frac{\partial\boldsymbol{\Phi}}{\partial\boldsymbol{x}_j^k}\Big)
\end{aligned} \tag{24}$$

Combining Eqs. (23) and (24) we can get the full expansion for each term of Eq. (16) in Eq. (25).

By reorganizing Eq. (25) and measuring at $\boldsymbol{\Phi}^*$ we can get a linear equation with respect to the target Jacobian $\partial\boldsymbol{\Phi}^*/\partial\boldsymbol{x}_j^k$. Since most terms can be computed

in a straightforward manner, we here only provide detailed solutions for computing the first-order derivatives $\partial \boldsymbol{l}(\boldsymbol{\Phi})/\partial \boldsymbol{\Phi}$ and second-order derivatives $\partial^2 \boldsymbol{l}(\boldsymbol{\Phi})/\partial \boldsymbol{\Phi}^2$ of the reprojected line with respect to the minimal parameters $\boldsymbol{\Phi}$.

Computation of First-order Derivatives $\partial \boldsymbol{l}(\boldsymbol{\Phi})/\partial \boldsymbol{\Phi}$. According to the chain rule, we have

$$\frac{\partial \boldsymbol{l}(\boldsymbol{\Phi})}{\partial \boldsymbol{\Phi}} = \frac{\partial \boldsymbol{l}(\boldsymbol{\Phi})}{\partial \boldsymbol{L}(\boldsymbol{\Phi})} \frac{\partial \boldsymbol{L}(\boldsymbol{\Phi})}{\partial \tilde{\boldsymbol{L}}(\boldsymbol{\Phi})} \frac{\partial \tilde{\boldsymbol{L}}(\boldsymbol{\Phi})}{\partial \boldsymbol{\Phi}}, \tag{26}$$

where the Jacobian from reprojected line to the 3D line in its Plücker coordinates is:

$$\frac{\partial \boldsymbol{l}(\boldsymbol{\Phi})}{\partial \boldsymbol{L}(\boldsymbol{\Phi})} = \boldsymbol{P}_l = \begin{bmatrix} \boldsymbol{K}_l & \boldsymbol{0} \end{bmatrix} \boldsymbol{H} \tag{27}$$

The Jacobian from normalized Plücker coordinates to unnormalied Plücker coordinates is:

$$\frac{\partial \boldsymbol{L}(\boldsymbol{\Phi})}{\partial \tilde{\boldsymbol{L}}(\boldsymbol{\Phi})} = \frac{1}{\|\tilde{\boldsymbol{d}}\|} (\boldsymbol{I}_{6\times 6} - \frac{1}{\|\tilde{\boldsymbol{d}}\|^2} \tilde{\boldsymbol{L}} \begin{bmatrix} \tilde{\boldsymbol{d}} & \boldsymbol{0} \end{bmatrix}) \tag{28}$$

In the following part, we derive $\partial \tilde{\boldsymbol{L}}/\partial \boldsymbol{\Phi}$. Although the derivation of this term has already been studied in previous literature [4,32], we will provide the solutions for completeness.

As stated in Eq. (10), the Plücker coordinates of an infinite line is

$$\tilde{\boldsymbol{L}}(\boldsymbol{\Phi}) = \tilde{\boldsymbol{L}}(\boldsymbol{\theta}, \rho) = \begin{bmatrix} w_1 \boldsymbol{u}_1 \\ w_2 \boldsymbol{u}_2 \end{bmatrix} = \begin{bmatrix} \cos(\rho) \boldsymbol{u}_1 \\ \sin(\rho) \boldsymbol{u}_2 \end{bmatrix} \tag{29}$$

Since we have $\boldsymbol{U} \in SO(3)$, we can compute its derivative in the Lie algebra, which is the space of skew-symmetric matrices

$$\mathfrak{so}(3) = \{\boldsymbol{\theta}^\wedge \in \mathbb{R}^{3\times 3} | \boldsymbol{\theta} \in \mathbb{R}\} \quad (\boldsymbol{\theta}^\wedge = [\boldsymbol{\theta}]_\times) \tag{30}$$

According to Rodrigues Formula, the closed-form expression for the exponential map from $\mathfrak{so}(3)$ to $SO(3)$ is

$$\boldsymbol{U} = \exp(\boldsymbol{\theta}^\wedge) = \boldsymbol{I} + (\frac{\sin\|\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|})\boldsymbol{\theta}^\wedge + (\frac{1-\cos\|\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|^2})\boldsymbol{\theta}^{\wedge 2} \tag{31}$$

Combining Eqs. (11) and (12), The logarithm map from $SO(3)$ to $\mathfrak{so}(3)$ is

$$\boldsymbol{\theta}^\wedge = \log \boldsymbol{U} = \frac{\|\boldsymbol{\theta}\|}{2\sin\|\boldsymbol{\theta}\|} (\boldsymbol{U} - \boldsymbol{U}^T) \tag{32}$$

According to Baker-Campbell-Hausdorff Formulas, the permutation in $\mathfrak{so}(3)$ and $SO(3)$ is related as

$$\begin{aligned} \boldsymbol{U}(\boldsymbol{\theta} + \boldsymbol{\delta\theta}) &= \exp((\boldsymbol{\theta} + \boldsymbol{\delta\theta})^\wedge) = \exp((\boldsymbol{J}_L \boldsymbol{\delta\theta})^\wedge) \exp(\boldsymbol{\theta}^\wedge) \\ &= (\boldsymbol{I} + [\boldsymbol{J}_L \boldsymbol{\delta\theta}]_\times) \boldsymbol{U}, \end{aligned} \tag{33}$$

$$\frac{\partial \tilde{\boldsymbol{L}}(\boldsymbol{\theta},\rho)}{\partial \boldsymbol{\theta}} = \frac{\tilde{\boldsymbol{L}}(\boldsymbol{\theta}+\boldsymbol{\delta\theta},\rho)-\tilde{\boldsymbol{L}}(\boldsymbol{\theta},\rho)}{\boldsymbol{\delta\theta}} = \begin{bmatrix} \frac{\cos(\rho)(\boldsymbol{I}+[\boldsymbol{J}_L\boldsymbol{\delta\theta}]_\times)\boldsymbol{u}_1-\cos(\rho)\boldsymbol{u}_1}{\boldsymbol{\delta\theta}} \\ \frac{\sin(\rho)(\boldsymbol{I}+[\boldsymbol{J}_L\boldsymbol{\delta\theta}]_\times)\boldsymbol{u}_2-\sin(\rho)\boldsymbol{u}_2}{\boldsymbol{\delta\theta}} \end{bmatrix} = \begin{bmatrix} \frac{-\cos(\rho)[\boldsymbol{u}_1]_\times \boldsymbol{J}_L\boldsymbol{\delta\theta}}{\boldsymbol{\delta\theta}} \\ \frac{-\sin(\rho)[\boldsymbol{u}_2]_\times \boldsymbol{J}_L\boldsymbol{\delta\theta}}{\boldsymbol{\delta\theta}} \end{bmatrix}$$

$$= \begin{bmatrix} -\cos(\rho)[\boldsymbol{u}_1]_\times \boldsymbol{J}_L \\ -\sin(\rho)[\boldsymbol{u}_2]_\times \boldsymbol{J}_L \end{bmatrix} = \begin{bmatrix} -w_1[\boldsymbol{u}_1]_\times \boldsymbol{J}_L \\ -w_2[\boldsymbol{u}_2]_\times \boldsymbol{J}_L \end{bmatrix} \tag{35}$$

$$\frac{\partial \tilde{\boldsymbol{L}}(\boldsymbol{\theta},\rho)}{\partial \boldsymbol{\rho}} = \frac{\tilde{\boldsymbol{L}}(\boldsymbol{\theta},\rho+\delta\rho)-\tilde{\boldsymbol{L}}(\boldsymbol{\theta},\rho)}{\delta\rho} = \begin{bmatrix} -\sin(\rho)\boldsymbol{u}_1 \\ \cos(\rho)\boldsymbol{u}_2 \end{bmatrix} = \begin{bmatrix} -w_2\boldsymbol{u}_1 \\ w_1\boldsymbol{u}_2 \end{bmatrix} \tag{36}$$

where $\boldsymbol{J}_L$ is the left Jacobian of $SO(3)$ matrix:

$$\boldsymbol{J}_L = \boldsymbol{I} + (\frac{1-\cos\|\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|^2})\boldsymbol{\theta}^\wedge + (\frac{\|\boldsymbol{\theta}\|-\sin\|\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|^3})\boldsymbol{\theta}^{\wedge 2}. \tag{34}$$

In this way, we can calculate the derivatives as in Eqs. (35) and (36). Combining the two equations we have the final Jacobian:

$$\frac{\partial \tilde{\boldsymbol{L}}(\boldsymbol{\Phi})}{\partial \boldsymbol{\Phi}} = \begin{bmatrix} -w_1[\boldsymbol{u}_1]_\times \boldsymbol{J}_L & -w_2\boldsymbol{u}_1 \\ -w_2[\boldsymbol{u}_2]_\times \boldsymbol{J}_L & w_1\boldsymbol{u}_2 \end{bmatrix} \tag{37}$$

Backsubstituting into Eq. (26) we can correctly compute $\partial \boldsymbol{l}(\boldsymbol{\Phi})/\partial\boldsymbol{\Phi}$ in the end.

Computation of Second-order Derivatives $\partial^2 \boldsymbol{l}(\boldsymbol{\Phi})/\partial\boldsymbol{\Phi}^2$. To be able to extend the derivation of the first-order derivatives, the only missing term from previous derivation is the second order derivatives of the unnormalized Plücker coordinates over its minimal parameters $\boldsymbol{\Phi}$, *i.e.* $\partial^2\tilde{\boldsymbol{L}}/\partial\boldsymbol{\Phi}^2$.

Here we derive $\partial^2\tilde{\boldsymbol{L}}/\partial\boldsymbol{\Phi}^2$. From Eq. (37) we have:

$$\begin{aligned} \frac{\partial \tilde{\boldsymbol{L}}(\boldsymbol{\Phi})}{\partial \boldsymbol{\Phi}} &= \begin{bmatrix} -w_1[\boldsymbol{u}_1]_\times \boldsymbol{J}_L & -w_2\boldsymbol{u}_1 \\ -w_2[\boldsymbol{u}_2]_\times \boldsymbol{J}_L & w_1\boldsymbol{u}_2 \end{bmatrix} \\ &= \begin{bmatrix} w_1\frac{\partial u_{11}}{\partial\theta_1} & w_1\frac{\partial u_{11}}{\partial\theta_2} & w_1\frac{\partial u_{11}}{\partial\theta_3} & \frac{\partial w_1}{\partial\rho}u_{11} \\ w_1\frac{\partial u_{21}}{\partial\theta_1} & w_1\frac{\partial u_{21}}{\partial\theta_2} & w_1\frac{\partial u_{21}}{\partial\theta_3} & \frac{\partial w_1}{\partial\rho}u_{21} \\ w_1\frac{\partial u_{31}}{\partial\theta_1} & w_1\frac{\partial u_{31}}{\partial\theta_2} & w_1\frac{\partial u_{31}}{\partial\theta_3} & \frac{\partial w_1}{\partial\rho}u_{31} \\ w_2\frac{\partial u_{12}}{\partial\theta_1} & w_2\frac{\partial u_{12}}{\partial\theta_2} & w_2\frac{\partial u_{12}}{\partial\theta_3} & \frac{\partial w_2}{\partial\rho}u_{12} \\ w_2\frac{\partial u_{22}}{\partial\theta_1} & w_2\frac{\partial u_{22}}{\partial\theta_2} & w_2\frac{\partial u_{22}}{\partial\theta_3} & \frac{\partial w_2}{\partial\rho}u_{22} \\ w_2\frac{\partial u_{32}}{\partial\theta_1} & w_2\frac{\partial u_{32}}{\partial\theta_2} & w_2\frac{\partial u_{32}}{\partial\theta_3} & \frac{\partial w_2}{\partial\rho}u_{32} \end{bmatrix} \end{aligned} \tag{38}$$

Then we have

$$\frac{\partial u_{i1}}{\partial\theta_j} = \frac{\frac{\partial\tilde{\boldsymbol{L}}}{\partial\boldsymbol{\Phi}}_{ij}}{w_1}(1\le i\le 3, 1\le j\le 3) \tag{39}$$

$$\frac{\partial u_{i2}}{\partial\theta_j} = \frac{\frac{\partial\tilde{\boldsymbol{L}}}{\partial\boldsymbol{\Phi}}_{(i+3)j}}{w_2}(1\le i\le 3, 1\le j\le 3) \tag{40}$$

$$\frac{\partial \boldsymbol{\theta}^{\wedge}}{\partial \theta_1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, \frac{\partial \boldsymbol{\theta}^{\wedge}}{\partial \theta_2} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}, \frac{\partial \boldsymbol{\theta}^{\wedge}}{\partial \theta_3} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{48}$$

Since for $-w_i[\boldsymbol{u}_i]_{\times}\boldsymbol{J}_L (i = 1, 2)$ we have:

$$\frac{\partial(-w_i[\boldsymbol{u}_i]_{\times}\boldsymbol{J}_L)}{\partial \boldsymbol{\theta}} = -w_i(\frac{\partial[\boldsymbol{u}_i]_{\times}}{\partial \boldsymbol{\theta}}\boldsymbol{J}_L + [\boldsymbol{u}_i]_{\times}\frac{\partial \boldsymbol{J}_L}{\partial \boldsymbol{\theta}}) \tag{41}$$

$$\frac{\partial(-w_i[\boldsymbol{u}_i]_{\times}\boldsymbol{J}_L)}{\partial \rho} = -\frac{\partial w_i}{\partial \rho}[\boldsymbol{u}_i]_{\times}\boldsymbol{J}_L \tag{42}$$

we can use Eqs. (39) and (40) to compute $\partial[\boldsymbol{u}_i]_{\times}/\partial\boldsymbol{\theta}$. Then, we only need to derive $\partial \boldsymbol{J}_L/\partial\boldsymbol{\theta}$. Furthermore, according to Eq. (34), we have:

$$\boldsymbol{J}_L = \boldsymbol{I} + f_g\boldsymbol{\theta}^{\wedge} + f_h\boldsymbol{\theta}^{\wedge 2}, \tag{43}$$

where f_g and f_h are defined as:

$$f_g = \frac{1 - \cos\|\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|^2}, f_h = \frac{\|\boldsymbol{\theta}\| - \sin\|\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|^3} \tag{44}$$

Thus, we can compute each component as follows:

$$\frac{\partial \boldsymbol{J}_L}{\partial \boldsymbol{\theta}} = \frac{\partial f_g}{\partial \boldsymbol{\theta}}\boldsymbol{\theta}^{\wedge} + f_g\frac{\partial \boldsymbol{\theta}^{\wedge}}{\partial \boldsymbol{\theta}} + \frac{\partial f_h}{\partial \boldsymbol{\theta}}\boldsymbol{\theta}^{\wedge 2} + f_h(\frac{\partial \boldsymbol{\theta}^{\wedge}}{\partial \boldsymbol{\theta}}\boldsymbol{\theta}^{\wedge} + \boldsymbol{\theta}^{\wedge}\frac{\partial \boldsymbol{\theta}^{\wedge}}{\partial \boldsymbol{\theta}}) \tag{45}$$

$$\frac{\partial f_g}{\partial \boldsymbol{\theta}} = (\frac{\sin\|\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|^3} + \frac{2(\cos\|\boldsymbol{\theta}\| - 1)}{\|\boldsymbol{\theta}\|^4})\boldsymbol{\theta}^T \tag{46}$$

$$\frac{\partial f_h}{\partial \boldsymbol{\theta}} = (\frac{1 - \cos\|\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|^4} + \frac{3(\sin\|\boldsymbol{\theta}\| - \|\boldsymbol{\theta}\|)}{\|\boldsymbol{\theta}\|^5})\boldsymbol{\theta}^T \tag{47}$$

where the partial derivative $\partial\boldsymbol{\theta}^{\wedge}/\partial\boldsymbol{\theta}$ can be computed as in Eq. (48).

Extension to Loss Kernels In practice the robust loss function is generally used to fight against the potential presence of outliers. In our system we use the Cauchy loss function following [14]. These robust loss functions are equivalent to applying a non-linear kernel function on top of the original residuals, which corresponds to the following objective:

$$\begin{aligned} \boldsymbol{\Phi}^* &= \arg\min_{\boldsymbol{\Phi}} E_g \\ &= \arg\min_{\boldsymbol{\Phi}} \frac{1}{2}\sum_{k}^{N_k}\sum_{j}^{s,e} g([D(\boldsymbol{x}_j^k, \Pi_k(\boldsymbol{L}(\boldsymbol{\Phi})]^2))). \end{aligned} \tag{49}$$

Here without loss of generality we denote $g(\cdot)$ a kernel function that operates on the squared error, with E_g denoting the full optimization objective. Following the same practice as in Eq. (20), we have:

$$\frac{\partial^2 E_g}{\partial \boldsymbol{\Phi} \partial \boldsymbol{x}_j^k} = \mathbf{0}. \tag{50}$$

Since the first-order derivative over $\boldsymbol{\Phi}$ can be written as:

$$\frac{\partial \left(\frac{1}{2} g([D(\boldsymbol{x}, \boldsymbol{l}_k(\boldsymbol{\Phi}))]^2) \right)}{\partial \boldsymbol{\Phi}} = \frac{1}{2} g'(\cdot) \frac{\partial [D(\boldsymbol{x}, \boldsymbol{l}_k(\boldsymbol{\Phi}))]^2}{\partial \boldsymbol{\Phi}}, \tag{51}$$

$$\begin{aligned} \frac{\partial (\frac{1}{2} g'(\cdot) \frac{\partial [D(\boldsymbol{x}, \boldsymbol{l}_k(\boldsymbol{\Phi}))]^2}{\partial \boldsymbol{\Phi}})}{\partial \boldsymbol{x}_j^k} =& (\frac{1}{2} \frac{\partial [D(\boldsymbol{x}, \boldsymbol{l}_k(\boldsymbol{\Phi}))]^2}{\partial \boldsymbol{\Phi}})^T \frac{\partial g'(\cdot)}{\partial \boldsymbol{x}_j^k} + g'(\cdot) \frac{\partial \frac{1}{2} \frac{\partial [D(\boldsymbol{x}, \boldsymbol{l}_k(\boldsymbol{\Phi}))]^2}{\partial \boldsymbol{\Phi}}}{\boldsymbol{x}_j^k} \\ =& (\frac{1}{2} \frac{\partial [D(\boldsymbol{x}, \boldsymbol{l}_k(\boldsymbol{\Phi}))]^2}{\partial \boldsymbol{\Phi}})^T g''(\cdot) \frac{\partial [D(\boldsymbol{x}, \boldsymbol{l}_k(\boldsymbol{\Phi}))]^2}{\partial \boldsymbol{x}_j^k} + \\ & \frac{1}{2} g'(\cdot) \frac{\partial^2 [D(\boldsymbol{x}, \boldsymbol{l}_k(\boldsymbol{\Phi}))]^2}{\partial \boldsymbol{\Phi} \partial \boldsymbol{x}_j^k} \end{aligned} \tag{52}$$

by expanding the LHS of Eq. (50) we have the form in Eq. (52). Note that the three derivative terms in Eq. (52) have all been discussed in the previous section with the naive loss function. Thus, we can safely extend the derivation from Sec. B.2 to complete the follow-up second-order sensitivity analysis to get the target Jacobian $\partial \boldsymbol{\Phi}^* / \partial \boldsymbol{x}_j^k$.

Validity Test Our analytical uncertainty propagation on both points and lines have all been validated with numerical tests. Specifically, we can make small perturbation on each input observation $\boldsymbol{x}_j^k$ and perform optimization on top of it, which enables us to compute the Jacobian $\partial \boldsymbol{\Phi}^* / \partial \boldsymbol{x}_j^k$ numerically. All the entries in the Jacobian matrix have less than 1% deviation between the numerical and analytical results.

We further perform a correlation test between the propagated 3D uncertainty and the map precision on the *delivery_ area* scene from ETH3D [24], as shown in Fig. 4 in the main paper. For each point track and line track, we not only compute its analytical 3D uncertainty but also measure its distance (distance from the nearest point for each line) to the groundtruth scans provided in the dataset. For points, we use the squared root of the maximum eigenvalue of the 3x3 point covariance matrix as the scalar-valued point uncertainty. For lines, we first propagate the 3D uncertainty on the optimal infinite line into the two endpoints as discussed in Sec. B.3 and take the squared root of the maximum eigenvalue between two endpoints as the scalar-valued line uncertainty. Then, we sort the points and lines into five bins with respect to its 3D uncertainty, and compute the precision for each bin at 1cm / 3cm / 5cm. Results from Fig. 4 in

the main paper show clear correlation between the propagated 3D uncertainty and the map precision.

B.3 Applications of the Propagated Covariance

Uncertainty Propagation for 3D Line Segments The propagated 3D uncertainty for each optimal line is represented in a 4x4 covariance matrix on its minimal parameters $\boldsymbol{\Phi}$. To get a geometrically meaningful uncertainty we can propagate the covariance matrix onto its 3D endpoints using the 3D point-to-line projection in Plücker form:

$$\boldsymbol{X}_{\perp} = \boldsymbol{X} + \boldsymbol{d} \times (\boldsymbol{m} + \boldsymbol{d} \times \boldsymbol{X}), \tag{53}$$

where $\boldsymbol{X}_{\perp}$ is the projection of $\boldsymbol{X}$ on the 3D line $\boldsymbol{L} = \begin{bmatrix} \boldsymbol{d} \, \boldsymbol{m} \end{bmatrix}$ in its Plücker form. Thus, we can easily compute the covariance of the 3D endpoint $\boldsymbol{X}_s$ (without loss of generality we consider the starting endpoint $\boldsymbol{X}_s$ here) using the operation of projecting it onto the line:

$$\Sigma_{\boldsymbol{X}_{s\perp}} = \frac{\partial \boldsymbol{X}_{s\perp}}{\partial \boldsymbol{\Phi}} \Sigma_{\boldsymbol{\Phi}} \frac{\partial \boldsymbol{X}_{s\perp}}{\partial \boldsymbol{\Phi}}^T \tag{54}$$

$$\frac{\partial \boldsymbol{X}_{s\perp}}{\partial \boldsymbol{\Phi}} = \frac{\partial \boldsymbol{X}_{s\perp}}{\partial \boldsymbol{L}} \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{\Phi}}. \tag{55}$$

Note that here we have $X_{s\perp} = X_s$ since X_s naturally lies on the infinite optimal line. Here we give the solution of the term $\partial \boldsymbol{X}_{\perp} / \partial \boldsymbol{L}$ as follows:

$$\frac{\partial \boldsymbol{X}_{\perp}}{\partial \boldsymbol{m}} = [\boldsymbol{d}]_{\times} \tag{56}$$

$$\frac{\partial \boldsymbol{X}_{\perp}}{\partial \boldsymbol{d}} = -[\boldsymbol{m}]_{\times} + (\boldsymbol{d}^T \boldsymbol{X}) \boldsymbol{I} + \boldsymbol{d} \boldsymbol{X}^T - 2 \boldsymbol{X} \boldsymbol{d}^T \tag{57}$$

In this way, we can get a 3x3 covariance matrix for each endpoint of the line through the whole uncertainty propagation from the noise of 2D endpoint observations.

Scale-Invariant 3D Uncertainty To be able to identify noisy points and lines, we can use the scalar-valued uncertainty. This can be computed as the squared root of the maximum eigenvalue of the point covariance matrix and endpoint covariance matrix (described in Sec. B.3) respectively.

However, to make it general, we need an uncertainty measurement that is relatively invariant to the scale of the scene, since the SfM reconstruction naturally exhibits the gauge ambiguity that makes the camera poses optimal only up to a similarity transformation. To deal with this issue, we propose to use the information from the supporting views for each track to rescale the scalar-valued uncertainty. Specifically, for each track, we divide the scalar-valued uncertainty by

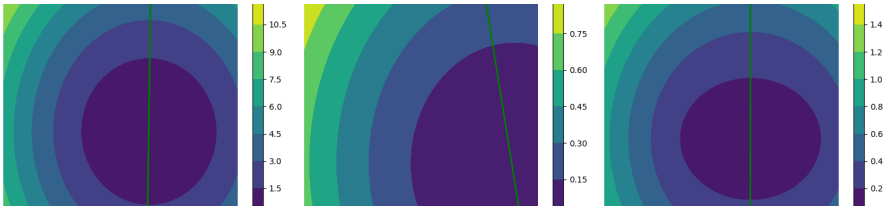

Fig. 1: Visualization of the uncertainty of reprojection error for 3D lines at different measured locations. Different from points, the uncertainty of the line reprojection error (point-to-line distance) depends on where it is measured.

the median value of "depth over focal length" across its supporting views. In this way, we get a scale-invariance 3D uncertainty that is in the unit of pixels. If the focal length is the same across views, this scale-invariance measurement can be geometrically interpreted as the reprojection uncertainty from the least reliable views at a certain distance. The measurement is used to identify unreliable tracks in the refinement (as shown in Fig. 10 in the main paper).

Reprojection Uncertainty As discussed in the main paper, a point/line track with large 3D uncertainty may still be reliable at certain views for localization. Thus, to reweight correspondences based on the 3D uncertainty from the map perspective, we employ the reprojection uncertainty with respect to an initial pose rather than the global one. This can be achieved by propagating the 3D uncertainty on the infinite line parameterized by $\boldsymbol{\Phi}$ to the uncertainty of the reprojection error, which is again formulated as the endpoint-to-line distance (as in Eq. (15)).

It is worth noting that different from points, the uncertainty of the reprojection error not only depends on the optimal 3D line, but also depends on the location of the measurement on the image. This is illustrated in Figure 1. The geometric interpretation is that: with the perturbation on the 3D infinite line with its uncertainty, the corresponding reprojected 2D line will move and rotate in the image plane. This can potentially form a relatively stationary region that has stably small reprojection error. Thus, measuring reprojection error in such regions with small reprojection uncertainty gives more reliable information on the reliability of the absolute pose proposal, therefore improving the accuracy and robustness of the localization results. The visualization of the reprojection uncertainty can be found in Figure 7 and Figure 11 in the main paper. We also show that the uncertainty-aware reweighting is able to contribute to consistent improvement under both point-alone and hybrid setup (as in Table 8 in the main paper).

C Integration of Structural Associations

Inspired by LIMAP [14], we integrate structural constraints at both triangulation and refinement by modeling point-line associations and VP-line associations. Specifically, the 2D point-line association graph for each image can be easily constructed with 2D point-to-line distance, and the 2D VP-line association graph naturally emerges from the VP estimation [29].

At triangulation, we directly follow the practice of LIMAP [14] to generate proposals from neighboring points and vanishing points to fight against the degeneracy issue of two-view line triangulation. At refinement, we can add soft constraints similarly to LIMAP [14] between points and lines, lines and VP by counting the connections of the corresponding supports on the 2D association graphs. While this appears to be beneficial on the quality of reconstruction and the pose accuracy, the integration of association residuals unfortunately breaks the blockwise property of the bundle adjustment Jacobian.

The general practice for efficient bundle adjustment involves matrix partitioning and the exploitation of Schur complement [2, 30], which largely benefits from the fact that the 3D map (points and lines) and the cameras form a bipartite graph in the optimization objective. However, the association residuals add edges among points and lines, making it unable to reorganize the Jacobian into the form where each point and line makes a block at the map side, inducing more off-diagonal entries in the corresponding submatrix of the Hessian for the map. While we can still perform the Schur complement trick in the larger block with connected component analysis, the relatively dense connections from the current design of using soft residuals can lead to huge blocks that significantly affect the efficiency. Nonetheless, since the Jacobian is still sparse and many soft residuals are not necessary, there is large room for efficiency improvement with more careful implementation. Also, the one without point-line association already achieves very promising results as shown in the main paper.

Note that the VP-line association does not suffer from this problem, since the VPs can be viewed as special cameras and thus moved to the camera side, keeping the graph from the optimization objective a bipartite.

D Absolute Pose Estimation with a Vanishing Point Correspondence

Since we additional construct and maintain the vanishing point (VP) tracks in the hybrid map, when registering a new image we have additional 2D-3D vanishing point correspondences from traversing the matching graph. This gives additional constraints on the rotation matrix. Specifically, one 2D-3D vanishing point correspondence gives constraints on 2 degrees of freedom on the absolute rotation, and a second one gives one another constraint on the final degree of freedom, which leaves a non-minimal overconstrained system. In this paper we only focuses on using a single vanishing point correspondence in the minimal estimation inside the hybrid RANSAC framework [5].

D.1 Relation to Gravity-Aligned Solvers

The presence of a single 2D-3D VP correspondence $(\boldsymbol{v}_{2d}, \boldsymbol{v}_{3d})$ gives the constraint on the rotation R as follows:

$$\boldsymbol{R}\boldsymbol{v}_{3d} = \boldsymbol{v}_{2d}, \tag{58}$$

where $\boldsymbol{v}_{2d}$ is the unnormalized VP direction in the local frame. This gives two degrees of freedom for the rotation matrix $\boldsymbol{R}$. Note that the constraint is equivalent to the gravity-aligned absolute pose estimation [13] with a tilted gravity direction. Thus, we can first rotate the system to align the 3D VP direction to the y-axis to employ all advances under the context of gravity-aligned absolute pose estimation. With two DOFs constrained in the rotation matrix, the absolute pose has 4 DOFs left, which can be reduced with 1) two 2D-3D point correspondences; 2) one 2D-3D point correspondence and one 2D-3D line correspondence. Note that two 2D-3D line correspondences do not work in this case due to dependent constraints on the rotation.

D.2 Gravity-Aligned Absolute Pose with Two Point Correspondences

The gravity-aligned absolute pose estimation with two point correspondences is initially studied in [13]. The main idea is to parameterize the final DOF of the rotation matrix in the polynomial form as follows:

$$\boldsymbol{R}(q) = \frac{1}{1+q^2}\begin{bmatrix} 1-q^2 & 0 & 2q \\ 0 & 1+q^2 & 0 \\ -2q & 0 & 1-q^2 \end{bmatrix}, \quad q \in \mathbb{R} \tag{59}$$

From one 2D-3D point correspondence $(\boldsymbol{x}, \boldsymbol{X})$ we have:

$$[\boldsymbol{x}]_\times(\boldsymbol{R}\boldsymbol{X} + \boldsymbol{t}) = 0 \tag{60}$$

With the availability of two such equations the problem can be finally reduced to a quadratic polynomials in q, which can be solved in closed form [13].

D.3 Gravity-Aligned Absolute Pose with One Point and One Line Correspondences

Similarly, given one point correspondence $(\boldsymbol{x}, \boldsymbol{X})$ and one line correspondence $(\boldsymbol{\ell}, \boldsymbol{L})$ it is also possible to recover the camera pose under known vertical direction. Let the 3D line $\boldsymbol{L}$ be parameterized as $t \mapsto \boldsymbol{X}_L + t\boldsymbol{V}_L$, then we get the following constraint on the camera rotation

$$\boldsymbol{\ell}^T\boldsymbol{R}\boldsymbol{V}_L = 0. \tag{61}$$

Using the same parameterization as in Eq. (59), this yields a quadratic equation in q which can be solved in closed form. Once the rotation is recovered, we have

three linearly independent equations left,

$$[\boldsymbol{x}]_\times(\boldsymbol{R}\boldsymbol{X}+\boldsymbol{t})=0, \tag{62}$$

$$\boldsymbol{\ell}^T(\boldsymbol{R}\boldsymbol{X}_L+\boldsymbol{t})=0, \tag{63}$$

from which we can recover the translation $\boldsymbol{t}$.

E More Experimental Details

E.1 Implementation Details

Our system is implemented in C++ with Python bindings. We follow the overall design of COLMAP [23] and use the same hyperparameters for the point part, which enables fair comparison between "point" and "hybrid" setup. For the scoring at line triangulation and hybrid bundle adjustment, we follow LIMAP [14] for the parameter choices. The line tracks are optimized with a Cauchy loss with parameter 0.25. For the registration, we use the same weight for points and lines at scoring and local optimization.

Similar to all existing SfM methods, those hyperparameters in our system can be changed by the users for both practical and research purpose, while using our default parameters at release should already work reasonable well on most in-the-wild cases.

We use the same hyperparameters across all the experiments. For the point feature, we use "SIFT" [15] + "NN-ratio" and "superpoint_max" [8] + "superglue outdoor" [21] from HLoc [20]. For the line feature, vanishing point estimation, and construction of 2D association graph, we follow the official implementation of LIMAP [14]. We use exhaustive matching for both the point-alone baseline COLMAP [23] and our method across all datasets.

E.2 Datasets and Evaluation

We test the proposed SfM system on multiple public datasets to verify its effectiveness. We mainly use the following two metrics at the evaluation:

- **Valid Registrations**: While the number of registered images is often used in the SfM evaluation, different methods can have different tolerance criteria on rejecting the registrations. This can potentially make the evaluation metric unfair when, for example, one system registers all the images with some of them totally deviating from the groundtruth. Motivated by these observations, we propose to evaluate using the valid registrations. Specifically, we first perform robust model alignment between the SfM predictions and the groundtruth and then count the number of images that are within 5cm / 5deg to the groundtruth. In our implementation we use the interfaces from COLMAP [23] for robust alignment.

– **Relative pose AUC**: Following the general practice on SfM evaluation [11], we compute relative pose errors over all image pairs exhaustively with respect to the groundtruth. For those pairs with one or both images not registered in the SfM reconstruction, we set a maximum relative pose error of 180 degree. AUC at different thresholds (in degrees) are reported on all the errors.

Hypersim [19] is a photorealistic synthetic dataset that is used for holistic indoor scene understanding. We follow the practice of LIMAP [14] to test on the first eight scenes and resize the image to a maximum dimension of 800. Each of the scene has around 100 images. The average across all the tested scenes are reported for each metric. For the evaluation of line reconstruction in Table 5, we use the evaluation tools in the code release of [14].

ETH3D [24] is a real world dataset that includes unordered images in both indoor and outdoor environments. We use the training split of DSLR images, which has a total of 13 scenes. We resize each image to a maximum dimension of 756, which is of the same size of the provided groundtruth depth images. This makes it faster to run our default line matcher GlueStick [18], while advances on the efficiency of line matching can further reduce the runtime at feature detection and matching. For evaluation we use the same two metrics as discussed with respect to the groundtruth poses provided in the data.

We further validate our method on the PhotoTourism data [27] from Image Matching Benchmark 2020 [11]. Specifically we test on the validation split which consists of three scenes in total: *Reichstag*, *Sacre Coeur*, and *Saint Peter's Square*. The official setup [11] runs COLMAP [23] on all images as a pseudo groundtruth and evaluates on a collection of small bags. Not only does this induce a small bias towards point-based methods, but the dataset is also with limited presence of structured line features. Nonetheless, we still show consistent improvements over point-alone methods with different types of features. Note that since the official groundtruth construction on this dataset employs a radial distortion model which does not favor line detection, we first perform undistortion on all the images and run SfM for both our system and the baseline COLMAP [23] with known intrinsics on the undistorted images.

E.3 Details on Visual Localization

To verify the effectiveness of our uncertainty-aware visual localization module, we perform visual localization experiments on two public datasets: Cambridge [12] and 7Scenes [26]. We follow the same experimental setup of HLoc [20] and LIMAP [14] to ensure fair comparison. Specifically, we use the triangulated point map from COLMAP [23] and line map from LIMAP [14]. We use NetVLAD [3] for image retrieval and SuperPoint [8] + SuperGlue [21] as the point feature for both datasets. For line features, following LIMAP [14] we use LSD [31] on Cambridge and SOLD2 [17] on 7Scenes [26].

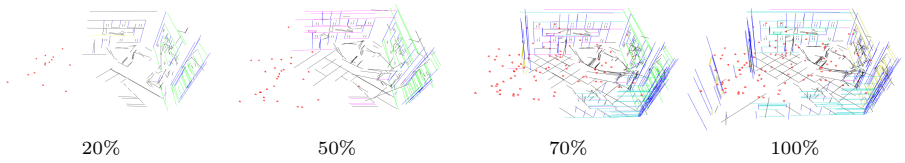

Fig. 2: Incremental line reconstruction during SfM. Parallel lines from line-VP association are colored the same.

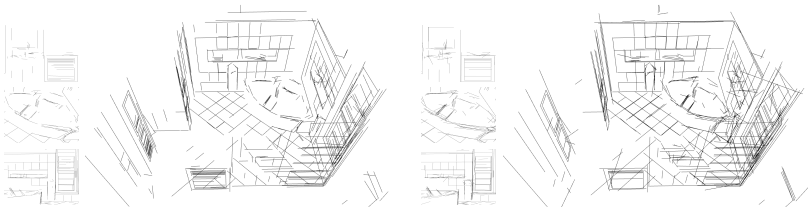

Fig. 3: Comparison of our proposed incremental line triangulator (**Right**) with the state-of-the-art global line triangulator (**Left**) [14]. Without the need of getting all the posed images beforehand, our incremental triangulator achieve robust line reconstruction of comparable quality with the global counterpart.

F Additional Results

We show qualitative results on the incremental process of hybrid reconstruction during SfM and the visual comparison with the global line triangulation from [14] in Fig. 2 and 3. We also show qualitative examples of our hybrid reconstruction on PhotoTourism [11]. Our method is able to reconstruct reasonable 3D maps of hybrid features from as few as 5 images.

Figure 5 shows an example on the effects of caching inactive supports and two-step refinement. The 3D lines are triangulated and removed repeatedly with the naive strategy, while the proposed mechanism keeps the temporarily unreliable supports and tracks while isolating them from the pose optimization.

We further show visual examples of our propagated 3D covariance on the reconstructed 3D lines in Fig. 6. With the principled uncertainty propagation our method can explicitly identify noisy lines from few views and degenerate configurations (horizontal lines cannot be reliably reconstructed with parallel horizontal motion). We also show another visual example of the effectiveness of the rejected uncertainty in Fig. 7.

To further verify the improved robustness of our proposed system, we run both COLMAP [23] and our hybrid SfM method on the recently introduced LaMAR dataset [22]. LaMAR [22] is a new large-scaled dataset that is captured using AR devices in diverse environments. We use the *hetrf* sensor from the HoloLens query validation data which consists of 12 sequences. Table 1 shows results with SuperPoint [8] + SuperGlue [21]. While the dataset is very challenging, we manage to get consistent improvement over COLMAP [23], thanks to the

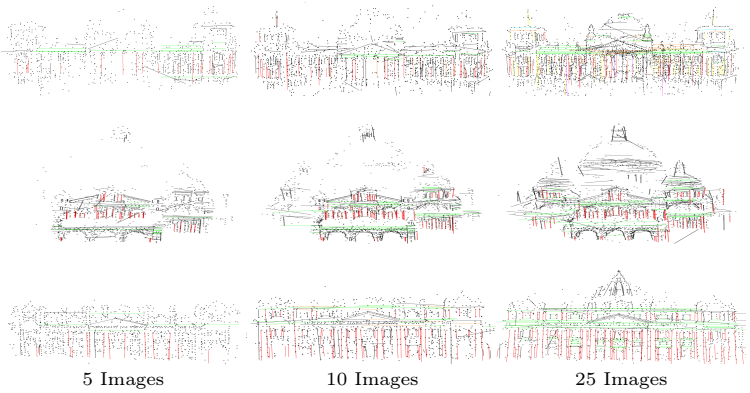

Fig. 4: Our hybrid reconstruction with different small image bags from PhotoTourism [11, 27]. Our method can achieve reasonable reconstruction from as few as 5 images.

Table 1: Structure-from-Motion results on LaMAR [22]. We report the relative pose AUC for both our system ("Hybrid") and COLMAP ("Point") [23] on SuperPoint [8] + SuperGlue [21]. While we achieve consistent improvement over COLMAP [23], the dataset is very challenging due to low overlap and fast motion.

Dataset	Point Feature	Method	AUC @ 3°/5°/10° ↑		
LaMAR	SP + SG	Point	2.8	7.9	14.5
		Hybrid	**4.0**	**10.8**	**18.5**

inclusion of hybrid features. Nonetheless, there is still large room for future improvement on this dataset, which can further benefit from temporal modeling. This is beyond the scope of this paper and is left as the next-step future work.

Finally, we apply our proposed SfM system to the widely studied application: view synthesis. Specifically, we run Nerfactos from NeRFStudio [16, 28] on both COLMAP [23] and our SfM predictions. Fig. 8 shows that our method enables better view synthesis quality due to more accurate and robust camera registrations.

References

1. Agarwal, S., Mierle, K.: Ceres solver. `http://ceres-solver.org`
2. Agarwal, S., Snavely, N., Seitz, S.M., Szeliski, R.: Bundle adjustment in the large. In: ECCV (2010)
3. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: Netvlad: Cnn architecture for weakly supervised place recognition. In: CVPR (2016)
4. Bartoli, A., Sturm, P.: Structure-from-motion using lines: Representation, triangulation, and bundle adjustment. Computer Vision and Image Understanding (CVIU) **100**(3), 416–441 (2005)

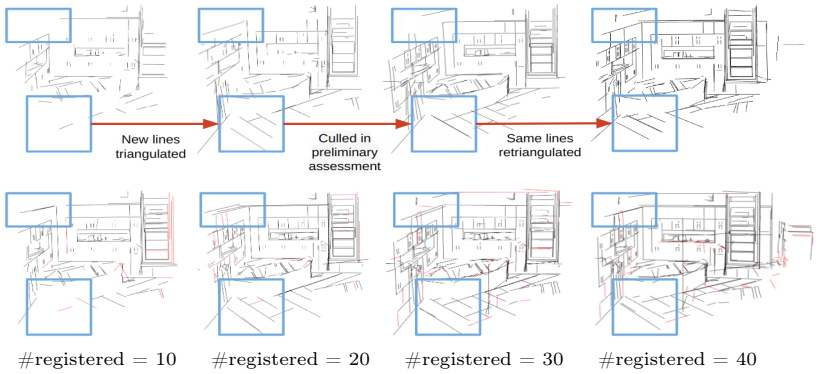

Fig. 5: Effects of caching inactive supports and two-step refinement. With the naive strategy (first row), lines are triangulated and removed repeatedly (as highlighted in the blue boxes), which largely slows down the reconstruction process. Our proposed mechanism (second row) keeps the unreliable supports and tracks (red) in the map while isolating them from the pose optimization.

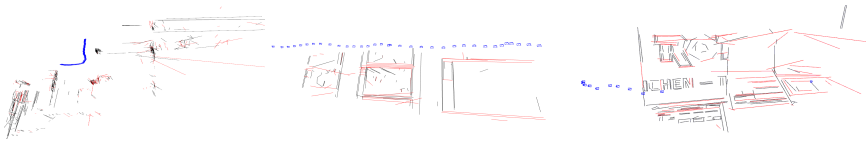

Fig. 6: Visualization of the scale-invariant reliability measurement based on the uncertainty propagation during SfM. Not only can our method identify noisy lines (red) flying around space, but it can also model unstable lines from degenerate view configurations which are not necessarily short of supporting views.

5. Camposeco, F., Cohen, A., Pollefeys, M., Sattler, T.: Hybrid camera pose estimation. In: CVPR (2018)
6. Chum, O., Matas, J., Kittler, J.: Locally optimized ransac. In: Joint Pattern Recognition Symposium (2003)
7. Dellaert, F., Kaess, M.: Factor Graphs for Robot Perception. Foundations and Trends in Robotics, Vol. 6 (2017), `http://www.cs.cmu.edu/~kaess/pub/Dellaert17fnt.pdf`
8. DeTone, D., Malisiewicz, T., Rabinovich, A.: Superpoint: Self-supervised interest point detection and description. In: Computer Vision and Pattern Recognition Workshops (CVPRW) (2018)
9. Fiacco, A.V., Ishizuka, Y.: Sensitivity and stability analysis for nonlinear programming. Annals of Operations Research **27**(1), 215–235 (1990)
10. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge university press (2003)
11. Jin, Y., Mishkin, D., Mishchuk, A., Matas, J., Fua, P., Yi, K.M., Trulls, E.: Image matching across wide baselines: From paper to practice. IJCV **129**(2), 517–547 (2021)
12. Kendall, A., Grimes, M., Cipolla, R.: PoseNet: A convolutional network for real-time 6-DoF camera relocalization. In: ICCV (2015)

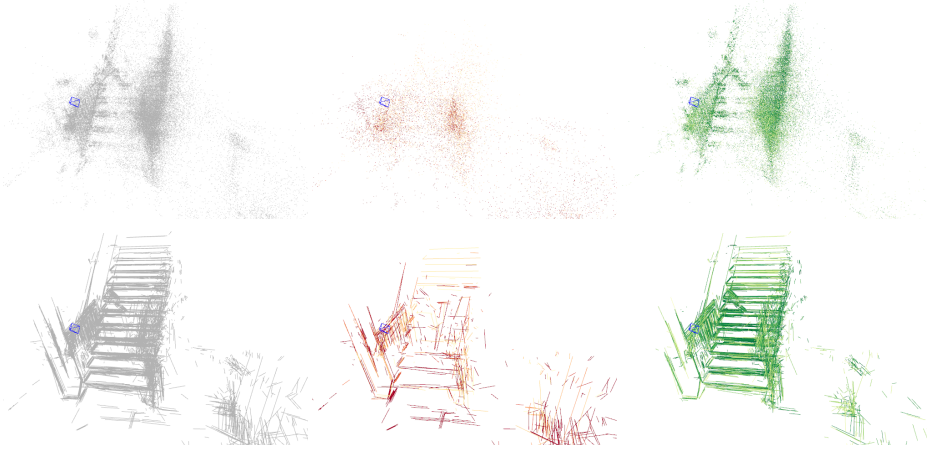

Fig. 7: Visualization on the reprojection uncertainty used in the uncertainty-aware localization on *stairs* from 7Scenes [26]. We show results on both points (**first row**) and lines (**second row**).

Fig. 8: With better accuracy and more valid registrations, our pipeline is able to improve robustness on neural rendering [16, 28].

13. Kukelova, Z., Bujnak, M., Pajdla, T.: Closed-form solutions to minimal absolute pose problems with known vertical direction. In: ACCV (2010)
14. Liu, S., Yu, Y., Pautrat, R., Pollefeys, M., Larsson, V.: 3d line mapping revisited. In: CVPR (2023)
15. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60**(2), 91–110 (2004)
16. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)
17. Pautrat, R., Lin, J.T., Larsson, V., Oswald, M.R., Pollefeys, M.: Sold2: Self-supervised occlusion-aware line description and detection. In: CVPR (2021)
18. Pautrat, R., Suárez, I., Yu, Y., Pollefeys, M., Larsson, V.: Gluestick: Robust image matching by sticking points and lines together. In: ICCV (2023)
19. Roberts, M., Ramapuram, J., Ranjan, A., Kumar, A., Bautista, M.A., Paczan, N., Webb, R., Susskind, J.M.: Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In: ICCV (2021)
20. Sarlin, P.E.: Visual localization made easy with hloc. `https://github.com/cvg/Hierarchical-Localization/`

21. Sarlin, P.E., DeTone, D., Malisiewicz, T., Rabinovich, A.: Superglue: Learning feature matching with graph neural networks. In: CVPR (2020)
22. Sarlin, P.E., Dusmanu, M., Schönberger, J.L., Speciale, P., Gruber, L., Larsson, V., Miksik, O., Pollefeys, M.: LaMAR: Benchmarking Localization and Mapping for Augmented Reality. In: ECCV (2022)
23. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR (2016)
24. Schops, T., Schonberger, J.L., Galliani, S., Sattler, T., Schindler, K., Pollefeys, M., Geiger, A.: A multi-view stereo benchmark with high-resolution images and multi-camera videos. In: CVPR (2017)
25. Seber, G.A., Wild, C.J.: Nonlinear regression. New Jersey: John Wiley & Sons **62**(63), 1238 (2003)
26. Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., Fitzgibbon, A.: Scene coordinate regression forests for camera relocalization in RGB-D images. In: CVPR (2013)
27. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3d. In: ACM SIGGRAPH (2006)
28. Tancik, M., Weber, E., Ng, E., Li, R., Yi, B., Kerr, J., Wang, T., Kristoffersen, A., Austin, J., Salahi, K., Ahuja, A., McAllister, D., Kanazawa, A.: Nerfstudio: A modular framework for neural radiance field development. In: ACM SIGGRAPH 2023 Conference Proceedings (2023)
29. Toldo, R., Fusiello, A.: Robust multiple structures estimation with j-linkage. In: ECCV (2008)
30. Triggs, B., McLauchlan, P.F., Hartley, R.I., Fitzgibbon, A.W.: Bundle adjustment—a modern synthesis. In: Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21–22, 1999 Proceedings (2000)
31. Von Gioi, R.G., Jakubowicz, J., Morel, J.M., Randall, G.: Lsd: A fast line segment detector with a false detection control. IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI) **32**(4), 722–732 (2008)
32. Zuo, X., Xie, X., Liu, Y., Huang, G.: Robust visual slam with point and line features. In: IROS (2017)