FASTMAP: Revisiting Dense and Scalable Structure from Motion

Jiahao Li[1*] Haochen Wang[1*] Muhammad Zubair Irshad[2] Igor Vasiljevic[2]
Matthew R. Walter[1] Vitor Campagnolo Guizilini[2†] Greg Shakhnarovich[1†]
[1]TTI-Chicago [2]Toyota Research Institute
{jiahao,whc,mwalter,greg}@ttic.edu
{zubair.irshad,igor.vasiljevic,vitor.guizilini}@tri.global

Abstract

We propose FASTMAP*, a new global structure from motion method focused on speed and simplicity. Previous methods like COLMAP and GLOMAP are able to estimate high-precision camera poses, but suffer from poor scalability when the number of matched keypoint pairs becomes large. We identify two key factors leading to this problem: poor parallelization and computationally expensive optimization steps. To overcome these issues, we design an SfM framework that relies entirely on GPU-friendly operations, making it easily parallelizable. Moreover, each optimization step runs in time linear to the number of image pairs, independent of keypoint pairs or 3D points. Through extensive experiments, we show that* FASTMAP *is faster than COLMAP and GLOMAP on large-scale scenes with comparable pose accuracy. Project webpage: https://jiahao.ai/fastmap.*

1. Introduction

Structure from Motion (SfM) is the task of recovering the camera parameters (intrinsics and extrinsics) and 3D structure (usually a sparse point cloud) given a collection of images. It is an essential step in a typical 3D reconstruction pipeline [1], and has been attracting more and more attention thanks to the recent development of neural reconstruction methods like NeRF [33] and Gaussian Splatting [21]. The 3D structure can be recovered through triangulation once we know the camera poses, but it can also facilitate the estimation of camera parameters. State-of-the-art systems [37, 45] jointly estimate the cameras and the point cloud with techniques such as bundle adjustment [50]. Another approach [14, 17, 55] is to directly estimate the camera parameters without the aid of 3D points, by mainly relying on globally aligning relative motions. In this paper we follow the latter paradigm, and aim to design a new SfM method that is both fast and accurate.

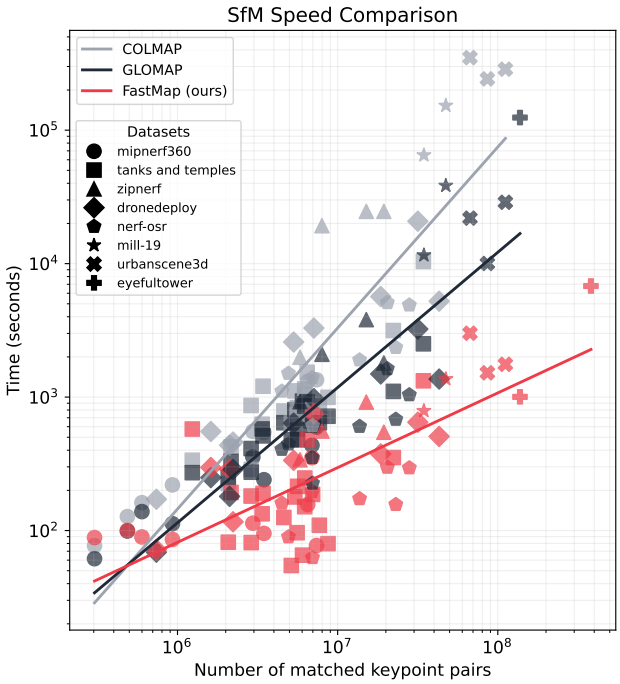

Figure 1. Timing of FASTMAP compared to COLMAP and GLOMAP on scenes from eight datasets, excluding the matching stage for all methods. Note the logarithmic time scale. Lines represent a least squares power function fit to timing across multiple datasets, as a function of the number of matched keypoint pairs.

Recent advances in learning-based SfM methods [7, 10, 47, 53] yield impressive results in the sparse regime, where one has a few images and attempts to recover approximate camera poses. However, their performance remains lacking in the dense and high-precision regime, where one has hundreds or thousands of images and aims for maximal accuracy. In such a scenario, which is our focus, the current state-of-the-art is still held by COLMAP [45]. Learning-based methods such as DUSt3R [53] rely on COLMAP to

*core contributors
†joint supervision & equal contributions

provide pseudo ground truth for training. But COLMAP is slow — processing a scene consisting of thousands of images could take multiple days. GLOMAP [37] improves upon COLMAP's speed through global (instead of incremental) SfM, but still takes many hours to converge on large scenes. For learning-based systems to efficiently scale up the training data, a fast and high quality pseudo ground truth annotator would be very helpful.

Previous SfM methods are slow in part due to their focus on the goal of sub-degree accuracy. Joint optimization of camera poses and 3D points with Quasi-second order Gauss-Newton type solvers [2] plays an essential role in driving down the "last one degree" of misalignment. This is particularly important for incremental SfM methods like COLMAP that may be prone to compounding errors during sequential camera registration. However, Gauss-Newton solvers scale poorly with the number of images and 3D points, and GPU implementations that can effectively exploit the sparsity in the Jacobians for improved scalability are less mature. As a result, the current software ecosystem remains largely CPU-bound—while COLMAP can use GPUs for SIFT [29] feature extraction and matching, camera registration remains restricted to CPUs.

We propose FASTMAP, a redesigned SfM framework that achieves fast, high-accuracy dense structure from motion. On large scenes with thousands of images, FASTMAP is up to one to two orders of magnitude faster than GLOMAP and COLMAP. See Fig. 1 for an efficiency comparison across many scenes. Importantly, FASTMAP achieves efficiency improvements while keeping comparable performance—extensive experiments on eight datasets demonstrate pose estimation accuracy and novel view synthesis quality close to GLOMAP and COLMAP.

FASTMAP is fast due to two major design choices. First, for all the iterative nonlinear optimization problems involved, we design algorithms such that the computational complexity of each iteration is only linear in the number of image pairs, not keypoint pairs or 3D points. This includes replacing the traditional bundle adjustment [50] present in previous SfM frameworks with a novel *re-weighting epipolar adjustment* algorithm, which is much more efficient. Second, throughout the entire framework, we formulate as many steps as possible as GPU-friendly dense tensor operations. This allows us to implement the entire method in PyTorch [39], which provides seamless GPU acceleration.

In summary, we introduce FASTMAP, a new SfM framework that is fully tensor-oriented and GPU-friendly. Our implementation, which we will release publicly as an open-source tool, achieves multi-camera pose estimation accuracy on par with state-of-the-art methods, while being more than an order of magnitude faster on large scenes.

2. Related Work

Global SfM Systems *Incremental* SfM methods like COLMAP [45] are state-of-the-art in accuracy and robustness, but *global* SfM systems are catching up [37]. These methods solve for all camera poses at once to avoid registering images sequentially, dramatically improving run time.

OpenMVG [34] and Theia [48] are two popular global SfM systems, using global rotation and translation averaging.They are fast, but trail COLMAP in accuracy and robustness [37]. HSfM [8] is a hybrid approach that combines incremental and global approaches, estimating rotations globally and translations incrementally.

Unlike prior global approaches that first perform translation averaging and then triangulation, GLOMAP [37] combines these steps, solving for camera translations and 3D points in one global step. They report results on par with COLMAP, but with large speed improvements both on small [46] and large-scale [44] benchmark.

Global Rotation Averaging [17] employs non-linear optimization to align global poses with local relative poses. Govindu [14] frames motion estimation as a global optimization problem, and Martinec and Pajdla [32] proposes to first solve for camera rotations using pairwise constraints and then obtain translations from a linear system using epipolar constraints. Wilson and Bindel [54] propose a more stable optimizer. In contrast, FASTMAP takes a GPU-friendly approach, obtaining global rotations through gradient descent on the geodesic distance between global and relative rotations after initialization [32].

Global Translation Estimation that comes after rotation averaging is challenging due to noisy measurements and the inherent lack of scale in relative translations. Many existing approaches struggle when baseline lengths differ significantly [37]. LUD [36] and Zhuang et al. [60] focus on improving stability and robustness in ill-conditioned scenarios. Jiang et al. [19] introduces a linearized approach that enforces constraints across camera triplets, ensuring consistency in multi-view configurations. Wilson and Snavely [55] propose to improve translation estimation via outlier removal alongside a simplified solver. Compared to prior works, FASTMAP (ours) adopts a much simpler design and optimizes global translation (camera locations) directly from estimated relative translations by gradient descent on normalized relative translation errors from random initialization, yielding effective and robust translation estimates.

End-to-end SfM Learning-based SfM methods vary in the degree of their departure from the traditional SfM pipeline and in their tradeoff between speed and accuracy. VGGSfM [52] hews closely to the traditional SfM methodology, building on point-tracking methods to propose a fully differentiable SfM method that includes bundle adjustment. Flowmap [47] uses pretrained optical flow and point tracking networks and a depth CNN to optimize per-scene global

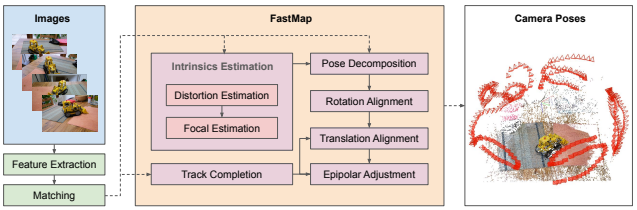

Figure 2. An overview of FASTMAP. Input images are processed using feature extraction and matching. Given the matching results, FASTMAP estimates the intrinsics and extrinsics of the cameras. Finally a sparse point cloud is generated by triangulation.

poses, calibration, and depth maps using gradient descent. Ace-Zero [7] uses a trained dense 3D scene coordinate regressor as an alternative to triangulation and registration in incremental SfM, instead incrementally relocalizing with the learned coordinate regressor.

The DUSt3R [53] architecture, which maps image pairs to *point maps*, initiated a new paradigm in learned SfM. DUSt3R pointmap estimates from image pairs can be used for camera calibration, depth estimation, correspondence, pose estimation and dense reconstruction. MASt3R [26] upgrades DUSt3R using dense correspondences from the predicted DUSt3R pointmap pairs; MASt3R-SfM [10] incorporates a global alignment stage, developing a full-fledged SfM system based on DUSt3R pointmap estimation. Fast3R [57] expands the DUSt3R paradigm using all-to-all attention to map arbitrary numbers of views to a global reconstruction and global camera poses, solving SfM in a feedforward pass over arbitrary image collections. FLARE [58] is a feedforward pointmap-based method that utilizes a coarse pose predictor to initialize camera geometry, forgoing DUSt3R-style post-training global optimization to improve performance. Matrix3D [30] uses a multimodal diffusion transformer backbone to generate novel views, camera pose, and depth maps for arbitrary images.

While learned methods (especially feedfoward methods) exhibit better speed—especially in settings where the number of images is relatively small and only approximate accuracy is desired—they still struggle when applied to hundreds to thousands of images and to the extent that they can ingest that amount of data, results become worse. Like these feedforward methods, FASTMAP is GPU-friendly and does not require second-order optimization, however it obtains better reconstructions, competitive with traditional SfM.

3. Method

Method overview The proposed FASTMAP (Fig. 2) consists of multiple stages roughly in sequential order. After extracting and matching keypoints, FASTMAP first estimates the distortion parameters (Sec. 3.1.1) and focal lengths (Sec. 3.1.2) for each camera. Using the estimated intrinsics, FASTMAP extracts the *relative* rotation and translation for each image pair, and performs *global* rotation alignment based on the relative rotations (Sec. 3.2). Using *track completion* (Sec. 3.3), FASTMAP generates new matched point pairs and image pairs, and proceeds to global translation alignment (Sec. 3.4). Finally, FASTMAP performs *re-weighting epipolar adjustment* (Sec. 3.5) to jointly refine the focal lengths and camera poses. This final step replaces the traditional bundle adjustment commonly used in previous work.

Matching FASTMAP's matching stage is exactly identical to that used in both COLMAP and GLOMAP: extracting and matching keypoints from the input images, followed by geometric verification of the resulting image pairs [45]. The output of the matching stage consists of the set of inlier keypoint pairs and either an estimated fundamental matrix $\mathbf{F}_{ij}$ or a homography matrix $\mathbf{H}_{ij}$ (the latter if it is consistent with sufficiently many inlier matches) for each image pair (i, j) with enough correspondences.

Camera assumptions We use a one-parameter radial distortion model (Sec. 3.1.1). The principal point is fixed to be the center, therefore the only intrinsics parameters to estimate are the distortion parameter and the focal length. We also assume that all the images are taken with a small number of cameras, and we know which images are from the same camera. In practice this can be inferred from image resolutions, EXIF tags, file names and directories, etc.

3.1. Intrinsics Estimation

FASTMAP assumes that images are generated according to a pinhole camera model parameterized by a 3×3 matrix $\mathbf{K}$, followed by radial distortion. We undistort keypoints using an estimate of the camera distortion (Sec. 3.1.1) and then estimate the focal length (Sec. 3.1.2) based on the essential matrices.

3.1.1. Distortion Estimation

We formulate distortion estimation as the problem of finding the distortion parameters that result in the most consistent two-view geometric model for each image pair (e.g., the fundamental matrix estimated from undistorted keypoints has the lowest epipolar error). We do so using the one-parameter division undistortion model [4, 11, 13, 18]

$$x_u = \frac{x_d}{1 + \alpha r_d^2} \qquad y_u = \frac{y_d}{1 + \alpha r_d^2}, \tag{1}$$

where (x_d, y_d) and (x_u, y_u) are the distorted and undistorted coordinates respectively, $r_d = \sqrt{x_d^2 + y_d^2}$ and $r_u = \sqrt{x_u^2 + y_u^2}$, and α is the distortion model parameter. The model can be inverted in closed form to apply distortion to keypoints [11]. We found this model to be more convenient than the commonly used but difficult to invert Brown-Conrady distortion model [9].

We use brute-force interval search to estimate the distortion parameter α. Given a set of image pairs that share the

same α, we sample set of candidate values from an interval $[\alpha_{\text{min}}, \alpha_{\text{max}}]$, and evaluate the average epipolar errors for each candidate after undistorting and re-estimating the fundamental matrices based on the sampled α (we ignore all the homography matrices at this stage). This method directly minimizes our objective (epipolar error) and takes into account information from multiple different image pairs, improving robustness to noise. Moreover, each candidate can be evaluated independently, making it highly parallelizable on a GPU. In the supplementary material, we provide more details regarding how to accelerate the above method with hierarchical sampling, and discuss generalizations to images with different distortion parameters.

3.1.2. Focal Length Estimation

We use the estimated distortion model to undistort all the keypoints, and re-estimate the fundamental and homography matrices. At this point, the only remaining unknown intrinsic parameter for each camera is the focal length. We estimate the focal length based on the re-estimated fundamental matrices from undistorted keypoints. While this is a well studied problem [3, 15, 20, 24, 49]. existing methods are susceptible to noise or require nonlinear optimization. Instead, we employ an interval search strategy similar to distortion estimation, but with a different objective.

Our method is based on the well-known property that a 3×3 matrix is an essential matrix if and only if its singular values are such that $\lambda_1 = \lambda_2$ and $\lambda_3 = 0$ ($\lambda_1 \geq \lambda_2 \geq \lambda_3$) [12, 16]. Given the correct fundamental matrix $\mathbf{F}$ and intrinsics matrix $\mathbf{K}$, the essential matrix $\mathbf{E} = \mathbf{K}^\top \mathbf{F} \mathbf{K}$ should satisfy $\frac{\lambda_1}{\lambda_2} = 1$. If all images share the same intrinsics, with a set of fundamental matrices $\{\mathbf{F}_i\}$, we can evaluate the accuracy of a candidate focal length f using the singular value ratio above. Letting $\lambda_1^{(i)} \geq \lambda_2^{(i)} \geq \lambda_3^{(i)}$ be the singular values of $\mathbf{K}^\top \mathbf{F}_i \mathbf{K}$, where $\mathbf{K}$ is a function of f, we can measure the validity of f as

$$v = \sum_i \exp\left(\frac{1 - \lambda_1^{(i)}/\lambda_2^{(i)}}{\tau}\right), \tag{2}$$

where τ is a temperature hyper-parameter. Intuitively, the above formula is close to one when $\lambda_1^{(i)}/\lambda_2^{(i)}$ is close to one, and decreases exponentially as $\lambda_1^{(i)}/\lambda_2^{(i)}$ increases.

We sample a set of focal length candidates and evaluate them using Eqn. 2. The candidate with the highest value (2) is chosen as the final estimate. This method can be easily generalized to deal with images with different focal lengths (see supplementary for details). After estimating the focal length, we transform the keypoints, fundamental matrices, and homography matrices using the estimated intrinsic matrices so that all components are calibrated.

3.2. Global Rotation

With estimated intrinsics, we can decompose the fundamental and homography matrices into relative rotations and translations [16, 31]. Given the set of image pairs $\mathcal{P} = \{(i,j)\}$ and the corresponding relative rotation matrices $\{\mathbf{R}^{i\to j}\}_{(i,j)\in\mathcal{P}}$, FastMap next estimates the world-to-camera global rotation $\mathbf{R}^{(i)}$ matrix for each image i. We formulate this as an optimization of a loss defined over all image pairs $\mathcal{P} = \{(i,j)\}$

$$\mathcal{L}_{\text{R}} = \frac{1}{|\mathcal{P}|} \sum_{(i,j)\in\mathcal{P}} d(\mathbf{R}^{(j)}, \mathbf{R}^{i\to j}\mathbf{R}^{(i)}), \tag{3}$$

where $d(\cdot,\cdot)$ is the geodesic distance between rotations

$$d(\mathbf{R}, \mathbf{R}') = \cos^{-1}\left(\frac{\text{Tr}(\mathbf{R}^T\mathbf{R}') - 1}{2}\right). \tag{4}$$

We parameterize the global rotation matrices $\mathbf{R}_i$ using a differentiable 6D representation [59], and optimize Eqn. 3 via gradient descent.

Unfortunately, directly optimizing the above objective from random initialization of $\mathbf{R}_i$ is prone to local minima. We use a slightly modified version of the method proposed by Martinec and Pajdla [32] to obtain a good initialization. The basic idea of the method is that although the column vectors in a rotation matrix are constrained by orthogonality, each column vector alone is only subject to a unit length constraint. If we consider one column at a time, we can formulate the optimization as as a least squares problem and solve it using SVD. The details of this initialization scheme is described in the supplementary.

3.3. Track Completion

Consider the graph in which 2D keypoints are nodes and pairwise edges denote keypoint matches. When a 3D scene point is observed in m different images, the projected 2D keypoints should ideally form a complete subgraph with m vertices. In practice, keypoint matching has low recall and tends to miss many point-pairs. A subgraph of related keypoints is often far from fully connected. Since the number (and quality) of matches is critical to the accuracy of pose estimation, we make up for low matching recall by *track completion*. A *track* is a connected component in the 2D keypoint connectivity graph and implies the existence of a shared 3D point. Tracks are used heavily in SfM [37, 45] to impose extra constraints, e.g., bundle adjustment [50] initializes 3D points based on tracks and minimizes the reprojection error of each 3D point with its track members.

FastMap avoids bundle adjustment and data structures containing both 3D scene points and 2D keypoints. Instead we explicitly convert tracks to additional matches with pairwise combinations of all keypoints in each track. This way

we still make use of the transitivity of matching and benefit from the extra constraints. These additional point pairs are only introduced after global rotation alignment, and are used in the global translation alignment and epipolar adjustment steps described below.

3.4. Global Translation

We now move to the step of estimating the 3D coordinates of the camera centers in a common (world) coordinate frame, up to a similarity transformation.

3.4.1. Relative Translation

Global translation alignment in our method relies heavily on relative translations between image pairs. Rather than using the translations determined via pose decomposition (Sec. 3.2), we first re-estimate the relative translations. We do so for two reasons. First, since we have estimated the global rotations, we can go back and re-compute the relative rotation of any image pair. The relative rotation computed in this way is much more accurate than those from relative pose decomposition. In turn, a better estimate of the relative rotation enables us to more accurately estimate relative translation. Second, after generating new point pairs from tracks, some image pairs that originally had no matches might have some now, and we can use these new point pairs to estimate the relative translation. We use 2D grid search to re-estimate the unit relative translation vectors. We first sample a set of candidates on the surface of the unit sphere, evaluate the mean epipolar error of each sample, and choose the candidate with the lowest error as the final estimate.

3.4.2. Global Translation Alignment

Given world-to-camera rotations $\{\mathbf{R}_i\}_{1\leq i\leq N}$ for N images and unit-length relative translations $\{\boldsymbol{t}^{i\to j}\}_{(i,j)\in\mathcal{P}}$ for a set of image pairs $\mathcal{P}$, we compute the normalized vector from the camera centers of image i to j in world coordinates

$$\boldsymbol{o}^{i\to j} = -\mathbf{R}_j^\top \boldsymbol{t}^{i\to j}. \tag{5}$$

We estimate the camera locations $\{\boldsymbol{o}_i\}_{1\leq i\leq N}$ in the world frame by minimizing the error between the normalized relative translation $\frac{\boldsymbol{o}_j-\boldsymbol{o}_i}{\|\boldsymbol{o}_j-\boldsymbol{o}_i\|_2}$ and the target $\boldsymbol{o}^{i\to j}$ above with gradient descent

$$\mathcal{L}_\text{t} = \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}}\left\|\frac{\boldsymbol{o}_j-\boldsymbol{o}_i}{\|\boldsymbol{o}_j-\boldsymbol{o}_i\|_2} - \boldsymbol{o}^{i\to j}\right\|_1 \tag{6}$$

Unlike global rotation optimization, optimizing this objective enables us to obtain accurate results using randomly initialized parameters. A similar observation is made in GLOMAP [37]; they optimize poses and 3D points jointly, and it is much more computationally expensive.

3.4.3. Multiple Initializations

Although random initialization works surprisingly well for the objective in Eqn. 6, it occasionally produces a small number of outliers. To deal with the problem, we propose to do multiple independent runs from different random initializations, and merge the solutions as the initialization for the final optimization loop. Since the global rotations are the same, different solutions can be aligned by moving the centroid to the origin and rescaling uniformly to have unit average norm. Then for each image the solution with the lowest average loss is chosen to be in the merged result.

3.5. Epipolar Adjustment

A typical SfM pipeline relies on *bundle adjustment* [50], the process of jointly refining the camera poses and inferred 3D points. Standard approaches to bundle adjustment are expensive due to a computational complexity that depends on the number of 3D points, which quantity often far exceeds the number of image pairs. Instead, we refine the poses from previous stages using *re-weighting epipolar adjustment*, an optimization method we introduce for which the computational complexity in each iteration is linear in the number of image pairs, not in the number of points.

In relative translation re-estimation, we obtain a set of image pairs with number of inliers above some threshold. We denote the set of such image pairs as $\mathcal{P} = \{(i_n, j_n)\}_{1\leq n\leq|\mathcal{P}|}$, where i_n and j_n are the indices of the first and second images in the pair. For an image pair $(i_n, j_n) \in \mathcal{P}$, we represent the set of point pairs as $\mathcal{Q}_n = \{(\boldsymbol{x}_{nm}^{(1)}, \boldsymbol{x}_{nm}^{(2)}) \in \mathbb{R}^2\times\mathbb{R}^2\}_{1\leq m\leq|\mathcal{Q}_n|}$, and let $\tilde{\mathcal{Q}}_n = \{(\tilde{\boldsymbol{x}}_{nm}^{(1)}, \tilde{\boldsymbol{x}}_{nm}^{(2)}) \in \mathbb{P}^2\times\mathbb{P}^2\}_{1\leq m\leq|\tilde{\mathcal{Q}}_n|}$ be the set of point pairs in normalized homogeneous coordinates.

Using estimated initializations from the previous stages, we optimize over the world-to-camera global rotations and translations to minimize the absolute epipolar error

$$\mathcal{L}_\text{e} = \frac{1}{Z}\sum_{n=1}^{|\mathcal{P}|}\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}|\tilde{\boldsymbol{x}}_{nm}^{(2)\top}\mathbf{E}_n\tilde{\boldsymbol{x}}_{nm}^{(1)}|, \tag{7}$$

where $Z = \sum_{n=1}^{|\mathcal{P}|}|\tilde{\mathcal{Q}}_n|$ is the total number of point pairs, and $\mathbf{E}_n$ is the essential matrix computed from the global rotations and translations for images i_n and j_n.

Evaluating Eqn. 7 for every iteration is expensive because it involves every point pair. However, if we replace the cost terms with L2 loss (as in Rodriguez et al. [41]), the overall objective can be re-organized to aggregate terms involving point pairs shared by the same image pair into a compact quadratic form (see the supplementary)

$$\mathcal{L}_\text{e} = \frac{1}{Z}\sum_{n=1}^{|\mathcal{P}|}\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}(\tilde{\boldsymbol{x}}_{nm}^{(2)\top}\mathbf{E}_n\tilde{\boldsymbol{x}}_{nm}^{(1)})^2 = \frac{2}{Z}\sum_{n=1}^{|\mathcal{P}|}\boldsymbol{e}_n^\top\mathbf{W}_n\boldsymbol{e}_n \tag{8}$$

where $\boldsymbol{e}_n = \text{flatten}(\mathbf{E}_n) \in \mathbb{R}^9$, and $\mathbf{W}_n \in \mathbb{R}^{9\times 9}$ is a matrix computed from all the point pairs in $\tilde{\mathcal{Q}}_n$. Note that only $\boldsymbol{e}_n$ is a function of the parameters to be optimized. The matrices $\mathbf{W}_n$ can be pre-computed for each image pair before optimization. With the precomputed $\mathbf{W}_n$, the cost of each optimization step is linear in the number of image pairs. A similar observation was made by Rodríguez et al. [42].

The expedience of Eqn. 8 comes with the side effect that the L2 loss is sensitive to outliers. To make the loss function robust but still preserve the benefit of pre-computation, we propose to use *iterative re-weighted least squares (IRLS)*. The intuition is that if we have an initialization close to the optimum, the L1 loss, which is more robust to outliers, can be approximated by a weighted L2 loss. In other words, for some differentiable computed scalar value z, we have $z^2 \approx z^2/|\hat{z}|$, where $\hat{z}$ is the value of z at initialization. In our case, after global translation alignment, we already have a good initialization of global poses, so we can compute the absolute epipolar error $|\hat{\epsilon}_{nm}|$ for each point pair, and use them to weight the L2 loss used above to get an approximate robust L1 loss

$$\hat{\mathcal{L}}_{\text{e}} = \frac{1}{Z}\sum_{n=1}^{|\mathcal{P}|}\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}\frac{(\tilde{\boldsymbol{x}}_{nm}^{(2)\top}\mathbf{E}_n\tilde{\boldsymbol{x}}_{nm}^{(1)})^2}{|\hat{\epsilon}_{nm}|} = \frac{2}{Z}\sum_{n=1}^{|\mathcal{P}|}\boldsymbol{e}_n^\top\hat{\mathbf{W}}_n\boldsymbol{e}_n, \tag{9}$$

where $\hat{\mathbf{W}}_n$ is similar to $\mathbf{W}_n$ in Eqn. 8, but computed from $\tilde{\mathcal{Q}}_n$ weighted by $|\hat{\epsilon}_{nm}|$.

After a round of optimization, we can better approximate the L1 loss by re-computing the weights and then do another optimization loop. To further reduce the impact of outliers, we periodically filter out point pairs with large epipolar error. We start from a relatively large initial threshold and gradually decrease it to a pre-determined minimum.

The above optimization problem only involves camera poses. We can also optimize the focal lengths by incorporating them into the computation of the essential matrices (in this case, the results are actually fundamental matrices).

4. Experiments

We compare FastMap against COLMAP [45] (commit `cf8f116`) and GLOMAP [37] (commit `18a70ee`) in terms of speed, pose accuracy, and the effect of pose estimation on novel view synthesis.

Implementation Details We implement FastMap entirely in PyTorch [39]. For all three methods we use the COLMAP matching algorithm. We enable shared intrinsics for COLMAP and GLOMAP if all images in a scene are from the same camera. All other hyper-parameters are set to their default values. We run COLMAP and GLOMAP on an AMD EPYC 9274F CPU (Zen 4) with 24 cores / 48 threads, and FastMap on a single A6000 (Ampere) GPU with 1 CPU core.

We use 3 levels of hierarchical sampling for distortion estimation (see supplementary), and each level has 10 samples. For focal length estimation, the search interval of FoV is $[20°, 160°]$, number of samples is 100, and the temperature τ in Eqn. 2 is set to 0.01. We use the Adam [22] optimizer for all the optimization problems in our method. The learning rate for rotation alignment is set to 1×10^{-4}, and translation alignment 1×10^{-3}. We use 3 initializations for translation alignment. For epipolar adjustment, the initial learning rate is 1×10^{-4}, and is decayed by a factor of 2 every time we prune outlier point pairs. We do 3 rounds of outlier pruning with a decreasing epipolar error threshold from 0.01 to 0.005. Between any two pruning steps, we do 3 iterations of re-weighting for IRLS.

4.1. Datasets

We experiment with eight datasets: MipNeRF360 [5], Tanks and Temples [23], ZipNeRF [6], NeRF-OSR [43], DroneDeploy [40], Mill-19 [51], Urbanscene3D [27], and Eyeful Tower [56]. Each dataset contains several scenes, covering a wide range of real-world scenarios and camera trajectory patterns. The number of images per scene ranges from around 200 to 6000.

With the exception of Tanks and Temples, each of these datasets includes author-provided reference camera poses. These reference poses are obtained through different means, including COLMAP (for MipNeRF360, ZipNeRF, NeRF-OSR), PixSfM [28] (for Mill-19 and Urbanscene3D), and commercial software (for DroneDeploy and Eyeful Tower). In the case of Urbanscene3D, we use the poses provided by Turki et al. [51]. For Tanks and Temples, we use COLMAP poses provided by Kulhanek and Sattler [25]. On one of the scenes from Tanks and Temples (Courthouse), we found that the reference poses are obviously inconsistent with the images, but decided to still treat them as ground-truth.

4.2. Metrics

We report wall-clock time in seconds, excluding the time required for feature extraction and matching (identical for all three methods and dominated by the SfM backend time for large scenes). We evaluate pose accuracy using the standard metrics [10, 37, 45]: ATE, RRA@δ, RTA@δ, and AUC@δ. ATE measures the accuracy of the overall predicted camera trajectory. RRA@δ and RTA@δ are the percentage of pairwise relative rotation and translation (derived from the estimated global poses) whose angular errors are below δ degrees. AUC@δ computes the the area under the error-recall curve up to the threshold δ, and then divides it by δ. Unlike RRA and RTA, which ignore the differnce once the error falls below the threshold, AUC@δ puts a stronger emphasis on high precision angle accuracy. Even with all pairwise relative motion angle errors less than δ, AUC@δ can still be very low.

		time (sec)			ATE↓			RTA@3↑			AUC-R&T @ 3 ↑			RTA@1↑			AUC-R&T @ 1 ↑		
	n_imgs	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP
mipnerf360 (9)	215.6	128	299	570	5.0e-4	3.3e-5	7.0e-4	99.8	100.0	99.9	96.9	98.2	98.1	99.5	100.0	99.9	91.4	94.6	94.4
tnt_advanced (6)	337.8	316	567	915	6.8e-3	1.2e-2	1.4e-3	62.4	78.7	98.7	33.1	75.2	95.9	32.9	77.5	97.7	10.5	70.1	91.0
tnt_intermediate (8)	268.6	116	598	687	9.2e-5	1.9e-5	2.3e-5	99.9	100.0	100.0	92.3	99.0	99.1	99.2	99.9	99.7	77.7	97.0	97.4
tnt_training (7)	470.1	323	941	2555	3.2e-3	1.1e-2	4.2e-4	87.3	88.7	99.8	74.9	87.8	99.3	79.4	88.6	99.8	56.7	86.3	98.5
nerf_osr (8)	402.8	178	672	2357	1.3e-3	1.1e-3	1.3e-3	92.0	92.0	92.1	70.9	71.9	71.7	71.5	71.9	71.6	42.3	45.2	44.6
drone_deploy (9)	524.7	376	942	4352	3.7e-3	4.4e-3	2.0e-3	98.4	98.2	91.2	79.7	81.1	65.8	89.8	91.5	74.1	50.4	53.5	40.7
zipnerf (4)	1527.2	589	2155	17628	5.3e-4	3.9e-3	3.4e-4	99.1	98.1	98.9	92.2	96.6	88.2	97.1	98.0	94.5	80.2	93.7	85.3
urban_scene (3)	3824.0	2096	20256	292734	1.8e-5	1.4e-5	1.4e-5	99.9	99.9	100.0	94.8	97.0	97.0	99.6	99.6	99.6	84.8	91.2	91.3
mill19_building	1920	1366	38428	152839	2.0e-5	1.3e-2	5.1e-3	100.0	0.1	81.7	95.5	0.0	76.9	99.5	0.0	80.7	87.0	0.0	68.5
mill19_rubble	1657	789	11571	64987	3.6e-5	8.0e-5	3.4e-5	99.9	99.8	99.9	93.5	94.5	94.6	98.7	98.6	98.7	81.5	84.7	84.8
eyeful_apartment	3804	1003	124310	> 7d	3.6e-3	9.4e-3	-	86.3	77.7	-	45.9	53.1	-	50.9	64.5	-	6.4	20.2	-
eyeful_kitchen	6042	6796	> 7d	> 7d	3.0e-3	-	-	83.6	-	-	37.6	-	-	45.7	-	-	4.4	-	-

Table 1. Speed and pose accuracy of FastMap, GLOMAP, and COLMAP. For datasets with more than two scenes, we denote the average metrics as `dataset-name(#scenes)`. In particular, Tanks and Temples [23] has three official splits, and we do the averaging separately for them. Mill-19 [51] and Eyeful Tower [56] scenes are listed separately. Metrics are color-coded in green, with color changes if the percentage gap >2% or ATE ratio >1.5. Red denotes complete failures and gray means the method did not finish in a week. Note the significant speedup of FastMap vs. previous work, especially on larger scenes.

		Absolute PSNR ↑			Relative to COLMAP	
		FastMap	GLOMAP	COLMAP	FastMap	GLOMAP
m360_bicycle	Zip-NeRF	25.60	25.78	25.86	-0.26	-0.08
	+ CamP	26.21	26.36	26.41	-0.21	-0.05
	GSplat	25.51	25.59	25.62	-0.11	-0.03
m360_bonsai	Zip-NeRF	34.78	34.91	34.47	0.31	0.44
	+ CamP	35.26	35.32	35.37	-0.12	-0.05
	GSplat	32.32	32.29	31.49	0.84	0.81
m360_counter	Zip-NeRF	28.97	28.95	29.18	-0.21	-0.23
	+ CamP	29.09	29.18	29.29	-0.20	-0.12
	GSplat	28.99	29.06	29.02	-0.02	0.04
m360_flowers	Zip-NeRF	22.05	22.29	21.89	0.15	0.40
	+ CamP	23.53	23.47	23.27	0.25	0.20
	GSplat	21.74	21.79	21.59	0.15	0.20
m360_garden	Zip-NeRF	28.10	28.20	28.20	-0.11	0.00
	+ CamP	28.54	28.49	28.54	0.00	-0.05
	GSplat	27.61	27.67	27.72	-0.11	-0.05
m360_kitchen	Zip-NeRF	32.29	32.43	32.31	-0.02	0.12
	+ CamP	32.47	32.19	32.21	0.27	-0.02
	GSplat	31.36	31.62	31.58	-0.22	0.05
m360_room	Zip-NeRF	32.81	32.94	32.93	-0.12	0.01
	+ CamP	32.51	32.48	32.44	0.07	0.04
	GSplat	31.77	31.71	31.67	0.11	0.04
m360_stump	Zip-NeRF	27.34	27.41	27.43	-0.09	-0.02
	+ CamP	28.10	28.03	28.03	0.07	0.00
	GSplat	26.97	26.89	26.84	0.13	0.05
m360_treehill	Zip-NeRF	23.73	24.05	24.04	-0.31	0.01
	+ CamP	25.73	25.74	25.99	-0.26	-0.25
	GSplat	22.71	22.88	22.80	-0.08	0.08
tnt_training (7)	InstantNGP	20.73	19.37	21.05	-0.32	-1.68
	GSplat	23.22	21.54	24.19	-0.97	-2.65
tnt_intermediate (8)	InstantNGP	22.29	22.51	22.38	-0.09	0.13
	GSplat	24.24	25.28	25.24	-1.00	0.03
tnt_advanced (6)	InstantNGP	16.59	16.94	17.55	-0.95	-0.60
	GSplat	18.82	18.74	22.06	-3.24	-3.32

Table 2. Novel view synthesis evaluation on MipNeRF360 [5] and Tanks and Temples [23]. Results for MipNeRF360 are listed separately for each scene, and those for Tanks and Temples are averaged over all scenes in each of the three splits. The color changes only if the PSNR difference >0.25. We report results for Zip-NeRF, Zip-NeRF + CamP optimization, and Gaussian Splatting.

	tnt_training (7)			tnt_intermediate (8)			tnt_advanced (6)		
	ATE	RTA@5	RRA@5	ATE	RTA@5	RRA@5	ATE	RTA@5	RRA@5
ACE-Zero [7]	1.2e-2	72.9	73.9	8.0e-3	74.0	67.5	2.8e-2	19.1	22.9
MAST3R-SfM [10]	6.2e-3	64.9	56.2	7.2e-3	57.5	50.8	2.0e-2	36.5	38.8
GLOMAP	1.1e-2	88.8	89.3	1.9e-5	100.0	100.0	1.2e-2	79.3	80.5
FastMap	3.2e-3	88.8	95.8	9.2e-5	100.0	100.0	6.8e-3	70.5	82.1

Table 3. Comparison against learning-based SfM on Tanks and Temples [23]. We use the COLMAP poses provided by [25] as reference. Numbers are averaged for scenes in each split.

One of the most common uses of SfM today is estimating camera poses for neural rendering methods like NeRF [33] and Gaussian Splatting [21]. For some of the datasets, we also evaluate the novel view synthesis quality trained on the output poses, intrinsics and triangulated point clouds.

4.3. Analysis

Pose accuracy Table 1 compares the three methods in average camera pose metrics on all the datasets (we include per-scene evaluation results in the supplementary). In general, our method is much faster than both GLOMAP and COLMAP. The speedup over GLOMAP is less dramatic when there are only a few hundred images (e.g., MipNeRF360), but it can be an order-of-magnitude or as much as two orders-of-magnitude faster on scenes with several thousand images (e.g., Urbanscene3D, Mill-19, and Eyeful Tower). On most datasets, FastMap is on par with GLOMAP and COLMAP in terms of RTA@3. There is a more prominent difference for stricter metrics (RTA@1, AUC@3, AUC@1). It shows that while FastMap succeeds in recovering the overall structures of camera trajectories, it does not squeeze the last bit of precision when the error goes down to one or two degrees.

None of the methods are perfect. FastMap performs particularly bad on the Advanced split of Tanks and Temples, probably because there are many erroneous matches due to repetitive patterns and symmetric structures in the scenes. This is a well-known problem of global SfM (i.e., GLOMAP also suffers a significant drop in performance compared to COLMAP), and incremental SfM methods like

COLMAP are more robust in these settings. On the building scene of the Mill-19 dataset, both COLMAP and GLOMAP fail catastrophically, however FASTMAP remains highly accurate. On DroneDeploy, none of the three methods are very good in terms of AUC@1 and RTA@1.

In Tab. 3, we compare FASTMAP to two representative learning-based methods, ACE-Zero [7] and MAST3R-SfM [10]. The results are quoted from MAST3R-SfM appendix Tab. 10. Both methods perform significantly worse than ours and GLOMAP. This indicates that while learning-based methods are promising, they are still far behind traditional methods in pose accuracy.

Novel view synthesis Table 2 evalutes the quality of novel view synthesis on the MipNeRF360 and Tanks and Temples datasets when using FASTMAP, COLMAP, and GLOMAP to estimate the camera poses. For MipNeRF360 dataset, we use ZipNeRF [6], a very high-quality NeRF method, and for Tank and Temples we use Instant-NGP [35], which has a better trade-off between quality and speed. While FASTMAP lags behind GLOMAP and COLMAP on most MipNeRF360 scenes, the PSNR difference is within 0.5. On Tanks and Temples, FASTMAP performs on par with GLOMAP, but both are worse than COLMAP. Here, again, a lower pose accuracy of FASTMAP under the strictest metrics does not prevent FASTMAP poses from yielding competitive PSNR. These results suggest that pose accuracy under a strict metric could be a misleading proxy for downstream view synthesis quality, and vice versa.

We also investigate the impact of different SfM poses on rendering with CamP [38], that simultaneously optimizes the radiance field and refines the camera poses. We include the results in Tab. 2 for comparison. In general, CamP improves the PSNR for all the three methods, and for some scenes (e.g., flowers, garden, kitchen, etc.) the gap in rendering quality is closed and sometimes even reversed.

FASTMAP does not explicitly materialize a 3D point cloud during pose estimation, but downstream applications such as Gaussian splatting [21] often require high quality 3D scene points as initialization. We describe in Appendix B.5 a simple post-processing triangulation step, and assess the quality of our sparse point cloud by Gaussian Splatting (GSplat). GSplat's rasterization routine demands all images to be formed by pinhole cameras. We undistort the images using our estimated parameters in Sec. 3.1.1. The results in Table 2 indicate a comparable point cloud quality to COLMAP and GLOMAP.

4.4. Ablations

Distortion estimation is the first step of FASTMAP, and its accuracy is critical to the performance. Table 4 presents the performance of FASTMAP with and without distortion estimation on the Intermediate split of Tanks and Temples. Without distortion estimation results drop, for some scenes catastrophically. We provide in the supplementary an additional insight into the effect of distortion estimate on the immediately following step of focal length estimation.

	AUC@3		AUC@10		RTA@3		RTA@10	
	w/	w/o	w/	w/o	w/	w/o	w/	w/o
Family	**95.1**	72.8	**98.5**	91.8	**100.0**	99.9	**100.0**	100.0
Francis	**95.5**	71.1	**98.6**	91.2	**99.9**	99.6	**100.0**	99.9
Horse	**96.8**	76.8	**99.0**	93.0	**100.0**	100.0	**100.0**	100.0
Lighthouse	**90.7**	4.6	**97.1**	42.2	**99.6**	46.5	**100.0**	98.5
M60	**95.6**	28.3	**98.7**	72.9	**99.9**	85.9	**100.0**	99.7
Panther	**93.0**	12.1	**97.9**	64.2	**99.9**	78.3	**100.0**	99.9
Playground	**84.6**	2.2	**95.4**	14.4	**100.0**	15.2	**100.0**	51.9
Train	**92.4**	54.2	**97.7**	86.0	**99.9**	99.6	**100.0**	99.9

Table 4. Effect of camera distortion estimation on pose accuracy.

		$m=1$	$m=2$	$m=3$	$m=4$	$m=8$
zipnerf_nyc	RTE@30	**0.59**	**0.09**	0.09	0.09	0.10
	RTA@3	98.59	99.45	99.45	99.45	99.45
zipnerf_alameda	RTE@30	**0.32**	**0.14**	0.13	0.09	0.09
	RTA@3	97.90	98.28	98.29	98.40	98.39
tnt_Train	RTE@30	**1.40**	**0.02**	0.02	0.02	0.29
	RTA@3	97.15	99.62	99.64	99.61	99.08
tnt_Lighthouse	RTE@30	**1.98**	**0.01**	0.01	0.02	0.01
	RTA@3	96.82	99.26	99.23	99.34	99.33
dploy_ruins3	RTE@30	**1.23**	**0.66**	0.67	0.92	0.69
	RTA@3	92.70	94.56	94.44	94.06	93.77
dploy_house4	RTE@30	**3.82**	**0.83**	1.00	0.83	0.83
	RTA@3	93.00	96.92	95.69	97.48	95.48

Table 5. Ablation of multiple translation initializations on selected scenes. The results are obtained right after translation alignment and before epipolar adjustment. We bold the RTE@30 entries for $m = 1$ and $m = 2$ initializations to highlight its effect.

		AUC@3	AUC@10	RTA@1	RTA@5	RRA@3
m360 (9)	FASTMAP	**97.2**	**99.1**	**99.8**	**100.0**	**100.0**
	w/o epipolar adjustment	75.0	90.8	85.5	99.5	94.5
	w/o track completion	80.4	86.4	83.3	91.4	83.6
alameda	FASTMAP	**95.2**	**98.1**	**99.0**	**99.3**	**99.9**
	w/o epipolar adjustment	86.7	95.5	94.9	99.2	99.8
	w/o track completion	94.8	**98.1**	**99.0**	99.4	**99.9**
berlin	FASTMAP	**81.6**	**93.2**	**92.8**	**99.2**	**97.5**
	w/o epipolar adjustment	70.4	89.5	82.4	98.8	95.7
	w/o track completion	60.4	81.3	70.4	90.8	90.3
london	FASTMAP	**96.6**	**98.8**	**99.6**	**99.9**	99.7
	w/o epipolar adjustment	90.1	96.6	97.8	99.7	99.3
	w/o track completion	96.1	98.7	**99.6**	**99.9**	**99.8**
nyc	FASTMAP	**94.6**	**98.1**	99.2	99.6	99.6
	w/o epipolar adjustment	89.7	96.7	97.0	99.7	99.6
	w/o track completion	93.9	98.0	**99.4**	**99.8**	**99.8**

Table 6. Epipolar adjustment and track completion ablation on the MipNeRF360 [5] and ZipNeRF [6] datasets. Results for MipNeRF360 are averaged over all the scenes, and for ZipNeRF each scene is listed separately.

Track completion In Table 6 we show the final performance of the FASTMAP with and without augmented point pairs from track completion (Sec. 3.3). Track completion significantly improves performance for MipNeRF360 scenes and some but not all ZipNeRF scenes.

Multiple translation initializations As shown in Table 5,

while a single initialization is prone to large-error outliers (see RTE@30 defined as 100.0 - RTA@30), increasing the number of initalizations improves performance. However, the effect plateaus with increased initializations and does not completely fix the outlier problem.

Epipolar adjustment Table 6 also presents the performance of FASTMAP with and without the final epipolar adjustment step. On all metrics, epipolar adjustment consistently improves over the poses from global translation alignment. The improvement is more prominent for stricter metrics (RTA@1), but less so for more tolerant metrics like RTA@5. This suggests that after translation alignment the cameras are already roughly in place, and epipolar adjustment continues to squeeze as much precision as it can.

5. Limitations and Conclusions

We introduce FASTMAP, a novel structure from motion method focused on simplicity and speed. FASTMAP is much faster than state-of-the-art methods (COLMAP and GLOMAP), while achieving comparable performance on pose accuracy and novel view synthesis quality. We attribute this significant speed gain to core algorithmic innovations that reduce the cost of each optimization step, and to an extensive use of GPU-friendly operations. These improvements do come with a few drawbacks. FASTMAP relies solely on fundamental matrices to estimate focal lengths, and might fail on scenes where most points lie on a single plane. FASTMAP is also more sensitive to incorrect matching induced by repetitive patterns and symmetric structures, when compared to GLOMAP. Nevertheless, we believe it is an important step towards highly efficient camera pose estimation and 3D reconstruction at scale. We will open-source our code and hope that it will benefit other researchers and practitioners.

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] Sameer Agarwal, Keir Mierle, and The Ceres Solver Team. Ceres Solver, 2023. 2

[3] Daniel Barath, Tekla Toth, and Levente Hajder. A minimal solution for two-view focal-length estimation using two affine correspondences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4

[4] João Pedro Barreto and Kostas Daniilidis. Fundamental matrix for cameras with radial distortion. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2005. 3

[5] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 6, 7, 8

[6] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-NeRF: Anti-aliased grid-based neural radiance fields. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023. 6, 8

[7] Eric Brachmann, Jamie Wynn, Shuai Chen, Tommaso Cavallari, Áron Monszpart, Daniyar Turmukhambetov, and Victor Adrian Prisacariu. Scene coordinate reconstruction: Posing of image collections via incremental learning of a relocalizer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 1, 3, 7, 8

[8] Hainan Cui, Xiang Gao, Shuhan Shen, and Zhanyi Hu. HSfM: Hybrid structure-from-motion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[9] C Brown Duane. Close-range camera calibration. *Photogramm. Eng*, 37(8), 1971. 3

[10] Bardienus Duisterhof, Lojze Zust, Philippe Weinzaepfel, Vincent Leroy, Yohann Cabon, and Jerome Revaud. MASt3R-SfM: A fully-integrated solution for unconstrained structure-from-motion. *arXiv preprint arXiv:2409.19152*, 2024. 1, 3, 6, 7, 8

[11] Bastian Erdnüß. A review of the one-parameter division undistortion model. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 1, 2021. 3

[12] Olivier Faugeras. *Three-dimensional computer vision: A geometric viewpoint*. MIT Press, 1993. 4

[13] Andrew W Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001. 3

[14] Venu Madhav Govindu. Combining two-view constraints for motion estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001. 1, 2

[15] Richard Hartley. Extraction of focal lengths from the fundamental matrix. *Unpublished manuscript*, 2, 1993. 4

[16] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2003. 4

[17] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *International Journal on Computer Vision*, 103, 2013. 1, 2

[18] Jose Henrique Brito, Roland Angst, Kevin Koser, and Marc Pollefeys. Radial distortion self-calibration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 3

[19] Nianjuan Jiang, Zhaopeng Cui, and Ping Tan. A global linear method for camera pose registration. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2013. 2

[20] Kenichi Kanatani and Chikara Matsunaga. Closed-form expression for focal lengths from the fundamental matrix. In *Proceedings of the Asian Conference on Computer Vision*, 2000. 4

[21] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 1, 7, 8

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[23] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4), 2017. 6, 7

[24] Viktor Kocur, Daniel Kyselica, and Zuzana Kukelova. Robust self-calibration of focal lengths from the fundamental matrix. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 4

[25] Jonas Kulhanek and Torsten Sattler. NeRFBaselines: Consistent and reproducible evaluation of novel view synthesis methods. *arXiv preprint arXiv:2406.17345*, 2024. 6, 7

[26] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with MASt3R. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 3

[27] Liqiang Lin, Yilin Liu, Yue Hu, Xingguang Yan, Ke Xie, and Hui Huang. Capturing, reconstructing, and simulating: the UrbanScene3D dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 6

[28] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-perfect structure-from-motion with featuremetric refinement. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 6

[29] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1150–1157, 1999. 2

[30] Yuanxun Lu, Jingyang Zhang, Tian Fang, Jean-Daniel Nahmias, Yanghai Tsin, Long Quan, Xun Cao, Yao Yao, and Shiwei Li. Matrix3d: Large photogrammetry model all-in-one. *arXiv preprint arXiv:2502.07685*, 2025. 3

[31] Ezio Malis and Manuel Vargas. *Deeper understanding of the homography decomposition for vision-based control*. PhD thesis, Inria, 2007. 4

[32] Daniel Martinec and Tomas Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. 2, 4, 12

[33] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1), 2021. 1, 7

[34] Pierre Moulon, Pascal Monasse, Romuald Perrot, and Renaud Marlet. OpenMVG: Open multiple view geometry. In *Proceedings of the International Workshop on Reproducible Research in Pattern Recognition*, 2017. 2

[35] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (TOG)*, 41(4), 2022. 8

[36] Onur Ozyesil and Amit Singer. Robust camera location estimation by convex programming. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2

[37] Linfei Pan, Daniel Barath, Marc Pollefeys, and Johannes Lutz Schönberger. Global structure-from-motion revisited. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 1, 2, 4, 5, 6

[38] Keunhong Park, Philipp Henzler, Ben Mildenhall, Jonathan T. Barron, and Ricardo Martin-Brualla. CamP: Camera preconditioning for neural radiance fields. *ACM Transactions on Graphics*, 2023. 8

[39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2, 6

[40] Nicholas Pilkington. DroneDeploy NeRF dataset. https://github.com/nickponline/dd-nerf-dataset, 2022. 6

[41] Antonio L Rodriguez, Pedro E López-de Teruel, and Alberto Ruiz. GEA optimization for live structureless motion estimation. In *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 2011. 5

[42] Antonio L Rodríguez, Pedro E López-de Teruel, and Alberto Ruiz. Reduced epipolar cost for accelerated incremental sfm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 6

[43] Viktor Rudnev, Mohamed Elgharib, William Smith, Lingjie Liu, Vladislav Golyanik, and Christian Theobalt. Nerf for outdoor scene relighting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 6

[44] Paul-Edouard Sarlin, Mihai Dusmanu, Johannes L Schönberger, Pablo Speciale, Lukas Gruber, Viktor Larsson, Ondrej Miksik, and Marc Pollefeys. LaMAR: Benchmarking localization and mapping for augmented reality. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 2, 15

[45] 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, 2, 3, 4, 6

[46] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 14

[47] Cameron Smith, David Charatan, Ayush Tewari, and Vincent Sitzmann. FlowMap: High-quality camera poses, intrinsics, and depth via gradient descent. *arXiv preprint arXiv:2404.15259*, 2024. 1, 2

[48] Christopher Sweeney, Tobias Hollerer, and Matthew Turk. Theia: A fast and scalable structure-from-motion library. In *Proceedings of the ACM International Conference on Multimedia MM*, 2015. 2

[49] Chris Sweeney, Torsten Sattler, Tobias Hollerer, Matthew Turk, and Marc Pollefeys. Optimizing the viewing graph for

structure-from-motion. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 4

[50] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21–22, 1999 Proceedings*, 2000. 1, 2, 4, 5

[51] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-NeRF: Scalable construction of large-scale NeRFs for virtual fly-throughs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 6, 7

[52] Jianyuan Wang, Nikita Karaev, Christian Rupprecht, and David Novotny. VGGSfM: Visual geometry grounded deep structure from motion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[53] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 3

[54] Kyle Wilson and David Bindel. On the distribution of minima in intrinsic-metric rotation averaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[55] Kyle Wilson and Noah Snavely. Robust global translations with 1DSfM. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014. 1, 2

[56] Linning Xu, Vasu Agrawal, William Laney, Tony Garcia, Aayush Bansal, Changil Kim, Samuel Rota Bulò, Lorenzo Porzi, Peter Kontschieder, Aljaž Božič, Dahua Lin, Michael Zollhöfer, and Christian Richardt. VR-NeRF: High-fidelity virtualized walkable spaces. In *SIGGRAPH Asia Conference Proceedings*, 2023. 6, 7

[57] Jianing Yang, Alexander Sax, Kevin J Liang, Mikael Henaff, Hao Tang, Ang Cao, Joyce Chai, Franziska Meier, and Matt Feiszli. Fast3r: Towards 3d reconstruction of 1000+ images in one forward pass. *arXiv preprint arXiv:2501.13928*, 2025. 3

[58] Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. FLARE: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. *arXiv preprint arXiv:2502.12138*, 2025. 3

[59] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4

[60] Bingbing Zhuang, Loong-Fah Cheong, and Gim Hee Lee. Baseline desensitizing in translation averaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

A. More Results

We show per-scene pose accuracy metrics for all the datasets from Tab. 7 to Tab. 11. The per-scene NeRF and Gaussian Splatting evaluation on Tanks and Temples is shown in Tab. 12.

B. Technical Details

B.1. Distortion Estimation

Hierarchical search In order to accelerate the interval search in distortion estimation, which scales linearly with the number of candidates, we employ a hierarchical search strategy that iteratively shrinks the interval. At each level of the hierarchy, after finding the solution, we set the left and right candidates as the endpoints of the new interval for the next level. The solution at the last level is the final estimate.

Multiple cameras If the two images in a pair do not share intrinsics, but the distortion parameter of one of the images is known, we can use a similar 1D search method to determine the distortion parameter of the other image. With this, we can extend the distortion estimation algorithm to deal with multiple different cameras, each of which corresponds to a known subset of images.

We say that an image pair is *ready* for a camera if either

1. both images correspond to that camera; or
2. only one of the images corresponds to that camera, but the distortion parameter of the other image is already estimated.

We estimate the distortion parameters for these cameras one by one. Each time, among the cameras whose distortion have not been estimated, we pick the one with the largest number of ready image pairs. The distortion parameter for this camera is then estimated using these image pairs. To do that, the only modification to the original algorithm (originally for a single image pair) is that for each candidate of α we compute the average epipolar error over all the point pairs in all the ready image pairs. The next camera is picked likewise, until the distortion parameters for all the cameras are estimated.

Importance of undistortion The most direct impact of incorrect distortion is on focal length estimation as illustrated in Figure 3, which visualizes focal length validity (smoothed over discrete samples) on two of the scenes with and without distortion estimation. After keypoints are undistorted (green), the validity score peaks at an accurate FoV estimation. Without distortion estimation (red), the total validity score decreases drastically, and the peak deviates from the correct FoV.

B.2. Focal Length Estimation

We adopt a similar strategy as distortion estimation Appendix B.1 to deal with multiple cameras. However we do not use hierarchical sampling because processing each candidate is relatively cheap, and we can afford to densely sample the interval.

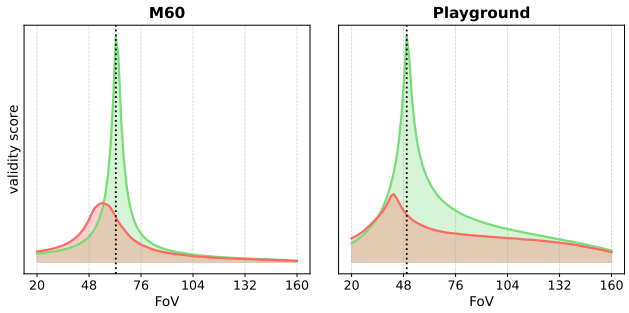

Figure 3. Effect of distortion on focal length estimation. Curves in green are with undistortion, and curves in red without. The dotted lines indicate the ground-truth FoVs.

B.3. Global Rotation

B.3.1. Initialization

In this section we discuss how to initialize the global rotation matrices for optimization. It is a modified version of Martinec and Pajdla [32].

Denote the set of images as $\mathcal{I}$ and the set of image pairs as $\mathcal{P} = \{(i,j)\}_{(i,j)\in\mathcal{I}\times\mathcal{I}}$, where each element has relative rotation matrix $\mathbf{R}^{i\to j}$. The goal is to construct a solution of global world to camera rotation matrices $\{\mathbf{R}_i\}_{i\in\mathcal{I}}$ to minimize the objective (note that in contrast to the final objective used in iterative optimization, this one uses L2 loss)

$$\mathcal{L} = \sum_{(i,j)\in\mathcal{P}} \left\| \mathbf{R}^{(j)} - \mathbf{R}^{i\to j}\mathbf{R}^{(i)} \right\|^2 \tag{10}$$

Unfortunately, this problem cannot be directly solved using simple least square techniques since $\mathbf{R}^{(i)} \in \mathrm{SO}(3)$ are 3×3 matrices with orthogonality constraints. However, we can decompose it into several sub-problems to circumvent the constraints. In the following we use $\mathbf{A}_{*,k}$ to denote the k^{th} solumn of a matrix $\mathbf{A}$. Note that each term inside the summation in Eqn. 10 can be splitted into three parts

$$\mathcal{L} = \sum_{(i,j)\in\mathcal{P}} \sum_{k=1,2,3} \left\| \mathbf{R}^{(j)}_{*,k} - \mathbf{R}^{i\to j}\mathbf{R}^{(i)}_{*,k} \right\|^2 \tag{11}$$

Since $\mathbf{R}^{(i)}$ is an orthogonal matrix, the column vectors $\mathbf{R}^{(i)}_{*,k}$ have unit length and are mutually orthogonal. These orthogonality constraints are difficult to deal with, but if we only look at one particular column, say $k=1$, and ignore the unit length constraint, the objective becomes an unconstrained least squares problem

$$\mathcal{L}^{(1)} = \sum_{(i,j)\in\mathcal{P}} \left\| \mathbf{R}^{(j)}_{*,1} - \mathbf{R}^{i\to j}\mathbf{R}^{(i)}_{*,1} \right\|^2 \tag{12}$$

		time (sec)			ATE↓			RTA@3↑			RRA@3↑			RTA@1↑			RRA@1↑			AUC-R&T @ 3 ↑			AUC-R&T @ 1 ↑		
	n_imgs	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP
m360_bicycle	194	89	138	161	7.9e-5	5.4e-5	5.6e-5	100.0	100.0	100.0	100.0	100.0	100.0	99.8	100.0	100.0	100.0	100.0	100.0	96.0	97.6	97.5	88.0	92.7	92.7
m360_bonsai	292	77	594	1347	3.9e-5	2.1e-5	5.9e-3	100.0	100.0	99.3	100.0	100.0	99.3	100.0	100.0	99.3	100.0	100.0	99.3	96.3	98.6	98.8	88.8	95.7	97.7
m360_counter	240	113	355	553	1.4e-5	2.5e-6	2.3e-6	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	98.9	99.8	99.8	96.7	99.5	99.5
m360_flowers	173	100	99	127	6.3e-5	6.0e-5	1.4e-4	100.0	100.0	100.0	100.0	100.0	100.0	99.8	99.9	99.7	100.0	100.0	100.0	96.2	96.4	93.5	88.7	89.3	80.5
m360_garden	185	95	240	543	7.6e-6	1.5e-5	1.5e-5	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	99.6	99.0	99.1	98.9	97.1	97.3
m360_kitchen	279	156	657	1309	4.4e-5	4.0e-5	3.9e-5	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	97.0	97.2	97.2	91.1	91.5	91.7
m360_room	311	351	435	795	4.1e-3	1.1e-5	9.2e-6	98.5	100.0	100.0	99.4	100.0	100.0	96.5	100.0	100.0	97.5	100.0	100.0	94.2	98.9	99.0	87.6	96.7	97.0
m360_stump	125	88	61	77	3.6e-5	1.9e-5	2.1e-5	100.0	100.0	100.0	100.0	100.0	100.0	99.9	100.0	100.0	100.0	100.0	100.0	98.8	99.4	99.3	96.5	98.1	98.0
m360_treehill	141	85	112	219	8.9e-5	7.7e-5	4.5e-5	100.0	100.0	100.0	100.0	100.0	100.0	99.7	99.8	99.9	100.0	100.0	100.0	95.4	96.8	98.3	86.4	90.6	94.9

Table 7. Per scene camera pose metrics on the MipNeRF360 Dataset.

		time (sec)			ATE↓			RTA@3↑			RRA@3↑			RTA@1↑			RRA@1↑			AUC-R&T @ 3 ↑			AUC-R&T @ 1 ↑		
	n_imgs	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP
tnt_advn_Auditorium	298	576	271	336	1.3e-2	3.3e-2	1.7e-3	6.9	91.1	99.3	5.3	91.4	99.3	0.8	89.8	99.2	0.5	90.2	99.3	0.1	86.6	98.0	0.0	79.9	95.4
tnt_advn_Ballroom	324	247	892	1154	1.8e-2	3.2e-2	6.5e-3	46.7	33.6	95.5	61.8	39.4	95.7	20.7	31.8	94.6	20.7	32.3	95.7	19.4	29.6	89.7	3.7	24.4	78.7
tnt_advn_Courtroom	301	189	512	1206	3.3e-3	4.3e-5	2.2e-5	61.1	99.9	99.9	94.8	100.0	100.0	14.3	99.3	99.8	30.0	100.0	100.0	21.2	96.8	98.6	1.3	91.0	96.1
tnt_advn_Museum	301	214	479	835	9.8e-4	5.5e-5	3.4e-5	89.3	100.0	100.0	100.0	100.0	100.0	33.3	99.7	99.9	79.3	100.0	100.0	48.8	97.2	98.6	9.6	91.9	95.9
tnt_advn_Palace	501	478	921	1534	3.5e-3	6.4e-3	1.5e-4	73.8	47.9	97.4	79.6	46.4	98.8	43.3	44.5	93.1	27.5	45.9	97.0	33.5	42.2	92.2	6.7	37.9	85.0
tnt_advn_Temple	302	192	329	423	1.8e-3	5.0e-5	7.2e-5	96.8	99.9	99.9	99.5	100.0	100.0	85.3	99.6	99.6	90.9	100.0	100.0	75.6	98.4	98.1	41.9	95.6	94.7
tnt_intrmdt_Family	152	81	273	287	9.1e-5	1.1e-5	1.0e-5	100.0	100.0	100.0	100.0	100.0	100.0	99.8	100.0	100.0	100.0	100.0	100.0	95.0	99.4	99.5	85.2	98.3	98.5
tnt_intrmdt_Francis	302	109	641	855	4.3e-5	6.2e-6	1.7e-6	99.9	100.0	100.0	100.0	100.0	100.0	99.6	99.9	100.0	100.0	100.0	100.0	95.5	99.5	99.9	86.9	98.5	99.7
tnt_intrmdt_Horse	151	81	250	225	6.9e-5	1.4e-5	4.2e-6	100.0	100.0	100.0	100.0	100.0	100.0	99.9	100.0	100.0	100.0	100.0	100.0	96.9	99.4	99.8	90.7	98.3	99.5
tnt_intrmdt_Lighthouse	309	186	617	806	1.5e-4	2.3e-5	8.9e-6	99.6	100.0	100.0	100.0	100.0	100.0	97.1	99.8	100.0	97.7	100.0	100.0	85.5	98.3	99.4	59.0	95.1	98.3
tnt_intrmdt_M60	313	125	642	785	5.0e-5	1.2e-5	6.7e-6	99.9	100.0	100.0	100.0	100.0	100.0	99.4	100.0	100.0	100.0	100.0	100.0	95.6	99.1	99.6	87.2	97.3	98.9
tnt_intrmdt_Panther	314	150	852	1046	7.9e-5	6.3e-5	1.4e-4	99.9	100.0	99.9	100.0	100.0	100.0	99.0	99.8	97.8	100.0	100.0	100.0	92.8	97.4	95.2	79.1	92.2	86.6
tnt_intrmdt_Playground	307	133	576	626	1.6e-4	1.8e-5	9.4e-6	100.0	100.0	100.0	100.0	100.0	100.0	99.3	99.9	100.0	92.3	100.0	100.0	84.6	99.2	99.5	55.7	97.6	98.6
tnt_intrmdt_Train	301	65	933	867	8.9e-5	7.8e-6	3.8e-6	99.9	100.0	100.0	100.0	100.0	100.0	99.5	99.9	100.0	100.0	100.0	100.0	92.4	99.4	99.8	77.6	98.2	99.4
tnt_trng_Barn	410	178	805	1112	1.2e-4	6.7e-6	4.2e-6	99.9	100.0	100.0	100.0	100.0	100.0	99.6	100.0	100.0	100.0	100.0	100.0	93.5	99.4	99.7	80.7	98.2	99.2
tnt_trng_Caterpillar	383	79	718	995	5.8e-5	5.6e-6	3.9e-6	99.8	100.0	100.0	100.0	100.0	100.0	99.5	99.9	99.9	100.0	100.0	100.0	95.7	99.4	99.7	87.7	98.4	99.3
tnt_trng_Church	507	352	1103	3152	8.0e-3	2.6e-2	7.9e-4	74.7	71.1	99.6	98.1	75.5	100.0	52.8	70.3	99.4	70.3	71.1	99.9	49.7	69.3	98.7	15.9	66.6	97.1
tnt_trng_Courthouse	1106	1324	2507	10386	1.2e-2	1.7e-2	1.3e-3	40.7	97.3	99.8	71.6	97.3	99.8	35.8	97.3	99.8	62.3	97.3	99.8	32.0	96.5	99.4	19.4	95.0	98.5
tnt_trng_Ignatius	263	96	565	736	5.8e-5	7.5e-6	1.9e-6	100.0	100.0	100.0	100.0	100.0	100.0	99.8	100.0	100.0	100.0	100.0	100.0	96.4	99.5	99.9	89.5	98.6	99.8
tnt_trng_Meetingroom	371	181	411	865	2.6e-3	1.4e-5	8.3e-4	95.8	100.0	99.4	99.9	100.0	98.9	68.2	99.9	99.2	70.3	100.0	98.9	62.1	99.0	98.0	18.4	97.0	96.3
tnt_trng_Truck	251	54	478	640	7.9e-5	3.4e-2	2.7e-6	100.0	52.7	100.0	100.0	52.6	100.0	99.8	52.6	100.0	100.0	52.6	100.0	94.9	51.8	99.9	84.9	50.3	99.6

Table 8. Per scene camera pose metrics on the Tanks and Temples Dataset.

Using techniques like SVD we can find a non-trivial solution that is not all zero. And since the relative rotation matrices $\mathbf{R}^{i\to j}$ are also orthogonal matrices, left multiplying it with a vector preserves the vector length. This means that in the solution, the three-dimensional vectors $\mathbf{R}^{(i)}_{*,1}$ should have similar lengths. We can normalize them to unit length and get a solution of the actual first columns of global rotation matrices $\{\mathbf{R}^{(i)}\}_{i\in\mathcal{I}}$.

Given already estimated first columns $\{\mathbf{R}^{(i)}_{*,1}\}_{i\in\mathcal{I}}$ of the global rotation matrices, we can estimate the second columns by minimizing the same objective but adding additional constraints enforcing orthogonality to the first columns

$$\begin{aligned}\mathcal{L}^{(2)} =& \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}}\left\|\mathbf{R}^{(j)}_{*,2}-\mathbf{R}^{i\to j}\mathbf{R}^{(i)}_{*,2}\right\|^2 \\ &+\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\left\|{\mathbf{R}^{(i)}_{*,1}}^{\top}\mathbf{R}^{(i)}_{*,2}\right\|^2,\end{aligned} \tag{13}$$

where $|\mathcal{P}|$ is the number of image pairs and $|\mathcal{I}|$ is the number of images, and they are used heuristically for controlling the relative weighting of the two terms. Note that in the above $\mathbf{R}^{(i)}_{*,1}$ are already fixed and the only free variables are $\{\mathbf{R}^{(i)}_{*,2}\}$. So this is still a least squares problem, and can be solved by applying SVD and then normalizing each 3-dimensional vector to unit length. A simple Gram-Schmidt process can be used for enforcing the orthogonality between the first and second columns.

We do not need to solve a least square problem again for the third columns because it can be directly computed by taking the cross product of the first two columns

$$\mathbf{R}^{(i)}_{*,3}=\mathbf{R}^{(i)}_{*,1}\times\mathbf{R}^{(i)}_{*,2} \tag{14}$$

B.3.2. Filtering

To improve the robustness of global rotation alignment, we filter out some image pairs whose number of inlier point pairs does not exceed certain threshold. Determining this threshold can be tricky: a low threshold might introduce a lot of outlier image pairs, but a high threshold could reduce the number of connections and even disconnect the images into several clusters. To alleviate this problem, we start from a large threshold, and reduce it by half if it leads to disconnected clusters. We do this iteratively until either all the images are connected, or a minimal threshold is reached. This partially solves the problem by making the threshold adaptive to the data. However it is still common that even at the minimal threshold, a few images are disconnected from the others. In that case we just consider them as outliers and ignore them in rotation alignment and later stages.

B.4. Epipolar Adjustment

In this section we derive the following equivalent form of the L2 epipolar adjustment objective (the re-weighting objective is similar)

$$\mathcal{L}=\frac{1}{Z}\sum_{n=1}^{|\mathcal{P}|}\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}(\tilde{\boldsymbol{x}}^{(2)\top}_{nm}\mathbf{E}_n\tilde{\boldsymbol{x}}^{(1)}_{nm})^2=\frac{2}{Z}\sum_{n=1}^{|\mathcal{P}|}\boldsymbol{e}_n^{\top}\mathbf{W}_n\boldsymbol{e}_n \tag{15}$$

		time (sec)			ATE↓			RTA@3↑			RRA@3↑			RTA@1↑			RRA@1↑			AUC-R&T @ 3 ↑			AUC-R&T @ 1 ↑		
	n_imgs	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP
nosr_europa	309	89	444	1504	9.3e-4	9.0e-4	8.8e-4	88.9	88.9	89.3	100.0	100.0	100.0	59.7	60.1	60.1	99.0	97.2	98.0	63.6	63.5	63.8	33.9	33.7	34.0
nosr_lk2	199	62	226	610	4.9e-4	5.0e-4	4.8e-4	97.5	97.2	97.2	100.0	100.0	100.0	89.2	88.5	88.1	99.8	100.0	99.0	83.9	84.8	84.4	61.7	65.1	64.1
nosr_lwp	354	195	351	1398	5.9e-4	6.0e-4	5.9e-4	96.1	96.1	96.2	100.0	100.0	100.0	74.5	75.7	76.2	100.0	100.0	100.0	73.6	75.9	76.1	41.5	48.3	48.4
nosr_rathaus	515	294	1039	4889	5.2e-4	5.1e-4	5.1e-4	88.5	88.1	88.5	100.0	100.0	100.0	56.0	55.1	56.0	99.2	100.0	100.0	59.8	60.5	61.0	24.9	28.4	28.6
nosr_schloss	379	173	602	1896	6.2e-4	6.1e-4	5.8e-4	92.2	92.1	92.2	100.0	100.0	100.0	72.9	73.2	73.8	99.9	100.0	100.0	71.8	72.3	72.6	43.5	44.8	45.1
nosr_st	397	160	405	1106	2.9e-3	3.0e-3	2.9e-3	96.4	96.2	96.0	98.0	98.5	98.0	85.3	86.4	83.2	97.0	95.6	96.7	78.4	80.3	78.5	48.7	54.8	51.3
nosr_stjacob	722	297	1629	5098	3.9e-3	1.7e-3	3.3e-3	91.3	93.0	92.2	98.6	99.7	98.9	76.3	77.9	76.8	98.6	99.2	98.9	74.2	75.8	74.8	49.1	50.6	49.5
nosr_stjohann	347	156	680	2351	7.9e-4	7.9e-4	7.9e-4	84.9	84.7	85.1	100.0	100.0	100.0	58.0	57.9	58.3	99.3	99.4	99.4	61.7	61.9	62.1	35.1	36.1	35.8

Table 9. Per scene camera pose metrics on the NeRF-OSR Dataset.

		time (sec)			ATE↓			RTA@3↑			RRA@3↑			RTA@1↑			RRA@1↑			AUC-R&T @ 3 ↑			AUC-R&T @ 1 ↑		
	n_imgs	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP
dploy_house1	220	116	287	460	7.9e-5	1.1e-4	1.1e-4	99.9	99.9	99.9	100.0	100.0	100.0	99.4	99.1	99.0	100.0	100.0	100.0	93.8	93.2	93.0	81.8	80.2	79.6
dploy_house2	725	375	1490	5666	6.8e-5	7.2e-5	1.3e-4	99.9	99.9	99.8	100.0	100.0	100.0	99.2	99.3	95.5	99.6	99.9	87.0	91.8	91.7	81.6	75.8	75.6	49.7
dploy_house3	180	287	180	436	5.2e-3	1.2e-2	1.2e-3	94.9	95.9	97.2	95.6	95.6	98.7	83.6	85.8	80.8	72.3	71.2	61.1	66.9	68.0	61.8	22.6	26.4	22.6
dploy_house4	349	296	249	552	9.6e-3	1.9e-2	8.8e-3	94.2	98.3	98.8	95.4	97.7	98.9	92.2	97.8	98.1	91.5	97.7	98.3	83.8	89.9	91.2	67.4	74.6	77.3
dploy_pipes1	97	71	68	171	6.7e-5	6.2e-5	6.1e-5	100.0	100.0	100.0	100.0	100.0	100.0	99.9	99.9	99.9	100.0	100.0	100.0	95.9	95.8	95.8	87.6	87.3	87.4
dploy_ruins2	1171	647	3232	20780	5.2e-5	4.1e-5	1.5e-4	99.9	100.0	99.2	100.0	100.0	100.0	98.9	99.4	87.7	97.4	100.0	90.4	88.2	92.1	75.5	65.7	76.5	35.2
dploy_ruins3	523	335	638	2585	1.2e-3	6.1e-3	3.5e-3	98.6	94.7	95.7	98.8	99.4	83.4	75.9	79.9	45.8	40.8	46.2	24.1	57.5	59.0	39.1	9.1	10.9	3.8
dploy_tower1	775	748	973	3290	1.7e-2	2.2e-3	2.9e-3	97.7	95.2	85.4	99.5	99.5	97.8	88.9	84.4	49.2	62.8	74.3	37.2	67.5	67.5	45.5	20.6	23.5	9.7
dploy_tower2	682	507	1365	5230	2.1e-4	1.6e-4	1.3e-3	99.9	99.8	44.5	100.0	100.0	25.6	70.3	77.6	10.7	100.0	100.0	9.5	71.6	72.9	8.5	23.1	26.2	0.8

Table 10. Per scene camera pose metrics on the DroneDeploy Dataset.

Note that each error term is linear in the essential matrix, so we can re-write it as a dot product of the a weight vector and a flattened version of the essential matrix

$$\mathcal{L} = \frac{1}{Z}\sum_{n=1}^{|\mathcal{P}|}\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}(\tilde{\boldsymbol{x}}_{nm}^{(2)\top}\mathbf{E}_n\tilde{\boldsymbol{x}}_{nm}^{(1)})^2 \tag{16a}$$

$$= \frac{1}{Z}\sum_{n=1}^{|\mathcal{P}|}\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}((\tilde{\boldsymbol{x}}_{nm}^{(2)}\tilde{\boldsymbol{x}}_{nm}^{(1)\top})\otimes\mathbf{E}_n)^2 \tag{16b}$$

$$= \frac{1}{Z}\sum_{n=1}^{|\mathcal{P}|}\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}(\boldsymbol{w}_{nm}^{\top}\boldsymbol{e}_n)^2, \tag{16c}$$

where $\otimes$ is the element-wise multiplication operator, $\boldsymbol{w}_{nm} = \text{flatten}(\tilde{\boldsymbol{x}}_{nm}^{(2)}\tilde{\boldsymbol{x}}_{nm}^{(1)\top}) \in \mathbb{R}^9$, and $\boldsymbol{e}_n = \text{flatten}(\mathbf{E}_n) \in \mathbb{R}^9$. Now re-arrange the terms to get the summation of a set of quadratic forms

$$\mathcal{L} = \frac{1}{Z}\sum_{n=1}^{|\mathcal{P}|}\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}(\boldsymbol{w}_{nm}^{\top}\boldsymbol{e}_n)^2 \tag{17a}$$

$$= \frac{1}{Z}\sum_{n=1}^{|\mathcal{P}|}\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}(\boldsymbol{w}_{nm}^{\top}\boldsymbol{e}_n)^{\top}(\boldsymbol{w}_{nm}^{\top}\boldsymbol{e}_n) \tag{17b}$$

$$= \frac{2}{Z}\sum_{n=1}^{|\mathcal{P}|}\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}\boldsymbol{e}_n^{\top}\boldsymbol{w}_{nm}\boldsymbol{w}_{nm}^{\top}\boldsymbol{e}_n \tag{17c}$$

$$= \frac{2}{Z}\sum_{n=1}^{|\mathcal{P}|}\boldsymbol{e}_n^{\top}\left(\sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}\boldsymbol{w}_{nm}\boldsymbol{w}_{nm}^{\top}\right)\boldsymbol{e}_n \tag{17d}$$

$$= \frac{2}{Z}\sum_{n=1}^{|\mathcal{P}|}\boldsymbol{e}_n^{\top}\mathbf{W}_n\boldsymbol{e}_n \tag{17e}$$

where $\mathbf{W}_n = \sum_{m=1}^{|\tilde{\mathcal{Q}}_n|}\boldsymbol{w}_{nm}\boldsymbol{w}_{nm}^{\top} \in \mathbb{R}^{9\times 9}$

B.5. Sparse Reconstruction

After pose estimation, we do a sparse reconstruction of the scene by triangulating the matched keypoint pairs from track completion. The 3D points corresponding to the same track are merged by averaging. To eliminate outliers, after merging the 3D points, we compute the re-projection error for each 2D keypoint, and mark those with large errors to be outlier keypoints. A 3D point is dropped if the number of inlier keypoints in the track is smaller than 3. We also filter out a 3D point if the maximal triangulation angle is smaller than some threshold.

C. Limitations

Sparse Views. Our method assumes that the input images densely cover the 3D scene. Many components in the pipeline implicitly assume that the coverage is dense so that the negative effect of outlier image or point pairs will be averaged out. If the coverage is sparse, the pipeline will be sensitive to outliers and likely break down. We tested our method on the ETH3D MVS (DSLR) [46], where each scene only contains a small number of images, and show the results in Tab. 13. While FastMap still succeeds on many scenes, it is less robust than GLOMAP.

Intrinsics Estimation. The intrinsics estimation algorithms in our method can fail under certain cases. Since the interval search used in both distortion and focal length estimation requires at least one image pair of images with shared intrinsics to begin with, it will not work if all the images have different intrinsics. It is also not robust if the number of images for each distinct camera is small. In addition, the focal length extraction method relies entirely on fundamental matrices, and is unreliable when the scene is dominated by homographies.

Homography. Apart from the impact on focal length estimation, too many homography image pairs can also jeop-

		time (sec)			ATE↓			RTA@3↑			RRA@3↑			RTA@1↑			RRA@1↑			AUC-R&T @ 3 ↑			AUC-R&T @ 1 ↑		
	n_imgs	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP
z_alameda	1734	917	3805	24641	5.6e-4	1.9e-4	6.7e-6	99.1	99.4	100.0	99.7	100.0	100.0	98.9	99.3	99.9	99.0	99.7	99.9	95.0	97.8	98.5	86.9	94.8	95.6
z_berlin	1511	545	1802	24648	1.0e-3	1.5e-2	1.1e-3	97.7	93.0	95.6	96.9	92.9	94.8	90.9	92.8	78.3	94.9	92.8	56.5	83.2	90.9	56.9	59.9	87.0	53.1
z_london	1874	556	2092	19238	5.7e-5	3.6e-4	2.8e-4	99.8	99.9	99.9	99.8	99.9	99.9	99.6	99.9	99.9	99.6	99.9	99.9	96.5	98.6	98.5	90.2	96.0	95.8
z_nyc	990	338	921	1988	4.7e-4	6.9e-6	5.3e-6	99.7	100.0	100.0	99.8	100.0	100.0	99.1	100.0	100.0	99.4	100.0	100.0	94.3	98.9	98.9	83.9	96.8	96.7
mill19_building	1920	1366	38428	152839	2.0e-5	1.3e-2	5.1e-3	100.0	0.1	81.7	100.0	10.5	81.7	99.5	0.0	80.7	100.0	2.5	78.6	95.5	0.0	76.9	87.0	0.0	68.5
mill19_rubble	1657	789	11571	64987	3.6e-5	8.0e-5	3.4e-5	99.9	99.8	99.9	100.0	99.9	100.0	98.7	98.6	98.7	100.0	99.9	100.0	93.5	94.5	94.6	81.5	84.7	84.8
urbn_Campus	5871	3009	21916	349490	1.2e-5	4.7e-6	4.9e-6	100.0	100.0	100.0	100.0	100.0	100.0	99.9	99.9	99.9	99.9	100.0	100.0	92.5	98.0	97.9	77.5	94.1	93.7
urbn_Residence	2582	1520	10028	242259	2.7e-5	2.7e-5	2.6e-5	99.8	99.9	99.9	100.0	100.0	100.0	98.8	98.9	99.0	100.0	100.0	100.0	94.6	95.2	95.4	84.7	86.3	86.8
urbn_Sci-Art	3019	1760	28824	286454	1.4e-5	9.9e-6	1.1e-5	100.0	100.0	100.0	100.0	100.0	100.0	99.9	99.9	99.9	100.0	100.0	100.0	97.4	97.7	97.8	92.2	93.1	93.5
eft_apartment	3804	1003	124310	> 7d	3.6e-3	9.4e-3	-	86.3	83.7	-	89.1	84.2	-	50.9	71.3	-	38.2	66.6	-	45.9	58.8	-	6.4	22.2	-
eft_kitchen	6042	6796	> 7d	> 7d	3.0e-3	-	-	83.6	-	-	85.1	-	-	45.7	-	-	26.0	-	-	37.6	-	-	4.4	-	-

Table 11. Per scene camera pose metrics on several large-scale datasets including ZipNeRF, Mill-19, Urbanscene3D and Eyeful Tower.

		Instant-NGP			Gaussian Splatting		
		FastMap	GLOMAP	COLMAP	FastMap	GLOMAP	COLMAP
training	Barn	23.37	23.68	23.69	26.17	27.81	27.79
	Caterpillar	20.17	20.20	20.23	23.30	23.47	23.59
	Courthouse	19.96	14.73	20.27	21.12	12.23	22.25
	Ignatius	18.11	18.14	18.26	21.42	22.04	21.86
	Meetingroom	21.61	22.59	22.41	23.71	25.35	25.17
	Truck	21.19	16.86	21.45	23.58	18.34	24.50
intermediate	Family	22.10	21.99	21.45	23.67	24.54	24.75
	Francis	23.68	23.73	23.48	26.94	27.30	27.59
	Horse	21.07	21.04	21.13	22.96	24.05	23.89
	Lighthouse	20.84	20.99	20.91	22.00	22.19	22.12
	M60	24.80	25.11	25.11	26.44	28.07	27.95
	Panther	25.25	26.01	25.74	27.13	28.27	27.97
	Playground	21.48	21.96	21.99	24.12	26.00	26.03
	Train	19.14	19.26	19.23	20.66	21.79	21.65
advanced	Auditorium	17.05	19.41	19.76	17.38	16.67	24.20
	Ballroom	17.94	13.92	19.12	19.17	11.91	23.64
	Courtroom	17.39	18.95	17.80	20.87	23.11	22.77
	Museum	14.58	14.73	14.53	20.00	20.97	20.96
	Palace	16.94	17.57	17.19	16.41	18.99	20.05
	Temple	15.65	17.10	16.87	19.08	20.80	20.73

Table 12. Per-scene novel view synthesis results on Tanks and Temples.

		ATE↓		RTA@3↑		AUC-R&T @ 3 ↑		AUC-R&T @ 1 ↑	
	n_imgs	FastMap	GLOMAP	FastMap	GLOMAP	FastMap	GLOMAP	FastMap	GLOMAP
botanical_garden	30	6.6e-1	1.3e-1	73.1	100.0	59.6	94.3	46.8	83.8
boulders	26	3.4e-1	2.2e-1	99.1	100.0	91.0	97.0	75.6	91.0
bridge	110	9.7e-1	1.1e-2	94.2	100.0	86.8	97.7	75.3	93.1
courtyard	38	2.2e-2	5.5e-1	32.7	100.0	14.3	96.0	5.3	88.2
delivery_area	44	1.3e-2	4.4e-1	29.1	100.0	17.6	97.8	8.0	93.3
door	7	6.6e-1	5.8e-0	95.2	100.0	89.9	98.0	79.3	94.1
electro	45	2.9e-1	2.4e-1	83.4	95.2	70.4	91.1	50.6	84.1
exhibition_hall	68	2.1e-2	1.5e-2	1.5	45.1	0.1	40.9	0.0	34.3
facade	76	5.7e-1	2.1e-2	74.1	100.0	67.0	97.4	59.9	92.4
kicker	31	2.6e-1	1.7e-1	98.3	93.8	85.9	91.7	62.7	88.1
lecture_room	23	6.1e-1	2.8e-1	76.3	100.0	60.9	95.0	42.6	85.7
living_room	65	4.5e-1	9.1e-1	99.8	99.8	95.2	96.2	86.9	89.2
lounge	10	2.2e-1	3.3e-0	33.3	33.3	32.4	32.7	30.4	31.4
meadow	15	1.4e-1	3.5e-0	11.4	86.7	7.2	80.2	5.1	68.2
observatory	27	1.8e-1	4.5e-1	99.1	99.1	84.8	86.5	59.4	63.9
office	26	2.0e-1	1.2e-1	52.3	95.7	42.3	82.7	32.0	61.2
old_computer	54	7.2e-1	3.3e-1	15.0	65.3	6.5	60.9	1.5	53.5
pipes	14	1.9e-1	4.2e-0	98.9	100.0	92.5	97.4	79.6	92.3
playground	38	2.7e-1	1.5e-1	99.9	99.9	94.3	97.2	83.3	91.7
relief	31	2.2e-1	2.5e-1	84.1	100.0	62.5	98.4	43.4	95.2
relief_2	31	2.7e-1	2.3e-1	99.8	100.0	94.3	98.4	83.6	95.1
statue	11	4.0e-1	7.4e-0	72.7	100.0	35.8	99.7	23.3	99.0
terrace	23	1.8e-1	1.5e-1	100.0	100.0	97.6	97.7	92.7	93.1
terrace_2	13	1.4e-1	1.5e-1	100.0	100.0	96.8	96.9	90.4	90.8
terrains	42	2.5e-1	2.2e-1	98.1	99.8	79.0	94.6	51.5	84.6

Table 13. Per scene camera pose metrics on ETH3D.

ardize relative pose decomposition. Both essential and homography decomposition produce four different solutions, and they are usually disambiguated with a cheirality check. But for some homography and keypoint pairs, cheirality check is not enough for determining a unique solution. Our current strategy is to simply pick the solution with the lowest index if there is a tie. This has potential issues if there are too many homography image pairs.

	Recall@1m↑		AUC@1m↑		AUC@5m↑	
	FastMap	GLOMAP	FastMap	GLOMAP	FastMap	GLOMAP
CAB	4.77	11.6	4.32	4.7	4.74	16.9
HGE	5.94	48.4	5.26	22.2	5.70	50.3
LIN	7.16	87.3	3.95	46.7	7.96	85.6

Table 14. Per scene camera pose metrics on LaMAR.

Repetitive Patterns and Symmetric Structures Non-learning based keypoint features and matching are not robust in the cases of repetitive patterns and symmetric structures in the scene. These wrong matches are hard to filter because they can have a lot of inlier point pairs with a very consistent two-view geometric model. Most traditional SfM methods are more or less impacted by these erroneous matches, and so is ours. In our experiments this problem is most prominent in the advanced split of the Tanks and Temples dataset (Tab. 8).

Degenerate Motions One important reason why bundle adjustment is popular in previous SfM methods is that it can utilize 3D points to resolve some ambiguities in global translation estimation. For example, when all the cameras are aligned in the same line, optimization methods that solely rely on relative motions or epipolar errors might fail because there is no way to uniquely (up to scale) determine the distance of any pair of cameras. Bundle adjustment uses tracks to impose extra constraints to solve this problem. This scenario is commonly seen in SLAM-like datasets. We tested our method on the large-scale LaMAR [44] dataset and show the results in the Tab. 14. Each scene in LaMAR consists of multiple trajectories of a moving VR headset or hand-held phone. These trajectories contain many straight and forward motions, and different trajectories only overlap sparsely. Our method does not work well compared to GLOMAP.

D. Version History

We document the changes of different ArXiv versions of the paper.

Version 2 We fixed a bug (commit 667cae1) in database reading that causes the method to fail completely on the calibrated ETH3D and LaMAR datasets. The new results are now included in Tab. 13 and Tab. 14.

Version 1 Initial release.