# Multi-Session SLAM with Differentiable Wide-Baseline Pose Optimization

Lahav Lipson
Princeton University
llipson@princeton.edu

Jia Deng
Princeton University
jiadeng@princeton.edu

## Abstract

*We introduce a new system for Multi-Session SLAM, which tracks camera motion across multiple disjoint videos under a single global reference. Our approach couples the prediction of optical flow with solver layers to estimate camera pose. The backbone is trained end-to-end using a novel differentiable solver for wide-baseline two-view pose. The full system can connect disjoint sequences, perform visual odometry, and global optimization. Compared to existing approaches, our design is accurate and robust to catastrophic failures. Code is available at* https://github.com/princeton-vl/MultiSlam_DiffPose

## 1. Introduction

Simultaneous Localization and Mapping (SLAM) is the task of estimating camera motion and a 3D map from video. The standard setup assumes a single continuous video. However, video data in the wild often consists of not a single continuous stream, but rather multiple disjoint sessions, either deliberately such as in collaborative mapping when multiple robots perform joint rapid 3D reconstruction, or inadvertently due to visual discontinuities in the video stream which can result from camera failures, extreme parallax, rapid turns, auto-exposure lag, dark areas, or extreme occlusion by dynamic objects. Handling such disjoint videos is important for many applications in AR and robotics, and gives rise to the task of Multi-Session SLAM.

In Multi-Session SLAM, the input consists of multiple disjoint video sequences and the goal is to estimate camera poses for all video frames under a single global reference. This is in contrast to "single-video SLAM", whose input is a single continuous video. In this work, we focus on the monocular, RGB-only Multi-Session SLAM setting.

Several approaches have been proposed to deal with Multi-Session SLAM, however existing solutions typically require additional sensor data in order to remove gauge freedoms and make tracking easier [16, 28, 37]. Only a small number of methods, notably CCM-SLAM [36] and ORB-SLAM3 [4], support Multi-Session SLAM from monocular video alone, due to the difficulty of aligning disjoint sequences under the 7-DOF gauge freedoms in monocular video. However, these approaches are based on classical feature descriptors, making them less accurate on average compared to recent designs based on deep networks.

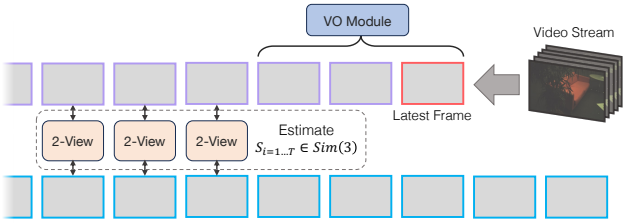


Figure 1. Our method estimates camera pose from multiple disconnected video streams.

In the standard SLAM setting, Teed and Deng [44] proposed to use a deep optical flow network (RAFT [43]) to track 2D motion, while jointly updating camera poses with a bundle adjustment layer. The method, DROID-SLAM, is accurate and avoids tracking failures, but the design assumes a continuous video stream and is not capable of the wide-baseline matching and non-local optimization necessary for Multi-Session SLAM. Deep Patch Visual Odometry [45] (DPVO) introduced a sparse visual-odometry-only analog of DROID-SLAM which achieves similar accuracy on single-video VO, but at much lower cost. However, DPVO also does not support Multi-Session SLAM for the same reasons as DROID-SLAM.

We propose a method for Multi-Session SLAM, capable of both wide-baseline relative pose and visual odometry using a single backbone architecture inspired by [45]. We introduce a differentiable solver layer which minimizes the symmetric epipolar distance (SED) from bi-directional optical flow. From this we construct a method for two-view pose which is capable of matching from far-apart views. This same design can be repurposed for visual odometry by swapping out the solver for bundle adjustment. By employing a unified backbone architecture for both tasks, we enable a simple approach to Multi-Session SLAM.

We evaluate our approach on challenging real-world datasets: EuRoC-MAV [2] and ETH3D [39]. Our sys-

tem is more accurate than prior approaches, and is robust to catastrophic failures. We also evaluate our two-view pose method in isolation on the Scannet and Megadepth datasets, and show that it is competitive with transformer-based matching networks. For pairs of far-apart views, our method is capable estimating accurate relative pose.

Our backbone predicts iterative updates to optical flow coupled with a differentiable solver layer for estimating camera pose. This framework, based on RAFT [43], has worked exceptionally well for Visual Odometry [45] and SLAM [44]. By leveraging this idea for wide-baseline matching, we can extend these methods to the Multi-Session setting without introducing substantial complexity.

## 2. Related Work

Most prior works treat Visual Simultaneous Localization and Mapping (Visual SLAM) as an optimization problem solving for a 3D scene model and camera trajectory which best explains the visual measurements [3].

**Indirect** approaches to SLAM perform keypoint matching as a pre-processing step, then estimate a 3D point cloud and camera poses by optimizing the 2D reprojection error over all matches [4, 20, 26, 27, 33, 38]. Reprojection-error is easier to optimize than photometric alignment, making indirect methods robust to lower camera hz [44]. Keypoint matching also enables a straightfoward approach to estimating two-view relative pose, a necessary step for Visual Multi-Session SLAM [14, 19, 25, 33, 46]. However, keypoint-based SLAM is less robust to low-texture environments compared to those which use photometric alignment [9, 10] or optical flow [44, 45].

**Semi-Indirect** approaches similarly optimize 2D reprojection error like indirect methods, but without requiring matches as input [42, 44, 45]. Instead, these approaches alternate between predicting optical flow residuals and performing bundle adjustment. Semi-indirect methods do not require repeatable keypoints across images, making them robust to low-texture settings while retaining the easier reprojection-error objective.

Our approach is most similar to Deep Patch Visual Odometry [45] (DPVO), which is a sparse analog of DROID-SLAM. DPVO predicts sparse optical flow instead of dense, performing similarly to DROID-SLAM while running faster and using half the memory.

**Differentiable solver layers** for camera pose estimation have been used in order to learn outlier rejection with data-driven training. For two-view relative pose, Ranftl and Koltun [30] used a deep network to learn an iteratively reweighted least-squares algorithm, which solved a weighted variant of the 8-point-algorithm [14] using confidences predicted by the network. [30] required matched points as input, wheras our approach can work from images alone. Roessle and Nießner [31] proposed an end-to-end architecture which used the weighted 8-point-algorithm from [30] on top of matches produced using Superglue [34], and supervised directly on the predicted pose. [31] works well, but is unable to outperform existing methods that use minimal solvers with LO-RANSAC [18] instead of differentiable solvers. In contrast, our approach is not built on top of an existing SOTA 2-view matcher; our design is also iterative, applying a recurrent module and solver layers multiple times to refine the prediction.

**Multi-Session SLAM** is the task of performing SLAM on multiple trajectories of the same scene. Like SLAM, Multi-Session SLAM is an online task where camera motion is estimated from a stream of images. However, in the Multi-Session setting, there are known breaks in the data-stream over which the small-baseline assumption no longer holds. While local optimization is sufficient for motion tracking from video, estimating wide-baseline camera pose is often non-convex [40] and more challenging due to large viewpoint changes. In monocular visual SLAM, the scale of each sequence is also ambiguous; aligning two trajectories requires estimating 7 degrees-of-freedom (translation, rotation, scale) for all sequences, excluding the first which can be considered the reference.

ORB-SLAM3 [4] performs Multi-Session SLAM by matching between ORB [32] descriptors. While tracking camera motion over each sequence, ORB-SLAM3 continually updates an *Atlas* of the scene - a lookup table between 2D keypoints and their 3D map point in the reference frame of their original sequence. To align disjoint sequences, candidate cross-sequence image pairs are identified using image-retrieval [12], the 3D map points are obtained from the Atlas, and the relative transformation is found using the Umeyama algorithm [46]. Several methods perform Multi-Session Visual Inertial SLAM [17, 28], however we focus on the visual-only monocular setting where the scale of each session is ambiguous and must be estimated when joining sequences.

CCM-SLAM [35, 36] and SLAMM [7] perform Multi-Session visual SLAM and are built on top of ORB-SLAM [27] as well, but are optimized for limited bandwidth and distributed processing, while ORB-SLAM3 performs better and is optimized for accuracy and speed.

## 3. Approach

**Overview:** We propose a backbone for matching between $\geq$2 views, and then construct a method for multi-session SLAM upon it. Our backbone approaches matching as optical flow; it borrows several ideas from RAFT [43], such as iteratively predicting flow residuals using a recurrent network, and the correlation feature pyramid. Our method maintains a running estimate of both camera pose and bi-directional optical flow, and uses the update operator to refine them both. We provide an overview in Fig. 2. An

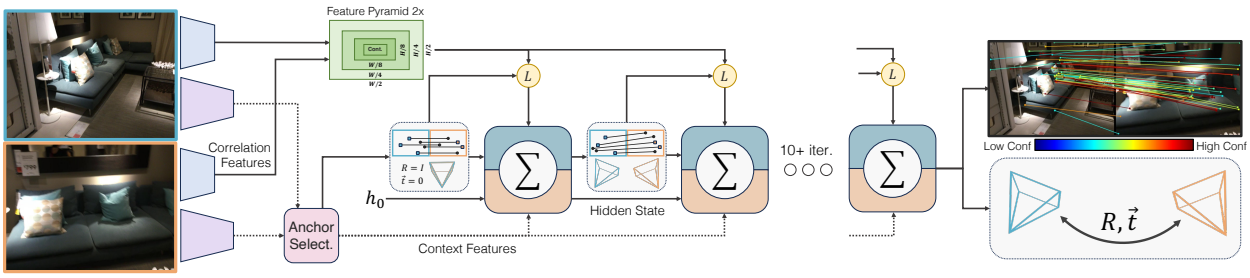


Figure 2. Overview of our backbone. Given a group of $\geq 2$ frames, our method jointly estimates bi-directional optical flow and camera poses. This module is applicable to (1) wide-baseline two-view matching and (2) visual odometry. We use our backbone for Multi-Session SLAM, which requires the ability to perform both (1) and (2). For each image, we select a set of 2D anchor points and initialize their depth and the camera poses trivially. Our approach then iteratively refines the matches for each anchor, similar to RAFT [43], while updating the camera poses. We alternate between matching and pose updates, where each one informs the update to the other. This entire procedure is repeated several times until convergence.

invariant of our backbone is that the matches are always clamped to plausible values given the pose estimates and an assumption of rigid scene geometry, therefore WLOG the matches can be considered a depth estimate.

At initialization, our backbone produces dense correlation feature pyramids for each image and context features which remain fixed throughout the forward pass. The update operator uses an RNN to predict an update to the matches, and a differentiable solver to update poses. Our backbone treats the two-view and multi-view settings differently: In the two-view setting, the solver updates the poses to minimize the symmetric epipolar distance (SED). In the multi-view setting, the solver updates both poses and depth to minimize the reprojection error. After the solver layer, the matches are adjusted to agree with the poses/depth.

### 3.1. Initialization

**Feature Extraction and Feature Pyramid:** Similar to RAFT, our method separately extracts context and correlation feature for each image. The context features are provided as input to the recurrent update operator, whereas the correlation features are used to evaluate the visual similarity between any two pixels using a dot product. The context feature maps are produced at $1/8$ resolution using a residual network. We then apply a linear-self-attention residual for long-range feature sharing. The correlation feature maps are also produced using a residual network, but with several exit ramps to produce features at $1/2$, $1/4$, and $1/8$ of the input image resolution. We also average-pool the last one three additional times to produce a correlation feature pyramid with 6 levels. We depict their architecture in the Appendix.

**Anchor-Point Selection:** Our method predicts sparse optical flow, where matches have one end anchored and the other end free to move in $\mathbb{R}^2$. We use a mix of detector [8]-chosen and randomly-chosen anchor points. Each point is assigned an initial match in the other image(s) at the same pixel coordinate. For the rest of the paper, we will refer to the $k^{th}$ anchor point as $a_k$, and its match in image $j$ as $m_{kj}$.

We index the context feature map at each $a_k$ to produce a unique context feature vector $ctx_k \in \mathbb{R}^{384}$, which is used by the update operator. Beyond this point, the full context feature map is not used and is discarded to save memory.

**Correlation:** Similar to other RAFT-based methods, we use correlation features to assess the visual alignment/similarity given by the current matching estimate. For each $(a_k, m_{kj})$ pair, we bilinearly sample the feature pyramids $f$ and $g$ for their respective frames at locations $a_k$ and $m_{kj}$ and take their inner product at each level, producing

$$\langle f_k^i, g_{kj}^i \rangle \in \mathbb{R} : i = 1...6 \quad (1)$$

To provide additional spatial context, we also perturb both $a_k$ and $m_{kj}$ in $3 \times 3$ and $7 \times 7$ grids, respectively, and calculate eq. 1 for all pairs. The resulting feature vector $\mathbf{C}_{kj} \in \mathbb{R}^{(6\times3\times3\times7\times7)}$ is then passed to the update operator.

### 3.2. Update Operator

The update operator produces a revision to all $m_{kj}$ and the estimated relative pose. It consists of three stages, depicted in Fig. 3: (1) An RNN predicts a 2D update to $m_{kj}$ and an associated confidence weight $w_{kj}$. (2) We solve for a pose estimate which is consistent with the newly predicted matches and confidence. (3) We adjust the matches to be physically plausible assuming rigid scene geometry. In the two-view setting, we clamp the matches to the epipolar lines, and in the multi-view setting we reproject the anchor points using the depth and poses from the solver.

Through the recurrent iterations, our method maintains a running state for each anchor-match $(a_k, m_{kj})$ pair, consisting of a hidden state vector $h_{kj}$, a confidence weight $w_{kj}$, and a match location $m_{kj}$ in frame $j$. This state is continually updated, during which $m_{kj}$ should ideally approach the true match location of $a_k$ in frame $j$, and $w_{kj}$ should approach 1. This is what we observe empirically.

**RNN Module:** The learnable component of the update operator is the recurrent network which predicts updates to

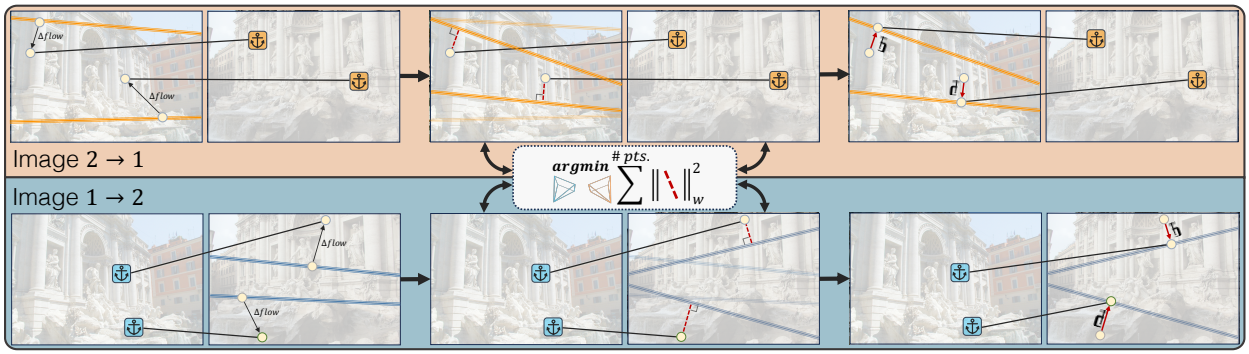


Figure 3. A single update iteration (two-view). For each anchor-match pair, we predict an update to the matches using the RNN. We then solve for an update to the camera poses which minimizes the symmetric epipolar distance (SED), producing a new set of epipolar lines. Finally, we clamp the matches back to the best-fit epipolar lines, and repeat the whole process again. In Fig. 8 and the Appendix, we visualize these iterations on real-world images.

$h_{kj}$, $m_{kj}$ and $w_{kj}$ for all anchor points. We visualize this operator in Fig. 4.

The input to the RNN are the context features, the previous hidden state, and the correlation features generated from the current matching estimate, all of which are added together and normalized with layernorm. A self-attention residual is also applied to edges with the same source and destination frame, followed by three gated-residual-units, whose architecture is depicted in the Appendix. The output is an updated hidden state, from which we predict an updated match and confidence using the flow head and confidence head, respectively, which are implemented as two-layer MLPs. The confidence head includes a softmax to restrict $w_{kj} \in (0, 1)$.

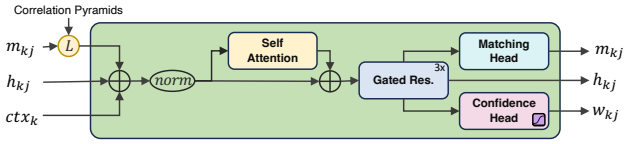


Figure 4. The RNN Module. For each anchor-match pair $(a_k, m_{kj})$, it predicts an update to $m_{kj}$ and an associated confidence $w_{kj}$. It also updates a hidden state $h_{kj} \in \mathbb{R}^{384}$. Internally, the RNN shares features via attention between updates with the same incoming and outgoing frame.

### 3.3. Two-View Solver and Pre-Conditioning

Our two-view solver aims to align the relative pose between two frames to the predicted matches by minimizing the distance between each match and its epipolar line. This strategy is accurate, however the objective function is non-convex, meaning it will only converge to the global minimum if it is initialized close to it. In contrast, the 8-point-algorithm, a common approach to this task, has the *opposite* problem; it does not suffer from local minimum since it is a homogeneous least squares, but the solution is sub-optimal. In Fig. 6, we visualize how our solver either converges to within a fraction of a degree, or not at all, meanwhile the 8-point-algorithm is more robust but less accurate.

The approach our method uses is to pre-condition the pose estimate using a weighted, dense variant of the 8-point algorithm [30], and then run our solver layer to refine the pose. This combined strategy obtains the best of both worlds, since the pre-conditioning will typically initialize the pose within the basin of convergence of our solver, which then refines the prediction. Fig. 6 visualizes this basin on the TartanAir [48] dataset using ground-truth flow.

**Pre-conditioning:** We solve the homogeneous least squares problem [14]

$$\arg\min_{\mathbf{F}} ||diag(\vec{w})\mathbf{M}\vec{\mathbf{F}}||^2 \quad s.t. \; ||\mathbf{F}||^2 = 1 \tag{2}$$

where $\mathbf{M}$ and $\vec{w}$ are constructed from the anchor-match pairs and confidence predictions. The points are normalized to $[-1, 1]$ beforehand. Afterwards, we reshape and uncalibrate $\mathbf{F}$ to obtain the essential matrix.

We obtain the four relative pose candidates following the procedure detailed in [13] and in the Appendix. During training, we select the pose candidate closest to the ground-truth on the $SE(3)$ manifold. During inference, we select the candidate by testing for chirality [13].

**SED Solver Layer:** The solver layer seeks to minimize the distance between each match $m_{kj}$, and the epipolar line induced by the current pose estimate and its respective anchor point $a_k$. The solver objective is bi-directional, since every image contains a unique set of anchor points; the objective is also known as the titular symmetric epipolar distance (SED) [11]. We parameterize the pose update as small rotations to the translation direction and orientation: $\xi_{\mathbf{t}}, \xi_{\mathbf{R}} \in \mathfrak{so}(3)$. These are the free variables. Let $K(i)$ be the set of anchor indices for frame $i$, $\mathbf{R}$ and $\mathbf{t}$ be the output of the pre-conditioning stage, and $epi(x, R, t)$ compute the epipolar line for a point $x$ given the relative pose $(R, t)$.

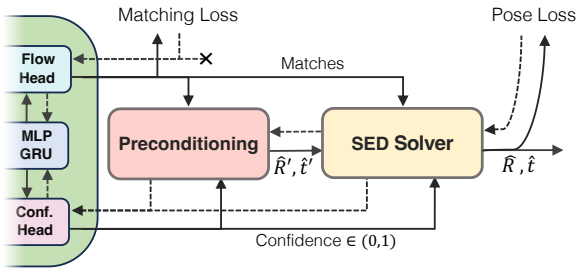


Figure 5. The flow of gradients through our solver. The updated matches from the RNN are supervised directly, and then detached from the gradient tape. The solver output is supervised with a pose loss. In the backward pass, gradients from the pose loss are used to supervise the confidence head in order to learn outlier rejection.

Formally, the minimization objective in our solver is

$$\begin{aligned} l_{kj} &= epi(a_k, (e^{\xi_{\mathbf{R}}}\mathbf{R})_{i\to j}, (e^{\xi_{\mathbf{t}}}\mathbf{t})_{i\to j}) : k \in K(i) \\ E_{i\to j} &= \sum_{k\in K(i)} w_{kj} \cdot \|err(m_{kj}, l_{kj})\|_2^2 \\ SED &= \underset{\xi_{\mathbf{R}},\xi_{\mathbf{t}}}{\arg\min} \left(E_{i\to j} + E_{j\to i}\right) \end{aligned} \tag{3}$$

where $err$ computes the 2D point-to-line error. The predicted weights $w_{kj}$ are included to allow the network to down-weight the contribution of any match which it deems unreliable. This solver layer is implemented in Pytorch, enabling gradients to propagate backward from the pose loss to the confidence head using autograd. We depict the gradient flow in Fig. 5.

To minimize eq. 3, we employ the Levenberg Marquardt algorithm. This requires computing the Jacobian for all terms. We provide the full derivations in the Appendix, but define the residual function $err$ in its entirety here.

Let $l$ be the epipolar line produced by $(a, e^{\xi_{\mathbf{R}}}\mathbf{R}, e^{\xi_{\mathbf{t}}}\mathbf{t})$. We can express each term of eq. 3 as:

$$err(m, l) = \left(\frac{l_x m_x + l_y m_y + l_z}{l_x^2 + l_y^2}\right) \begin{bmatrix} l_x \\ l_y \end{bmatrix} \tag{4}$$

and the epipolar line as

$$\begin{aligned} \mathbf{E} &= (e^{\xi_{\mathbf{R}}}\mathbf{R})^\top [e^{\xi_{\mathbf{t}}}\mathbf{t}]_\times \\ l = \mathbf{F}\begin{bmatrix} a_x \\ a_y \\ 1 \end{bmatrix} &= \begin{bmatrix} \\ (K_2^\top)^{-1}\mathbf{E}K_1^\top \\ \\ \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ 1 \end{bmatrix} \end{aligned} \tag{5}$$

where $\mathbf{F}$ and $\mathbf{E}$ are the fundamental and essential matrices.

### 3.4. Adapting our backbone to Visual Odometry

To adapt our backbone to VO, we make several changes:

**Multi-view Solver (BA):** Our local optimizer minimizes reprojection error and treats depth as a separate variable.

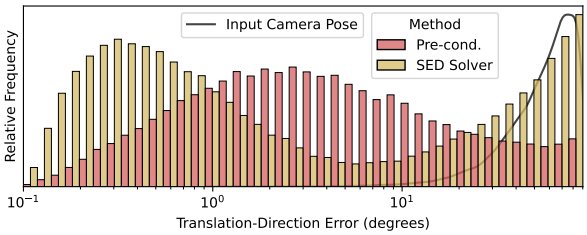


Figure 6. Convergence of our solver, given ground-truth matches (from TartanAir [48]). Our SED solver is accurate, but only converges to the global minimum when initialized close to it. We initialize the pose using the 8-pt-algorithm to ensure it is within the SED convergence basin.

This is identical to the bundle adjustment from DPVO. Formally, let $F$ be the set of connected frames, $G$ be the global poses, $d_k$ be the depth estimate for anchor $k$, and $\Pi(\cdot)$ be the $3D \to 2D$ projection function. The bundle adjustment objective is:

$$\underset{G,\mathbf{d}}{\arg\min} \sum_{(i,j)\in F} \sum_{k\in K(i)} w_{kj} \cdot \left\| \Pi[G_j^{-1} G_i \Pi^{-1}(a_k, d_k)] - m_{kj} \right\|^2 \tag{6}$$

The preconditioning stage is not necessary as we can linearly-extrapolate the camera pose estimates from previous frames to achieve good initialization.

**Clamping:** Since epipolar lines do not make sense in the multi-view setting, we reset the optical flow using the pose and depth from the bundle adjustment.

**RNN changes:** We include mechanisms in the RNN to share latent features between updates if they stem from the same anchor point. Specifically, we use the message passing and temporal convolutions from DPVO [45].

### 3.5. Trajectory Alignment

In the Multi-Session SLAM problem, we estimate a relative Sim(3) between pairs of disjoint trajectories in order to align them. All anchor points already have a depth estimate $d_k$ as result of the bundle adjustment (eq. 6) from the VO system. We use our two-view method to estimate the relative rotation and translation direction, and compute the translation magnitude and relative scaling by comparing the $d_k$'s to the depth from the triangulated two-view matches.

*1) Estimate relative rotation, translation direction, and matches.* We first retrieve a candidate image pair $(i, j)$ using NetVLAD [1], one from each trajectory, and apply our two-view model to estimate their relative rotation and translation direction.

*2) Align the depth from the two-view and VO operators.* We previously defined $d_k$ as the depth output of the VO system. Let $d'_k$ be the triangulated depth obtained from our two-view matches:

$$d'_k = \begin{cases} triang(\mathbf{R}_{i\to j}, \mathbf{t}_{i\to j}, a_k, m_{kj}) & : k \in K(i) \\ triang(\mathbf{R}_{j\to i}, \mathbf{t}_{j\to i} a_k, m_{ki}) & : k \in K(j) \end{cases} \tag{7}$$

We solve

$$\underset{s_i \in \mathbb{R}>0}{\arg\max} \sum_{k \in K(i)} \mathbf{1}\left[\frac{1}{\lambda} < \frac{d_k}{s_i \cdot d'_k} < \lambda\right] \tag{8}$$

to recover the translation magnitude, where the hyperparameter $\lambda = 1.05$. In layman terms, eq. 9 seeks *what translation magnitude would align the most triangulated points to the existing 3D map?* We can obtain a reasonably good solution to eq 9 by brute-force checking $s_i = d_k/d'_k \quad \forall k \in K(i)$, similar to RANSAC. If the number of inliers is too small, we retry on a new candidate pair. Conversely, the scale-difference is $s_i/s_j$, where $s_j$ is estimated the same way, but for anchor points in $K(j)$:

$$\underset{s_j \in \mathbb{R}>0}{\arg\max} \sum_{k \in K(j)} \mathbf{1}\left[\frac{1}{\lambda} < \frac{d_k}{s_j \cdot d'_k} < \lambda\right] \tag{9}$$

Given the rotation $\mathbf{R}_{j\to i}$ and translation $\mathbf{t}_{j\to i}$ from the two-view solver layers, the resulting transformation can be used to align the two trajectories:

$$\mathcal{S}_{j\to i} = \begin{bmatrix} \frac{s_i}{s_j}\mathbf{R} & s_j\mathbf{t} \\ 0 & 1 \end{bmatrix} \in \text{Sim}(3) \tag{10}$$

The graphs are then merged by concatenating all buffers.

### 3.6. Training

We train our backbone separately for VO and for two-view pose. It is trained using a matching loss and a pose loss. The pose loss for the two-view training

$$\mathcal{L}_{pose2V} = \sum_{t=1}^{12} cos^{-1}\left(\bar{\mathbf{t}}^t \cdot \bar{\mathbf{t}}^{\mathbf{gt}}\right) + \alpha \left\|[(\mathbf{R}^t)^T \mathbf{R}^{\mathbf{gt}}]\right\|_{SO(3)} \tag{11}$$

penalizes angle error for both the predicted translation direction and predicted orientation. The pose loss for the VO training is the SE3 manifold distance between the predicted and ground-truth poses:

$$\mathcal{L}_{poseVO} = \sum_{t=1}^{T} \left\|[(\mathbf{G}^t)^{-1} \mathbf{G}^{\mathbf{gt}}]\right\|_{SE(3)} \tag{12}$$

The matching loss

$$\mathcal{L}_{matching} = \frac{1}{|N|} \sum_{a}^{N} \sum_{t=1}^{T} \left\|m_a^t - m_a^{\mathbf{gt}}\right\|_2 \tag{13}$$

is a standard endpoint-error [24, 43], applied only on pixels which have valid depth and are verified to be visible in both images. Our update operator is applied 12 times per training example; supervision is applied after every update (See Fig. 5). The final loss is $\mathcal{L} = \mathcal{L}_{pose} + \beta \cdot \mathcal{L}_{matching}$.

Following prior work [23], we pre-train our two-view method on synthetic homographies (without pose loss) for two epochs on the Oxford-Paris 1M Distractors dataset [29]. We then tune our model on a $^{50}/_{50}$ mixture of Scannet [6] and Megadepth [21] for 100,000 steps and a batch size of 120, using 10 A6000 GPUs for 5 days. The VO backbone is trained using the procedure from [45] on TartanAir.

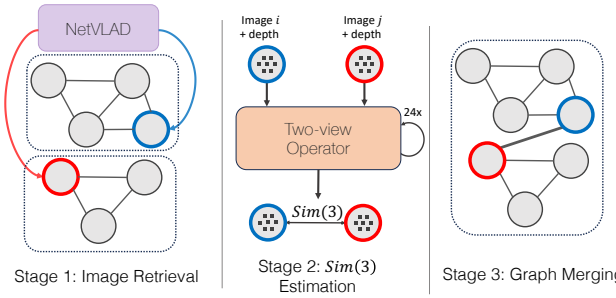


Figure 7. High-level overview of our approach to aligning disjoint trajectories. We use NetVLAD [1] to retrieve two co-visible images, one from each trajectory, and apply our two-view update operator to estimate a relative Sim(3) alignment, following the procedure detailed in Sec 3.5. We then transform the second trajectory into the reference frame of the first, and merge all buffers.

### 3.7. Multi-Session SLAM System

**Overview:** To perform Multi-Session SLAM, we use our VO-trained backbone to perform visual odometry and global optimization; our two-view backbone connects disjoint trajectories following the procedure in Sec 3.5. Our full system builds the pose graph incrementally: new video frames inserted into the factor graph as they are received, and unjoined trajectories are aligned and merged as soon as a connection is found.

**VO Frontend:** Our VO frontend extracts features and estimates camera poses and depth for incoming frames, operating on a sliding window covering the most recently observed 22 keyframes. Keyframing occurs retroactively on the $4^{th}$ oldest keyframe if it is found to be redundant. New poses are initialized using a linear-motion model, and depth is copied from the previous frame. Like DPVO [45], we initialize after observing 8 frames with significant motion.

**Global Optimization:** We perform global optimization by introducing proximity factors using the existing pose and depth measurements, following the approach from DROID-SLAM [44], and update the matches and poses/depth using the VO backbone. We only run the backend periodically, and after joining trajectories.

**Trajectory Joining:** We query the database of NetVLAD descriptors to find co-visible frames. If several pairs are retrieved, we compute relative poses in a mini-batch and use the estimate with the highest inlier-ratio.

## 4. Experiments

We evaluate the performance of our two-view method in isolation, and our Multi-Session SLAM system as a whole.

### 4.1. Two-View Evaluation

We evaluate our two-view method on two popular relative pose benchmarks: The Scannet [6] 1500 and

| | Overall Approach | Pose error AUC [%] ↑ | | |
|---|---|---|---|---|
| | | @5° | @10° | @20° |
| Superglue [34] | Matching ↓ RANSAC [15] (LO-RANSAC [18]) | 16.2(17.7) | 33.8(35.6) | 51.8(54.2) |
| LightGlue [23] | | 16.5(19.4) | 33.4(36.9) | 50.1(53.5) |
| LoFTR [41] | | 22.1(25.7) | 40.1(45.0) | 47.6(61.4) |
| MatchFormer [47] | | 24.3(27.3) | 43.9(47.6) | 61.4(64.9) |
| ASpanFormer [5] | | 25.6(28.4) | 46.0(48.8) | 63.3(65.8) |
| Roessle&Nießner [31] | Matching→W8PA | 20.7 | 41.6 | 61.7 |
| Roessle&Nießner [31] | Matching→W8PA→BA | 25.7 | 47.2 | 66.4 |
| Ours | Optical-Flow ↗ ↘ Clamp ← Solver | **30.5** | **50.9** | **67.5** |

Table 1. Two-view results on Scannet [6]. Existing methods perform matching as pre-processing, then use a solver; our method is iterative. Results in parenthesis use the LO-RANSAC [18] estimator. "W8PA→BA" = weighted 8-point-algorithm and bundle-adjustment. Our design outperforms prior approaches. We use the same model weights as in Tab. 2.

| | Overall Approach | Pose error AUC [%] ↑ | | |
|---|---|---|---|---|
| | | @5° | @10° | @20° |
| Superglue [34] | Matching ↓ RANSAC [15] (LO-RANSAC [18]) | 49.7(65.8) | 67.1(78.7) | 80.6(87.5) |
| LightGlue [23] | | 49.9(66.7) | 67.0(79.3) | 80.1(87.9) |
| LoFTR [41] | | 52.8(66.4) | 69.2(78.6) | 81.2(86.5) |
| MatchFormer [47] | | 53.3(66.5) | 69.7(78.9) | 81.8(87.5) |
| ASpanFormer [5] | | 55.3(**69.4**) | 71.5(**81.1**) | 83.1(**88.9**) |
| Roessle&Nießner [31] | Matching→W8PA | 46.9 | 62.8 | 76.3 |
| Roessle&Nießner [31] | Matching→W8PA→BA | 61.2 | 74.9 | 85.0 |
| Ours | Optical-Flow ↗ ↘ Clamp ← Solver | 60.2 | 72.3 | 81.0 |

Table 2. Two-view results on Megadepth [21]. Existing methods perform matching as pre-processing, then use a solver; our method is iterative. "W8PA→BA" = weighted 8-point-algorithm and bundle-adjustment. Results in parenthesis use the LO-RANSAC [18] estimator. Our radically different design leads to better results indoors (Tab. 1), but is less competitive in photo-tourism settings where high-volume matching is easier and therefore RANSAC-based methods fare better. We use the same model weights as in Tab. 1.

MegaDepth [21] 1500 test datasets. We compare to existing matching networks [5, 23, 34, 41, 47], which are typically in service of improving COLMAP [38]-based SfM pipelines [22, 33]. In contrast, our approach is in service of Multi-Session SLAM. We report pose error AUC, as is done in prior matching work (higher is better). We use the same configuration and model weights on both datasets.

Most existing methods perform matching as a pre-processing step using a deep network, then estimate pose with a minimal solver using RANSAC [15, 18]. Roessle and Nießner [31] replace the RANSAC optimizer with a weighted-8-point algorithm and bundle adjustment. In contrast, our two-view method couples the prediction of pose and matches, refining both over many iterations, and regresses optical flow instead of keypoint affinities.

**Scannet:** We report two-view results on Scannet [6] in Tab. 1. Our approach outperforms existing methods on Scannet, which contains fewer salient keypoints and many texture-less surfaces and motion blur.

**MegaDepth:** We report two-view results on Megadepth [21] in Tab. 2. On Megadepth, the matching problem is substantially easier since photo-tourism images contain many salient keypoints, resulting in better performance across all methods; our approach is on-par with existing approaches using the OpenCV RANSAC pose estimator [15], but not the LO-RANSAC [18] estimator.

**Ablations:** We ablate on various aspects of our proposed architecture in Tab. 3 and show that they lead to improved performance. Specifically, we show that pre-conditioning and the SED solver lead to better accuracy when used together, and that clamping the matches to the epipolar lines is also important.

### 4.2. Multi-Session SLAM evaluation

We evaluate our approach to Multi-Session SLAM on the EuRoC-MAV [2] and ETH3D [39] datasets, since they provide camera poses under a single global reference. Following the evaluation for single-video SLAM, we align the final predictions to the ground truth by computing a global 7-DOF alignment to account for the guage freedoms. We report RMSE ATE[m] in metric units. We sample 96 anchors in each video frame using a mix of random and Superpoint [8] keypoints. Our two-view method uses those same anchors to join pairs of trajectories. We use the same weights and configuration on both datasets.

**EuRoC-MAV:** In Tab. 9 and Fig. 10, we report results on the EuRoC MAV dataset [2]. Following the evaluation from [4], we compare on four groups of disjoint trajectories across the three distinct environments: 3 in the Machine Hall, 5 in the Machine Hall, 3 in Vicon 1, and 3 in Vicon 2. Ground-truth poses are obtained using a laser tracker and Vicon cameras. These video sequences are long, with the entirety of the Machine-Hall group being 11.5 minutes of video (13.7K frames) spread across 5 sequences. Our approach achieves significantly lower error than ORB-SLAM3 on all groups, including 11x lower error on the Vicon 2 sequences (0.024 vs 0.284). Video is recorded at 20-FPS; all methods run in real-time. We also compare to CCM-SLAM [36], which only provides results on the on MH01-03 sequence, the mono-inertial ORB-SLAM3, and the mono-inertial VINS [28]. We outperform the other approaches on all trajectory groups.

**ETH3D:** In Tab. 4, we report results on the training set from ETH3D [39]. We compare across 5 unique scenes composed of multiple trajectories: 1-4 from *Sofa*, 3&4 from *Table*, 1-3 from *Plant Scene*, 1&2 from *Einstein*, and 2&3 from *Planar*. Sequences are excluded if they belong to the official ETH3D test set (with no ground-truth) or are from a different scene altogether. The sequences are trimmed

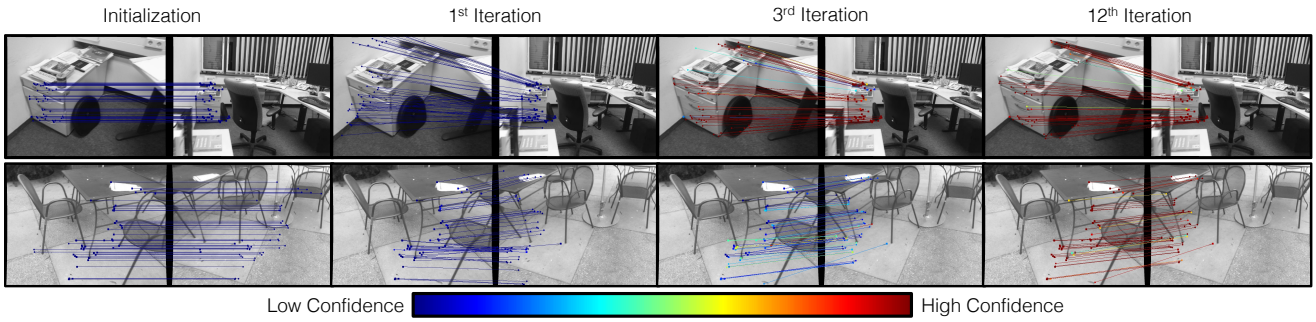


Figure 8. Qualitative results on Scannet. Our two-view variant is able to estimate accurate relative poses across wide camera baselines. It initializes all matches with uniform depth and identity relative pose. Progressive applications of our update operator lead to more accurate matches and higher predicted confidence. Some image pairs (row 2) take more iterations to converge than others (row 1).

| Scene name | MH01-03 | MH01-05 | V101-103 | V201-203 | Cam. |
|---|---|---|---|---|---|
| # Disjoint Trajectories | 3 | 5 | 3 | 3 | hz |
| Ours Mono-Visual | **0.022** | **0.036** | **0.031** | **0.024** | 20 |
| CCM-SLAM [36] Mono-Visual | 0.077 | - | - | - | **38** |
| ORB-SLAM3 [4] Mono-Visual | 0.030 | 0.058 | 0.058 | 0.284 | 20 |
| VINS [28] Mono-Inertial | - | 0.210 | - | - | - |
| ORB-SLAM3 Mono-Inertial | 0.037 | 0.065 | 0.040 | 0.048 | 20 |

Figure 9. Multi-Session SLAM evaluation on EuRoC-MAV [2] using RMSE ATE[m] ↓. We compare to existing visual and inertial Multi-Session SLAM approaches, all using monocular video input. Our method outperforms existing approaches on all sequence groups. Results for baseline approaches were obtained from [4].

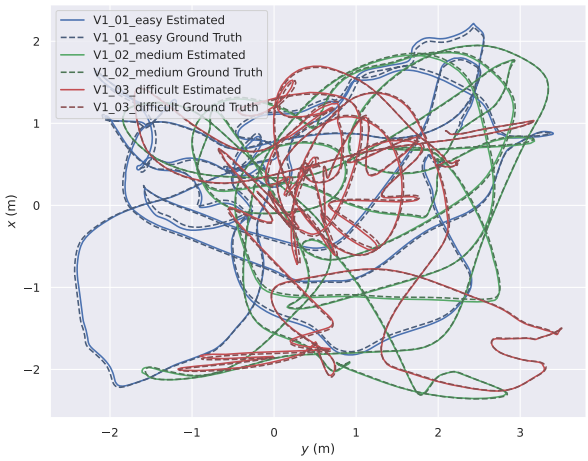


Figure 10. Our prediction on the Vicon1 [2] sequences. Our method is accurate and robust to chaotic motion.

from a single, longer video; to make joining the trajectories non-trivial, we reverse every other sequence, ensuring that there is a sufficient disconnect between subsequent videos. This is a novel benchmark, so [28, 36] do not report results.

| Ablation Experiment | | Pose error AUC @10°[%] ↑ | |
|---|---|---|---|
| | | Megadepth Val | Scannet Val |
| Baseline | Full model | **53.3** | **33.9** |
| Architecure | Shallower correlation pyramid | 44.3 | 27.1 |
| | No attention in update-op | 48.6 | 30.5 |
| | No ReLU-attn in feat-extractor | 47.4 | 29.7 |
| Solver | No pre-conditioning | 35.5 | 20.6 |
| | No SED solver | 23.8 | 13.1 |
| | No clamping step | 21.9 | 10.7 |

Table 3. Two-view ablation experiments on two validation sets. Our proposed RNN and feature extractor changes from DPVO improve the result. Removing components from our proposed solver design decreases accuracy, as does removing the clamping step.

| Scene name | Sofa | Table | Plant Scene | Einstein | Planar |
|---|---|---|---|---|---|
| # Disjoint Trajectories | 4 | 2 | 3 | 2 | 2 |
| ORB-SLAM3 [4] Mono-Visual | FAIL (no init) | 0.018 | FAIL (init→lost) | FAIL (init→lost) | **0.010** |
| Ours Mono-Visual | **0.010** | **0.010** | **0.021** | **0.032** | 0.047 |

Table 4. Multi-Session SLAM evaluation on ETH3D [39] using RMSE ATE[m] ↓. Our method is robust to catastrophic failures.

ORB-SLAM3 fails on *Sofa*, *Plant Scene*, and *Einstein* for various reasons (couldn't initialize, or lost the feature tracks). Our approach outperforms ORB-SLAM3 on 4/5 groups, and does not fail on any. ORB-SLAM3 only succeeds on 2/5 groups.

## 5. Conclusion

We introduce a new method for mono-visual Multi-Session SLAM. Our system utilizes a novel backbone which can estimate two-view pose and perform visual odometry. We leverage a novel differentiable solver which minimizes the symmetric epipolar distance. We compare against existing approaches and show strong performance across several datasets. This work was partially supported by IARPA and the National Science Foundation.

## References


[1] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016. 5, 6

[2] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35 (10):1157–1163, 2016. 1, 7, 8

[3] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016. 2

[4] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021. 1, 2, 7, 8

[5] Hongkai Chen, Zixin Luo, Lei Zhou, Yurun Tian, Mingmin Zhen, Tian Fang, David Mckinnon, Yanghai Tsin, and Long Quan. Aspanformer: Detector-free image matching with adaptive span transformer. In *European Conference on Computer Vision*, pages 20–36. Springer, 2022. 7

[6] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 6, 7

[7] Hayyan Afeef Daoud, Aznul Qalid Md. Sabri, Chu Kiong Loo, and Ali Mohammed Mansoor. Slamm: Visual monocular slam with continuous mapping using multiple maps. *PloS one*, 13(4):e0195878, 2018. 2

[8] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018. 3, 7

[9] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014. 2

[10] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017. 2

[11] Mohammed E Fathy, Ashraf S Hussein, and Mohammed F Tolba. Fundamental matrix estimation: A study of error criteria. *Pattern Recognition Letters*, 32(2):383–391, 2011. 4

[12] Dorian Gálvez-López and Juan D Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, 2012. 2

[13] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003. 4, 1

[14] Richard I Hartley. In defense of the eight-point algorithm. *IEEE Transactions on pattern analysis and machine intelligence*, 19(6):580–593, 1997. 2, 4

[15] Itseez. Open source computer vision library. https://github.com/itseez/opencv, 2015. 7

[16] Mathieu Labbé and François Michaud. Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of field robotics*, 36(2):416–446, 2019. 1

[17] Mathieu Labbé and François Michaud. Multi-session visual slam for illumination-invariant re-localization in indoor environments. *Frontiers in Robotics and AI*, 9:801886, 2022. 2

[18] Viktor Larsson and contributors. PoseLib - Minimal Solvers for Camera Pose Estimation, 2020. 2, 7

[19] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Ep n p: An accurate o (n) solution to the p n p problem. *International journal of computer vision*, 81:155–166, 2009. 2

[20] Stefan Leutenegger, Paul Furgale, Vincent Rabaud, Margarita Chli, Kurt Konolige, and Roland Siegwart. Keyframe-based visual-inertial slam using nonlinear optimization. *Proceedings of Robotis Science and Systems (RSS) 2013*, 2013. 2

[21] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 6, 7

[22] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-perfect structure-from-motion with featuremetric refinement. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5987–5997, 2021. 7

[23] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. LightGlue: Local Feature Matching at Light Speed. In *ICCV*, 2023. 6, 7

[24] Lahav Lipson, Zachary Teed, and Jia Deng. Raft-stereo: Multilevel recurrent field transforms for stereo matching. In *2021 International Conference on 3D Vision (3DV)*, pages 218–227. IEEE, 2021. 6

[25] Xiao Xin Lu. A review of solutions for perspective-n-point problem in camera pose estimation. In *Journal of Physics: Conference Series*, page 052009. IOP Publishing, 2018. 2

[26] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017. 2

[27] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 2

[28] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018. 1, 2, 7, 8

[29] F. Radenović, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *CVPR*, 2018. 6

[30] Rene Ranftl and Vladlen Koltun. Deep fundamental matrix estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2, 4

[31] Barbara Roessle and Matthias Nießner. End2end multi-view feature matching with differentiable pose optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 477–487, 2023. 2, 7

[32] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011. 2

[33] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12716–12725, 2019. 2, 7

[34] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. 2, 7

[35] Patrik Schmuck and Margarita Chli. Multi-uav collaborative monocular slam. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3863–3870. IEEE, 2017. 2

[36] Patrik Schmuck and Margarita Chli. Ccm-slam: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams. *Journal of Field Robotics*, 36(4):763–781, 2019. 1, 2, 7, 8

[37] Thomas Schneider, Marcin Dymczyk, Marius Fehr, Kevin Egger, Simon Lynen, Igor Gilitschenski, and Roland Siegwart. maplab: An open framework for research in visual-inertial mapping and localization. *IEEE Robotics and Automation Letters*, 3(3):1418–1425, 2018. 1

[38] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 7

[39] Thomas Schöps, Torsten Sattler, and Marc Pollefeys. BAD SLAM: Bundle adjusted direct RGB-D SLAM. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 7, 8

[40] Hauke Strasdat, J Montiel, and Andrew J Davison. Scale drift-aware large scale monocular slam. *Robotics: science and Systems VI*, 2(3):7, 2010. 2

[41] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8922–8931, 2021. 7

[42] Zachary Teed and Jia Deng. Deepv2d: Video to depth with differentiable structure from motion. *arXiv preprint arXiv:1812.04605*, 2018. 2

[43] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020. 1, 2, 3, 6

[44] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021. 1, 2, 6

[45] Zachary Teed, Lahav Lipson, and Jia Deng. Deep patch visual odometry. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 1, 2, 5, 6

[46] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04): 376–380, 1991. 2

[47] Qing Wang, Jiaming Zhang, Kailun Yang, Kunyu Peng, and Rainer Stiefelhagen. Matchformer: Interleaving attention in transformers for feature matching. In *Proceedings of the Asian Conference on Computer Vision*, pages 2746–2762, 2022. 7

[48] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4909–4916. IEEE, 2020. 4, 5

# Appendix

## A. Gradient Computation for the SED Solver

$$d \coloneqq l_x^2 + l_y^2 \tag{14}$$

$$\zeta \coloneqq l_x m_x + l_y m_y + l_z \tag{15}$$

$$\frac{\partial err(m,l)}{\partial l} = \begin{bmatrix} \left(\frac{-2l_x^2\zeta}{d^2} + \frac{l_x m_x}{d} + \frac{\zeta}{d}\right) & \left(\frac{-2l_x l_y \zeta}{d^2} + \frac{l_x m_y}{d}\right) & \left(\frac{l_x}{d}\right) \\ \left(\frac{-2l_y l_x \zeta}{d^2} + \frac{l_y m_x}{d}\right) & \left(\frac{-2l_y^2\zeta}{d^2} + \frac{l_y m_y}{d} + \frac{\zeta}{d}\right) & \left(\frac{l_y}{d}\right) \end{bmatrix} \tag{16}$$

Let $\bar{a}$ be the anchor location in homogenous coordinates. The partials for $l$ w.r.t. $\mathbf{F}$ are

$$\frac{\partial l}{\partial \mathbf{F}} = \frac{\partial \mathbf{F}\bar{a}}{\partial \mathbf{F}} = \bar{a}^\top \otimes \mathbb{I} \in \mathbb{R}^{3\times(3\times3)} \tag{17}$$

The partials for the fundamental matrix w.r.t. the essential matrix, given the known calibration matrix $K$:

$$\frac{\partial \mathbf{F}}{\partial \mathbf{E}} = \frac{\partial\left[(K^{-1})^\top \mathbf{E} K^{-1}\right]}{\partial \mathbf{E}} = (K^{-1})^\top \otimes K^{-1} \in \mathbb{R}^{(3\times3)\times(3\times3)} \tag{18}$$

The $3 \times 3$ essential matrix in terms of the input rotation $\mathbf{R}$ and translation $\mathbf{t}$, and the local updates $\xi_\mathbf{R}$ and $\xi_\mathbf{t}$, is

$$\mathbf{E} = (\mathbf{e}^{\xi_\mathbf{R}}\mathbf{R})^\top [\mathbf{e}^{\xi_\mathbf{t}}\mathbf{t}]_\times \tag{19}$$

The derivatives of $\xi_\mathbf{R}$ and $\xi_\mathbf{t}$ are taken at the identity, so they are equal to 0 when treated as a constant. To compute the partial of this essential matrix w.r.t. $\xi_\mathbf{R}$ at 0:

$$\mathbf{tx} \coloneqq [\mathbf{t}]_\times \in \mathbb{R}^{3\times3} \tag{20}$$

$$\frac{\partial \mathbf{E}}{\partial \xi_\mathbf{R}} = \frac{\partial (e^{\xi_\mathbf{R}}\mathbf{R})^\top [\mathbf{t}]_\times}{\xi_\mathbf{R}} = \begin{bmatrix} \mathbf{R}^\top[\mathbf{tx}_{c1}]_\times \\ \mathbf{R}^\top[\mathbf{tx}_{c2}]_\times \\ \mathbf{R}^\top[\mathbf{tx}_{c3}]_\times \end{bmatrix}^\top \in \mathbb{R}^{(3\times3)\times3} \tag{21}$$

Similarly, the partial w.r.t. $\xi_\mathbf{t}$ at 0:

$$\frac{\partial \mathbf{E}}{\partial \xi_\mathbf{t}} = \frac{\partial \mathbf{R}^\top [\mathbf{e}^{\xi_\mathbf{t}}\mathbf{t}]_\times}{\partial \xi_\mathbf{t}} = \frac{\partial \mathbf{R}^\top [\mathbf{e}^{\xi_\mathbf{t}}\mathbf{t}]_\times}{\partial (e^{\xi_\mathbf{t}}\mathbf{t})} \frac{\partial (e^{\xi_\mathbf{t}}\mathbf{t})}{\partial \xi_\mathbf{t}} = \mathbf{R}^\top \frac{\partial [\vec{n}]_\times}{\partial \vec{n}} (-\mathbf{tx}) \in \mathbb{R}^{(3\times3)\times3} \tag{22}$$

Putting it together with the chain rule:

$$\frac{\partial err(m,l)}{\partial \xi_\mathbf{R}} = \frac{\partial err(m,l)}{\partial l} \frac{\partial l}{\partial \mathbf{F}} \frac{\partial \mathbf{F}}{\partial \mathbf{E}} \frac{\partial \mathbf{E}(e^{\xi_\mathbf{R}}\mathbf{R}, e^{\xi_\mathbf{t}}\mathbf{t})}{\partial \xi_\mathbf{R}} \in \mathbb{R}^{2\times3} \tag{23}$$

$$\frac{\partial err(m,l)}{\partial \xi_\mathbf{t}} = \frac{\partial err(m,l)}{\partial l} \frac{\partial l}{\partial \mathbf{F}} \frac{\partial \mathbf{F}}{\partial \mathbf{E}} \frac{\partial \mathbf{E}(e^{\xi_\mathbf{R}}\mathbf{R}, e^{\xi_\mathbf{t}}\mathbf{t})}{\partial \xi_\mathbf{t}} \in \mathbb{R}^{2\times3} \tag{24}$$

$$\tag{25}$$

## B. Extracting Rotation and Translation from the Essential Matrix

To obtain the relative pose given the essential matrix $\mathbf{E}$, we follow the procedure prescribed in [13].

$$W \coloneqq \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{26}$$

$$Z \coloneqq \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{27}$$

$$U, \Sigma, V^\top = \text{SVD}(\mathbf{E}) \tag{28}$$

$$t = UZU^\top \quad (29) \qquad R1 = UWV^\top \quad (30) \qquad R2 = UW^\top V^\top \quad (31)$$

The four plausible solutions are

$$[(t, R1), (t, R2), (-t, R1), (-t, R2)] \quad (32)$$

During training, we choose the solution closest to the ground-truth. During inference, we choose the pose which triangulates the most points in front of the camera.

## C. Architecture Details

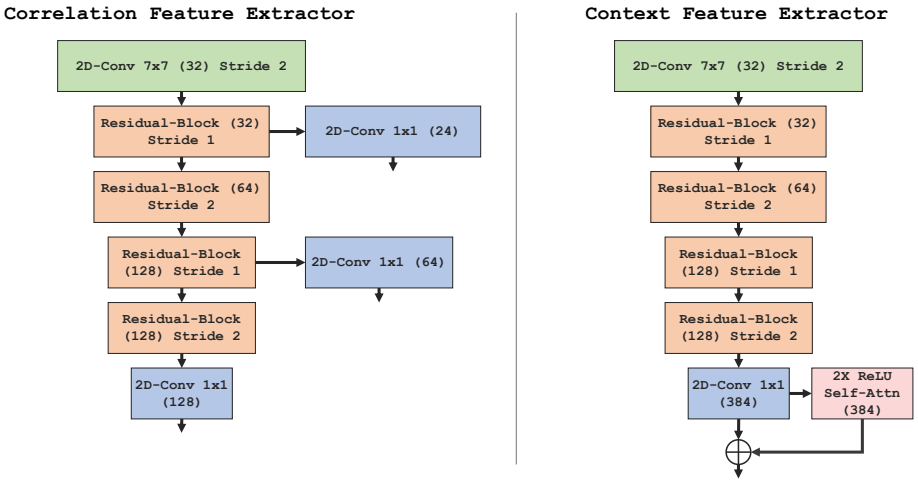


Figure 11. The architecture of our context and correlation feature extractors. Both feature extractors are residual networks. The context feature extractor also uses ReLU self-attention to propagate information across the image. The correlation features are used to evaluate visual similarity at multiple spatial resolutions. The numbers in parenthesis are the output feature dimensions.

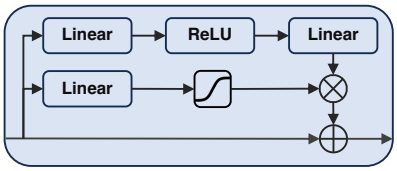


Figure 12. The gated residual unit.

**MLP-GRU** We depict the gated residual unit in Fig. 12. This is the same design from DPVO [45].

**Feature Extractors** We visualize the feature extractors in Fig. 11. The correlation features are produced at $1/2$, $1/4$ and $1/8$ the image resolution. The context features are extracted only at $1/8$ resolution, and have additional self-attention layers to propagate information over the image.

## D. Additional Qualitative Results


Figure 13. Additional Qualitative results on Scannet.