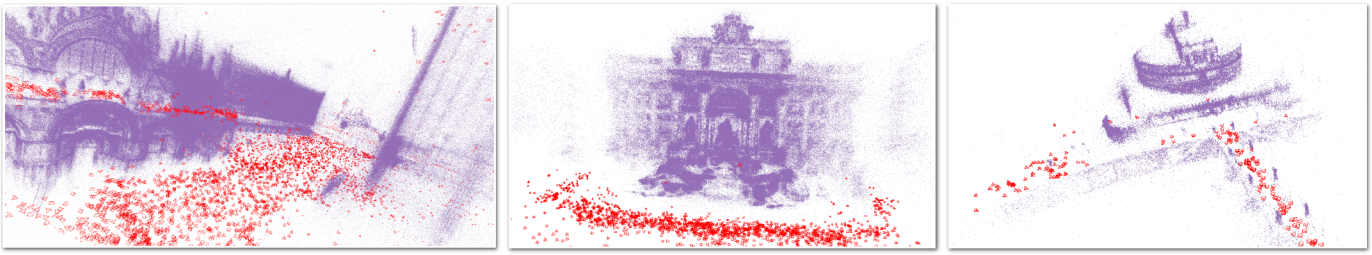
Robotics: Science and Systems 2025
Los Angeles, CA, USA, June 21–25, 2025

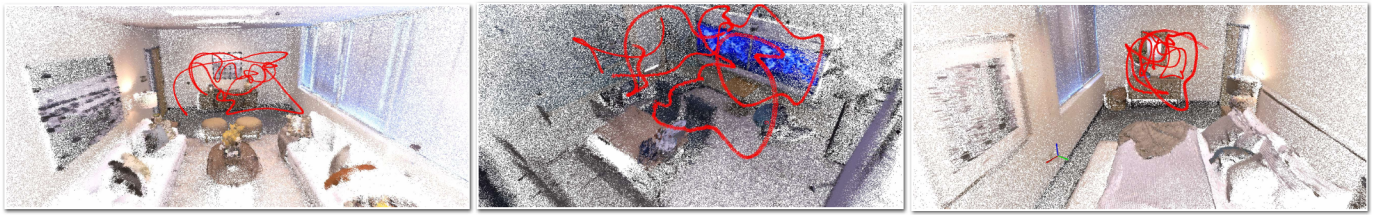
Building Rome with Convex Optimization

Haoyu Han and Heng Yang
School of Engineering and Applied Sciences, Harvard University

https://computationalrobotics.seas.harvard.edu/XM

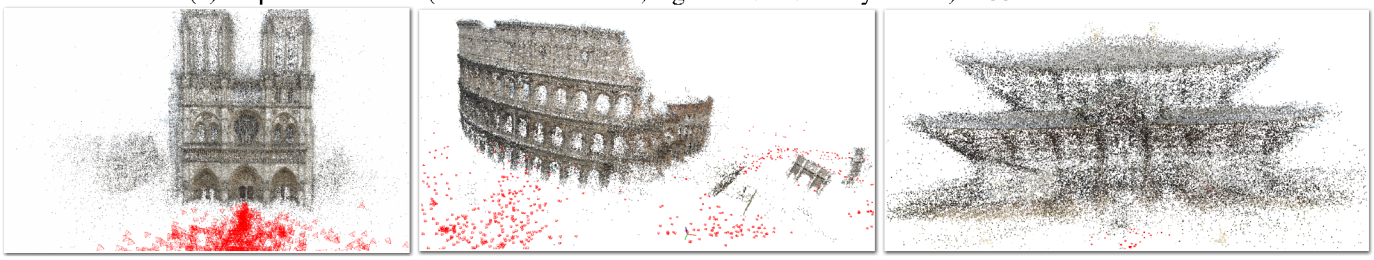


(a) BAL Dataset. From left to right: 10155, 1934, and 392 camera frames.

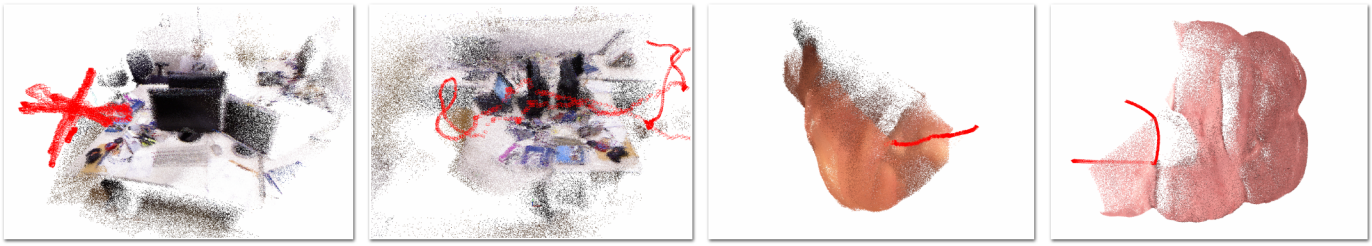


(b) Replica Dataset. All three cases have 2000 camera frames.

(c) Mip-Nerf Dataset (left: reconstruction, right: novel view synthesis). 185 camera frames.



(d) IMC Dataset. From left to right: 3765, 2063, and 904 camera frames.



Left (e) TUM Dataset (798 and 613 camera frames); Right (f) C3VD Dataset (370 and 613 camera frames).

Fig. 1: Faster, scalable, and initialization-free 3D reconstruction powered by conveX bundle adjustMent (XM).

***Abstract*—Global bundle adjustment is made easy by depth prediction and convex optimization. We (*i*) propose a scaled bundle adjustment (SBA) formulation that lifts 2D keypoint measurements to 3D with learned depth, (*ii*) design an empirically tight convex semidefinite programming (SDP) relaxation that solves SBA to certifiable global optimality, (*iii*) solve the SDP relaxation at extreme scale with Burer-Monteiro factorization and a CUDA-based trust-region Riemannian optimizer (dubbed XM), (*iv*) build a structure from motion pipeline with XM as the optimization engine and show that XM-SfM compares favorably with existing pipelines in terms of reconstruction quality while being significantly faster, more scalable, and initialization-free.**

I. Introduction

At the heart of modern structure from motion (SfM) and simultaneous localization and mapping (SLAM) sits *bundle adjustment* (BA), the procedure of reconstructing camera poses and 3D landmarks from 2D image keypoints.

Classical BA formulation. Consider a so-called *view graph* illustrated in Fig. 2 with two types of nodes: 3D points $p_k \in \mathbb{R}^3, k = 1, \dots, M$ and camera poses $(R_i, t_i) \in \mathrm{SE}(3), i = 1, \dots, N$. The set of edges $\mathcal{E}$ contains visibility information. An edge $(i,k) \in \mathcal{E}$ indicates the k-th 3D point is visible to the i-th camera, and a 2D keypoint measurement $u_{ik} \in \mathbb{R}^2$ has been obtained regarding the projection of the point onto the camera frame.[1] The bundle adjustment problem consists of estimating points and poses (i.e., p_k's and (R_i, t_i)'s on the nodes) using the 2D keypoint measurements (i.e., u_{ik}'s on the edges). This is often formulated as an optimization problem:

$$\min_{\substack{p_k \in \mathbb{R}^3, k=1,\dots,M \\ (R_i,t_i) \in \mathrm{SE}(3), i=1,\dots,N}} \sum_{(i,k)\in\mathcal{E}} \left\| u_{ik} - \pi(R_i p_k + t_i) \right\|^2, \qquad (1)$$

where $R_i p_k + t_i$ transforms the point p_k to the i-th camera frame, $\pi(v) := [v_1; v_2]/v_3$ divides the first two coordinates by the depth and projects the 3D point to 2D, and the sum of squared errors evaluates how well the reprojected 2D points agree with the measurements u_{ik} for all $(i,k) \in \mathcal{E}$. Problem (1) assumes the availability of a view graph, which is often obtained through feature detection and matching in SfM and SLAM pipelines (to be detailed in §IV). An important observation is that problem (1) can only be solved *up to scale*. This is because $\pi(R_i p_k + t_i) \equiv \pi(R_i(\alpha p_k) + (\alpha t_i))$ for any scalar $\alpha > 0$, i.e., scaling the points and translations by a factor of α does not change the objective value of problem (1).

Optimization challenges. Problem (1) is intuitive to formulate but extremely difficult to optimize. The difficulty comes from two challenges. First, problem (1) is highly nonconvex. The nonconvexity comes from both the nonconvex feasible set $\mathrm{SE}(3)$ and the nonconvex objective function due to the 2D reprojection function $\pi(\cdot)$. Second, problem (1) can have an extremely large scale. Both the number of camera poses N and the number of points M can range from hundreds to tens of thousands (*cf.* examples in Fig. 1). Due to these challenges, BA solvers such as Ceres [3], GTSAM [17], and accelerated variants [19, 36] require good initializations and can often get stuck in poor local minima (*cf.* §V-A). To address this, the popular SfM pipeline COLMAP [39] employs an *incremental* strategy which starts by reconstructing only two views (i.e., (1) with $N = 2$) and then sequentially registers additional camera images and associated 3D structure. Incremental SfM ensures stable initialization but makes the pipeline slow and hard to scale to large datasets (*cf.* §V where COLMAP requires several hours runtime). The recent *global* SfM pipeline GLOMAP [32] replaces the incremental process with rotation averaging and global positioning, but such initialization can still be time-consuming (see results in §V).

[1] We consider BA with calibrated cameras, i.e., u_{ik} has been normalized by camera intrinsics.

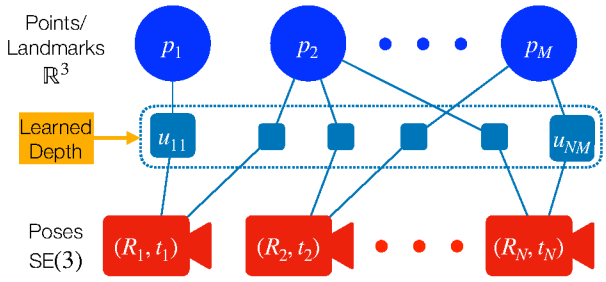


Fig. 2: A view graph for the bundle adjustment formulation (1). We propose a *scaled bundle adjustment* formulation (3) by lifting the 2D keypoints to 3D with learned depth.

Therefore, the motivating question of this paper is:

Can we design an algorithm that solves the global bundle adjustment problem (1) *without initialization and at scale?*

BA with learned depth. We start by designing an approximate formulation to problem (1) that is simpler to optimize. The key insight is to lift the 2D keypoints u_{ik} as approximate (and noisy) 3D keypoints leveraging off-the-shelf large-scale pretrained monocular depth prediction models, such as [35, 49, 10]. Formally, let $d_{ik} > 0$ be the predicted depth of the 2D keypoint u_{ik}, we generate a 3D keypoint measurement

$$\tilde{u}_{ik} = d_{ik} \begin{bmatrix} u_{ik} \\ 1 \end{bmatrix}, \quad \forall (i,k) \in \mathcal{E}. \qquad (2)$$

In our work, we select a *metric depth* model, which is designed to produce depth estimates that match the true depth when the model is perfectly trained. Further, due to imperfect depth prediction caused by potentially out-of-distribution test images, we incorporate a scaling parameter that allows subsequent optimization to correct erroneous predictions, leading to the following *scaled bundle adjustment* (SBA) formulation

$$\min_{\substack{p_k \in \mathbb{R}^3, k=1,\dots,M \\ s_i > 0, i=1,\dots,N \\ (R_i,t_i) \in \mathrm{SE}(3), i=1,\dots,N}} \sum_{(i,k)\in\mathcal{E}} w_{ik} \left\| R_i(s_i \tilde{u}_{ik}) + t_i - p_k \right\|^2, \qquad (3)$$

where s_i scales the predicted 3D keypoint $\tilde{u}_{ik}$ in (2), (R_i, t_i) transforms the scaled 3D keypoint to the global frame,[2] and the squared errors in the objective computes 3D distances to the landmarks p_k without the reprojection $\pi(\cdot)$. In (3), we allow weighting the squared errors by confidence values ($w_{ik} > 0$) from the depth prediction model.

Remark 1 (Connection to Other Perception Problems). *It is worth noting that (i) without the per-frame scaling, problem* (3) *recovers the multiple point cloud registration problem [15, 26]; (ii) when $N = 2$ (two frames), problem* (3) *reduces to (scaled) point cloud registration [25, 48].*

Through depth prediction, we removed the reprojection function $\pi(\cdot)$ from (1) and resolved one challenge. However, nonconvexity and large scale remain in problem (3).

[2] Although we used the same notation (R_i, t_i) in both problems (1) and (3), the two transformations are inverse to each other. (R_i, t_i) in (1) transforms landmarks from global frame to camera frame, while (R_i, t_i) in (3) transforms landmarks from camera frame to global frame.

Contributions. First, we show the remaining nonconvexity in problem (3) is "benign". Our strategy is to first rewrite (3) as a quadratically constrained quadratic program (QCQP), and then design a convex semidefinite program (SDP) relaxation, in the spirit of a growing family of SDP-enabled certifiable algorithms [47, 38, 52, 28, 6, 33, 44]. We show the SDP relaxation is empirically *tight*, i.e., globally optimal solutions of the nonconvex (3) can be computed from the convex SDP relaxation with optimality certificates. Second, we show the convex SDP relaxations can be solved at *extreme scale and speed*—faster and more scalable than even the best local solvers such as CERES. The enabling technique is to exploit the low-rankness of (tight) SDP optimal solutions via Burer-Monteiro (BM) factorization [13] and solve the resulting Riemannian optimization using a trust-region algorithm [11]. *For the first time, we implemented the trust-region Riemannian optimizer directly in C++/CUDA* and show the GPU implementation is up to 100 times faster than the state-of-the-art CPU-based MANOPT package [12] and can solve extreme-scale problems beyond the reach of MANOPT (e.g., $N > 10,000$ camera frames, see Fig. 1). We name our GPU solver XM (conveX bundle adjustMent). Third, we build a full SfM pipeline with XM as the optimization engine and various techniques from prior work, such as feature matching from COLMAP, view graph creation from GLOMAP, CERES refinement (i.e., warmstart CERES with XM's solutions for solving (1)), and outlier-robust filtering and estimation schemes [4, 20, 31, 40]. We test XM-SfM across six popular datasets and demonstrate that XM-SfM compares favorably with existing SfM pipelines in terms of reconstruction quality while being significantly faster, more scalable, and initialization-free, thanks to convex optimization and GPU-based implementation.

In summary, our contributions are:

- designing an empirically tight convex SDP relaxation for the scaled bundle adjustment problem (3);
- solving the convex SDP at extreme scales using BM factorization paired with a trust-region Riemannian optimizer directly implemented in C++/CUDA, *i.e.,* XM;
- creating a full SfM pipeline called XM-SfM that "builds Rome with convex optimization".

Paper organization. We derive the QCQP formulation for (3) and design the SDP relaxation in §II. We present BM factorization and the CUDA-based trust-region Riemannian optimizer in §III. We describe the XM-SfM pipeline in §IV and present experimental results in §V. We conclude in §VI.

II. QCQP AND CONVEX SDP RELAXATION

In this section, let us focus on solving the SBA problem (3) to certifiable global optimality. We proceed in two steps. In §II-A, we simplify the original formulation as a quadratically constrained quadratic program (QCQP) through a sequence of mathematical manipulations. In §II-B, we apply Shor's semidefinite relaxation to "convexify" the nonconvex QCQP.

Before we get started, we remove the ambiguity of problem (3) through anchoring.

Anchoring. Observe that one can choose $s_i \to 0, \forall i = 1,\dots,N$, $t_1 = \cdots = t_N = p_1 = \cdots = p_M = \mathbf{0}$, and the objective value of (3) can be set arbitrarily close to zero. Additionally, multiplying an arbitary rotation matrix on R_i, t_i, p_i does not change the objective of (3), leading to infinitely many solutions. To resolve these issues, we anchor the first frame and set $R_1 = \mathbf{I}_3, t_1 = \mathbf{0}, s_1 = 1$.

A. QCQP Formulation

We first show that the unconstrained variables in (3), namely the translations t_i and the 3D landmarks p_k, can be "marginalized out", leading to an optimization problem only concerning the scaling factors and 3D rotations.

Proposition 2 (Scaled-Rotation-Only Formulation)**.** *Problem (3) is equivalent to the following optimization*

$$\begin{aligned} \rho^\star = \min \quad & \operatorname{tr}\left(QU^\mathsf{T}U\right) && \text{(4a)}\\ \textit{subject to} \quad & U = \begin{bmatrix} \mathbf{I}_3 & s_2R_2 & \cdots & s_NR_N \end{bmatrix} && \text{(4b)}\\ & s_i > 0, R_i \in \mathrm{SO}(3), \quad i = 2,\cdots,N && \text{(4c)} \end{aligned}$$

where $Q \in \mathbb{S}^{3N}$ is a constant and symmetric "data matrix" whose expression is given in Appendix A-A.

Let $U^\star$ represent the optimal solution of (4) and let $t = [t_1;\dots;t_N] \in \mathbb{R}^{3N}$ be the concatenation of translations, $p = [p_1;\dots;p_M] \in \mathbb{R}^{3M}$ be the concatenation of landmark positions. The optimal translations $t^\star$ and landmark positions $p^\star$ of problem (3) can be recovered from $U^\star$ as follows:

$$\begin{bmatrix} t^\star \\ p^\star \end{bmatrix} = (A \otimes \mathbf{I}_3)\operatorname{vec}(U^\star). \tag{5}$$

where the expression of A can be found in Appendix A-A.

Proof: See Appendix A-A. ∎

Proposition 2 reformulates the SBA problem (3) as a lower-dimensional problem (4). However, problem (4) is not a QCQP because the objective function is a degree-four polynomial in s_i and R_i. In the next step, we show that it is possible to combine the "scaled rotation" s_iR_i together as a new variable, effectively reducing the degree of the polynomial.

Proposition 3 (QCQP Formulation)**.** *Define the set of scaled orthogonal group as*

$$\mathfrak{s}\mathrm{O}(3) = \{\bar{R} \in \mathbb{R}^{3\times 3} \mid \exists s > 0, R \in \mathrm{O}(3) \text{ s.t. } \bar{R} = sR\}. \tag{6}$$

The set $\mathfrak{s}\mathrm{O}(3)$ can be described by quadratic constraints

$$\bar{R} = \begin{bmatrix} c_1 & c_2 & c_3 \end{bmatrix} \in \mathfrak{s}\mathrm{O}(3) \iff \begin{cases} c_1^\mathsf{T}c_1 = c_2^\mathsf{T}c_2 = c_3^\mathsf{T}c_3 \\ c_1^\mathsf{T}c_2 = c_2^\mathsf{T}c_3 = c_3^\mathsf{T}c_1 = 0. \end{cases} \tag{7}$$

Consider the following QCQP

$$\begin{aligned} \rho^\star_{\mathsf{QCQP}} = \min \quad & \operatorname{tr}\left(QU^\mathsf{T}U\right) && \text{(8a)}\\ \textit{subject to} \quad & U = \begin{bmatrix} \mathbf{I}_3 & \bar{R}_2 & \cdots & \bar{R}_N \end{bmatrix} && \text{(8b)}\\ & \bar{R}_i \in \mathfrak{s}\mathrm{O}(3),\ i = 2,\dots,N. && \text{(8c)} \end{aligned}$$

and let $U^\star = [\mathbf{I}_3, \bar{R}_2^\star, \dots, \bar{R}_N^\star]$ be a global optimizer. If

$$\det \bar{R}_i^\star > 0, i = 2,\dots,N, \tag{9}$$

then $U^\star$ is a global minimizer to problem (4).

Proof: See Appendix A-B. ■

It is clear that

$$\rho^\star_{\mathsf{QCQP}} \leq \rho^\star \tag{10}$$

because from (4) to (8) we have relaxed the $\mathrm{SO}(3)$ constraint to $\mathrm{O}(3)$. After solving (8), if there exists some index i such that $\det \bar{R}_i^\star < 0$, we can extract a feasible solution to (4) by projecting onto $\mathrm{SO}(3)$ after extracting the scaling from $\bar{R}_i^\star$.

Remark 4 (Bad Local Minimum)**.** *Problem (8) is a nonconvex QCQP. In fact, it is a smooth Riemannian optimization by identifying $\mathfrak{so}(3)$ as a product manifold of the positive manifold and the orthogonal group. Therefore, one can directly use, e.g.,* MANOPT *to solve (8) (and also (4)). However, as we will show in §V on the Mip-Nerf 360 dataset [7], directly solving (8) can get stuck in bad local minima.*

This motivates and necessitates convex relaxation.

B. Convex SDP Relaxation

For any QCQP, there exists a convex relaxation known as Shor's semidefinite relaxation [46, Chapter 3]. The basic idea is fairly simple: by creating a matrix variable $X := U^\mathsf{T} U$ that is quadratic in the original variable U, problem (8) becomes *linear* in X. The convex relaxation proceeds by using convex positive semidefinite constraints and linear constraints to properly enforce the matrix variable X.

Proposition 5 (SDP Relaxation)**.** *The following semidefinite program (SDP)*

$$\rho^\star_{\mathsf{SDP}} = \min_{X \in \mathbb{S}^{3N}} \ \operatorname{tr}(QX) \tag{11a}$$

$$\text{subject to} \quad X = \begin{bmatrix} \mathbf{I}_3 & \cdots & * \\ \vdots & \ddots & \vdots \\ * & \cdots & \alpha_N \mathbf{I}_3 \end{bmatrix} \succeq 0 \tag{11b}$$

is a convex relaxation to (8), i.e.,

$$\rho^\star_{\mathsf{SDP}} \leq \rho^\star_{\mathsf{QCQP}}. \tag{12}$$

Let $X^\star$ be a global minimizer of (11).

If $\operatorname{rank}(X^\star) = 3$, then $X^\star$ can be factorized as $X^\star = (\bar{U}^\star)^\mathsf{T} \bar{U}^\star$, and it holds[3]

$$\bar{U}^\star = \begin{bmatrix} \bar{R}_1^\star & \bar{R}_2^\star & \cdots & \bar{R}_N^\star \end{bmatrix} \in O(3) \times \mathfrak{so}(3)^{N-1}. \tag{13}$$

Define

$$U^\star = (\bar{R}_1^\star)^\mathsf{T} \bar{U}^\star, \tag{14}$$

then $U^\star$ is a global optimizer to (8). In this case, we say the relaxation is tight *or* exact*.*

Proof: See Appendix A-C. ■

If $\operatorname{rank}(X^\star) > 3$, then we can extract feasible solutions to (8) by taking the top three eigenvectors of $X^\star$ and project the corresponding entries to $\mathfrak{so}(3)$ and further to scaled rotations, a step that is typically called *rounding*. Denote $\hat{U}$ as the rounded solution that is feasible for (4), we can evaluate the objective of (4) at $\hat{U}$ and denote it $\hat{\rho}$. Combining the chain of inequalities from (10) and (12), we get

$$\rho^\star_{\mathsf{SDP}} \leq \rho^\star_{\mathsf{QCQP}} \leq \rho^\star \leq \hat{\rho}, \tag{15}$$

where the last inequality follows from $\hat{U}$ is a feasible solution for the minimization problem (4). Assuming we can solve the convex SDP, we compute $\rho^\star_{\mathsf{SDP}}$ and $\hat{\rho}$ at both ends of the inequality (15), allowing us to evaluate a relative suboptimality:

$$\eta = \frac{\hat{\rho} - \rho^\star_{\mathsf{SDP}}}{1 + |\hat{\rho}| + |\rho^\star_{\mathsf{SDP}}|}. \tag{16}$$

$\eta \to 0$ certifies global optimality of the rounded solution $\hat{U}$, and tightness of the SDP relaxation.

Summary

From (3) to (4), we first eliminated translations and landmark positions. From (4) to (8), we formulated a QCQP by creating the new constraint set $\mathfrak{so}(3)$. From (8) to (11), we applied Shor's semidefinite relaxation. This sequence of manipulations and relaxations allows us to focus on solving the convex SDP problem (11) while maintaining the ability to certify (sub)optimality of the original nonconvex problem (3), through the inequalities established in (15).

Remark 6 (Connection to Prior Work)**.** *For readers familiar with SDP relaxations, this section should not be surprising at all—that is the reason why we kept this section very brief and only focus on milestone results listed in the propositions. The closest two works to ours are* SE-Sync *[38] and* SIM-Sync *[52], and the SDP relaxation technique traces back to at least the work by Carlone et al. [14]. The novelty of our formulation are twofold: (a) we estimate scaling factors with the motivation to correct learned depth while* SE-Sync *estimates rotations and translations only; (b) we jointly estimate 3D landmarks and (scaled) camera poses while* SIM-Sync *does not estimate 3D landmarks. These differences allow us to solve the long-standing bundle adjustment problem with convex optimization.*

We shall focus on how to solve the convex SDP (11).

III. BURER-MONTEIRO FACTORIZATION AND CUDA-BASED RIEMANNIAN OPTIMIZER

Let us first write the SDP (11) in standard primal form:

$$\min_{X \in \mathbb{S}^{3N}} \ \operatorname{tr}(QX) \tag{17a}$$

$$\text{subject to} \quad \langle A_i, X \rangle = b_i, i = 1, \ldots, m \tag{17b}$$

$$X \succeq 0 \tag{17c}$$

where we have rewritten the constraint (11b) as a positive semidefinite (PSD) constraint (17c) and $m = 5N + 1$ linear equality constraints (17b). To see why this reformulation is true, note that the diagonal 3×3 blocks of X are either $\mathbf{I}_3$ or scaled $\mathbf{I}_3$—the former can be enforced using 6 linear equalities

[3] $\bar{R}_1^\star$ is in $O(3)$ because we restrict the scale of the first frame to be 1.

and the latter can be enforced using 5, summing up to $5N+1$ linear equalities. Associated with the primal standard SDP (17) is the following dual standard SDP:

$$\max_{y\in\mathbb{R}^m} \quad \sum_{i=1}^{m} b_i y_i \tag{18a}$$

$$\text{subject to} \quad Z(y) := Q - \sum_{i=1}^{m} y_i A_i \succeq 0. \tag{18b}$$

Since Slater's condition holds ($X = \mathbf{I}_{3N} \succ 0$ is feasible for the primal (17)), we know strong duality holds between primal (17) and dual (18), i.e., their optimal values are equal to each other [46, 45].

Scalability. Once the SDP (11) is formulated in standard forms, it can be solved by off-the-shelf SDP solvers such as MOSEK [5], provided that the SDP's scale is not so large. Our SDP relaxation leads to a matrix variable with size $3N \times 3N$. This means that when N is in the order of hundreds, MOSEK can solve the SDP without any problem. However, when N is in the order of thousands or tens of thousands, which is not uncommon in bundle adjustment problems, MOSEK will become very slow or even runs out of memory (e.g., MOSEK becomes unresponsive when $N > 2000$). Therefore, we decided to customize a solver for our SDP relaxation.

A. Burer-Monteiro Factorization

The key structure we will leverage is, as stated in Proposition 5, the optimal solution $X^\star$ has its rank equal to three when the SDP relaxation is tight. In other words, the effective dimension of $X^\star$ can be $3 \times 3N$ instead of $3N \times 3N$.

To exploit the low-rank structure, we will leverage the famous Burer-Monteiro (BM) factorization [13].

Proposition 7 (BM Factorization)**.** *For a fixed rank $r \geq 3$, the Burer-Monteiro factorization of (11) and (17) reads:*

$$\rho^\star_{\mathsf{BM},r} = \min_{\bar{R}_i\in\mathbb{R}^{r\times 3}, i=1,\ldots,N} \quad \operatorname{tr}\left(QU^\mathsf{T}U\right) \tag{19a}$$

$$\textit{subject to} \quad U = \begin{bmatrix} \bar{R}_1 & \ldots & \bar{R}_N \end{bmatrix} \tag{19b}$$

$$\bar{R}_i^\mathsf{T}\bar{R}_i = \begin{cases} \mathbf{I}_3 & i = 1 \\ \alpha_i \mathbf{I}_3 & i \geq 2 \end{cases}. \tag{19c}$$

A few comments are in order. First, by factorizing $X = U^\mathsf{T}U$, $X \succeq 0$ holds by construction. Second, note that when $r = 3$, problem (19) is exactly the same as the original QCQP (8) (up to the difference in $\bar{R}_1$), and thus is NOT convex. Third, as long as $r \geq r^\star$ where $r^\star$ is the minimum rank of all optimal solutions of the SDP (11), the nonconvex BM factorization has the same global minimum as the convex SDP (11) [46, 13]. Note that the factor U is a matrix of size $r \times 3N$, i.e., a flat matrix as shown in Fig. 3 bottom right.

Counterintuitive? The reader might find this confusing. In §II, we applied a series of techniques to relax a nonconvex optimization problem into a convex SDP, enabling us to solve it to global optimality. Surprisingly, the BM factorization appears to "undo" this effort, pulling us back into nonconvex optimization. While the low-rank factorization offers a clear scalability advantage, it remains uncertain whether this benefit outweighs the drawbacks of reintroducing nonconvexity.

Rank "staircase". The secret ingredient of BM factorization to tackle nonconvexity is that we will solve the factorized problem (19) at *increasing ranks*, like stepping up a "staircase" (*cf.* Fig. 3 bottom right). We will start with the lowest rank $r = 3$ and solve problem (19) using local optimization. One of two cases will happen. (a) The local optimizer is the same as the global optimizer of the SDP, in which case we can leverage the dual SDP (18) to certify global optimality and declare victory against the SDP. (b) The local optimizer is not the same as the global optimizer of the SDP, in which case we can again leverage the dual SDP (18) to "escape" the bad local minimum via increasing the rank, i.e., going up the staircase.

We formalize this in Algorithm 1 and prove its correctness.

Algorithm 1: Riemannian Staircase

1: **Input:** the data matrix Q
2: **Output:** optimal solution $U^\star$
3: `# Initialization`
4: Set $r = 3$, $U_r^0 = [\mathbf{I}_3, \ldots, \mathbf{I}_3]$
5: **while** True **do**
6: `# Local optimization of (19)`
7: $U_r^\star = \textsc{LocalOptimizer}(U_r^0)$
8: `# Compute dual certificate`
9: $y_r, Z(y_r) \leftarrow \textsc{Solve}$ (21)
10: `# Certify global optimality`
11: **if** $Z(y_r) \succeq 0$ **then**
12: **return** U_r
13: `# Escape local minimum`
14: $v \leftarrow \textsc{LeastEigenvector}(Z(y_r))$
15: $U_{r+1} = \begin{bmatrix} U_r^\mathsf{T} & \alpha v \end{bmatrix}^\mathsf{T}$ with $\alpha = 1$
16: `# Line search`
17: **while** $\operatorname{tr}\left(QU_{r+1}U_{r+1}^\mathsf{T}\right) \geq \operatorname{tr}\left(QU_rU_r^\mathsf{T}\right)$ **do**
18: $\alpha = \alpha/2$
19: $U_{r+1} = \begin{bmatrix} U_r^\mathsf{T} & \alpha v \end{bmatrix}^\mathsf{T}$
20: $r \leftarrow r+1$, $U_{r+1}^0 \leftarrow U_{r+1}$

Certify global optimality. At every iteration of Algorithm 1, it first computes a local optimizer of the BM factorization problem (19) in line 7 using an algorithm to be described in §III-B. Denote the local optimizer as $U_r^\star$. To check whether $(U_r^\star)^\mathsf{T}U_r^\star$ is the optimal solution to the convex SDP (11), we need to compute the dual optimality certificate.

Theorem 8 (Dual Optimality Certificate)**.** *Let $U_r^\star$ be a locally optimal solution of (19). Then, the linear independence constraint qualification (LICQ) must hold at $U_r^\star$, i.e.,*

$$\nabla_U(\langle A_i, (U_r^\star)^\mathsf{T}U_r^\star\rangle - b_i) = 2A_i(U_r^\star)^\mathsf{T}, \quad i = 1, \ldots, m \tag{20}$$

are linearly independent. Hence, there must exist a unique dual variable $y^\star$ such that

$$Z(y^\star)(U_r^\star)^\mathsf{T} = \mathbf{0}. \tag{21}$$

If further,

$$Z(y^\star) \succeq 0, \tag{22}$$

then $U_r^\star$ is a global optimizer of (19), and $(U^\star)^\mathsf{T}U^\star, y^\star, Z(y^\star)$ are optimal for the SDP (17) and its dual (18).

Proof: In general, LICQ may not hold at a locally optimal solution of the BM factorization. However, in Appendix A-D, we show that the unique structure of our XM SDP relaxation leads to guaranteed satisfaction of LICQ. The rest of the Theorem is standard and follows from [13] or [46]. ■

Theorem 8 provides a simple recipe to certify global optimality by solving the linear system of equations in (21) (recall $Z(y^\star)$ is linear in $y^\star$ from (18b)), forming the dual matrix $Z(y^\star)$, and checking its positive semidefinite-ness.

Escape local minimum. If $Z(y^\star)$ is not PSD, then $(U_r^\star)^\mathsf{T}U_r^\star$ is not optimal for the SDP. In this case, the following theorem states that the eigenvector of $Z(y^\star)$ corresponding to the minimum eigenvalue provides a descent direction.

Theorem 9 (Descent Direction)**.** *Let $U_r^\star$ be a local minimizer of problem (19) and $y^\star$ be the corresponding dual variable. Suppose $Z(y^\star)$ is not PSD and v is an eigenvector of $Z(y^\star)$ corresponding to a negative eigenvalue. Then, consider the BM factorization (19) at rank $r+1$. The direction*

$$D = \begin{bmatrix} 0 \\ v^T \end{bmatrix} \tag{23}$$

is a descent direction at the point

$$\hat{U} = \begin{bmatrix} U_r^\star \\ 0 \end{bmatrix}. \tag{24}$$

Proof: See [13] or [46]. ■

Theorem 9 states that if Algorithm 1 gets stuck at rank r, it can escape the local minimum by increasing the rank. Since D is a descent direction, we perform line search in lines 18-19 until a point with lower objective value is found.

Global convergence. Algorithm 1 is guaranteed to converge to the optimal solution of the SDP, as stated below.

Theorem 10 (Global Convergence)**.** *Algorithm 1 finds a globally optimal solution of the SDP pair (17)-(18) regardless of the initialization point U_r^0 at $r = 3$.*

Proof: We show that, in the worst case where $r = 3N$, every local optimizer of the BM factorization (19) corresponds to a globally optimal solution of the SDP [27, Corollary 8]. To see this, if the locally optimal solution at $r = 3N$ is rank-deficient, then global optimality is guaranteed by the "rank deficiency" Lemma [13, Proposition 4]. Conversely, if the solution is full-rank, then $Z(y^\star)$ must be zero—hence also positive semidefinite—as a result of solving (21), which again implies global optimality of the SDP. ■

This property ensures that our approach is initialization-free. To verify the correctness of the theory, we conducted experiments on the BAL-93 dataset using random initializations, and after 1000 trials, our method achieved a 100% success rate.

It is worth noting that, in the worst case when r is increased to $3N$, the BM factorization does not have any scalability advantage over the original SDP. Fortunately, in almost all numerical experiments, Algorithm 1 terminates when $r = 3$ or 4, bringing significant scalability advantage to solving large-scale SDP relaxations.

B. C++/CUDA-based Riemannian Optimization

Everything in Algorithm 1 is clear except line 7 where one needs to locally optimize the nonconvex BM factorization (19).

Riemannian structure. At first glance, problem (19) looks like a nonconvex constrained optimization problem. However, the constraints of (19) indeed define a smooth manifold.

Proposition 11 (BM Riemannian Optimization)**.** *The BM factorization problem (19) is equivalent to the following* unconstrained *Riemannian optimization problem*

$$\begin{aligned} \min \quad & \operatorname{tr}\left(QU^\mathsf{T}U\right) && \text{(25a)} \\ \textit{subject to} \quad & U = \begin{bmatrix} R_1 & s_2R_2 & \cdots & s_NR_N \end{bmatrix} && \text{(25b)} \\ & s_i \in \mathcal{M}_p, i = 2, \ldots, N, && \text{(25c)} \\ & R_i \in \mathcal{M}_s^{(r)}, i = 1, \ldots, N && \text{(25d)} \end{aligned}$$

where $\mathcal{M}_p$ is the positive manifold defined as

$$\mathcal{M}_p := \{s \mid s > 0\}, \tag{26}$$

and $\mathcal{M}_s^{(r)}$ is the Stiefel manifold of order r:

$$\mathcal{M}_s^{(r)} = \{R \in \mathbb{R}^{r\times 3} \mid R^\mathsf{T}R = \mathbf{I}_3, \}. \tag{27}$$

Proof: By inspection. ■

We solve problem (25) using the Riemannian trust-region algorithm with truncated conjugate gradient (Rtr-tCG). Details of this algorithm can be found in [1, 11].

C++/CUDA implementation. The Rtr-tCG algorithm is readily available through the MANOPT package [12]. However, to boost efficiency and enable fast solution of the SDP relaxation, we implement the Rtr-tCG algorithm directly in C++/CUDA, using analytical hessian-vector product.

- **Conjugate gradient method**. The conjugate gradient method involves only Hessian-vector products and vector addition. The Hessian-vector product can be decomposed into two components: (a) the Euclidean Hessian-vector product and (b) the Riemannian projection onto the tangent space. The first component primarily requires matrix-vector multiplication, which can be efficiently implemented using `cuBLAS`. The second component involves batched small matrix-matrix multiplications, matrix-matrix inner products, scalar-matrix multiplications, all of which are implemented using custom CUDA kernels. For vector addition, we directly utilize the `cublasDaxpy` function from `cuBLAS`.
- **Retraction**. The retraction operation aims to map a point from the tangent space back to the manifold. In our case, the retraction operation happens both on the positive manifold and the Stiefel manifold. The retraction on the positive manifold is a simple custom kernel, while the retraction on the Stiefel manifold involves QR decomposition. We directly apply Gram-Schmidt process on every batch of $3 \times r$ matrices, which is implemented using custom CUDA kernels.

In §V, we show our GPU-based implementation achieves up to 100 times speedup compared to the CPU-based MANOPT.

Summary

We focused on developing a customized solver capable of solving large-scale SDP relaxations in (11) (and (17)). We applied the Burer-Monteiro factorization method to exploit low-rankness of the optimal SDP solutions (*cf.* problem (19)), and leveraged the Staircase Algorithm 1 to solve the nonconvex BM factorization to global optimality. We pointed out the BM factorization problem is indeed an unconstrained Riemannian optimization problem (*cf.* problem (25)) and developed a C++/CUDA-based implementation that is significantly faster than MANOPT.

Remark 12 (Connection to Prior Work)**.** *This is not the first time BM factorization has been applied in robotics.* SE-Sync *and related works [18, 37] pioneered the application of BM factorization for solving the pose graph optimization problem. Several works [22, 21, 23, 24] utilized the dual optimality certificate result in Theorem 8 to develop fast certifiers. Our novelty lies in developing the first C++/CUDA-based implementation of the Riemannian trust-region algorithm to push the scalability limitations of BM factorization.*

IV. STRUCTURE FROM MOTION WITH XM

In this section, we present our SfM pipeline with XM as the optimization engine, illustrated in Fig. 3.

View graph. To construct view graph for an image set, we first run COLMAP's *feature extractor* and *exhaustive matcher* to extract 2D correspondences. The feature extractor employs SIFT [30] for feature detection and description, while the exhaustive matcher matches every image pair. After matching, we apply GLOMAP's *track establishment* to produce a four-column file where the first two columns represent feature point coordinates, the third indicates the image index, and the fourth corresponds to the 3D landmark indices. Currently, we use the original implementation from COLMAP and GLOMAP. However, we remark that it is possible to speed up the processes further using C++ and GPU implementation. We leave this improvement for future work.

Depth estimation. We use the depth estimation model UNIDEPTH [35] to calculate the metric depth of a given image, and lift the view graph from 2D to 3D. It is worth emphasizing that our test datasets do **NOT** belong to the training dataset of UNIDEPTH. If given the confidence map, we use it to update the weight of different observations. We also tried other depth prediction models in Appendix B.

Filter from two-view estimation. Using 2D observations, we estimate the relative pose between two images. Based on this pose, we filter out 3D landmarks with large Euclidean distance errors. Specifically, landmarks with distance errors exceeding three times the median are removed.

XM **solver**. We then use the lifted 3D measurements and the view-graph to form a Q matrix as shown in (4). We solve the SDP problem in (11) using our XM solver. If needed, we also delete the 10% measurements with the largest residuals and re-run the XM solver. This corresponds to a greedy heuristic for outlier removal [4, 40] and we call it XM^2 (running twice).

CERES refinement. Usually the depth predictions are quite noisy, leading to inaccurate estimations of XM. We therefore also feed the estimated poses and landmarks to CERES as a warmstart to solve the original bundle adjustment problem (1). As we will show, the solution of XM always provide a strong warmstart for CERES to quickly optimize (1).

V. EXPERIMENTS

We evaluate the XM solver and the XM-SfM pipeline on diverse datasets. XM is benchmarked against the leading bundle adjustment solver CERES and XM-SfM is compared against the widely adopted SfM pipelines COLMAP and GLOMAP. We further analyze the convergence rate and evaluate robustness to depth estimation noise in Appendix F.

Experiments run on a Lambda Vector workstation with 64-core AMDő Ryzen Threadripper Pro 5975WX CPUs and dual NVIDIAő RTX 6000 GPUs, using CUDA 12.4 and Python 3.11.10. All dependencies are carefully set up to leverage multi-CPU and GPU acceleration.

A. BAL Dataset

We first evaluate on the Bundle Adjustment in the Large (BAL) dataset [2]. This dataset contains reconstruction results from Flickr photographs using Bundler. On BAL we focus on XM solver performance rather than the full XM-SfM pipeline.

Setup and baselines. For input, we use 2D observations for CERES and 3D observations for XM. The 2D keypoint measurements come directly from the BAL dataset, while the 3D keypoint measurements are lifted by appending z-coordinates to the 2D measurements. Though accurate, these 3D observations incorporate slight noise, making them a refined yet imperfect ground truth. To showcase XM's efficiency in solving the SDP relaxation of the SBA problem (3), we also evaluate MANOPT using the same input as XM.

Metrics. We evaluate performance based on the runtime and the median of Absolute Trajectory Error (ATE) and Relative Pose Error (RPE). More details can be found in Appendix G. Additionally, we report the suboptimality and the minimum eigenvalue of the $Z(y)$ matrix in (18) to demonstrate that XM achieves global optimality. Note that the minimum eigenvalue of $Z(y)$ has been used as a metric for global optimality in previous works as well [14]. Runtime is split into preprocessing time and solver time. The former primarily involves constructing the Q matrix in (4), which is implemented in Python, while the latter corresponds to the GPU solver. We separate preprocessing time and solver time because there are still ways to further reduce the preprocessing time (while the solver time, to the best of our understanding, has been pushed to the limit). For example, as building the Q matrix requires large dense matrix multiplications, a GPU implementation is expected to accelerate this step by 10 to 100 times. However, we leave this as a future step.

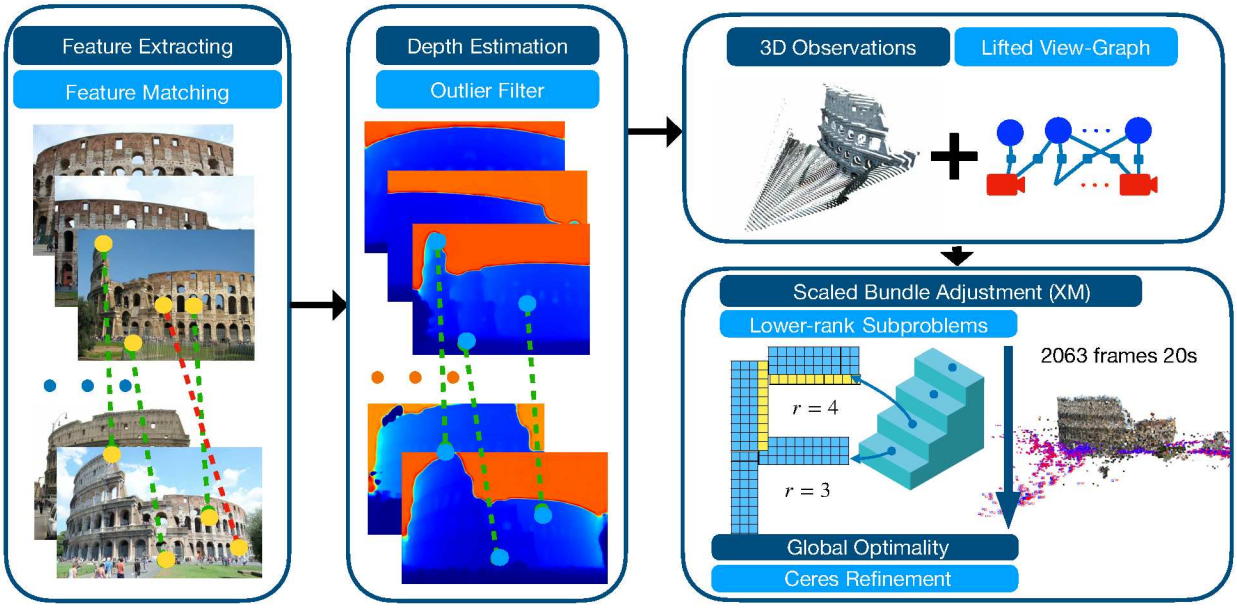


Fig. 3: XM-SfM: structure from motion pipeline with XM.

TABLE I: Results on the BAL dataset. We report the ATE, RPE and running time for CERES, MANOPT, and our proposed XM solver. The evaluation is conducted on four BAL datasets with varying numbers of frames to demonstrate that our method is both fast and accurate across datasets of different scales (e.g., BAL-10155 indicates there are 10155 camera frames to be reconstructed). CERES fails at a bad local minimum without a good initial guess, while MANOPT performs significantly slower and even fails to solve within ten hours on the largest dataset.

Datasets	BAL-93 (6033 landmarks)			BAL-392 (13902 landmarks)			BAL-1934 (67594 landmarks)			BAL-10155 (33782 landmarks)		
Metrics	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T
CERES	**0.06** 4.21	19.47° 0.18	15.46° 0.29	0.37 23.31	23.49° 0.43	21.80° 0.7	3.85 227.15	177.20° 0.32	31.50° 0.63	2.41 50.97	84.29° 0.71	80.19° 1.25
CERES-GT	0.06 0.36	**0.0°** **0.0**	**0.0°** 0.01	**0.36** 3.59	**0.0°** 0.01	**0.0°** 0.01	3.78 16.39	**0.0°** 0.01	**0.0°** **0.01**	**2.4** **21.49**	**0.0°** **0.0**	**0.0°** **0.0**
CERES-GT-0.01	0.06 11.61	1.72° 0.01	2.87° 0.02	0.38 25.74	4.58° 0.04	5.73° 0.07	**3.73** 427.79	**0.0°** 0.01	**0.0°** **0.01**	2.45 380.4	2.87° 0.04	5.15° 0.07
CERES-GT-0.1	0.06 9.24	40.29° 0.56	34.38° 1.01	0.37 21.34	22.37° 0.37	30.38° 0.73	3.76 70.43	**0.0°** 0.82	9.17° 1.3	2.38 35.06	20.57° 0.33	37.78° 0.67
MANOPT	0.94 0.56	**0.0°** **0.0**	**0.0°** **0.0**	2.7 21.43	**0.0°** **0.0**	**0.0°** **0.0**	69.15 236.46	**0.0°** **0.0**	**0.0°** **0.01**	336.76 **	** **	** **
XM	0.94 **0.07**	**0.0°** **0.0**	**0.0°** **0.0**	2.7 **0.55**	**0.0°** **0.0**	**0.0°** **0.0**	69.15 **2.09**	**0.0°** **0.0**	**0.0°** **0.01**	336.76 4322.77	0.73° 0.02	0.5° 0.02

TABLE II: Results on the BAL dataset (suboptimality and min-eig). We report the suboptimality gap and minimum eigenvalue of Z matrix in (18) for our proposed XM method.

Datasets	BAL-93	BAL-392	BAL-1934	BAL-10155
Suboptimality-Gap	4.8×10^{-4}	4.2×10^{-3}	1.0×10^{-2}	6.2×10^{-1}
Min-eig	-8.8×10^{-5}	1.3×10^{-7}	1.2×10^{-6}	-2.1×10^{1}

Results. Table I summarizes the comparison between XM and other methods. We tested several versions of CERES. "CERES" indicates running CERES without any initialization. "CERES-GT" indicates starting CERES at the groundtruth estimation. "CERES-GT-0.01" means adding noise to the groundtruth with standard deviation 0.01. We make several observations. (a) Without good initialization, CERES does not work, as shown by the failures of CERES and CERES-GT-0.1. (b) XM is up to 100 times faster than MANOPT, showing the superior efficiency of our GPU implementation. Notably, XM's solver time is below a second for N in the order of hundreds, and XM scales to $N > 10,000$ camera frames.

Table II presents the suboptimality gap and minium eigenvalue of XM. As we can see, except the largest instance with $N = 10155$ camera frames, XM solved all the other instances to certifiable global optimality. The reason why XM did not solve the largest instance to global optimality is because we restricted its runtime to one hour (XM indeed achieve global optimality if allowed four hours of runtime).

Fig. 4 visualizes the 3D reconstructions. Since real images are not available in BAL, all 3D landmarks have the same purple color. The reconstructed cameras are shown in red.

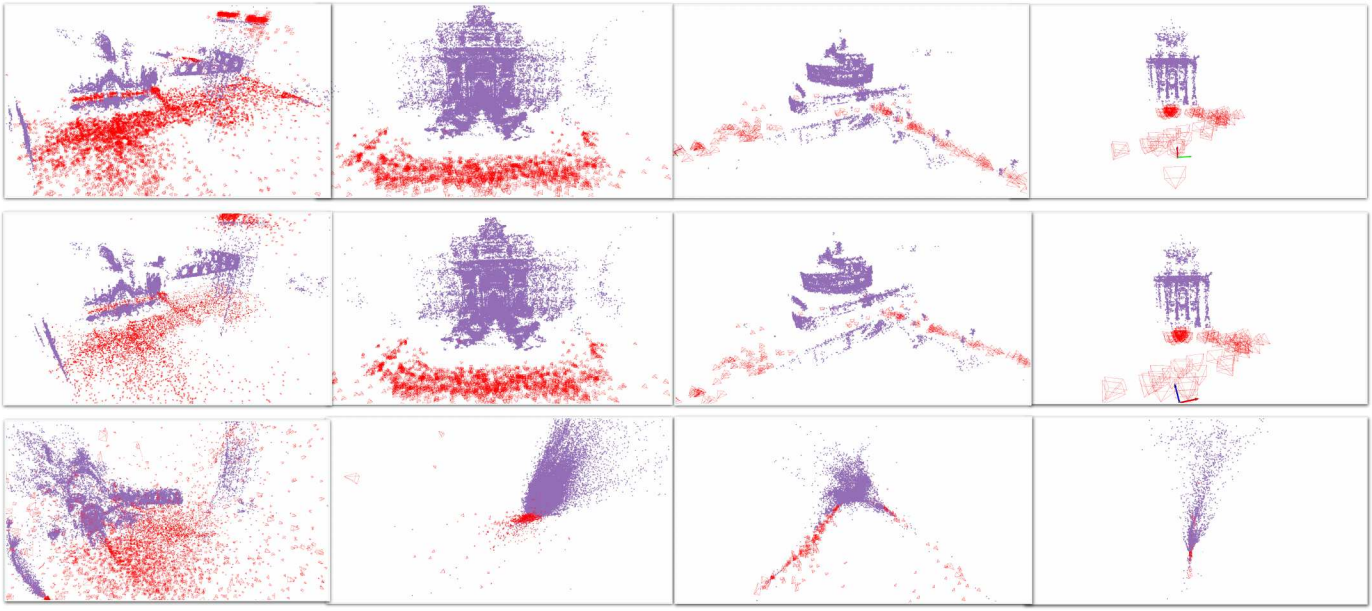


Fig. 4: Visualization of BAL datasets. **Top:** Our XM solver. **Middle:** CERES-GT-0.01. **Bottom:** CERES-GT-0.1. Both our XM solver and CERES-GT-0.01 accurately recover the ground truth camera poses and landmarks, whereas CERES-GT-0.1 fails.

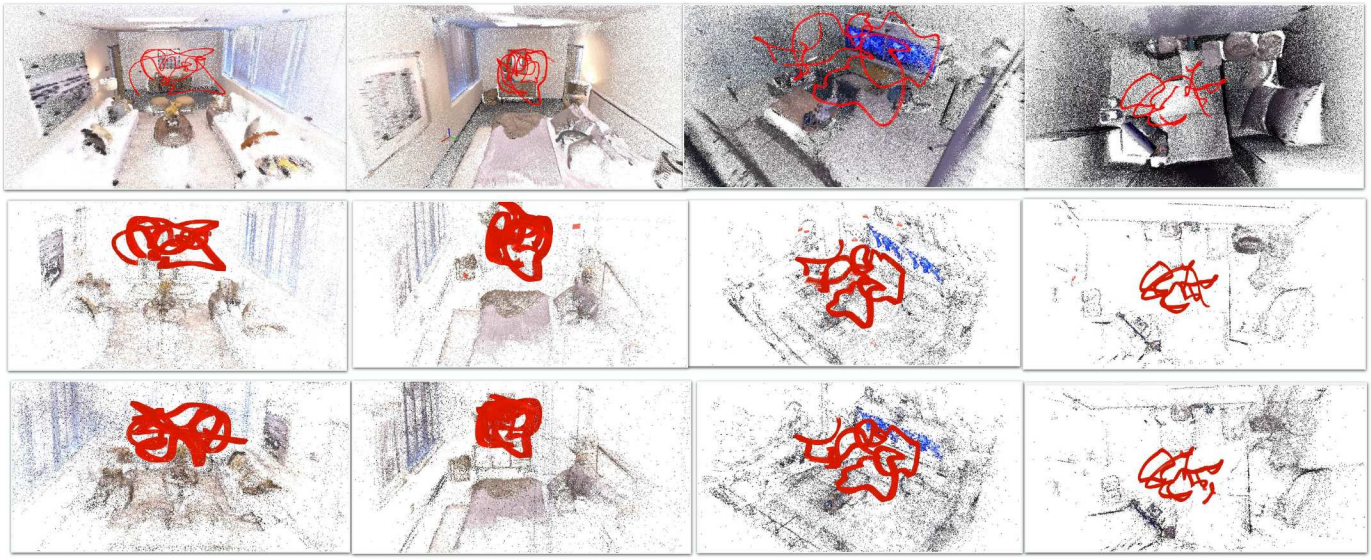


Fig. 5: Visualization of Replica datasets. **Top:** Our XM solver. **Middle:** GLOMAP. **Bottom:** COLMAP. All methods achieve high accuracy, producing nearly identical reconstruction results. GLOMAP sometimes produce outliers (see column 2 and 3).

Takeaway

- The SBA problem (3) is easier to solve than the BA problem (1)—we can design efficient convex relaxations. While CERES needs good initialization for solving (1), XM requires no initializations for solving (3).
- With GT depth and outlier-free matchings, solving SBA with XM produces the same result as solving BA with CERES or COLMAP.
- XM is fast and scalable.

B. Replica Dataset

We then test on the Replica dataset [53, 43, 41], which contains synthetic images of different virtual scenes.

Setup, baselines, metrics. We use the groundtruth depth map but employ the full XM-SfM pipeline. We compare XM-SfM, both with and without two-view filtering, against COLMAP and GLOMAP. The evaluation metrics remain the same. For runtime analysis, we categorize all components preceding our XM solver—including Matching, Indexing, Depth Estimation, Filtering, and Matrix Construction—as preprocessing time. The indexing, filtering and matrix construction components can be further accelerated in CUDA. Similarly, for GLOMAP and COLMAP, all steps prior to global positioning and bundle adjustment are counted as preprocessing time.

Results. Results are presented in Table III and Table IV. Each Replica dataset contains 2000 frames, but for a diverse

Fig. 6: Visualization of Mip-Nerf datasets. **Top:** COLMAP. **Bottom:** Our XM solver. 3D-gaussian renderings are the same.

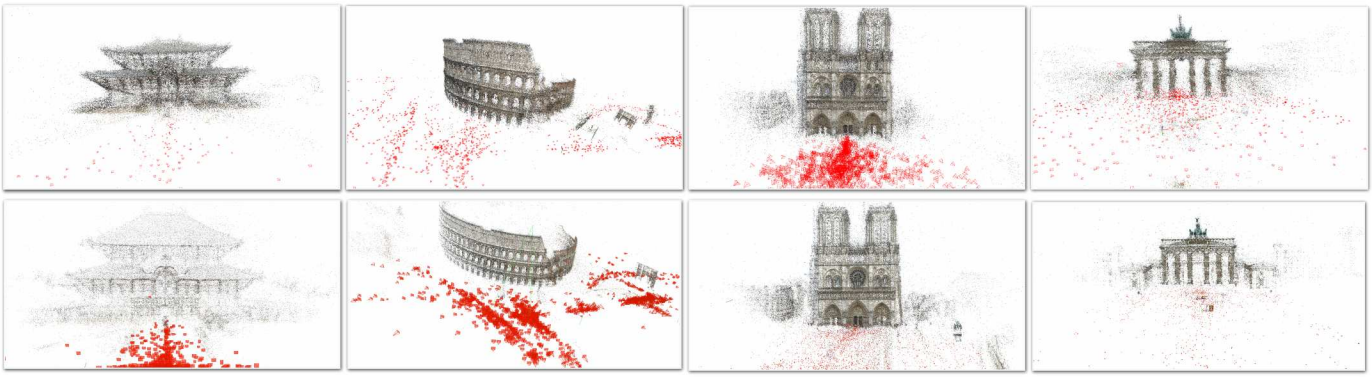


Fig. 7: Visualization of IMC2023 datasets. **Top:** Our XM solver. **Bottom:** GLOMAP.

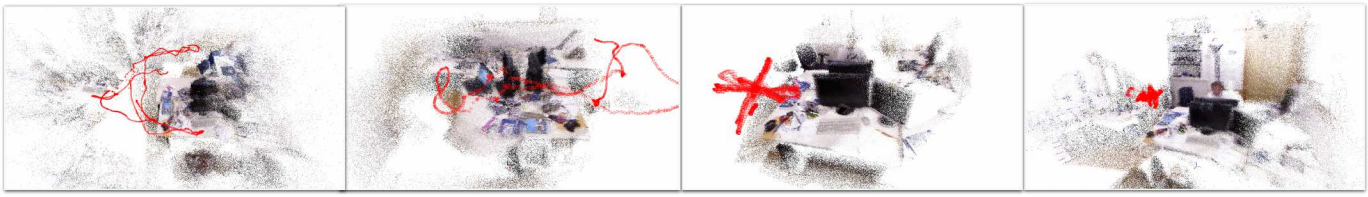


Fig. 8: Visualization of TUM datasets using our XM solver.

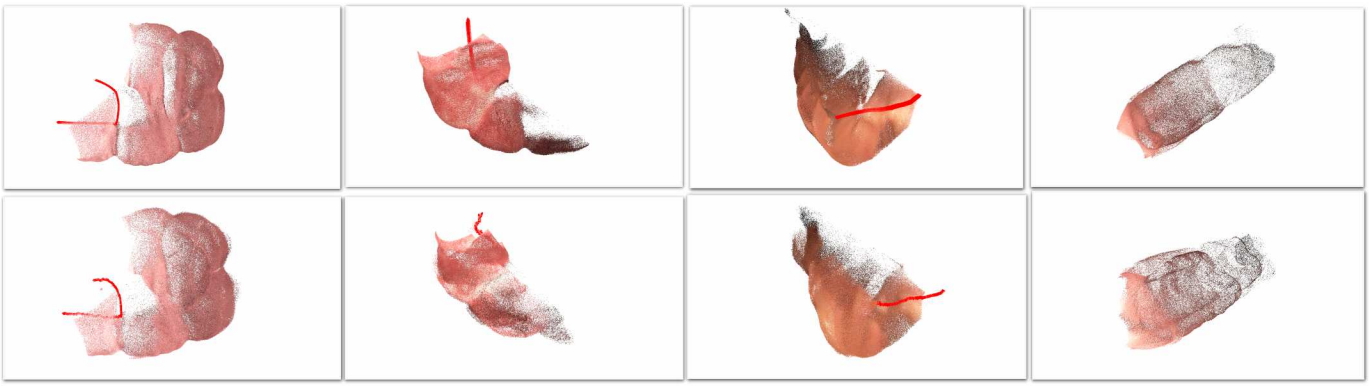


Fig. 9: Visualization of C3VD medical datasets. **Top:** With ground truth depth. **Bottom:** With learned depth.

TABLE III: Results on the Replica dataset. COLMAP, while highly stable, is extremely slow, taking over 20 hours for datasets with 2000 frames. GLOMAP improves speed but still requires several hours for the solving stage and occasionally produces outliers. In comparison, our solver achieves similar accuracy in just 10 seconds for the same dataset size.

Datasets	XM²			FILTER + XM²			GLOMAP			COLMAP		
Metrics	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T
Room0-100	10.52 **0.12**	**0.323°** 0.005	0.083° 0.007	13.71 **0.12**	1.025° 0.005	0.077° 0.008	6.4 39.57	0.469° **0.003**	0.05° **0.004**	**1.71** 100.61	0.582° **0.003**	**0.038°** **0.004**
Room0-2000	1142.74 **8.5**	1.38° 0.004	0.44° 0.006	1868.24 16.26	0.379° 0.003	0.242° 0.005	957.31 5371.98	0.136° **0.001**	0.043° **0.001**	**120.02** 74827.56	**0.078°** **0.001**	**0.068°** **0.001**
Room1-100	13.08 **0.14**	1.847° **0.009**	0.16° **0.013**	16.19 0.14	2.056° 0.01	0.176° 0.014	8.75 53.95	**1.409°** 0.014	**0.122°** 0.019	**1.99** 144.39	10.765° 0.06	0.726° 0.072
Room1-2000	1156.6 7.37	**0.005°** 0.006	**0.018°** 0.009	1651.12 **6.18**	0.428° 0.005	0.418° 0.008	911.0 3806.94	0.166° **0.001**	0.066° **0.002**	**379.45** 64912.5	0.156° 0.002	0.06° **0.002**
Office0-100	12.75 **0.15**	3.554° 0.04	0.128° 0.051	16.53 0.12	3.385° 0.024	0.127° 0.033	8.67 27.99	**0.559°** **0.015**	**0.04°** **0.019**	**3.58** 49.02	1.53° 0.019	0.067° 0.028
Office0-2000	730.15 **4.6**	**0.012°** 0.016	**0.015°** 0.024	1181.42 5.61	0.053° 0.016	0.051° 0.019	613.85 3241.77	2.822° 0.024	0.058° 0.034	**168.36** 35682.24	0.184° **0.003**	0.06° **0.004**
Office1-100	8.13 **0.13**	2.708° 0.022	0.2° 0.03	10.81 0.15	2.032° 0.015	0.138° 0.023	4.37 18.7	**1.423°** **0.006**	**0.021°** **0.008**	**1.36** 61.5	3.366° 0.014	0.039° 0.021
Office1-2000	759.43 **23.1**	10.516° 0.077	3.802° 0.113	1270.29 230.25	5.08° 0.038	1.643° 0.052	666.34 2838.54	**0.037°** **0.001**	**0.045°** **0.001**	**140.53** 25512.84	13.882° 0.078	0.157° 0.112

TABLE IV: Results on the Replica dataset (min-eig and suboptimality). All the min-eig and suboptimality-gap are small, which means our solver find the global minima.

Method	XM²		FILTER + XM²	
Metric	Min-eig	Suboptimality-gap	Min-eig	Suboptimality-gap
Room0-100	1.1×10^{-5}	1.9×10^{-6}	1.1×10^{-5}	7.4×10^{-8}
Room0-2000	3.5×10^{-8}	2.2×10^{-5}	-6.3×10^{-8}	5.9×10^{-6}
Room1-100	2.6×10^{-6}	7.9×10^{-4}	1.3×10^{-5}	1.8×10^{-6}
Room1-2000	5.8×10^{-8}	2.2×10^{-4}	-4.6×10^{-7}	4.5×10^{-6}
Office0-100	-8.0×10^{-7}	1.4×10^{-7}	-1.0×10^{-6}	1.1×10^{-7}
Office0-2000	4.7×10^{-7}	5.2×10^{-4}	-1.5×10^{-6}	6.7×10^{-6}
Office1-100	-8.0×10^{-7}	1.0×10^{-7}	8.5×10^{-7}	2.4×10^{-8}
Office1-2000	-4.2×10^{-8}	2.3×10^{-5}	6.8×10^{-8}	3.8×10^{-4}

comparison across different dataset sizes, we sample the first 100 frames from each dataset as a separate experiment. "Room0-100" refers to the first 100 frames, while "Room0-2000" represents the full dataset.

XM *consistently outperforms baselines by 100 to 1000 times in solver speed*, solving almost all 2000-frame datasets within 10 seconds. At the same time, XM maintains high accuracy, achieving a median translation error of just 1%. In practice, a 0.1% and 1% translation error yield nearly identical reconstruction quality, as illustrated in Fig. 5.

Additionally, we provide a runtime breakdown for XM in Appendix C. The solver time is negligible, appearing as only a thin bar in the chart. Matching, indexing, and filtering are the most time-consuming components, with the latter two planned for CUDA implementation as future work.

Takeaway

- With ground truth depth and COLMAP matchings, minor filter refinement achieves results comparable to COLMAP and GLOMAP.
- XM remains highly efficient and scalable.

TABLE V: Results on the Mip-Nerf and Zip-Nerf datasets. COLMAP is very slow, while both XM and GLOMAP achieve comparable accuracy. However, XM is significantly faster.

Method	Metrics	Datasets				
		Kitchen-279 -187865	Garden-185 -106858	Bicycle-194 -41866	Room-311 -111783	Alameda-1724 -416665
FILTER + XM² + CERES	Solver Time (XM + CERES)	**0.8 +** **19.19**	**0.65 +** **3.98**	**0.58 +** **22.47**	**1.01 +** **24.35**	**62.0 +** **218.13**
	Processing Time	229.82	174.57	145.75	223.44	3348.46
	ATE-T ATE-R	0.008 0.072°	0.002 0.021°	0.019 0.229°	0.002 0.06°	0.009 0.317°
	RPE-T RPE-R	0.005 0.054°	0.003 0.025°	0.028 0.219°	0.002 0.084°	0.014 0.493°
GLOMAP	Solver Time	363.22	193.86	77.76	269.39	1169.83
	Processing Time	124.23	90.7	79.84	96.26	2631.69
	ATE-T ATE-R	0.016 0.107°	0.013 0.061°	0.039 0.51°	0.003 0.083°	0.001 0.074°
	RPE-T RPE-R	0.023 0.135°	0.02 0.067°	0.056 0.114°	0.003 0.029°	0.002 0.053°
COLMAP	Solver Time	1422.96	486.84	273.84	958.5	79278.36
	Processing Time	**97.26**	**78.1**	**75.16**	**62.47**	**2494.29**
	ATE-T ATE-R	0.0 0.0°	0.0 0.0°	0.0 0.0°	0.0 0.0°	0.0 0.0°
	RPE-T RPE-R	0.0 0.0°	0.0 0.0°	0.0 0.0°	0.0 0.0°	0.0 0.0°

TABLE VI: Results on the Mip-Nerf and Zip-Nerf datasets (min-eig and suboptimality). All datasets are solved to global minimum.

Metric	Kitchen	Garden	Bicycle	Room	Alameda
Min-eig	1.1×10^{-10}	-3.6×10^{-9}	4.8×10^{-8}	-4.6×10^{-7}	-3.8×10^{-7}
Suboptimality-gap	8.9×10^{-8}	8.8×10^{-9}	2.5×10^{-5}	4.5×10^{-6}	8.1×10^{-2}

C. Mip-Nerf and Zip-Nerf Dataset

Mip-Nerf and Zip-Nerf [7, 8] are real-world image datasets around a single object. We evaluate learned depth and downstream novel view synthesis tasks on these datasets.

Setup, baselines, metrics. As these datasets are generated by COLMAP, we use its camera poses as ground truth to benchmark XM against COLMAP and GLOMAP. To address inaccuracies in learned depth, we apply CERES refinement, incorporating its runtime into the solver time, denoted as "XM + CERES." All other evaluation metrics remain unchanged.

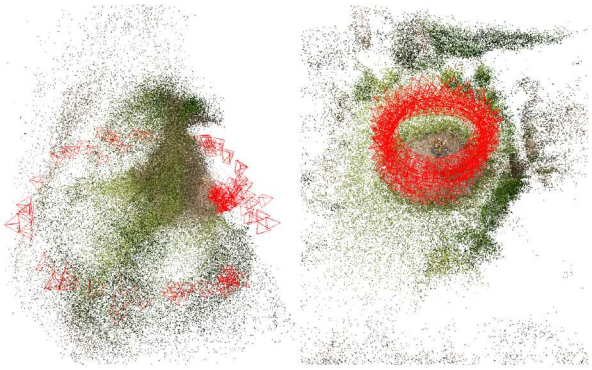


Fig. 10: Illustration of local minimum on the Mip-Nerf dataset. **Left:** Solution of (19) with $r = 3$, where the solver gets stuck in a poor local minimum. **Right:** Solution after increasing the rank to 4, which successfully escapes the local minimum.

Results. Results are presented in Table V, with suboptimality detailed in Table VI. *While adding* CERES *increases solver time, the overall runtime remains 10 to 100 times faster than the baselines.* On the garden and room datasets, we achieve a 0.2% error, demonstrating the same accuracy as the baselines, while on others, the error may be slightly higher. However, as shown in Fig. 6, this has no noticeable impact on downstream 3D Gaussian Splatting tasks [29].

A runtime breakdown for XM is provided in Appendix C. Matching and indexing remain slow, while depth estimation now takes even longer. This is due to (a) foundation models requiring much time for estimation and (b) depth estimation runtime scales linearly with the number of frames, whereas Mip-Nerf datasets are relatively small.

The need for convex relaxation. In Fig. 10, we demonstrate that directly solving (19) with rank 3 (equivalently (8)) can lead to a local minimum. Specifically, the solver terminates with a minimal eigenvalue of $Z(y)$ at -6.2×10^2, resulting in a messy reconstruction. However, by increasing the rank to 4, the solver escapes the local minimum and achieves the global minimum, indicating that while the relaxation remains tight, the Burer-Monteiro method requires a higher rank to find the global minimum.

Takeaway

- With CERES refinement to mitigate depth errors, XM achieves nearly the same accuracy as COLMAP and GLOMAP. Moreover, this has no impact on downstream novel view synthesis tasks.
- XM maintains a significant speed advantage, even with learned depth and real-world data.
- BM factorization and the Riemannian staircase effectively escape local minimum.

D. IMC, TUM and C3VD Datasets

Followed by Mip-Nerf, we step to the IMC PhotoTourism dataset [16], TUM dataset [42], and C3VD medical dataset [9]. These datasets are quite challenging because of varying environments and low-quality images.

TABLE VII: Results on the IMC datasets. XM-SfM achieves comparable accuracy while being much more scalable.

Datasets	FILTER + XM² + CERES			GLOMAP		
Metrics	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T
Rome-2063 -203244	4877.89 **19.17 + 300.16**	1.252° 0.021	0.72° 0.035	**3459.04** 9813.04	**0.09°** **0.003**	**0.069°** **0.005**
Gate-1363 -62561	1349.74 **2.92 + 99.66**	1.329° 0.038	2.022° 0.089	**712.56** 2263.79	**0.41°** **0.018**	**0.19°** **0.047**
Temple-904 -78026	1026.65 **1.34 + 93.5**	0.545° **0.008**	0.254° **0.013**	**600.4** 1560.86	**0.276°** 0.012	**0.216°** 0.03
Paris-3765 -314422	16645.16 **22.17 + 304.49**	0.473° **0.008**	0.561° **0.017**	**10598.8** 74560.1	**0.117°** 0.009	**0.087°** **0.017**

TABLE VIII: Results on the TUM datasets. The error is slightly higher due to low-resolution images.

Metrics	Solver Time / Processing Time	ATE-T / ATE-R	RPE-T / RPE-R
fr1/xyz-798-26078	0.63 + 9.69 / 498.49	0.035 / 3.148°	0.05 / 1.134°
fr1/rpy-723-26071	0.88 + 18.8 / 329.84	0.023 / 6.784°	0.038 / 2.495°
fr1/desk-613-38765	0.63 + 7.95 / 201.17	0.024 / 2.831°	0.041 / 2.277°
fr1/room-1362-86634	10.17 + 182.29 / 464.4	0.049 / 3.718°	0.086 / 2.196°

Setup, baselines, metrics. We only compare our XM solver against GLOMAP on IMC datasets. In these three datasets we add both outlier filtering and XM². In C3VD we use the ground truth depth map and the learned depth from a medical-specific depth prediction model [34].

Results. The results are presented in Table VII, Table VIII, and Table IX. The IMC datasets consist of large image collections capturing some famous landmarks. As a result, GLOMAP requires an extremely long runtime, exceeding 20 hours for the largest dataset. Our accuracy is comparable to GLOMAP, with visualizations provided in Fig. 7. The TUM and C3VD datasets produce high-quality reconstructions, though accuracy is affected by the complexity of the environment and insufficient lighting. Visualizations of these reconstructions are provided in Fig. 8 and Fig. 9.

Takeaway

XM remains efficient and scalable across diverse datasets, achieving results comparable to GLOMAP while being significantly faster.

VI. CONCLUSION

We proposed XM, a scalable and initialization-free solver for global bundle adjustment, leveraging learned depth and convex optimization. By relaxing scaled bundle adjustment as a convex SDP and solving it efficiently with Burer-Monteiro factorization and a CUDA-based trust-region Riemannian optimizer, XM achieved certifiable global optimality at extreme scales. Integrated into the XM-SfM pipeline, it maintains the accuracy of existing SfM methods while being significantly faster and more scalable.

Limitation and future work. First, while our XM solver outperforms baselines in speed, it can be sensitive to noise

TABLE IX: Results on the C3VD datasets. Ground truth depth is significantly more accurate than learned depth, while they both fail on the last dataset because of dark environment.

Datasets	FILTER + XM^2 + GT-DEPTH			FILTER + XM^2		
Metrics	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T
medical5	55.45 0.31	3.284° 0.009	0.647° 0.015	30.19 0.34	11.64° 0.109	3.191° 0.169
medical6	171.35 0.46	4.102° 0.014	0.851° 0.019	166.19 0.47	116.79° 0.262	4.729° 0.352
medical7	261.58 0.81	3.69° 0.01	1.231° 0.013	250.48 0.89	131.686° 0.047	3.564° 0.062
medical8	56.84 0.23	66.107° 0.72	0.062° 1.196	38.25 0.22	159.909° 0.452	0.091° 0.704

and outliers. Future work includes refining the filtering process and developing better methods to handle outliers. Second, our GPU solver is built on the `cuBLAS` dense matrix-vector multiplication library, whereas SLAM camera sequences often have sparse patterns. Extending the XM solver to support sparse matrix-vector multiplications would enhance its applicability to SLAM. Third, our XM-SfM pipeline still has components that potentially can be accelerated 100 to 1000 times through CUDA implementation, e.g., filtering and matrix construction. Lastly, we acknowledge that XM relies on a well-trained depth prediction model. While generic models are available, training effective depth predictors can be particularly challenging in data-scarce domains such as medical or space applications.

ACKNOWLEDGEMENTS

We thank Shucheng Kang for the help on CUDA programming; Xihang Yu and Luca Carlone for discussions about depth prediction models; and members of the Harvard Computational Robotics Group for various explorations throughout the project.

REFERENCES

[1] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008. 6, 19

[2] Sameer Agarwal, Noah Snavely, Steven M Seitz, and Richard Szeliski. Bundle adjustment in the large. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part II 11*, pages 29–42. Springer, 2010. 7

[3] Sameer Agarwal, Keir Mierle, et al. Ceres solver: Tutorial & reference. *Google Inc*, 2(72):8, 2012. 2

[4] Pasquale Antonante, Vasileios Tzoumas, Heng Yang, and Luca Carlone. Outlier-robust estimation: Hardness, minimally tuned algorithms, and applications. *IEEE Transactions on Robotics*, 38(1):281–301, 2021. 3, 7

[5] Mosek ApS. Mosek optimization toolbox for matlab. *Users Guide and Reference Manual, Version*, 4(1), 2019. 5

[6] Timothy D Barfoot, Connor Holmes, and Frederike Dümbgen. Certifiably optimal rotation and pose estimation based on the cayley map. *The International Journal of Robotics Research*, page 02783649241269337, 2023. 3

[7] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 4, 11

[8] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. *ICCV*, 2023. 11

[9] Taylor L Bobrow, Mayank Golhar, Rohan Vijayan, Venkata S Akshintala, Juan R Garcia, and Nicholas J Durr. Colonoscopy 3d video dataset with paired depth from 2d-3d registration. *Medical Image Analysis*, page 102956, 2023. 12

[10] Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R Richter, and Vladlen Koltun. Depth pro: Sharp monocular metric depth in less than a second. *arXiv preprint arXiv:2410.02073*, 2024. 2, 17

[11] Nicolas Boumal. *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023. 3, 6, 19

[12] Nicolas Boumal, Bamdev Mishra, P-A Absil, and Rodolphe Sepulchre. Manopt, a matlab toolbox for optimization on manifolds. *The Journal of Machine Learning Research*, 15(1):1455–1459, 2014. 3, 6

[13] Samuel Burer and Renato DC Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical programming*, 95(2):329–357, 2003. 3, 5, 6

[14] Luca Carlone, David M Rosen, Giuseppe Calafiore, John J Leonard, and Frank Dellaert. Lagrangian duality in 3d slam: Verification techniques and optimal solutions. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 125–132. IEEE, 2015. 4, 7

[15] Kunal N Chaudhury, Yuehaw Khoo, and Amit Singer. Global registration of multiple point clouds using semidefinite programming. *SIAM Journal on Optimization*, 25(1):468–501, 2015. 2

[16] Ashley Chow, Eduard Trulls, HCL-Jevster, Kwang Moo Yi, lcmrll, old ufo, Sohier Dane, tanjigou, WastedCode, and Weiwei Sun. Image matching challenge 2023. https://kaggle.com/competitions/image-matching-challenge-2023, 2023. Kaggle. 12

[17] Frank Dellaert and GTSAM Contributors. borglab/gtsam, 2022. URL https://github.com/borglab/gtsam. 2

[18] Frank Dellaert, David M Rosen, Jing Wu, Robert Mahony, and Luca Carlone. Shonan rotation averaging: Global optimality by surfing so (p)^ n so (p) n. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 292–308. Springer, 2020. 7

[19] Taosha Fan, Joseph Ortiz, Ming Hsiao, Maurizio Monge, Jing Dong, Todd Murphey, and Mustafa Mukadam. Decentralization and acceleration enables large-scale bundle adjustment. *arXiv preprint arXiv:2305.07026*, 2023. 2

[20] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 3

[21] Mercedes Garcia-Salguero and Javier Gonzalez-Jimenez. Certifiable planar relative pose estimation with gravity prior. *Computer Vision and Image Understanding*, 239: 103887, 2024. 7

[22] Mercedes Garcia-Salguero, Jesus Briales, and Javier Gonzalez-Jimenez. Certifiable relative pose estimation. *Image and Vision Computing*, 109:104142, 2021. 7

[23] Connor Holmes and Timothy D Barfoot. An efficient global optimality certificate for landmark-based slam. *IEEE Robotics and Automation Letters*, 8(3):1539–1546, 2023. 7

[24] Connor Holmes, Frederike Dümbgen, and Timothy D Barfoot. Sdprlayers: Certifiable backpropagation through polynomial optimization problems in robotics. *arXiv preprint arXiv:2405.19309*, 2024. 7

[25] Berthold KP Horn. Closed-form solution of absolute orientation using unit quaternions. *Josa a*, 4(4):629–642, 1987. 2

[26] José Pedro Iglesias, Carl Olsson, and Fredrik Kahl. Global optimality for point set registration using semidefinite programming. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8287–8295, 2020. 2

[27] Michel Journée, Francis Bach, P-A Absil, and Rodolphe Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 2010. 6

[28] Shucheng Kang, Xiaoyang Xu, Jay Sarva, Ling Liang, and Heng Yang. Fast and certifiable trajectory optimization. *arXiv preprint arXiv:2406.05846*, 2024. 3

[29] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/. 12

[30] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999. 7

[31] Carl Olsson, Anders Eriksson, and Richard Hartley. Outlier removal using duality. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1450–1457. IEEE, 2010. 3

[32] Linfei Pan, Dániel Baráth, Marc Pollefeys, and Johannes L Schönberger. Global structure-from-motion revisited. In *European Conference on Computer Vision*, pages 58–77. Springer, 2025. 2

[33] Alan Papalia, Andrew Fishberg, Brendan W O'Neill, Jonathan P How, David M Rosen, and John J Leonard. Certifiably correct range-aided slam. *IEEE Transactions on Robotics*, 2024. 3

[34] Akshay Paruchuri, Samuel Ehrenstein, Shuxian Wang, Inbar Fried, Stephen M Pizer, Marc Niethammer, and Roni Sengupta. Leveraging near-field lighting for monocular depth estimation from endoscopy videos. *arXiv preprint arXiv:2403.17915*, 2024. 12

[35] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. Unidepth: Universal monocular metric depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10106–10116, 2024. 2, 7, 17

[36] Jie Ren, Wenteng Liang, Ran Yan, Luo Mai, Shiwen Liu, and Xiao Liu. Megba: A gpu-based distributed library for large-scale bundle adjustment. In *European Conference on Computer Vision*, pages 715–731. Springer, 2022. 2

[37] David M Rosen. Scalable low-rank semidefinite programming for certifiably correct machine perception. In *Algorithmic Foundations of Robotics XIV: Proceedings of the Fourteenth Workshop on the Algorithmic Foundations of Robotics 14*, pages 551–566. Springer, 2021. 7

[38] David M Rosen, Luca Carlone, Afonso S Bandeira, and John J Leonard. Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group. *The International Journal of Robotics Research*, 38(2-3):95–125, 2019. 3, 4

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

[40] Kristy Sim and Richard Hartley. Removing outliers using the ℓ_∞ norm. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 485–494. IEEE, 2006. 3, 7

[41] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 9

[42] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. 12

[43] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6229–6238, 2021. 9

[44] Yulun Tian, Kasra Khosoussi, David M Rosen, and Jonathan P How. Distributed certifiably correct pose-graph optimization. *IEEE Transactions on Robotics*, 37 (6):2137–2156, 2021. 3

[45] Henry Wolkowicz, Romesh Saigal, and Lieven Vanden-

berghe. *Handbook of semidefinite programming: theory, algorithms, and applications*, volume 27. Springer Science & Business Media, 2012. 5

[46] Heng Yang. Semidefinite optimization and relaxation. *Working draft edition, https://hankyang.seas.harvard.edu/Semidefinite/*, 2024. 4, 5, 6

[47] Heng Yang and Luca Carlone. Certifiably optimal outlier-robust geometric perception: Semidefinite relaxations and scalable global optimization. *IEEE transactions on pattern analysis and machine intelligence*, 45(3):2816–2834, 2022. 3

[48] Heng Yang, Jingnan Shi, and Luca Carlone. Teaser: Fast and certifiable point cloud registration. *IEEE Transactions on Robotics*, 37(2):314–333, 2020. 2, 19

[49] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10371–10381, 2024. 2

[50] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv:2406.09414*, 2024. 17

[51] Wei Yin and Mu Hu. Metric3D: A toolbox for zero-shot metric depth estimation. https://github.com/YvanYin/Metric3D, 2024. 17

[52] Xihang Yu and Heng Yang. Sim-sync: From certifiably optimal synchronization over the 3d similarity group to scene reconstruction with learned depth. *IEEE Robotics and Automation Letters*, 2024. 3, 4

[53] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12786–12796, 2022. 9

APPENDIX A
PROOFS

A. Proof of Proposition 2

Proof: We begin by expressing the objective function in (3) in its vectorized form:

$$\sum_{(i,k)\in\mathcal{E}} w_{ik} \left\| R_i(s_i\tilde{u}_{ik}) + t_i - p_k \right\|^2$$
$$= \sum_{(i,k)\in\mathcal{E}} w_{ik} \left\| (\tilde{u}_{ik}^{\mathsf{T}} \otimes \mathbf{I}_3)\mathrm{vec}\left(s_i R_i\right) + t_i - p_k \right\|^2$$

where w_{ik} denotes the weight assigned to each term.

Let $e_i \in \mathbb{R}^N$ be a vector of zeros except for the i-th entry, which is set to 1. Similarly, let $\bar{e}_k \in \mathbb{R}^M$ be a vector of zeros except for the k-th entry, which is set to 1. Furthermore, define the following concatenated vectors: $t = [t_1; \ldots; t_N] \in \mathbb{R}^{3N}$ for translations, $p = [p_1; \ldots; p_M] \in \mathbb{R}^{3M}$ for landmark positions, $r = [\mathrm{vec}\left(s_1 R_1\right); \ldots; \mathrm{vec}\left(s_N R_N\right)] \in \mathbb{R}^{9N}$ for vectorized scaled rotations.

With these definitions, we simplify the objective function as $L(t, p, r)$.

$$\sum_{(i,k)\in\mathcal{E}} w_{ik} ||((e_i^{\mathsf{T}} \otimes \tilde{u}_{i,k}) \otimes \mathbf{I}_3)r + (e_i^{\mathsf{T}} \otimes \mathbf{I}_3)t - (\bar{e}_k^{\mathsf{T}} \otimes \mathbf{I}_3)p||^2$$

$$= r^{\mathsf{T}}\left(Q_1 \otimes \mathbf{I}_3\right) r + t^{\mathsf{T}}\left(Q_2 \otimes \mathbf{I}_3\right) t + p^{\mathsf{T}}\left(Q_3 \otimes \mathbf{I}_3\right) p + 2r^{\mathsf{T}}\left(V_1 \otimes \mathbf{I}_3\right) t - 2r^{\mathsf{T}}\left(V_2 \otimes \mathbf{I}_3\right) p - 2t^{\mathsf{T}}\left(V_3 \otimes \mathbf{I}_3\right) p$$

where $Q_1, Q_2, Q_3, V_1, V_2, V_3$ are as follows

$$Q_1 = \sum_{(i,k)\in\mathcal{E}} w_{ik}(e_i \otimes \tilde{u}_{i,k})(e_i^{\mathsf{T}} \otimes p_{i,k}^{\mathsf{T}}) \in \mathbb{R}^{3N\times 3N}, \quad (28)$$

$$Q_2 = \sum_{(i,k)\in\mathcal{E}} w_{ik}(e_i e_i^{\mathsf{T}}) \in \mathbb{R}^{N\times N}, \quad (29)$$

$$Q_3 = \sum_{(i,k)\in\mathcal{E}} w_{ik}(\bar{e}_k \bar{e}_k^{\mathsf{T}}) \in \mathbb{R}^{M\times M}, \quad (30)$$

$$V_1 = \sum_{(i,k)\in\mathcal{E}} w_{ik}(e_i \otimes \tilde{u}_{i,k}) e_i^{\mathsf{T}} \in \mathbb{R}^{3N\times N}, \quad (31)$$

$$V_2 = \sum_{(i,k)\in\mathcal{E}} w_{ik}(e_i \otimes \tilde{u}_{i,k}) \bar{e}_k^{\mathsf{T}} \in \mathbb{R}^{3N\times M}, \quad (32)$$

$$V_3 = \sum_{(i,k)\in\mathcal{E}} w_{ik} e_i \bar{e}_k^{\mathsf{T}} \in \mathbb{R}^{N\times M}, \quad (33)$$

We now solve out t and p by taking the gradient of $L(t, p, r)$ w.r.t. t and p respectively. Let $y = \begin{bmatrix} t \\ p \end{bmatrix}$. We can represent $L(t, p, r)$ using y, r as follows:

$$L(y, r) = r^{\mathsf{T}}\left(Q_1 \otimes \mathbf{I}_3\right) r - 2r^{\mathsf{T}}\left(V_{tp} \otimes \mathbf{I}_3\right) y + y^{\mathsf{T}}\left(Q_{tp} \otimes \mathbf{I}_3\right) y \quad (34)$$

where Q_{tp}, V_{tp} are as follows:

$$Q_{tp} = \begin{bmatrix} Q_2 & -V_3 \\ -V_3^{\mathsf{T}} & Q_3 \end{bmatrix} \in \mathbb{R}^{(N+M)\times(N+M)} \quad (35)$$

$$V_{tp} = \begin{bmatrix} -V_1 & V_2 \end{bmatrix} \in \mathbb{R}^{3N\times(N+M)}. \quad (36)$$

set $\nabla_y L(y, r) = 0$ we obtain

$$(Q_{tp} \otimes \mathbf{I}_3) y = (V_{tp} \otimes \mathbf{I}_3)^{\mathsf{T}} r. \quad (37)$$

Let $\mathcal{G}$ denote the view-graph. Next we prove Q_{tp} can be represented as:

$$Q_{tp} = L(\mathcal{G}) \quad (38)$$

where $L(\mathcal{G})$ is the Laplacian of $\mathcal{G}$:

Lemma 13. *Q_{tp} is the Laplacian of $\mathcal{G}$.*

Proof: Note that $\mathcal{G}$ is a weighted undirected graph. Calling $\delta(q)$ for the set of edges incident to a vertex q, and $w_e = w_{qp}$ for $e = (q, p)$, the Laplacian of $\mathcal{G}$ is:

$$L(\mathcal{G})_{qp} = \begin{cases} \sum_{e\in\delta(q)} w_e & \text{if } q = p, \\ -w_{qp} & \text{if } (q, p) \in \mathcal{E}, \\ 0 & \text{if } (i, j) \notin \mathcal{E}. \end{cases} \quad (39)$$

On the other hand, by expanding (35) and compare it with (39), we finish the proof. □

Calling $\bar{y} = [t_2; \ldots; t_N; p_1; \ldots; p_{N+M}] \in \mathbb{R}^{3(N+M)-3}$ and $\bar{Q}_{tp} = [c_2; \ldots; c_{N+M}] \in \mathbb{R}^{(N+M)\times(N+M-1)}$ where c_i is the i-th column of Q_{tp}. We prove when fix $s_1 = 1$, $R_1 = \mathbf{I}_3$, $t_1 = 0$, y has a unique solution:

$$(\bar{Q}_{tp} \otimes \mathbf{I}_3)\bar{y} = -(V_{tp} \otimes \mathbf{I}_3)^{\mathsf{T}} r \quad (40)$$

Since $\mathrm{rank}\left(L(\mathcal{G})\right) = N + M - 1$ and $\sum_{i=1,..,N} c_i = \mathbf{0}_N$, then $span(c_1, ..., c_n) = span(c_2, ..., c_n)$ and $\mathrm{rank}\left(\bar{Q}_{tp}\right) = N + M - 1$ which implies that $\bar{Q}_{tp}$ has full column rank. Hence, by taking inverse and rearrange, we obtain

$$\bar{y} = \left(\bar{A} \otimes \mathbf{I}_3\right) r \quad (41)$$

where

$$\bar{A} = -(\bar{Q}_{tp}^{\mathsf{T}}\bar{Q}_{tp})^{-1}\bar{Q}_{tp}^{\mathsf{T}} V_{tp}^{\mathsf{T}} \quad (42)$$

Together with $t_1 = 0$,

$$y = \left(A \otimes \mathbf{I}_3\right) r \quad (43)$$

with

$$A = \begin{bmatrix} \mathbf{0}_{1\times 3N} \\ -(\bar{Q}_{tp}^{\mathsf{T}}\bar{Q}_{tp})^{-1}\bar{Q}_{tp}^{\mathsf{T}} V_{tp}^{\mathsf{T}} \end{bmatrix} \in \mathbb{R}^{(N+M)\times 3N} \quad (44)$$

Now we have a closed form solution of y. Plug in the solution of y into (34), we obtain:

$$L(r) = r^{\mathsf{T}}\left(\left(A^{\mathsf{T}} Q_{tp} A + V_{tp} A + A^{\mathsf{T}} V_{tp}^{\mathsf{T}} + Q_1\right) \otimes \mathbf{I}_3\right) r$$

Note that

$$r = [\mathrm{vec}\left(s_1 R_1\right); \ldots; \mathrm{vec}\left(s_N R_N\right)] = \mathrm{vec}\left(U\right)$$

Then $L(r)$ is equivalent to

$$\mathrm{vec}\left(U\right)^{\mathsf{T}}\left(\left(A^{\mathsf{T}} Q_{tp} A + V_{tp} A + A^{\mathsf{T}} V_{tp}^{\mathsf{T}} + Q_1\right) \otimes \mathbf{I}_3\right) \mathrm{vec}\left(U\right)$$

Rewriting this in a more compact matrix representation yields:

$$\rho^\star = \min_R \mathrm{tr}\left(QU^{\mathsf{T}}U\right) \quad (45)$$

$$Q := A^{\mathsf{T}} Q_{tp} A + V_{tp} A + A^{\mathsf{T}} V_{tp}^{\mathsf{T}} + Q_1 \in \mathbb{S}^{3N} \quad (46)$$

concluding the proof. ■

TABLE X: Results of different Depth Estimation Model on the Mip-Nerf datasets.

Method	Unidepth			Depth-pro			DepthAnything-v2			Metric3D-v2		
Metrics	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T	Processing Time Solver Time	ATE-R ATE-T	RPE-R RPE-T
Kitchen-279 -187865	229.82 **0.8 + 19.19**	0.154° 0.018	0.201° 0.027	436.19 1.13 + 21.54	**0.051°** **0.006**	**0.062°** **0.008**	224.25 0.9 + 29.52	0.384° 0.053	0.478° 0.072	**216.0** 1.66 + 107.85	0.621° 0.064	0.785° 0.099
Garden-185 -106858	**174.57** 0.65 + 3.98	0.021° **0.002**	**0.025°** **0.003**	322.73 0.54 + 16.96	0.061° 0.007	0.035° 0.01	179.88 **0.61 + 3.82**	**0.02°** **0.002**	**0.025°** **0.003**	197.72 1.01 + 21.74	0.059° 0.007	0.035° 0.009
Bicycle-194 -41866	**145.75** 0.58 + 22.47	0.229° 0.019	0.219° 0.028	299.62 0.57 + 11.61	0.05° **0.003**	0.039° 0.006	152.37 **0.53 + 3.44**	**0.034°** **0.003**	**0.035°** **0.004**	168.63 1.11 + 7.0	19.257° 1.123	7.533° 1.466
Room-311 -111783	223.44 **1.01 + 24.35**	0.06° 0.002	0.084° **0.002**	429.43 1.18 + 33.49	**0.053°** **0.001**	**0.06°** **0.002**	190.9 0.99 + 37.35	0.067° 0.002	0.095° **0.002**	**180.55** 1.08 + 86.68	2.54° 0.052	1.584° 0.083

B. Proof of Proposition 3

Proof: We show that under (9), if $U^\star$ is a global optimizer of (8), then it is also a global optimizer of (4).

First, note that (8) is a relaxation of (4), because $\mathrm{O}(3) \subseteq \mathrm{SO}(3)$. Thus, we have:

$$\mathrm{tr}\left(Q(U^\star)^\mathsf{T} U^\star\right) = \rho^\star_{\mathsf{QCQP}} \leq \rho^\star. \tag{47}$$

From (9) and the definition $\bar{R}_i^\star = s_i^\star R_i^\star$, we obtain

$$\det(R_i^\star) = \frac{\det(\bar{R}_i^\star)}{(s_i^\star)^3} > 0. \tag{48}$$

This implies that $U^\star$ is also a feasible solution to (4). By (47), $U^\star$ is therefore the global optimizer of (4). ■

C. Proof of Proposition 5

Proof: We first establish that (11) is a relaxation of (8). Let $X = U^\mathsf{T} U$. By definition, we have $x^\mathsf{T} X x = (Ux)^\mathsf{T} Ux \geq 0$ for all $x \in \mathbb{R}^{3N}$, which implies that $X \succeq 0$.

Furthermore, since $\bar{R}_i \in \mathfrak{s}\mathrm{O}(3)$, it follows that $\bar{R}_i^\mathsf{T} \bar{R}_i = s_i^2 \mathbf{I}_3 \triangleq \alpha \mathbf{I}_3$. This ensures that the feasible set of (11) is broader than that of (8), confirming that it is indeed a relaxation.

Next we prove if $\mathrm{rank}\,(X^\star) = 3$, we can extract the global minimizer of (8) from $X^\star$. Let $X^\star = (\bar{U}^\star)^\mathsf{T} \bar{U}^\star$, then $\bar{U}^\star$ is rank 3. We can decompose $\bar{U}^\star$ as

$$\bar{U}^\star = \begin{bmatrix} \bar{R}_1^\star & \bar{R}_2^\star & \dots & \bar{R}_N^\star \end{bmatrix} \tag{49}$$

Since X is feasible, we have $\bar{R}_i^\star \bar{R}_i^\star = \alpha \mathbf{I}_3 = s_i^2 \mathbf{I}_3$, which implies that $\bar{R}_i^\star / s_i \in \mathrm{O}(3)$.

Define $U^\star = \bar{R}_1^{\star T} \bar{U}^\star$, making $U^\star$ a feasible solution to (8). Moreover, since

$$\mathrm{tr}\left(Q(U^\star)^\mathsf{T} U^\star\right) = \rho^\star_{\mathsf{SDP}} \leq \rho^\star_{\mathsf{QCQP}}, \tag{50}$$

it follows that $U^\star$ is the global minimizer of (8). ■

D. Proof of LICQ condition

Proof: Since $U_r = [R_1, \dots, R_n], R_k \in \mathbb{R}^{r\times 3}$ satisfies (19), we have $R_k^T R_k = \alpha_k I_3, \alpha_k > 0$. Therefore, $\mathrm{rank}(R_k) = 3, \forall k$, i.e., R_k is column full-rank.

We show the structure of the $\{A_i\}_{i=1}^m$ matrices guarantees LICQ. Recall $\{A_i\}_{i=1}^m$ enforces the diagonal 3×3 blocks of X in (19) to be scaled identity matrices (for the first block an additional constraint enforces the diagonal entries to be 1). Therefore, the k-th group of $\{A_i\}_{i=1}^m$ has the following form

$$A_{5k-4+\ell} = \begin{bmatrix} 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & B^\ell & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 \end{bmatrix}, \ell = 1, ..., 5$$

where the matrix is all zero except in the k-th 3×3 block:

$$B^1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B^2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}, B^3 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B^4 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, B^5 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

Due to this structure, $A_{5k-4+\ell} U_r^T$ is entirely zero except for rows $3k-2$ through $3k$. So $A_{5k-4+\ell} U_r^T$ for different values of k are clearly linearly independent since their nonzero regions do not overlap. Hence, we only need to investigate LICQ given a fixed k.

Proof by contradiction: suppose there exists $\gamma_\ell, \ell = 1, \dots, 5$ such that

$$(\sum_{\ell=1}^{5} \gamma_\ell B^\ell) R_k^T = 0,$$

then, because R_k^T is row full-rank, we have $\sum_{\ell=1}^5 \gamma_\ell B^\ell = 0$, and thus $\gamma_\ell = 0, \forall \ell$. This proves that $\{A_i U_r^T\}_{i=1}^m$ must be linearly independent. ■

Appendix B
Depth Models

We compare the performance of different depth estimation models on the Mip-Nerf datasets. The models we consider are **Unidepth** [35], **Depth-pro** [10], **DepthAnything-V2** [50], and **Metric3D-v2** [51]. We evaluate these models using ATE/RTE and runtime as detailed in Section V. The results are shown in Table X.

All four models produce accurate results, demonstrating the robustness of our XM-SfM pipeline to different depth models. Unidepth is the fastest, while DepthAnything-V2 and Metric3D-v2 are slightly slower but occasionally more accurate. Depth-Pro is the slowest yet consistently stable in accuracy.

Appendix C
Breakdown Plot of Run Time

We present time breakdown plots (Fig. 11, Fig. 12, Fig. 13), for three different cases. Depth estimation scales linearly with

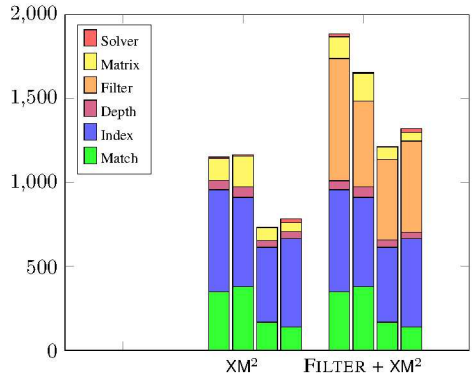


Fig. 11: Breakdown Results on the Replica dataset (2000 frames)

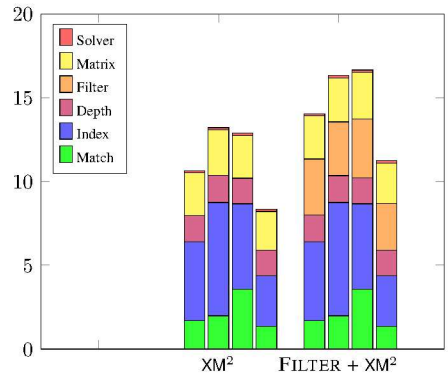


Fig. 12: Breakdown Results on the Replica dataset (100 frames)

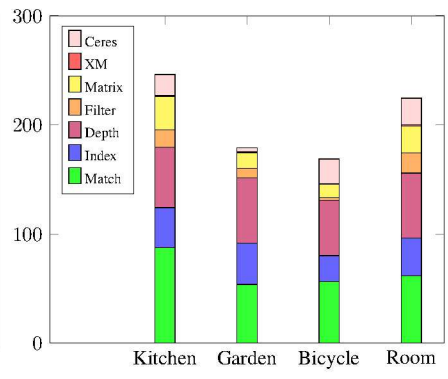


Fig. 13: Breakdown Results on the Mip-Nerf dataset

the number of frames, making it more significant in smaller datasets. Matching and indexing become dominant in larger datasets, as exhaustive matching time grows quadratically with the number of frames. Solver time remains consistently minimal across all datasets.

APPENDIX D
SCALE REGULARIZATION

Due to inaccuracies in real-world data, the scale of frames $2, \ldots, N$ is often significantly smaller than that of the first frame[4]. As a result, the global minimum tends to collapse all cameras $2, \ldots, N$ into a single point to avoid large errors in their estimates. To mitigate this, we introduce a scale regularization term in the objective function of (11).

$$\begin{aligned} &\min_{X \in \mathbb{S}^{3N}} \operatorname{tr}(QX) + \lambda \sum_{i=2}^{N} (X_{3i,3i} - 1)^2 \\ &\text{subject to } \langle A_i, X \rangle = b_i, \forall i \in 1, \ldots, 5N+1 \\ &\qquad X \succeq 0 \end{aligned} \tag{51}$$

This modification does not violate the previously established theorems; Slaters Condition and strong duality remain valid, and the optimality condition remains unchanged. However, the $Z(y)$ matrix is modified to

$$Z(y) = Q - \sum_{i=1}^{5N+1} y_i A_i + \lambda \nabla \Big(\sum_{i=2}^{N} (X_{3i,3i} - 1)^2 \Big) \tag{52}$$

To prove this, we first write (51) in the standard form:

$$\min \quad \langle Q, X \rangle + F(X) \tag{53}$$
$$s.t. \quad X \succ 0 \tag{54}$$
$$\langle A_i, X \rangle = b_i, \quad \forall i \tag{55}$$

[4]Anchored to be 1.

$F(X)$ is a quadratic term defined by $F(X) = \lambda \sum_{i=2}^{N} (X_{3i,3i} - 1)^2$. This is still a convex problem and the Lagrangian is:

$$L(X, y, Z) = \langle Q, X \rangle + F(X) - \tag{56}$$
$$\sum_i y_i (\langle A_i, X \rangle - b_i) - \langle Z, X \rangle \tag{57}$$
$$= \langle Q - \sum_i y_i A_i - Z, X \rangle + F(X) + \sum_i y_i b_i \tag{58}$$

The KKT condition is:

$$Z = Q + \nabla F(X) - \sum_i y_i A_i \succ 0 \tag{59}$$
$$\langle A_i, X \rangle = b_i, \quad \forall i \tag{60}$$
$$X \succ 0 \tag{61}$$
$$\langle Z, X \rangle = 0 \tag{62}$$

Note that L is linear in the off-diagonal elements of X and quadratic in the diagonal elements. To ensure a finite minimum, we require $(\nabla L)_{ij} = 0$ for $i \neq j$. For the diagonal elements, achieving the minimum requires $(\nabla L)_{ii} = 0$. Thus, together, we obtain $\nabla L = 0$, i.e.

$$Q - \sum_i y_i A_i - Z = -\nabla F(X) \tag{63}$$

From the definition of $F(x)$ we know $X_{3j,3j} = -\frac{(Q - \sum_i y_i A_i - Z)_{3j,3j}}{2\lambda} + 1$. So we plug in X then we get the dual problem:

$$\max \quad \sum_i b_i y_i - \frac{\sum_{j=2}^{N} (C - \sum_i y_i A_i - Z)^2_{3j,3j}}{4\lambda} \tag{64}$$
$$+ \sum_{j=2}^{N} (Q - \sum_i y_i A_i - Z)_{3j,3j} \tag{65}$$
$$s.t. \quad Z \succ 0 \tag{66}$$
$$(Q - \sum_i y_i A_i - Z)_{ij} = 0, \quad i \neq j. \tag{67}$$

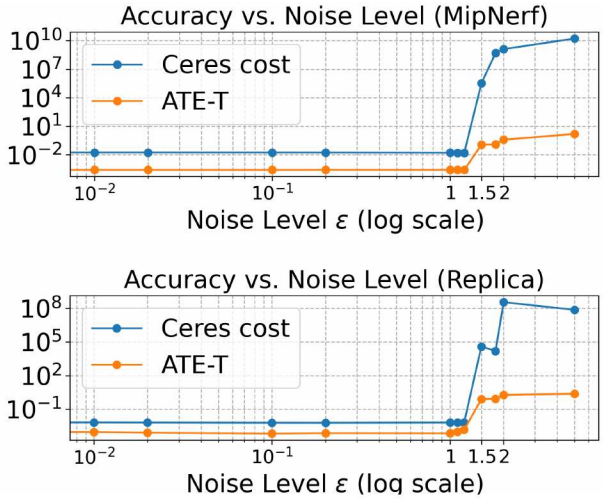


Fig. 14: Accuracy (ATE-T) and the final Ceres objective value of Replica and Mip-NeRF datasets as the depth noise level ϵ increases. When $\epsilon < 1.5$ they fall into the same local minima. The SDP relaxation breaks at $\epsilon = 5$ for Replica and never for Mip-NeRF.

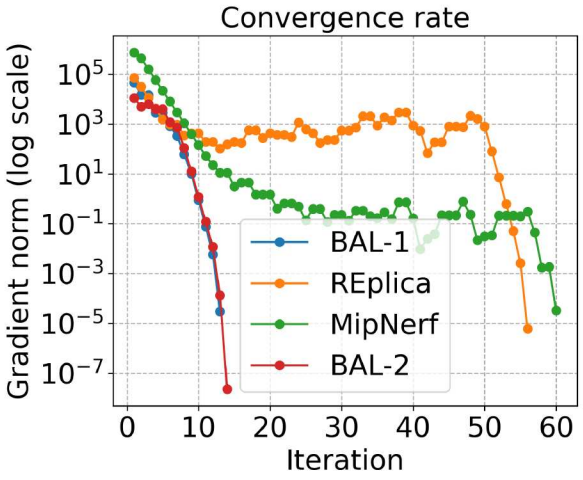


Fig. 15: Convergence rate of XM on different datasets. BAL datasets show quadratic convergence. The other two enter quadratic convergence after a slow convergence region.

APPENDIX E
SUBOPTIMALITY FOR SDP

In (16), we established suboptimality when (11) is solved to global optimality. However, in practice, numerical errors arise, and variations in gradient tolerance settings can lead to increased suboptimality. Here, we provide a rigorous proof for computing suboptimality in SDP problems and apply it directly to the scale regularization problem in (51).

Theorem 14. *Given the primal problem (51) we have*

$$\rho_{\text{SDP}} \geq \rho^{\star}_{\text{SDP}} \geq max(0, \lambda_{min}(Z))tr(X) + \rho_{dual} \tag{68}$$

Proof: for all feasible X we have:

$$\langle Q, X\rangle + F(X) \geq \langle Q - \sum_i y_i A_i + \nabla F(x), X\rangle \tag{69}$$

$$+ \underbrace{\sum_i y_i b_i + F(X) - \langle \nabla F(X), X\rangle}_{\text{dual value}} \tag{70}$$

$$= \langle Z, X\rangle + \rho_{\text{dual}} \tag{71}$$

$$\geq \max(0, \lambda_{\min}(Z))\text{tr}(X) + \rho_{\text{dual}} \tag{72}$$

and we have $\rho_{\text{SDP}} \geq \rho^{\star}_{\text{SDP}}$. ∎

ρ_{SDP} and $\max(0, \lambda_{\min}(Z))\text{tr}(X) + \rho_{\text{dual}}$ can be directly evaluated within the algorithm. A small gap between them indicates that ρ_{SDP} is close to $\rho^{\star}_{\text{SDP}}$, confirming that the problem has been solved to global optimality. We define this new suboptimality gap as:

$$\eta = \frac{\hat{\rho} - \max(0, \lambda_{\min}(Z))\text{tr}(X) - \rho_{\text{dual}}}{1 + |\hat{\rho}| + |\max(0, \lambda_{\min}(Z))\text{tr}(X) + \rho_{\text{dual}}|}. \tag{73}$$

APPENDIX F
CONVERGENCE AND NOISE ANALYSIS

A. Convergence Rate

We show the convergence rate of our algorithm on different datasets. From Riemannian trust-region algorithm with truncated conjugate gradient (Rtr-tCG) we already know that it has local quadratic convergence [11, 1]. Our experimental results (Fig. 15) verified this.

B. Noise Analysis

We test our pipeline to investigate tolerance for noise in depth estimation. Specifically, we add a noisy scaling factor s to the depth of each observation, defined as $s = (1+\epsilon)^x, x \sim \mathcal{U}(-1, 1)$, where ϵ controls the noise level. Fig. 14 right plots the relationship between ϵ and the reconstruction accuracy. The pipeline breaks at $\epsilon = 1.5$ which shows robustness against inaccurate depth. The SDP relaxation does not break until $\epsilon = 5$, a stronger level of robustness.

APPENDIX G
METRICS

The classical absolute Trajectory Error (ATE) and Relative Pose Error (RPE) are defined as:

$$\text{ATE}_i = \|\hat{T}_i - T_i\|_F^2, \tag{74}$$

$$\text{RPE}_{ij} = \|(\hat{T}_j\hat{T}_i)^{-1}T_iT_j^{-1}\|_F^2, \tag{75}$$

where T_i represents ground truth pose and $\hat{T}_i$ represents estimated transformation. Because of the potential transformation between ground truth and estimation, an alignment between the two trajectories is required. The alignment is done by solving a point-cloud registration problem[5] between two clusters of cameras.

[5]We ignore the rotation in each camera frame and align position of cameras through TEASER [48].