Abstract

This paper derives a novel linear position constraint for cameras seeing a common scene point, which leads to a direct linear method for global camera translation estimation. Unlike previous solutions, this method does not require connected camera-triplet graph, and works on images with weaker association. The final linear formulation does not involve the coordinates of scene points, which makes it efficient even for large scale data. We solve the linear equation based on robust $L_1$ norm, which makes our system more robust to outliers in essential matrices and feature correspondences. We experiment this method on both sequentially captured data and unordered Internet images. The experiments demonstrate its strength in robustness, accuracy, and efficiency.

1. Introduction

Structure-from-motion (SfM) aims to estimate scene structure and camera motion from multiple images. Usually it starts with relative motion estimation between camera pairs or triplets (e.g. [31, 42]), and registers all cameras and 3D scene points in a global coordinate system. It then applies the bundle adjustment (BA) [43] to optimize the final solution. This paper focuses on the step of registering all cameras into a common global coordinate system.

Many registration methods have been proposed, which can be roughly divided into three categories: sequential, hierarchical, and global. Sequential methods (e.g. Bundler [39]) add cameras/images one by one to the global coordinate system. Hierarchical methods (e.g. [11, 24]) iteratively merge partial reconstructions obtained from short image sequences. Both sequential and hierarchical methods rely on repetitively calling intermediate bundle adjustment to minimize error accumulation and drifting. So global methods (e.g. [29, 9, 20]) which can register all cameras simultaneously in a single step are more attractive for both efficiency and accuracy.

Most of the global methods solve the camera orientations and positions separately. The camera orientation estimation has been well studied with robust and efficient solutions such as [17]. The camera position estimation is more challenging. Many methods [13, 5, 3] rely on pairwise relative translation encoded in essential matrices to infer camera positions. While these two-view based methods achieve simple linear formulation, they often degenerate with collinear cameras because an essential matrix does not tell the scale of relative translation (i.e. the baseline length). To address this degeneracy, some methods [37, 30] exploit motion constraints from camera triplets to infer camera positions. Essentially, they propagate the scales of translation across camera triplets with a common edge. Along this direction, a direct linear formulations is derived recently in [20]. However, these triplet-based methods only reconstruct cameras in a strongly connected triplet graph, where two triplets are connected by their common edge. The 3D reconstruction will distort or break into disconnected components when such strong association among images does not exist.

This problem is exemplified in Figure 1 with the method in [20]. The Street example on the top accidently has large camera separation at some points (see the two highlighted
neighboring images), which breaks the whole image set to three disconnected triplet graphs and generates three isolated reconstructions. In the Seville example on the bottom, those Internet images are mostly captured from two viewpoints (see the two representative sample images) with weak affinity between images at different viewpoints. This weak data association causes seriously distorted reconstruction.

To address this problem, we exploit position constraints among cameras linked by a feature track, so that the scales of translation can easily propagate to further cameras. In this way, we avoid degeneracy at collinear motion and deal with weakly associated data at the same time. Our method is based on a novel constraint arising in a triangle formed by two camera centers and scene point. According to this constraint, the positions of cameras linked by a feature track should satisfy a linear equation. We solve these linear equations collected from different feature tracks together to recover all camera positions simultaneously. Furthermore, the final linear equation does not include any scene point, which significantly reduces the equation size and makes our method efficient.

At the same time, this elegant direct linear formulation lends us sophisticated optimization tools, such as $L_1$ norm optimization [6, 12, 4, 34]. We minimize the $L_1$ norm when solving the linear equation of camera positions. In this way, our method can tolerate a larger amount of outliers in both essential matrices and feature correspondences. The involved $L_1$ optimization is nontrivial. We derive a linearization of the alternating direction method of multipliers (ADMM) algorithm [4] to address it.

2. Related Work

Traditional approaches. Most of well-known SfM systems register cameras sequentially [36, 39, 40, 35, 2] or hierarchically [11, 24, 19] from pairwise relative motions. In order to minimize error accumulation, frequent intermediate bundle adjustment is required for both types of methods, which significantly reduces computation efficiency. The performance of sequential methods relies heavily on the choice of the initial image pair and the order of subsequent image additions [16].

Global rotation estimation. Global SfM methods solve all camera poses simultaneously. Most of these methods take two steps. Typically, they solve camera orientations first and then positions. The orientation estimation is well studied with an elegant rotation averaging algorithm presented in [17]. The basic idea was first introduced by Govindu [13], and then developed in several following works [14, 29, 17, 7].

Global translation estimation. The translation estimation is more challenging, since an essential matrix does not tell the scale of translation. Some pioneer works [13, 5, 15] solved camera positions by enforcing consistency between pairwise camera translation directions and those encoded in essential matrices. Arie-Nachimson et al. [3] computed camera positions with a novel decomposition of the essential matrix. However, these two-view based methods all degenerate with collinear camera motion.

This degeneracy can be avoided by exploiting relative motion constraints from camera triplets. Sim et al. [37] computed ‘triad direction vectors’ from camera triplets to determine all camera positions simultaneously. For triplet graph with a cycle, Courchay et al. [8] glued individual triplet reconstructions by loop analysis and nonlinear optimization. Sinha et al. [38] reconstructed triplets by aligning pairwise reconstructions, and then solved camera positions from them. Moulon et al. [30] estimated camera positions and the magnitudes of their relative translations together by linear programming. Jiang et al. [20] derived a direct linear formulation from triplets based constraints to solve all camera positions. While these triplet-based methods can avoid degenerated camera motion, they often require strong association among images – a connected triplet graph, where camera triplets are connected by common edges.

In comparison, our method is based on constraints derived from feature tracks. It has a unique strength that it does not require a connected camera triplet graph and does not degenerate on collinear motion.

Some global methods estimate camera positions and 3D points together. Kahl and Hartley [22] minimized the $L_{\infty}$-norm of reprojection error and solved the problem with quasi-convex optimization. This formulation, while being computationally and memory expensive, produces globally optimal estimation. Later works proposed to speed up the computation by selecting only representative points [29], using fast optimization algorithms [33, 1, 25], or customized cost function and optimization procedure [49]. However, the $L_{\infty}$-norm is known to be sensitive to outliers and careful outlier removal is necessary [32, 10]. Randall et al. [9] combined the strength of discrete and continuous optimization to solve camera poses from camera-camera and camera-point position constraints. It is demonstrated on impressive city-scale reconstruction with some aid from GPS. Generally speaking, involving scene points improves the robustness/accuracy of camera registration, but also complicates the optimization.

Dealing with incorrect epipolar geometries. The estimated pairwise relative motions (i.e. essential matrices) often contain significant amount of outliers due to feature matching failures or repetitive scene structures. Our robust estimation is related to works that filter out incorrect epipolar geometries (EG). ‘Missing correspondences’ analysis among image triplets was proposed in [47] to identify wrong image association. Zach et al. [48] used loop consistency analysis to infer the validity of each EG. Wilson
et al. [44] constructed a visibility graph of the feature tracks and adopted a network analysis for detecting unlikely links in the visibility graph. Jiang et al. [21] searched for the correct reconstruction by iterative reconstructions and reasoning missing correspondences. Chatterjee et al. [7] proposed a robust algorithm for global rotation estimation using modern $L_1$ optimization and reweighted least squares, while they did not address the translation estimation. Wilson et al. [45] removed EG outliers by solving a 1D ordering problem after computing global rotations using [7]. In general, a robust SFM method should try to remove EG outliers as more as possible, and adopt the robust solver at the same time.

3. Global Translation Estimation

Given an essential matrix between two images $i, j$ (e.g. computed by the five-point algorithm [31, 26]), we obtain the relative rotation $R_{ij}$ and translation direction $t_{ij}$ between the two cameras. Here, $R_{ij}$ is a $3 \times 3$ orthonormal matrix and $t_{ij}$ is a $3 \times 1$ unit vector. We further denote the global orientation and position of the $i$-th ($1 \leq i \leq N$) camera as $R_i$ and $c_i$, respectively. These camera poses are constrained by the following equations

$$R_j = R_{ij} R_i, \quad R_j(c_i - c_j) \simeq t_{ij}. \quad (1)$$

Here, $\simeq$ means equal up to a scale.

Like most global methods, we compute camera orientations first and solve camera positions after that. We adopt the global rotation estimation method in [7]. In order to enhance the robustness, we remove outlier essential matrices by loop verification [48] and careful initialization, which are detailed in Section 5. Our translation estimation is based on a linear position constraint arising from a triangle formed by two camera centers and a scene point. By this constraint, the positions of different camera pairs linked by a feature point should satisfy a linear equation. Essentially, it propagates the scales of translation among all camera pairs on a feature track.

3.1. Linear constraint in a triangle

The 3D position of a scene point $p$ can be triangulated from its projections in two cameras. As shown in Figure 2, we compute $p$ as the middle point of the mutual perpendicular line segment $AB$ of the two rays passing through $p$’s image projections. Specifically, it is computed as

$$p = \frac{1}{2}(A + B) = \frac{1}{2}(c_i + s_i x_i + c_j + s_j x_j). \quad (2)$$

Here, $c_i$ and $c_j$ are the two camera centers. The two unit vectors $x_i$ and $x_j$ origin from the camera centers and point toward the image projections of $p$. $s_i$ and $s_j$ are the distances from the points $A, B$ to $c_i, c_j$ respectively, i.e. $A = c_i + s_i x_i$ and $B = c_j + s_j x_j$.

[Diagram of Figure 2: The scene point $p$ is the middle point of the mutual perpendicular line segment $AB$ of the rays passing through $p$’s image projections. The positions of $p, c_i,$ and $c_j$ satisfy a linear constraint detailed in Section 3.1.]

Following [20], we compute $x_i$ and $x_j$ by rotating the relative translation direction $c_{ij}$ between $c_i$ and $c_j$, which gives us

$$p = \frac{1}{2} \left( c_i + s_i R(\theta_i) \frac{c_j - c_i}{\|c_j - c_i\|} + c_j + s_j R(\theta_j) \frac{c_i - c_j}{\|c_i - c_j\|} \right). \quad (3)$$

The two 3D rotation matrices $R(\theta_i)$ and $R(\theta_j)$ rotate the relative translation direction $c_{ij}$ to the directions $x_i$ and $x_j$ respectively. Both rotations can be computed easily. In addition, the two ratios $s_i/\|c_j - c_i\|$ and $s_j/\|c_j - c_i\|$ can be computed easily by the middle-point algorithm [18]. Specifically, assuming unit baseline length, in the local coordinate system attached to one of the cameras, $c_i, c_j, x_i,$ and $x_j$ are all known. Thus, we can solve $s_i$ and $s_j$ (they are actually the two ratios in Equation 3 for general baseline length) from

$$(c_i + s_i x_i - c_j - s_j x_j) \times (x_i \times x_j) = 0. \quad (4)$$

Here, $\times$ is the cross product of vectors. Thus, Equation 3 becomes,

$$p = \frac{1}{2} \left( (A^i_j - A^j_i)(c_i - c_j) + c_i + c_j \right) \quad (5)$$

where $A^i_j = s_i/\|c_j - c_i\| R(\theta_i)$ and $A^j_i = s_j/\|c_i - c_j\| R(\theta_j)$ are known matrices. This equation provides linear constraints among positions of two camera centers and a scene point.

3.2. Constraints from a feature track

If the same scene point $p$ is visible in two image pairs $c_i, c_j$ and $c_k, c_l$, we obtain two linear equations about $p$’s position according to Equation 5. We can eliminate $p$ from these equations to obtain a linear constraint among four camera centers as the following,

$$(A^i_j - A^j_i)(c_i - c_j) + c_i + c_j$$

$$= (A^k_l - A^l_k)(c_k - c_l) + c_k + c_l. \quad (6)$$
Equation 6 elegantly correlates the scales of translation (i.e., baseline length $\|c_i - c_j\|$ and $\|c_k - c_l\|$) of far apart camera pairs sharing a common scene point. In other words, the scale information is propagated through feature tracks. In comparison, triplet-based methods (e.g., [37, 20, 30]) can only propagate the scale information over camera triplets sharing an edge. Therefore, this new formulation can reconstruct images with much weak association. For example, in the Seville data in Figure 1 (bottom row), $c_1, c_2$ could come from one popular viewpoint of the building, and $c_k, c_l$ could come from a different viewpoint. As long as there is a feature track linking them together, Equation 6 provides position constraints among these far apart cameras. The difference between our method and [20] can be easily seen from a simple case of three cameras. The method in [20] requires essential matrices of all three camera pairs, while our method requires only two of them. Furthermore, since there are abundant feature tracks, we can select the most reliable ones to solve camera positions, which is desired for system robustness. More detail of such selection will be discussed in Section 5.

4. Robust Estimation by $L_1$ Norm

Our linear global method requires solving a linear system like $Ax = 0$ to estimate camera centers. (Note, we abuse the symbol $x$ in this section to keep the denotation simple.) Here, $x$ is a vector formed by concatenating all camera centers, and $A$ is the coefficient matrix obtained by stacking Equation 6 from feature tracks. The 3D reconstruction process is noisy and involves many outliers, both in the pairwise relative motion (i.e., essential matrices) and feature correspondences. We enhance the system robustness by minimizing the $L_1$ norm, instead of the conventional $L_2$ norm. In other words, we solve the following optimization problem,

$$\arg\min_x \|Ax\|_1, \quad s.t. \quad x^\top x = 1. \quad (7)$$

This problem might be solved by iterative reweighted total least square, which is often slow and requires good initialization. We capitalize on the recent alternating direction method of multipliers (ADMM) [4] for better efficiency and large convergence region. Due to the quadratic constraint, i.e., $x^\top x = 1$, the ADMM algorithm cannot be directly applied to our problem. We linearize the optimization problem in the inner loop of ADMM to solve Equation 7.

Let $e = Ax$, the augmented Lagrangian function of Equation 7 is

$$L(e, x, \lambda) = \|e\|_1 + \langle \lambda, Ax - e \rangle + \frac{\beta}{2} \|Ax - e\|^2, \quad s.t. \quad x^\top x = 1, \quad (8)$$

where $\lambda$ is the Lagrange multiplier, $\langle \cdot, \cdot \rangle$ is the inner product, and $\beta > 0$ is a parameter controlling the relaxation. We then iteratively optimize $e, x, \lambda$ in Equation 8. In each iteration, we update $e_{k+1}, x_{k+1}, \lambda_{k+1}$ according to the following scheme,

$$e_{k+1} = \arg\min_e L(e, x_k, \lambda_k) = \arg\min_e \|e\|_1 + \langle \lambda_k, Ax_k - e \rangle + \frac{\beta}{2} \|Ax_k - e\|^2, \quad (9)$$

$$x_{k+1} = \arg\min_x L(e_{k+1}, x, \lambda_k) = \arg\min_x \langle \lambda_k, Ax - e_{k+1} \rangle + \frac{\beta}{2} \|Ax - e_{k+1}\|^2, \quad (10)$$

$$\lambda_{k+1} = \lambda_k + \beta (Ax_{k+1} - e_{k+1}), \quad (11)$$

where $\Omega := \{x^\top x = 1 | x \in \mathbb{R}^n\}$.

A closed-form solution [27] exists for the minimization of Equation 9. (Please see Appendix A for the formula.) However, solving Equation 10 is hard because of the quadratic constraint on $x$ (i.e., $x^\top x = 1$). Therefore, we linearize Equation 10 at $x_k$ to solve it efficiently. After simple derivation (please see Appendix B for more details), we have a closed-form solution for $x_{k+1}$ as,

$$x_{k+1} = \frac{C}{\|C\|^2}, \quad (12)$$

where $C = x_k - \frac{1}{\beta_k} A^\top (Ax_k - e_{k+1}) - \frac{1}{\beta_k^2} A^\top \lambda_k$.

In order to speed up convergence [28], we adopt a dynamic parameter $\beta$ as,

$$\beta_{k+1} = \min\{\beta_{\text{max}}, \rho \beta_k\}, \quad (13)$$

where $\rho > 1$. We set $\rho$ as 1.01 in our experiments.

Algorithm 1 summarizes our linearized ADMM algorithm.

Algorithm 1. Our linearized ADMM for Equation 7.

1. **Initialize**: Set $x_0$ as to the $L_2$ solution (i.e., the eigenvector with smallest eigenvalue of $A$), $e_0 = 0, \lambda_0 = 0, \beta_0 = 10^{-6}$.
2. **while** not converged, **do**
   3. **Step 1**: Update $e$ by solving Equation 9;
   4. **Step 2**: Update $x$ by solving Equation 12;
   5. **Step 3**: Update $\lambda$ by solving Equation 11;
   6. **Step 4**: Update $\beta$ by solving Equation 13;
   7. **end while**
5. Additional Robust Issues

To apply our algorithm to reconstruct a set of images, we first build a ‘match graph’, where each camera/image is a vertex and two vertices are connected if an essential matrix can be computed between them. Our method can reconstruct a connected component of the match graph. This section contains additional implementation details to build a robust system.

5.1. Filtering outlier EGs

There are often incorrect essential matrices caused by feature matching failure or repetitive scene structures. This problem is more severe when processing unordered Internet images. Without intermediate bundle adjustments, global methods are more sensitive to these outliers. Although the L1 optimization tolerates more outliers than the conventional L2 solution, we find it is still necessary to pre-filter the essential matrices.

**Filtering before orientation estimation.** We adopt the loop verification [48] on the pairwise relative camera orientations. We only analyze loops of three cameras for simplicity. Specifically, we chain the relative rotations along a three-camera loop as \( R = R_{ij} R_{jk} R_{ki} \), and compute the angular difference [17] between \( R \) and the identity matrix. If the difference is larger than a threshold \( \phi_1 \) (3 or 5 degrees for sequential data or unordered Internet data), all three essential matrices are discarded.

**Initializing orientation estimation.** We then apply the L1 method in [7] to estimate camera orientations from remaining essential matrices. We initialize camera orientations by chaining the relative rotations on the maximum spanning tree of the match graph, where graph edges are weighted by the number of feature correspondences.

**Filtering with estimated orientations.** We further filter essential matrices with the computed camera orientations. Specifically, for each camera pair, we compute their relative rotation from their orientations estimated by [7], and compare it with the one encoded in their essential matrix. If the difference is larger than a threshold \( \phi_2 \) (set to 5 or 10 degrees), we discard that essential matrix.

5.2. Feature tracks selection

Since there are usually abundant feature tracks to solve camera positions, we carefully choose the most reliable ones to enhance system robustness. For better feature matching quality, we only consider feature correspondences that are inliers of essential matrix fitting. We sort all feature tracks by their lengths in descending order, and add tracks to our system one by one. A feature track is skipped, if all cameras on it have been covered by previously added tracks for at least \( K \) times. (We set \( K = 30 \) in our experiments.)

For a feature track with \( L \) camera pairs on it, we can choose any two pairs to build the Equation 6. In this way, we can generate \( L(L - 1)/2 \) linear equations. In fact, only \( L - 1 \) of them are independent. So we select the most reliable ones. We consider the match graph formed by these cameras. Since we only consider feature correspondences passing essential matrix verification, this graph has only one connected component. We weight each graph edge by \( M + \alpha \frac{1}{\theta_p} \), where \( M \) is the number of feature matches between two images, and \( \theta_p \) is the angle between their principal axes. The combination weight \( \alpha \) is fixed at 0.1 in all experiments. We take the maximum spanning tree of this graph and use neighboring tree edges to build linear equations for camera centers from Equation 6.

6. Experiments

We evaluate our method with various data. All our experiments were run on a machine with two 2.4GHz Intel Xeon E5645 processors and 48G RAM. We use ARPACK [23] to solve the sparse eigenvalue problem.

6.1. Evaluation on synthetic data

We first evaluate the performance of our algorithm on synthetic data with known ground truth. The synthetic data consists of three cameras and 500 scene points generated according to the following setting. As illustrated in Figure 3, camera \( c_0 \) is placed at the world origin and camera \( c_2 \) is at a random location away from \( c_0 \) by 0.2 unit. The position of \( c_1 \) is generated in different ways to evaluate different aspects of the SfM system. Each camera has a field of view of 45° and image resolution of 352 \( \times \) 288 pixels. The scene points are generated randomly within the viewing volume of the first camera, spanning a depth range of about 0.5 unit. The scene distance from the first camera is about 1 unit. We compute the essential matrices between each camera pair using the five-point algorithm [31].

Two metrics are used to evaluate the accuracy of computed camera poses. The error of relative translation direc-

![Figure 3. Configuration for synthetic data generation. We randomly place cameras \( c_0, c_1, \) and \( c_2 \) and 500 scene points as illustrated here. We then test each method 100 times on the randomly created images, and evaluate their accuracy with respect to the ground truth.](image-url)
tions $t_{err}$ is the mean angular difference (in degrees) between the estimated and the true baseline directions. The error of camera positions $c_{err}$ is the mean Euclidean distance between the estimated and the true camera centers. We report the average results of 100 random trials for each method.

**Comparison with [3].** We compare our method with the method in [3] to demonstrate the robustness of our algorithm with collinear camera motion. We randomly sample $c_1$ to vary the angle $\angle c_1c_0c_2$ from 0.1 to 5 degrees while ensuring $c_1$ keep equal distances from $c_0$ and $c_2$. Image coordinates of the projected 3D points are perturbed by zero mean Gaussian noise with standard deviation $\sigma = 0.4$ pixels. A comparison of the results from these two methods are shown in Figure 4. According to both error metrics $t_{err}$ and $c_{err}$, the performance of our method is quite stable under collinear camera motion.

**Comparison with [20].** We compare our method with the method in [20] on data with weak association. In this experiment, we fix the $c_2$ at the point $(-0.2, 0, 0)$ and randomly sample $c_1$ with $\angle c_1c_0c_2$ fixed at 45 degrees. Moreover, we fix the number of feature correspondences between $c_1$ and $c_2$ at 10, and adjust the standard deviation $\sigma$ of the Gaussian noise on feature positions from 0.1 to 1 pixel. This setup simulates situations of weak data association (i.e., the essential matrix between $c_1$, $c_2$ is poor). The reconstruction accuracies of both methods are reported in Figure 5. It is clear that as $\sigma$ increases, our method produces more accurate results than [20] according to both metrics. Note that our method can work even when there is 0 feature correspondence between $c_1$ and $c_2$, while the method in [20] requires the essential matrix between them.

### 6.2. Evaluation on benchmark data

We compare our method with a typical incremental SfM solution, the VisualSFM [46], and several global SfM methods on the benchmark data provided in [41]. We use both ground truth camera intrinsics and approximate intrinsics from EXIF in the experiment. The results on VisualSFM [46] is obtained by running the software provided by its author. The results for other methods are either obtained from their authors or cited from their publications.

We summarize all the results in Table 1 and Table 2. All results are evaluated after the final bundle adjustment. We only report the metric $c_{err}$ since the results on $t_{err}$ is not available for [30] and [3]. Since [45] includes a random initialization, we test it with 50 trials and take the result with smallest error. We notice that [45] has poor performance for these datasets and fail for castle-P30. We believe this is because the method in [45] is designed for Internet images with $O(n^2)$ image pairs. Sequential data with $O(n)$ pairs is hard for [45].

To further evaluate the $L_1$ norm optimization, we also experiment the conventional $L_2$ norm optimization with our formulation. The results are indicated as $L_1$ and $L_2$ respectively in these two tables. Our method generally produces the smallest errors with both ground truth intrinsics or approximate ones from EXIF. The $L_2$ and $L_1$ methods produce similar results on the fountain-P11 and Herz-Jesu-P25 data, since both of them have few outliers. But the $L_1$ norm outperforms $L_2$ significantly on the castle-P30 data, whose essential matrix estimation suffers from repetitive scene structures. Note the $L_2$ method works reason-
able on the castle-P30 data when the ground truth intrinsics are used. We find it is due to the final bundle adjustment, and the $L_2$ method generates much larger error than the $L_1$ method before bundle adjustment.

### 6.3. Evaluation on general data

We further evaluate our algorithm with five median to large scale image collections, including both sequentially captured images and unordered Internet images. The Building data is obtained from [48], which consists of 128 sequentially captured images. There are outlier essential matrices due to the repetitive windows. The Street data contains 168 sequential street-view images captured by ourselves with relatively large separation between cameras. The Trevi Fountain data consists of 1259 images downloaded from Flickr.com. The Notre Dame data is obtained from [39] and consists of 568 images (with EXIF tags). The Seville data is from [44], which we take 301 images in the largest connected component of the match graph. This data is heavily contaminated with wrong feature matches. Further, many of these images are captured from two far apart popular tourist viewpoints. The association between images from different viewpoints is weak. We use the feature tracks provided by the authors of [44].

We further evaluated our method on the noisy datasets published in [45] and the Quad data in [9]. These datasets are quite noisy and contain many incorrect EGs and repetitive features. The Quad data is in particular difficult, which consists of 6514 images and is the only large-scale dataset with known ‘ground truth’ camera motions. Our method successfully reconstruct those examples as shown in Figures 6 – 8, which demonstrate the robustness of the proposed method.

**Comparison with [46].** We compare our method with VisualSFM [46] by the numbers of reconstructed cameras and reconstructed 3D points as well as the running time. We also quantify the difference between our reconstructions and those from VisualSFM. The results are shown in Table 3.

Generally speaking, both methods reconstruct similar number of cameras and 3D scene points. We also record the running time of both algorithms. In Table 3, $t_{reg}$ is the time spent on camera registration only, and $t_{total}$ is $t_{reg}$ plus the time spent on bundle adjustment (BA) (both intermediate and final BA). Time spent on feature matching and essential matrices estimation is excluded, since that is the same for both methods and can be trivially parallelized. Our method is a few times ($2 – 8$ times) faster than the VisualSFM on $t_{total}$. We further compare the results from our method and VisualSFM. $\Delta R$ and $\Delta c$ are the mean camera orientation difference and position difference respectively. $\Delta c$ is normalized by the largest distance between two cameras. As shown in Table 3, our method generates consistent results with VisualSFM. The result consistency on the Street data is missing, because VisualSFM (with default parameters) breaks it into 6 disconnected reconstructions since the separation between cameras is relatively big.

**Comparison with [20].** As shown in Figure 1, [20] generates large distortion for Seville and breaks Street into 3 reconstructions. This is a fundamental drawback of triplet-based methods like [20, 30], since they rely on triplets with common edges to propagate the scale information. Our method propagates scale on feature tracks and does not suffer from such limitation. Our method is slightly slower than [20], because of the $L_1$ norm solution.

**Comparison with [45].** We provide quantitative comparison with the method in [45] in Table 4. As there are repetitive features and many incorrect EGs, [45] designed a smart filtering method to exclude incorrect EGs before solve the global SFM problem. In contrast, we adopt straightforward filtering and propose a more sophisticated translation estimation algorithm. As shown in Table 4, our method has smaller initial median errors than [45] and comparable results after BA. Our method fails on three datasets including Piccadilly, Union Square, and Roman Forum due to our straightforward filtering of EGs.

### 7. Conclusion

We derive a novel linear method for global camera translation estimation. This method is based on a novel position constraint on cameras linked by feature tracks, which allows us to propagate the scale information across far apart camera pairs. In this way, our method works well even on weakly associated images. The final linear formulation does not involve coordinates of scene points, so it is easily scalable and computationally efficient. We further develop a $L_1$ optimization method to make the solution robust to outlier essential matrices and feature correspondences. Experiments on various data and comparison with recent works demonstrate the effectiveness of this new algorithm.

| Data and # of input images | # of reconstructed cameras | # of reconstructed points | Registration time $t_{reg}(s)$ | Total time $t_{total}(s)$ | Results difference |
|---------------------------|---------------------------|---------------------------|-------------------------------|--------------------------|--------------------|
| Building (128)            | 128                       | 78752                     | 11                            | 77                       | 1.1                |
| Seville (301)             | 298                       | 79800                     | 20                            | 532                      | 3.0                |
| Notre Dame (568)          | 473                       | 150082                    | 45                            | 882                      | 1.8                |
| Trevi Fountain (1259)     | 1259                      | 339365                    | 136                           | 2444                     | 1.1                |
| Street (168)              | 168                       | 47352                     | -                             | -                        | -                  |

Table 3. Comparison with VisualSFM [46] on general data shown in Figure 8. Please see the text for more details.
Table 4. Comparison with [45] on challenging data shown in Figure 6. $N_c$ is the number of reconstructed cameras by our method, and $N_p$ is the number of reconstructed 3D points. $\tilde{x}_0$, $\tilde{x}_{BA}$, and $\bar{x}_{BA}$ denote the median error before BA, the median error after BA, and the average error after BA respectively, where errors are the distances to corresponding cameras computed by an incremental SfM method [39].

| Data and # of input images | Meters | $N_c$ | $N_p$ | $\tilde{x}_0$ | $\tilde{x}_{BA}$ | $\bar{x}_{BA}$ | $\tilde{x}_0$ | $\tilde{x}_{BA}$ | $\bar{x}_{BA}$ |
|---------------------------|--------|-------|-------|--------------|----------------|--------------|--------------|----------------|--------------|
| Alamo                     | 70     | 500   | 85523 | 1.1          | 0.6            | 0.3          | 0.3          | 0.5            | 3e7          |
| Ellis Island              | 180    | 211   | 60922 | 3.7          | 3.1            | 3.0          | 3.0          | 1.8            | 3e7          |
| Metropolis                | 200    | 281   | 33438 | 9.9          | 35.4           | 10.0         | 10.0         | 50.7           | 3e7          |
| Montreal N.D.             | 30     | 426   | 116725| 2.5          | 0.8            | 1.0          | 1.0          | 1.1            | 3e7          |
| NYC Library               | 130    | 288   | 33187 | 2.5          | 1.4            | 0.4          | 0.9          | 1.0            | 6.9          |
| Piazza del Popolo         | 60     | 294   | 48511 | 3.1          | 2.6            | 2.2          | 2.4          | 2e2            | 3.2          |
| Tower of London           | 300    | 393   | 65211 | 11.0         | 4.4            | 1.0          | 1.1          | 40.0           | 6.2          |
| Vienna Cathedral          | 120    | 578   | 171061| 6.6          | 6.5            | 0.4          | 2.6          | 2e4            | 4.0          |
| Yorkminster               | 150    | 341   | 27626 | 3.4          | 3.7            | 0.1          | 3.8          | 5e2            | 14.1         |

Figure 6. Selected reconstruction results on the data released by [45].

Figure 7. The reconstruction result for Quad. The rotations from Bundler [39] are used for this dataset.
Figure 8. Reconstruction results on general data with various difficulties including repetitive scene structures, large camera separation, unbalanced camera distribution, noisy feature correspondences. These examples are referred as (a) Building, (b) Trevi Fountain, (c) Street, (d) Notre Dame, (e) Seville. Our results on these examples demonstrate the robustness of the proposed method.
Appendix

A. Solution for Equation 9. From Equation 9, we have
\[
e_{k+1} = \arg \min_{\beta} \frac{1}{\beta} \|e\|_1 + \frac{1}{2} \|Ax_k - e + \frac{\lambda_k}{\beta}\|_2^2,
\]
\[
= \arg \min_{\epsilon} \epsilon \|e\|_1 + \frac{1}{2} \|e - u\|^2,
\]
where \(\epsilon = \frac{1}{\beta}\), and \(u = Ax_k + \frac{\lambda_k}{\beta}\). According to [27], the solution for Equation 14 is
\[
e_{k+1} = \begin{cases} u - \epsilon, & \text{if } u > \epsilon, \\ u + \epsilon, & \text{if } u < -\epsilon, \\ 0, & \text{otherwise.} \end{cases}
\]

B. Derivation of Equation 12. We linearize the quadratic term \(\frac{\beta}{2} \|Ax - e_{k+1}\|^2\) in Equation 10 at \(x_k\), which gives
\[
x_{k+1} = \arg \min_{x \in \Omega} \left(\beta^T(Ax_k - e_{k+1}), x - x_k\right)^T + \frac{\beta_n}{2} \|x - x_k\|^2 = \arg \min_{x \in \Omega} \frac{\beta_n}{2} \|x - C\|^2,
\]
where \(\Omega := \{x^T x = 1| x \in \mathbb{R}^n\}\), \(C = x_k - \frac{1}{\beta} A^T(Ax_k - e_{k+1}) - \frac{1}{\beta^2} \lambda^k\), and \(\gamma = \sigma_{\max}(A^T A)\) is a proximal parameter. Therefore, we can get Equation 12 directly.

References

[1] S. Agarwal, N. Snavely, and S. Seitz. Fast algorithms for \(l_\infty\) problems in multiview geometry. In Proc. CVPR, pages 1–8, 2008. 2
[2] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski. Building home in a day. In Proc. ICCV, 2009. 2
[3] M. Arie-Nachimson, S. Z. Kovalsky, I. Kemelmacher-Shlizerman, A. Singer, and R. Basri. Global motion estimation from point matches. In Proc. 3DPVT, 2012. 1, 2, 6
[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn., 3(1):1–122, 2011. 2, 4
[5] M. Brand, M. Antone, and S. Teller. Spectral solution of large-scale extrinsic camera calibration as a graph embedding problem. In Proc. ECCV, 2004. 1, 2
[6] E. J. Candes and T. Tao. Decoding by linear programming. Information Theory, IEEE Transactions on, 51(12):4203–4215, 2005. 2
[7] A. Chatterjee and V. Govindu. Efficient and robust large-scale rotation averaging. In Proc. ICCV, pages 521–528, 2013. 2, 3, 5
[8] J. Courchay, A. Dalalyan, R. Keriven, and P. Sturm. Exploiting loops in the graph of trifocal tensors for calibrating a network of cameras. In Proc. ECV, pages 85–99, 2010. 2
[9] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In Proc. CVPR, pages 3001–3008, 2011. 1, 2, 7
[10] A. Dalalyan and R. Keriven. L1-penalized robust estimation for a class of inverse problems arising in multiview geometry. In NIPS, 2009. 2
[11] A. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. Proc. ECCV, pages 311–326, 1998. 1, 2
[12] T. Goldstein and S. Osher. The split bregman method for l1-regularized problems. SIAM Journal on Imaging Sciences, 2(2):323–343, 2009. 2
[13] V. M. Govindu. Combining two-view constraints for motion estimation. In Proc. CVPR, pages 218–225, 2001. 1, 2
[14] V. M. Govindu. Lie-algebraic averaging for globally consistent motion estimation. In Proc. CVPR, 2004. 2
[15] V. M. Govindu. Robustness in motion averaging. In Proc. ACCV, 2006. 2
[16] S. Haner and A. Heyden. Covariance propagation and next best view planning for 3d reconstruction. In Proc. ECCV, pages 545–556, 2012. 2
[17] R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. IJCV, pages 1–39, 2013. 1, 2, 5
[18] R. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, 2003. 3
[19] M. Havlena, A. Torii, J. Knopp, and T. Pajdla. Randomized structure from motion based on atomic 3d models from camera triplets. In Proc. CVPR, pages 2874–2881, 2009. 2
[20] N. Jiang, Z. Cui, and P. Tan. A global linear method for camera pose registration. In Proc. ICCV, 2013. 1, 2, 3, 4, 6, 7
[21] N. Jiang, P. Tan, and L. Cheung. Seeing double without confusion: Structure-from-motion in highly ambiguous scenes. In Proc. CVPR, pages 1458–1465, 2012. 3
[22] F. Kahl and R. Hartley. Multiple view geometry under the \(l_\infty\)-norm. IEEE Trans. PAMI, 30:1603–1617, 2007. 2
[23] R. Lehoucq and J. Scott. An evaluation of software for computing eigenvalues of sparse nonsymmetric matrices. Preprint MCS-P547, 1195, 1996. 5
[24] M. Lhuillier and L. Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. IEEE Trans. PAMI, 27(3):418–433, 2005. 1, 2
[25] H. Li. Efficient reduction of l-infinity geometry problems. In Proc. CVPR, pages 2695–2702, 2009. 2
[26] H. Li and R. Hartley. Five-point motion estimation made easy. In Proc. ICPR, pages 630–633, 2006. 3
[27] Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. arXiv preprint arXiv:1009.5055, 2010. 4, 10
[28] Z. Lin, R. Liu, and Z. Su. Linearized alternating direction method with adaptive penalty for low-rank representation. In Advances in neural information processing systems, pages 612–620, 2011. 4
[29] D. Martinec and T. Pajdla. Robust rotation and translation estimation in multiview reconstruction. In Proc. CVPR, pages 1–8, 2007. 1, 2
[30] P. Moulin, P. Monasse, and R. Marlet. Global fusion of relative motions for robust, accurate and scalable structure from motion. In Proc. ICCV, 2013. 1, 2, 4, 6, 7
[31] D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE Trans. PAMI*, 26:756–777, 2004. 1, 3, 5
[32] C. Olsson, A. Eriksson, and R. Hartley. Outlier removal using duality. In *Proc. CVPR*, pages 1450–1457, 2010. 2
[33] C. Olsson, A. Eriksson, and F. Kahl. Efficient optimization for l_∞ problems using pseudoconvexity. In *Proc. ICCV*, 2007. 2
[34] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2013. 2
[35] M. Pollefeys, D. Nistér, J.-M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S.-J. Kim, P. Merrell, et al. Detailed real-time urban 3d reconstruction from video. *IJCV*, 78(2-3):143–167, 2008. 2
[36] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera. *IJCV*, 59:207–232, 2004. 2
[37] K. Sim and R. Hartley. Recovering camera motion using l∞ minimization. In *Proc. CVPR*, pages 1230–1237, 2006. 1, 2, 4
[38] S. Sinha, D. Steedly, and R. Szeliski. A multi-stage linear approach to structure from motion. In *ECCV Workshop on Reconstruction and Modeling of Large-Scale 3D Virtual Environments*, 2010. 2
[39] N. Snavely, S. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. *ACM Trans. on Graph.*, 25:835–846, 2006. 1, 2, 7, 8
[40] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *IJCV*, 80(2):189–210, 2008. 2
[41] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Proc. CVPR*, 2008. 6
[42] P. Torr and A. Zisserman. Robust parameterization and computation of the trifocal tensor. *Image and Vision Computing*, 15:591–605, 1997. 1
[43] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. *Lecture Notes in Computer Science*, pages 298–375, 2000. 1
[44] K. Wilson and N. Snavely. Network principles for sfm: Disambiguating repeated structures with local context. In *Proc. ICCV*, 2013. 2, 7
[45] K. Wilson and N. Snavely. Robust global translations with ldsfm. In *Proc. ECCV (3)*, pages 61–75, 2014. 3, 6, 7, 8
[46] C. Wu. Towards linear-time incremental structure from motion. In *Proc. 3DV*, 2013. 6, 7
[47] C. Zach, A. Irschara, and H. Bischof. What can missing correspondences tell us about 3d structure and motion? In *Proc. CVPR*, 2008. 2
[48] C. Zach, M. Klopschitz, and M. Pollefeys. Disambiguating visual relations using loop constraints. In *Proc. CVPR*, 2010. 2, 3, 5, 7
[49] C. Zach and M. Pollefeys. Practical methods for convex multi-view reconstruction. In *Proc. ECCV; Part IV*, pages 354–367, 2010. 2