Very Large-Scale Global SfM by Distributed Motion Averaging

Siyu Zhu*[1], Runze Zhang *[1], Lei Zhou[1], Tianwei Shen[1], Tian Fang†[2], Ping Tan[3], and Long Quan[1]

[1]The Hong Kong University of Science and Technology [2]Altizure.com [3]Simon Fraser University

Abstract

Global Structure-from-Motion (SfM) techniques have demonstrated superior efficiency and accuracy than the conventional incremental approach in many recent studies. This work proposes a divide-and-conquer framework to solve very large global SfM at the scale of millions of images. Specifically, we first divide all images into multiple partitions that preserve strong data association for well-posed and parallel local motion averaging. Then, we solve a global motion averaging that determines cameras at partition boundaries and a similarity transformation per partition to register all cameras in a single coordinate frame. Finally, local and global motion averaging are iterated until convergence. Since local camera poses are fixed during the global motion average, we can avoid caching the whole reconstruction in memory at once. This distributed framework significantly enhances the efficiency and robustness of large-scale motion averaging.

1. Introduction

Structure-from-Motion (SfM) has been intensively investigated in computer vision. Earlier methods are mostly incremental, where images are reconstructed one by one [1, 34, 35, 37, 42, 50] . Recent studies [3, 5, 6, 7, 17, 18, 20, 27, 33] suggest that a global approach, reconstructing all images together, leads to better accuracy and efficiency. However, global SfM has so far only been demonstrated with relatively small-scale data-sets at the order of a few thousand images [9, 11, 21, 33, 49]. This work is the first to push global SfM to the scale of millions of input images, larger than all previous works.

The key to the global SfM methods is motion averaging. The time and spatial complexity of state-of-the-art motion averaging methods [3, 5, 6, 7, 17, 18, 20, 27, 33] following the gradient and Hessian-based optimization approach is cubic and square respectively in the number of input images [2]. Therefore, global motion averaging quickly reaches the memory and efficiency bottleneck as the number of input images drastically increases.

*Siyu Zhu and Runze Zhang equally contribute to the work as the first authors. Siyu Zhu is with Alibaba A.I. Labs since 2017.

†Tian Fang is with Shenzhen Zhuke Innovation Technology since 2017.

To conquer the problems above, we propose a distributed and robust motion averaging method which is inspired by the nested dissection algorithm [25]. We formulate the large-scale motion averaging problem on a camera graph, where each camera is a vertex and cameras with relative motion constraints are linked by edges. By dividing the original camera graph into multiple small-scale sub-graphs, we group the variables of each sub-graph and order them first in the Hessian matrix, while variables called separators [25] which connect multiple sub-graphs are ordered second. Since sub-graphs excluding separators are independent with each other, we can first solve the Cholesky factorization of each sub-graph excluding separators in a distributed manner and then the factorization of separators. To further reduce the communication overhead among sub-graphs, we iterate each sub-graph till convergence before solving the separators. In this paper, we call the optimization process of each sub-graph as *local motion averaging* and that of separators as *global motion averaging*. We also introduce a similarity transformation to parameterize the camera poses of each sub-graph, so that the linearization stays the same in global motion averaging. Since only the separators, namely the cameras connecting multiple sub-graphs, and the similarity transformations are considered in global motion averaging, the entire reconstruction is avoided to be cached in core memory at once.

Dividing the camera-graph into strongly associated sub-graphs also improves the robustness of global SfM. Previous motion averaging methods often assume uniform accuracy of relative poses, which degrades reconstruction accuracy when there are both strong and weak associations among cameras. Our framework clusters strong affiliated cameras together and fixes their relative motions in the global motion averaging, which can be applied to many previous motion averaging algorithms, like [11, 28, 33, 45, 48], to further improve their performance.

In experiments, we demonstrate our framework on sequential, Internet, and challenging city-scale data-sets. Remarkably, we are even able to average the camera poses of a city-scale data-set with more than one million high-resolution images in parallel. We further apply our framework to enhance the previous prestigious motion averaging works [33, 45, 48] on both efficiency and robustness.

2. Related Works

Thanks to the massive image data, city-scale 3D reconstruction [1, 16, 15, 39, 38, 47, 51, 53, 54, 55, 56, 57, 58] has been a hot research topic in computer vision, in which Structure-from-Motion (SfM) is the pivotal point. The incremental SfM methods [1, 34, 35, 37, 43, 50] progressively recover the camera pose of the next-best-view [13, 19]. However, the redundant intermediate bundle adjustment leads to low efficiency and drifting errors. In comparison, the global SfM methods compute all the camera poses simultaneously from the available epipolar geometry [3, 5, 6, 7, 17, 18, 20, 27, 33] or trifocal tensor [8, 21, 28, 41]. Such global methods are highly efficient and can compensate for severe drifting errors. A hybrid formulation [4, 44, 58] exploiting incremental SfM to initialize partitions of accurate and robust local camera poses and motion averaging to globally merge them together is naturally presented.

Most of global SfM methods solve camera rotations and translations separately. Govindu *et al.* [17] simultaneously compute all the camera rotations while this work is limited by the fact that the rotation manifold has a non-trivial topology [20]. Martinec *et al.* [27] ignore the manifold constraint and obtain a linear algorithm. On the other hand, the translation averaging method is divided into two types. The essential matrix based methods [3, 5, 17, 33] suffers from the fact that essential matrices can only determine the directions of relative translations and is limited to a parallel rigid graph [33]. The trifocal tensor based methods [8, 21, 28, 40] require strong association among images. To address this issue, some other methods optimize camera poses together with scene points [9, 23, 27, 36, 41, 49].

However, a standard motion averaging problem that considers all the relative poses at once gradually becomes both memory and time consuming as the number of relative motions rises sharply. This problem becomes more obvious in translation averaging [10, 11, 21, 28, 49] that considers the relative translations between cameras and 3D points as well. To address the large scale motion averaging problem in a distributed manner, we propose a divide-and-conquer framework similar to nested dissection [25]. Some works [14, 29, 54] also try to solve the large scale bundle adjustment problem in an out-of-core or distributed manner.

To guarantee robust motion averaging, the work in [18] adopts a $\ell 1$ optimization [7] and follows the Lie-algebra rotation representation to achieve better rotation averaging results. The works in [11, 33] also apply a $\ell 1$ solver to translation averaging. Meanwhile, other works focus on the filters [11, 22, 49, 52] to effectively discard erroneous epipolar geometry and feature correspondences and provide well-posed initialization for motion averaging. In contrast, we first solve the well-conditioned sub-problems of strongly affiliated images and then globally merge them together to obtain a robust system.

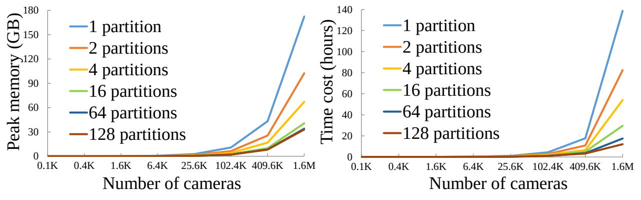

Figure 1: The comparisons of the estimated peak memory and time cost of motion averaging given different numbers of partitions. The line chart is estimated on the city-scale data-set. If all the cameras are in one partition, our approach degenerates to the traditional motion averaging.

3. Motion Averaging Review

We first give the notations and definitions of this paper. We specify the absolute pose of a camera C_i as $\mathbf{P}_i = [\mathbf{R}_i | \mathbf{t}_i]$, where $\mathbf{R}_i \in \mathbb{R}^{3\times 3}$ is a 3D rotation matrix denoting the camera orientation, $\mathbf{t}_i = -\mathbf{R}_i \mathbf{c}_i$ specifies a 3D translation vector, and $\mathbf{c}_i \in \mathbb{R}^{3\times 1}$ is the position of the camera optical center. For each pair of cameras C_i and C_j, there are two relative motion constraints:

$$\mathbf{R}_{ij} = \mathbf{R}_j \mathbf{R}_i^\top, \tag{1}$$

$$\lambda_{ij} \hat{\mathbf{t}}_{ij} = \mathbf{R}_j (\mathbf{c}_i - \mathbf{c}_j), \tag{2}$$

where $\mathbf{R}_{ij} \in \mathbb{R}^{3\times 3}$ denotes a relative rotation matrix, $\hat{\mathbf{t}}_{ij} \in \mathbb{R}^{3\times 1}$ represents a unit vector of the relative translation direction, and λ_{ij} is a scale factor. We also denote the sparse 3D points as $\mathcal{X} = \{\mathbf{x}_i\}$, where $\mathbf{x}_i \in \mathbb{R}^{3\times 1}$ is the 3D position of a scene point.

Given a reference frame and a set of relative rotations $\mathcal{R}_{\text{rel}} = \{\mathbf{R}_{ij}\}$, a global rotation averaging algorithm obtains the camera rotations $\mathcal{R} = \{\mathbf{R}_i\}$ by solving the following minimization problem:

$$\arg\min_{\mathcal{R}} \sum_{\mathbf{R}_{ij} \in \mathcal{R}_{\text{rel}}} d^{\mathbf{R}}(\mathbf{R}_{ij}, \mathbf{R}_j \mathbf{R}_i^\top)^p, \tag{3}$$

where the variable $p = 1, 2$ chooses $\ell 1$ or $\ell 2$ norm and the distance measure $d^{\mathbf{R}}(\mathbf{S}, \mathbf{R})$ is defined on $SO(3)$, e.g. angular distance, chordal distance, quaternion distance *etc.* [20].

Given the fixed global orientations $\mathcal{R} = \{\mathbf{R}_i\}$ and some known camera-to-camera relative translations $\mathcal{T}_{\text{rel}} = \{\mathbf{t}_{ij}\}$ and camera-to-point relative translations $\mathcal{U}_{\text{rel}} = \{\mathbf{u}_{ij}\}$, a translation averaging problem computes global camera positions $\mathcal{T} = \{\mathbf{c}_i\}$ by minimizing the following function,

$$\begin{aligned} \arg\min_{\mathcal{T}} & \sum_{\mathbf{t}_{ij} \in \mathcal{T}_{\text{rel}}} d^{\mathbf{T}}(\mathbf{t}_{ij}, \mathbf{R}_j(\mathbf{c}_i - \mathbf{c}_j))^p \\ & + \sum_{\mathbf{u}_{ij} \in \mathcal{U}_{\text{rel}}} d^{\mathbf{T}}(\mathbf{u}_{ij}, \mathbf{R}_i^\top(\mathbf{x}_j - \mathbf{c}_i))^p, \end{aligned} \tag{4}$$

where the dissimilarity measure $d^{\mathbf{T}}(\mathbf{u}, \mathbf{v})$ can be a Euclidean distance, angular distance, chordal distance

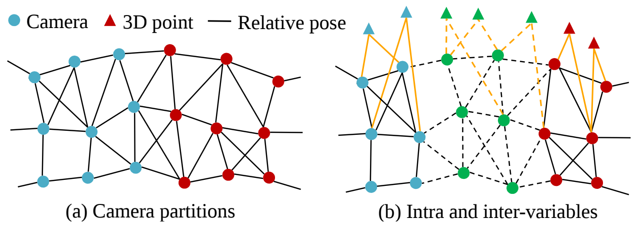

Figure 2: An illustration of intra and inter variables (cameras, relative poses, and 3D points), where (b) shows the intra and inter variables corresponding to the camera partition in (a). The blue and red dots in (b) represent *intra-cameras*, and the green dots are *inter-cameras*. The blue and red triangles represent *intra-3D-points*, and the green triangles are *inter-3D-points*. The solid and dashed lines represent *intra-relative-poses* and *inter-relative-poses* respectively.

etc. [49], and different norms $p = 1, 2$ can be considered. The relative translations between cameras and 3D points are also introduced to avoid generating disconnected models in less photographed scenes [10, 49].

The above two minimization problems are often solved by gradient and Hessian-based optimization methods, which have a computational complexity of $O((m + n)^3)$ for each iteration and a memory requirement of $O(mn(m+n))$ [2], where m is the number of cameras and n is the number of scene points in translation averaging. The computational complexity and memory requirement gradually become the bottleneck for very large-scale motion averaging, especially when solving problems that involve millions of images. Figure 1 demonstrates the growth of peak memory and time cost of motion averaging along with the number of input cameras.

4. Distributed Motion Averaging

4.1. Problem Formulation

Our goal is to compute the global poses $\mathcal{P} = \{\mathbf{P}_i\}$ of a great number of cameras $\mathcal{C} = \{C_i\}$ from the relative rotations $\mathcal{R}_{\text{rel}} = \{\mathbf{R}_{ij}\}$ and translations $\mathcal{T}_{\text{rel}} = \{\mathbf{t}_{ij}\}$ in a distributed manner. We assume that most of the erroneous epipolar geometry and feature correspondences have been discarded by the epipolar filters [11, 22, 49, 52].

Some terminologies are necessary to facilitate our following discussion, which are better explained by referring to Figure 2. In this figure, we define a camera graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where each vertex $V_i \in \mathcal{V}$ is a camera C_i, an edge $E_{ij} \in \mathcal{E}$ will link two cameras C_i and C_j if the relative motion between them is known. Figure 2 (a) shows a camera partition, where cameras in the same partitions have the same color. If a camera is only linked to the cameras in the same partition, we name it as *intra-camera*. The set of all these cameras is $\mathcal{C}_{\text{intra}}$. The edges among cameras in $\mathcal{C}_{\text{intra}}$ are referred as *intra-relative-poses*. If a camera is linked to those in other partitions, we name it *inter-cameras*, which are the *separators* in the nest dissection [25]. They form a set $\mathcal{C}_{\text{inter}}$. The edges involving a camera in $\mathcal{C}_{\text{inter}}$ are referred as *inter-relative-poses*. Moreover, the 3D points visible only by intra-cameras are defined as *intra-3D-points* denoted by $\mathcal{X}_{\text{intra}}$, and the others as *inter-3D-points* termed as $\mathcal{X}_{\text{inter}}$. Such a categorization of cameras and points is illustrated in Figure 2 (b).

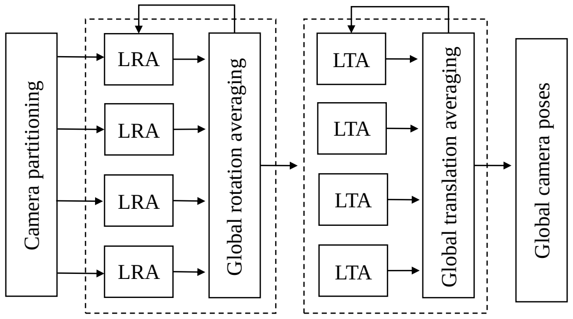

Figure 3: The system architecture of our distributed motion averaging. We abbreviate local rotation averaging to "LRA" and local translation averaging to "LTA".

After camera partitioning, each partition is reconstructed by a local motion averaging, whose complexity can be easily controlled by the number of partitions. A compact parameterization is necessary to make the following global motion averaging manageable. For a camera partition $\mathcal{C}^k = \{C_i^k\}$, the *intra*-cameras and 3D points within it are parametrized by a similarity transformation:

$$\mathbf{S}^k = [\alpha^k \mathbf{Q}^k \,|\, \mathbf{l}^k], \tag{5}$$

where α^k is the scale factor, $\mathbf{Q}^k$ is the rotation matrix, and $\mathbf{l}^k$ is the translation vector. In this way, the global optimization is simplified to consider only *inter-cameras* and the similarity transformations of different partitions, which significantly reduces the number of involved parameters.

Now we can have the following notations. The rotation and position of an intra-camera $C_i^k \in \mathcal{C}_{\text{intra}}^k$ is denoted by $\mathbf{R}_i^k$ and $\mathbf{c}_i^k$ respectively within a local coordinate frame. This local coordinate frame is registered to the global coordinate frame by a similarity transformation $\mathbf{S}^k = [\alpha^k \mathbf{Q}^k \,|\, \mathbf{l}^k]$. Therefore, the rotation and position of C_i^k in the global coordinate frame are $\mathbf{R}_i = \mathbf{R}_i^k {\mathbf{Q}^k}^T$ and $\mathbf{c}_i = \alpha^k \mathbf{Q}^k \mathbf{c}_i^k + \mathbf{l}^k$ respectively. The rotation and position of an inter-camera $C_j \in \mathcal{C}_{\text{inter}}$ are denoted as $\mathbf{R}_j$ and $\mathbf{c}_j$ with respect to the global coordinate frame.

4.2. System Architecture

Figure 3 shows our system architecture. We first divide the input images into some partitions based on the association among them (Section 4.3). Then the motion averaging will be completed by two steps, namely rotation averaging (Section 4.4) and translation averaging (Section 4.5).

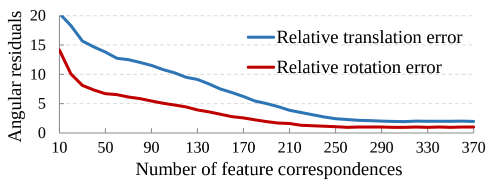

Figure 4: The accuracy of relative rotations and translations measured by the median angular distance in degrees for different numbers of feature correspondences of camera pairs. The statistics are based on the Internet data-set [49].

In each step, our system iterates between the distributed local motion averaging and the global motion averaging until the convergence criterion is reached. In the first iteration of local motion averaging, we follow the traditional motion averaging pipeline revisited in Section 3 to reconstruct camera rotations or translations of each partition. From the second iteration, we optimize the intra-camera poses while fixing all the inter-camera poses and similarity transformations. In the global motion averaging, the inter-camera poses and similarity transformations associated with each partition are optimized by the inter-relative-poses with all the intra-camera poses fixed in their local coordinate frames.

4.3. Camera Partitioning

We start with the camera graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ and recursively apply normalized-cut [12] to partition the camera graph into two sub-graphs until the local motion averaging corresponding to each sub-graph can be solved by a single computer. Normalized-cut [12] also encourages a balanced partition for a high degree of parallelism. Next, we define the edge weight $w(e_{ij})$ between two cameras C_i and C_j as the number of their inlier feature correspondences. As shown in Figure 4, camera pairs with strong association indicate robust relative poses. In this way, camera pairs with more accurate relative motions tend to be grouped together. Finally, we have a set of camera partitions $\{\mathcal{C}^k \,|\, \mathcal{C}^k = \{C_i^k\}\}$. A sample partition is shown in Figure 5 (a).

4.4. Rotation Averaging

Local rotation averaging For every partition of cameras $\mathcal{C}^k = \{C_i^k\}$, we fix the rotations of similarity transformations $\mathcal{Q} = \{\mathbf{Q}^k\}$ and the rotations of inter cameras $\mathcal{R}_{\text{inter}} = \{\mathbf{R}_j\}$, and obtain associate rotations of intra-cameras $\mathcal{R}_{\text{intra}} = \{\mathcal{R}^k_{\text{intra}} \,|\, \mathcal{R}^k_{\text{intra}} = \{\mathbf{R}_i^k\}\}$ with respect to their local coordinate frames by the minimization problem:

$$\forall \mathcal{C}^k : \; L^{\mathbf{R}}(\mathcal{R}^k) = L^{\mathbf{R}}_{\text{intra}}(\mathcal{R}^k) + L^{\mathbf{R}}_{\text{betw}}(\mathcal{R}^k). \tag{6}$$

First, $L^{\mathbf{R}}_{\text{intra}}(\mathcal{R}^k)$ is the local rotational error function of intra-cameras, which is defined as:

$$L^{\mathbf{R}}_{\text{intra}}(\mathcal{R}^k) = \sum_{\mathbf{R}_i^k, \mathbf{R}_j^k \in \mathcal{R}^k_{\text{intra}}} d^{\mathbf{R}}(\mathbf{R}_{ij}, \mathbf{R}_j^k {\mathbf{R}_i^k}^T)^p. \tag{7}$$

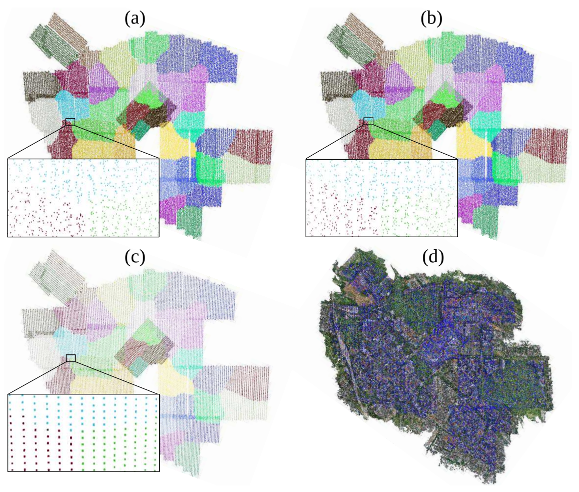

Figure 5: The intermediate results of the City-B data-set. (a) The partitions of cameras. (b) The camera poses after the first global motion averaging. (c) The camera poses after the second local motion averaging. (d) The final camera poses and sparse 3D points after bundle adjustment.

Second, $L^{\mathbf{R}}_{\text{betw}}(\mathcal{R}^k)$ is the local rotational error function between intra-cameras and inter-cameras. Given that $\mathbf{R}_i = \mathbf{R}_i^k {\mathbf{Q}^k}^T$, we can define $L^{\mathbf{R}}_{\text{betw}}(\mathcal{R}^k)$ as:

$$L^{\mathbf{R}}_{\text{betw}}(\mathcal{R}^k) = \sum_{\substack{\mathbf{R}_i^k \in \mathcal{R}^k_{\text{intra}} \\ \mathbf{R}_j \in \mathcal{R}_{\text{inter}}}} d^{\mathbf{R}}(\mathbf{R}_{ij}, \mathbf{R}_j (\mathbf{R}_i^k {\mathbf{Q}^k}^T)^T)^p. \tag{8}$$

Global rotation averaging With the rotations of intra-cameras $\mathcal{R}_{\text{intra}} = \{\mathcal{R}^k_{\text{intra}} \,|\, \mathcal{R}^k_{\text{intra}} = \{\mathbf{R}_i^k\}\}$ fixed, we define the global rotational error $G^{\mathbf{R}}(\mathcal{S}, \mathcal{R})$ as:

$$G^{\mathbf{R}}(\mathcal{S}, \mathcal{R}) = G^{\mathbf{R}}_{\text{betw}}(\mathcal{S}, \mathcal{R}) + G^{\mathbf{R}}_{\text{inter}}(\mathcal{R}). \tag{9}$$

First, $G^{\mathbf{R}}_{\text{betw}}(\mathcal{S}, \mathcal{R})$ is the rotational error function between intra-cameras and inter-cameras. Since we have $\mathbf{R}_i = \mathbf{R}_i^k {\mathbf{Q}^k}^T$, $G^{\mathbf{R}}_{\text{betw}}(\mathcal{S}, \mathcal{R})$ is defined by:

$$G^{\mathbf{R}}_{\text{betw}}(\mathcal{S}, \mathcal{R}) = \sum_k \sum_{\substack{\mathbf{R}_i^k \in \mathbf{R}^k_{\text{intra}} \\ \mathbf{R}_j \in \mathbf{R}_{\text{inter}}}} d(\mathbf{R}_{ij}, \mathbf{R}_j (\mathbf{R}_i^k {\mathbf{Q}^k}^T)^T)^p. \tag{10}$$

Moreover, $G^{\mathbf{R}}_{\text{inter}}(\mathcal{R})$ is the global rotational error function of inter-cameras and it is defined as:

$$G^{\mathbf{R}}_{\text{inter}}(\mathcal{R}) = \sum_{\mathbf{R}_i, \mathbf{R}_j \in \mathcal{R}_{\text{inter}}} d(\mathbf{R}_{ij}, \mathbf{R}_j \mathbf{R}_i^T)^p. \tag{11}$$

We set the rotation of any one camera $\mathbf{R}_i = \mathbf{I}_{3\times 3}$ to fix the gauge freedom.

4.5. Translation Averaging

After obtaining the global rotations, we continue to average camera translations while regarding all the rotations as known parameters.

Local translation averaging With inter-camera positions $\mathcal{T}_{\text{inter}} = \{\mathbf{c}_j\}$ and similarity transformations $\mathcal{S} = \{\mathbf{S}^k \mid \mathbf{S}^k = [\alpha^k \mathbf{Q}^k \mid \mathbf{l}^k]\}$ fixed, we can obtain the positions of intra-cameras $\mathcal{T}_{\text{intra}} = \{\mathcal{T}^k_{\text{intra}} \mid \mathcal{T}^k_{\text{intra}} = \{\mathbf{c}^k_i\}\}$ with respect to their local coordinate frames by minimizing the following error function considering both camera-to-camera and camera-to-point relative translations in terms of camera partitions $\{\mathcal{C}^k \mid \mathcal{C}^k = \{C^k_i\}\}$. That is

$$\forall \mathcal{C}^k : \; L^{\mathbf{T}}(\mathcal{T}^k) = L^{\mathbf{T}}_{\text{intra}}(\mathcal{T}^k) + L^{\mathbf{T}}_{\text{betw}}(\mathcal{T}^k). \tag{12}$$

Specifically, the local positional error function of intra-cameras denoted by $L^{\mathbf{T}}_{\text{intra}}(\mathcal{T}^k)$ is defined as:

$$L^{\mathbf{T}}_{\text{intra}}(\mathcal{T}^k) = \sum_{\mathbf{c}^k_i, \mathbf{c}^k_j \in \mathcal{T}^k_{\text{intra}}} d^{\mathbf{T}}(\mathbf{t}_{ij}, \mathbf{R}^k_j(\mathbf{c}^k_i - \mathbf{c}^k_j))^p. \tag{13}$$

Since we have $\mathbf{c}_i = \alpha^k \mathbf{Q}^k \mathbf{c}^k_i + \mathbf{l}^k$, the local positional error function between intra-cameras and inter-cameras denoted by $L^{\mathbf{T}}_{\text{betw}}(\mathcal{T}^k)$ is defined as:

$$L^{\mathbf{T}}_{\text{betw}}(\mathcal{T}^k) = \sum_{\substack{\mathbf{c}^k_i \in \mathcal{T}^k_{\text{intra}} \\ \mathbf{c}_j \in \mathcal{T}_{\text{inter}}}} d^{\mathbf{T}}(\mathbf{t}_{ij}, \mathbf{R}_j((\alpha^k \mathbf{Q}^k \mathbf{c}^k_i + \mathbf{l}^k) - \mathbf{c}_j))^p \tag{14}$$

Here, we omit the camera-to-point relative translation constraints, which take a similar form as the camera-to-camera constraints in Equation 13 and 14.

Global translation averaging Fixing the intra-camera positions $\mathcal{T}_{\text{intra}} = \{\mathcal{T}^k_{\text{intra}} \mid \mathcal{T}^k_{\text{intra}} = \{\mathbf{c}^k_i\}\}$, we define the global positional error function $G^{\mathbf{T}}(\mathcal{S}, \mathcal{T})$ as:

$$G^{\mathbf{T}}(\mathcal{S}, \mathcal{T}) = G^{\mathbf{T}}_{\text{betw}}(\mathcal{S}, \mathcal{T}) + G^{\mathbf{T}}_{\text{inter}}(\mathcal{T}). \tag{15}$$

Here, $G^{\mathbf{T}}_{\text{betw}}(\mathcal{S}, \mathcal{T})$ is the global positional error function between intra-cameras and inter-cameras. Given that $\mathbf{c}_i = \alpha^k \mathbf{Q}^k \mathbf{c}^k_i + \mathbf{l}^k$, the global positional error is defined as:

$$G^{\mathbf{T}}_{\text{betw}}(\mathcal{S}, \mathcal{T}) = \sum_k \sum_{\substack{\mathbf{c}^k_i \in \mathcal{T}^k_{\text{intra}} \\ \mathbf{c}_j \in \mathcal{T}_{\text{inter}}}} d^{\mathbf{T}}(\mathbf{t}_{ij}, \mathbf{R}_j((\alpha^k \mathbf{Q}^k \mathbf{c}^k_i + \mathbf{l}^k) - \mathbf{c}_j))^p. \tag{16}$$

Finally, we define the global positional error of inter-cameras as:

$$G^{\mathbf{T}}_{\text{inter}}(\mathcal{T}) = \sum_{\mathbf{c}_i, \mathbf{c}_j \in \mathcal{T}_{\text{inter}}} d^{\mathbf{T}}(\mathbf{t}_{ij}, \mathbf{R}_j(\mathbf{c}_i - \mathbf{c}_j))^p. \tag{17}$$

Likewise, we omit the camera-to-point relative translation constraints in Equation 16 and 17. We fix the gauge freedom by setting the position of any one camera $\mathbf{c}_i = \mathbf{0}_{3\times 1}$ and the scale of any one similarity transformation $\alpha^k = 1$.

4.6. Implementation Details

This section briefly expounds on the implementation of our large-scale motion averaging system. It needs to be emphasized that the proposed distributed and robust motion averaging framework is also applicable to the state-of-the-art motion averaging approaches [3, 5, 6, 7, 17, 18, 20, 27, 33] with different choices of distance measures and norms.

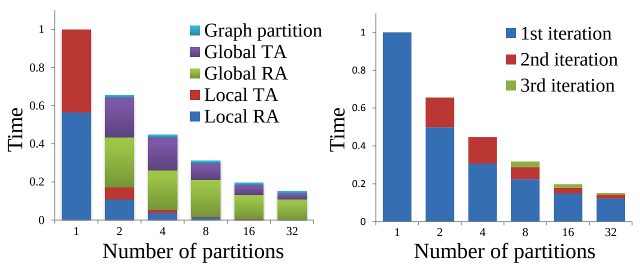

Figure 6: The comparison of the motion averaging running time between the traditional and our distributed approaches with different numbers of partitions. The time of one partition corresponds to that of the traditional method. The running time is calculated on the city-scale data-sets on the average of 10 runs using our distributed computing system.

Camera partitioning Figure 1 shows that as the number of camera partitions increases, both the memory and time consumption first remarkably decrease and then stabilize. That means over-partitioning does not bring additional benefits. Therefore, the data-sets handled by a single computer are partitioned according to the computer core number. In our experiments, we limit the number of cameras within a partition ($\leq$ 3000) so that each partition can be fit into a single computer for the subsequent operations.

Relative motions In each partition, we follow the work in [4, 58] and utilize robust local incremental SfM to obtain a partial reconstruction and associate relative poses. The introduction of local incremental SfM is helpful in two aspects. First, incremental SfM employs RANSAC based filters [24, 30] and repeated intermediate bundle adjustment [46] to discards most of the erroneous epipolar geometry and feature correspondences. More importantly, the relative translation between cameras C^k_i and C^k_j with the baseline length can be obtained, and we denote it as $\mathbf{t}^k_{ij}$. Since $\mathbf{t}^k_{ij}$ is also up to the similarity transformation $\mathbf{S}^k = [\alpha^k \mathbf{Q}^k \mid \mathbf{l}^k]$ in the global coordinate frame, we have $\mathbf{t}_{ij} = \alpha^k \mathbf{t}^k_{ij}$. Therefore, we can directly use the Euclidean distance between two relative translations as the dissimilarity measure of translation averaging. To introduce the constraints of camera-to-point relative translations into translation averaging, we choose a subset of scene points by greedily selecting 3D points visible by the most number of cameras until each camera sees at least 20 points [49].

Distance measures Given that most of the erroneous relative motions have been discarded by local incremental SfM, we directly refer to the robust non-linear loss function for a fast and robust convergence. First, we use the angular distance, the most natural metric on $SO(3)$ [20], to measure the distance between two rotation matrices $\mathbf{S}$ and $\mathbf{R}$, namely $d^{\mathbf{R}}(\mathbf{R}, \mathbf{S}) = ||\log(\mathbf{R}\mathbf{S}^T)||_2$. Moreover, we solve the translation averaging by minimizing Euclidean distances between relative translations with baseline lengths, which is

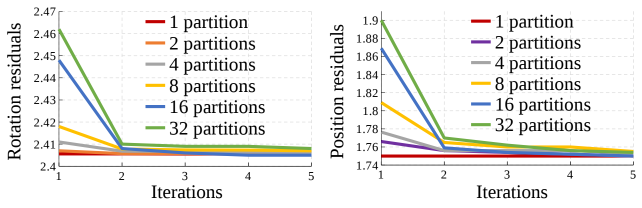

Figure 7: The convergence rate of the rotational and positional errors provided different numbers of partitions. The line chart is obtained on the city-scale data-sets [49] on the average of 10 runs.

a much well-posed similarity averaging according to [11]. That measures the distance between two vectors $\mathbf{u}, \mathbf{v}$ as $d^{\mathbf{T}}(\mathbf{u}, \mathbf{v}) = L_\delta(\mathbf{u} - \mathbf{v})$ and $L_\delta(x) = \delta(\sqrt{1 + (\frac{x}{\delta})^2} - 1)$ is the Pseudo-Huber loss function with slope δ ($\delta = 0.1$). The slope is set according to the real GPS scale and we fix the scale of one relative translation of two cameras which have real GPS positions from the EXIF tag.

Non-linear optimization To solve all the non-linear least square objective functions defined in Equation (6), (9), (12) and (15), we use the standard Levenberg-Marquardt algorithm [32]. The efficient package [2] is adopted as the solver. To initialize the similarity transformations, we follow the work in [4] that uses sampled epipolar geometry across partitions to roughly merge partial sparse reconstructions together.

Since the local motion averaging reaches convergence before global motion averaging, we have a very small number of iterations of local and global motion averaging before convergence. More importantly, the communications are remarkably reduced, and local motion averaging becomes inherently highly parallel. We use a standard stopping criteria adopted in non-linear optimization. In both rotation averaging and translation averaging, an iteration stops when the root mean square error of their corresponding distance measurement decreases by less than 1%. The histogram in Figure 6 shows that our approach with 32 partitions takes only 15% of the time of the traditional approach.

As shown in Figure 7, we regard the residual of the traditional motion averaging (i.e. 1 partition) as the baseline. We can see that our algorithm converges in two global iterations with 2 and 4 partitions and three global iterations with 8, 16 and 32 partitions. After the first iteration, the rotational and positional residuals drop to approximately $0.012\% - 2.2\%$ and $0.90\% - 8.6\%$ above the minimum residuals respectively. After the second iteration, the rotational and positional residuals are about $0.012\% - 0.57\%$ and $0.34\% - 0.57\%$ above the minimum residuals.

5. Experiments

The experiments of the city-scale data-sets are run on a distributed computing system consisted of 12 computers with 12 core 3.6 GHz processors and 64 GB RAM. The Internet data-set [49] and the sequential data-set [22] are run on a single computer with the same configuration. We also use a computer with 40 core 3.6 GHz processors and 512 GB RAM to test the traditional methods on the city-scale data-sets. As for the implementation of our SfM pipeline, we use SIFT [26] to detect scale-invariant features, the method in [31] to retrieve candidate image pairs for putative feature matching, and distributed bundle adjustment [54] for the final non-linear optimization of both camera poses and 3D points.

Internet data-sets Table 1 demonstrates the statistical comparison between the global SfM methods [33, 45, 49] and our implementation with and without the proposed distributed formulation on the Internet data-set [49]. After applying our framework to the global SfM methods [33, 45, 49], they obviously reconstruct more cameras. We can verify that the divide-and-conquer optimization, where the well-posed subproblems are tackled first and subsequently merged together, encourages a robust convergence indeed, especially for the data-set consisted of images captured in a wild. The improvement of efficiency is significant after adopting our distributed formulation. The works in [33, 45, 49] and our implementation (denoted as T^*_{MA}) under the proposed framework are $2.6 - 9.9$ times more efficient than the original methods (T_{MA}). We further test the robustness of our pipeline on the challenging ArtsQuad6K and Dubrovnik6K data-sets. The overview and zoom-in figures of these data-sets are shown in Figure 9.

Sequential data-sets In Figure 8, we further demonstrate the robustness of our method on the Temple of Heaven data-set [22] with 341 sequential images under different levels of Gaussian noise in the relative motions. The sparse reconstruction from the traditional motion averaging approach is regarded as the ground-truth for the absolute measurement of rotational and positional errors. A Gaussian noise $\mathcal{N}(0, \sigma_{\mathbf{r}}^2)$ is added to the relative rotations, more specifically to the angle of the angle-axis representation, where $\sigma_{\mathbf{r}} = 1$ (in degrees). Another Gaussian noise $\mathcal{N}(0, \sigma_{\mathbf{t}}^2)$ is added to the relative translations, where $\sigma_{\mathbf{t}} = 1$ (in centimeters). At the noise level of $7\sigma_{\mathbf{r}}$ and $16\sigma_{\mathbf{t}}$, our approach succeeds to reconstruct the close-loop while the traditional method fails. Figure 8(e) and (f) also show the absolute camera rotation and position error curves. The errors of the traditional method fluctuate largely as the noise level increases, while those of our method increase much more stably. In the Figure 10, we add an experiments to confirm the necessity of intra variables.

City-scale data-sets Finally, we turn to the challenging city-scale data-sets, in which the largest City-A data-set contains 1.21 million images of 50 mega-pixels. The peak

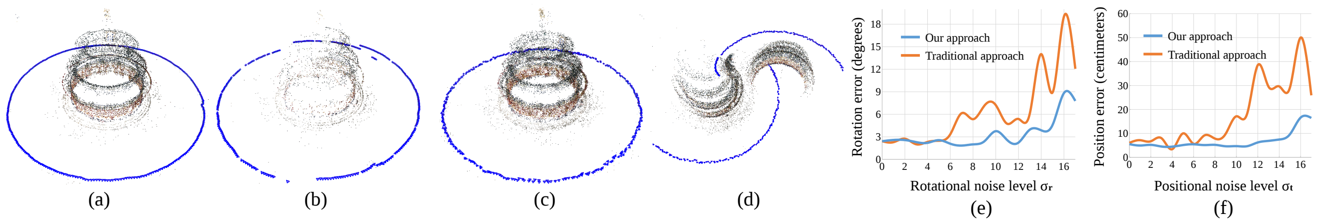

Figure 8: The results of the sequential data-set. (a) are camera poses from our method at the rotational Gaussian noise level of $7\sigma_{\mathbf{r}}$ and (b) are from traditional methods. (c) are camera poses from our method at the positional Gaussian noise level of $16\sigma_{\mathbf{t}}$ and (d) are from traditional methods. (e) and (f) are comparisons of the absolute rotational and positional errors given different levels of rotational and positional Gaussian noises.

Datasets	# images	Number of reconstructed cameras								Motion averaging time (sec)							
		LUD [33]		1DSfM [49]		Sweeney [45]		Ours		LUD [33]		1DSfM [49]		Sweeney [45]		Ours	
		N_c	N_c^*	N_c	N_c^*	N_c	N_c^*	N_c	N_c^*	T_{MA}	T_{MA}^*	T_{MA}	T_{MA}^*	T_{MA}	T_{MA}^*	T_{MA}	T_{MA}^*
Alamo	577	547	**551**	529	**542**	533	**547**	549	**559**	133	20	752	97	892	112	173	24
Ellis Island	227	207	**213**	214	**221**	203	**217**	221	**224**	76	11	139	25	155	20	26	13
Metropolis	341	**288**	287	291	**307**	272	**291**	298	**322**	120	19	201	28	233	32	88	16
Montreal N.D.	450	435	**442**	427	**439**	416	**433**	445	445	167	29	1135	142	1236	156	167	27
Notre Dame	553	536	**538**	**507**	504	501	**519**	514	**542**	126	24	1445	184	1596	200	246	32
NYC Library	332	320	**327**	295	**307**	294	**304**	290	**312**	54	11	392	56	437	57	79	12
Piazza del Popolo	350	305	**321**	308	**327**	302	**331**	334	**342**	31	12	191	31	224	31	72	16
Piccadilly	2152	1953	**2077**	1956	**2099**	1928	**2047**	2114	**2122**	2224	284	2425	303	3455	433	932	173
Roman Forum	1084	901	**1021**	989	**1042**	**966**	959	1079	1079	1243	157	1245	161	1415	192	604	89
Tower of London	572	425	**482**	414	**504**	409	**507**	458	**510**	86	18	606	79	643	82	320	64
Union Square	789	698	**717**	**710**	704	701	**712**	720	**732**	264	33	340	45	442	60	145	35
Vienna Cathedral	836	750	**786**	770	**792**	771	**803**	793	**807**	208	34	2837	360	3135	395	712	72
Yorkminster	437	404	**417**	401	**417**	409	**422**	407	**413**	148	23	777	102	876	109	199	29

Table 1: The statistics of the Internet data-sets [49]. Here, N_c^* and N_c denote the number of reconstructed cameras from an approach with and without the the proposed distributed and robust framework. T_{MA}^* and T_{MA} represent the time of motion averaging with and without the proposed framework. We implement the work [33, 45] and obtain the statistics.

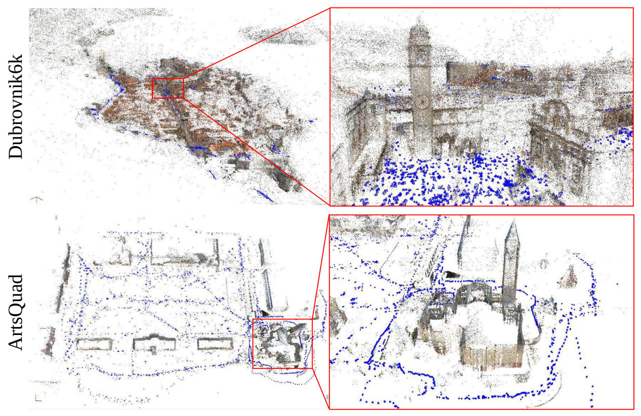

Figure 9: The SfM results of the challenging ArtsQuad6K and Dubrovnik6K data-sets from our pipeline.

memory of the traditional motion averaging approach denoted as "TMA" on the City-A data-set is 134.01 GB, which runs out of memory on our distributed computing system with 64 GB RAM, and it takes 86.7 hours on a single computer to finish a traditional motion averaging. In contrast, our approach completes motion averaging in 9.44 hours with only 7.06 GB peak memory on the distributed computing system. As shown in the table of Figure 12, the peak memory of our approach on the city-scale data-sets is only $4.2\% - 7.7\%$ of the traditional motion averaging method, and the time cost is $10.9\% - 26.0\%$ of the traditional method.

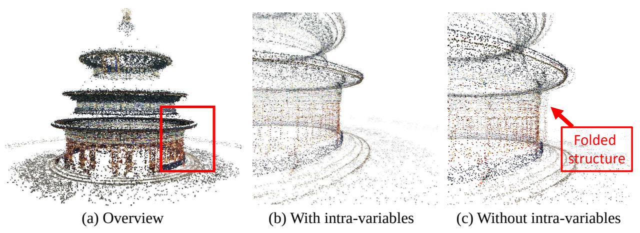

Figure 10: The sparse reconstruction from motion averaging with and without intra variables. As (c) shows, the sparse reconstruction without intra variables (merged by similarity transformations) will introduce folded structures, namely poor camera poses.

Limitations We can see from the plot in Figure 12 that as the number of images drastically increases, more partitions are introduced and the global optimization gradually dominates the memory requirement. Although the peak memory of our largest City-A data-set is only 7.06 GB and there is still a lot of headroom left, global motion averaging comes

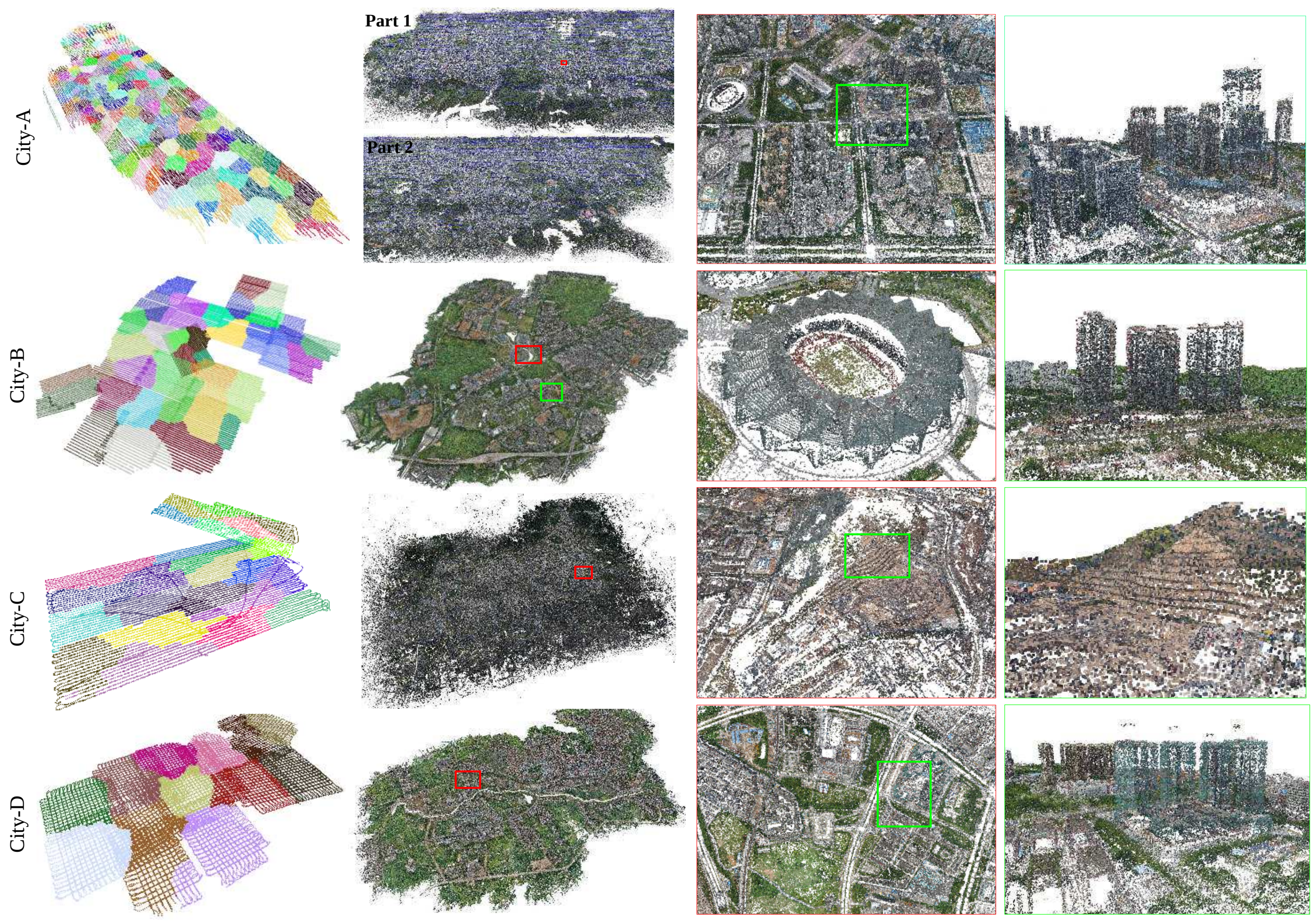

Figure 11: The visual SfM results of our city-scale data-sets. Figures from left to right are respectively camera partitions, the final SfM results after bundle adjustment, and the detailed SfM results visualized from different viewpoints.

Data-set	# images	Resolution	# partitions	# cameras	# relative poses	# points	Peak memory [GB]		Time [hours]	
							TMA	Ours	TMA	Ours
City-A	1210106	50 Mpixel	364	1207472	263.4M	1.72B	134.01	7.06	86.70	9.44
City-B	138200	24 Mpixel	35	138193	29.3M	100.2M	13.76	0.72	17.85	2.45
City-C	91732	50 Mpixel	24	91714	21.3M	76.2M	9.12	0.38	10.95	1.95
City-D	36480	36 Mpixel	13	36428	8.1M	27.8M	3.62	0.28	3.65	0.95

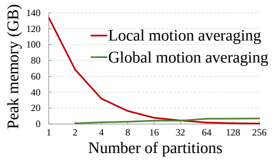

Figure 12: Left: the statistics of the city-scale data-sets. “Traditional motion averaging” is abbreviated as “TMA”. Right: the peak memory of local and global motion averaging given different numbers of partitions on the City-A data-set.

to be the bottleneck of scalability of our approach along with the continuously enlarged reconstruction scale.

6. Conclusion

Finally, the contributions of this paper can be summed up in two points. First, we introduce a divide-and-conquer framework to handle large-scale motion averaging problems in a distributed manner with almost a magnitude reduction in both memory and computation time. Second, the recursive partitioning reorders the optimization of camera poses in a more robust manner. Remarkably, the proposed framework is applicable to the majority of the state-of-the-art motion averaging methods to boost their scalability and robustness. Future work includes further investigation of the numerical methods of parallel and distributed computation. We also intend to propose a truly distributed SfM system with acceptable machine-machine communications able to handle more large-scale motion averaging problems.

Acknowledgement

This work is supported by Hong Kong RGC 16208614, Hong Kong ITC PSKL12EG02, T22-603/15N, and China 973 program, 2012CB316300. The city-scale data-sets are provided by Altizure.com.

References

[1] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski. Building rome in a day. *Commun. ACM*, 54(10):105–112, 2011. 1, 2

[2] S. Agarwal, K. Mierle, and Others. Ceres solver. http://ceres-solver.org. 1, 3, 6

[3] M. Arie-Nachimson, S. Z. Kovalsky, I. Kemelmacher-Shlizerman, A. Singer, and R. Basri. Global motion estimation from point matches. In *3DIMPVT*, 2012. 1, 2, 5

[4] B. Bhowmick, S. Patra, A. Chatterjee, V. M. Govindu, S. Banerjee, B. Bhowmick, S. Patra, A. Chatterjee, V. M. Govindu, and S. Banerjee. Divide and conquer: Efficient large-scale structure from motion using graph partitioning. 2014. 2, 5, 6

[5] M. Brand, M. Antone, and S. Teller. Spectral solution of large-scale extrinsic camera calibration as a graph embedding problem. In *ECCV*, 2004. 1, 2, 5

[6] L. Carlone, R. Tron, K. Daniilidis, and F. Dellaert. Initialization techniques for 3d slam: a survey on rotation estimation and its use in pose graph optimization. In *ICRA*, 2015. 1, 2, 5

[7] A. Chatterjee and V. M. Govindu. Efficient and robust large-scale rotation averaging. In *ICCV*, 2013. 1, 2, 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 *ECCV*, 2010. 2

[9] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. SfM with MRFs: Discrete-continuous optimization for large-scale structure from motion. *PAMI*, 2013. 1, 2

[10] Z. Cui, N. Jiang, C. Tang, and P. Tan. Linear global translation estimation with feature tracks. In *BMVC*, 2015. 2, 3

[11] Z. Cui and P. Tan. Global structure-from-motion by similarity averaging. In *ICCV*, 2015. 1, 2, 3, 6

[12] I. S. Dhillon, Y. Guan, and B. Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *PAMI*, 29(11):1944–1957, 2007. 4

[13] E. Dunn and J. Frahm. Next best view planning for active model improvement. In *BMVC*, 2009. 2

[14] A. Eriksson, J. Bastian, T.-J. Chin, and M. Isaksson. A consensus-based framework for distributed bundle adjustment. In *CVPR*, 2016. 2

[15] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys. Building rome on a cloudless day. In *ECCV*, 2010. 2

[16] Y. Furukawa, B. Curless, S. M. Seitz, R. Szeliski, and G. Inc. R.: Towards internet-scale multiview stereo. In *CVPR*, 2010. 2

[17] V. M. Govindu. Combining two-view constraints for motion estimation. In *CVPR*, 2001. 1, 2, 5

[18] V. M. Govindu. Lie-algebraic averaging for globally consistent motion estimation. In *CVPR*, 2004. 1, 2, 5

[19] S. Haner and A. Heyden. Covariance propagation and next best view planning for 3d reconstruction. In *SSBA*, 2012. 2

[20] R. I. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *IJCV*, 103(3):267–305, 2013. 1, 2, 5

[21] N. Jiang, Z. Cui, and P. Tan. A global linear method for camera pose registration. In *ICCV*, 2013. 1, 2

[22] N. Jiang, P. Tan, and L. F. Cheong. Seeing double without confusion: Structure-from-motion in highly ambiguous scenes. In *ICCV*, 2012. 2, 3, 6

[23] F. Kahl and R. Hartley. Multiple-view geometry under the L_infty-norm. *PAMI*, 30(9):1603–1617, 2008. 2

[24] L. Kneip, D. Scaramuzza, and R. Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *CVPR*, 2011. 5

[25] R. J. Lipton, D. J. Rose, and R. E. Tarjan. Generalized nested dissection. *SIAM Journal on Numerical Analysis*, 16(2):346–358, 1979. 1, 2, 3

[26] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 6

[27] D. Martinec and T. Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *ICPR*, 2007. 1, 2, 5

[28] P. Moulon, P. Monasse, and R. Marlet. Global fusion of relative motions for robust, accurate and scalable structure from motion. In *ICCV*, 2013. 1, 2

[29] K. Ni, D. Steedly, and F. Dellaert. Out-of-core bundle adjustment for large-scale 3d reconstruction. In *ICCV*, 2007. 2

[30] D. Nistér. An efficient solution to the five-point relative pose problem. *PAMI*, pages 756–770, 2004. 5

[31] D. Nistér and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006. 6

[32] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006. 6

[33] O. Özyesil and A. Singer. Robust camera location estimation by convex programming. In *CVPR*, 2015. 1, 2, 5, 6, 7

[34] M. Pollefeys, D. Nistér, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3d reconstruction from video. *IJCV*, 78(2-3):143–167, 2008. 1, 2

[35] 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(3):207–232, 2004. 1, 2

[36] C. Rother. *Multi-View Reconstruction and Camera Recovery using a Real or Virtual Reference Plane*. PhD thesis, 2003. 2

[37] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 1, 2

[38] T. Shen, J. Wang, T. Fang, S. Zhu, and L. Quan. Color correction for image-based modeling in the large. In *ACCV*, 2016. 2

[39] T. Shen, S. Zhu, T. Fang, R. Zhang, and L. Quan. Graph-based consistent matching for structure-from-motion. In *ECCV*, 2016. 2

[40] K. Sim and R. Hartley. Recovering camera motion using l∞ minimization. In *CVPR*, 2006. 2

[41] S. N. Sinha, D. Steedly, and R. Szeliski. A multi-stage linear approach to structure from motion. In *ECCV-workshop RMLE*, 2010. 2

[42] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring image collections in 3d. *SIGGRAPH*, 2006. 1

[43] N. Snavely, S. M. Seitz, and R. Szeliski. Skeletal graphs for efficient structure from motion. In *CVPR*, 2008. 2

[44] C. Sweeney, V. Fragoso, T. Höllerer, and M. Turk. Large scale sfm with the distributed camera model. In *3DV*, 2016. 2

[45] C. Sweeney, T. Sattler, T. Hollerer, M. Turk, and M. Pollefeys. Optimizing the viewing graph for structure-from-motion. In *ICCV*, 2015. 1, 6, 7

[46] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *LNCS*, 2000. 5

[47] J. Wang, T. Fang, Q. Su, S. Zhu, J. Liu, S. Cai, C. Tai, and L. Quan. Image-based building regularization using structural linear features. *TVCG*, 22(6):1760–1772, 2016. 2

[48] K. Wilson, D. Bindel, and N. Snavely. When is rotations averaging hard? In *ECCV*, 2016. 1

[49] K. Wilson and N. Snavely. Robust global translations with 1dsfm. In *ECCV*, 2014. 1, 2, 3, 4, 5, 6, 7

[50] C. Wu. Towards linear-time incremental structure from motion. In *3DV*, 2013. 1, 2

[51] Y. Yao, S. Li, S. Zhu, T. Fang, H. Deng, and L. Quan. Relative camera refinement for accurate dense reconstruction. In *3DV*, 2017. 2

[52] C. Zach, A. Irschara, and H. Bischof. What can missing correspondences tell us about 3d structure and motion? In *CVPR*, 2008. 2, 3

[53] R. Zhang, S. Li, T. Fang, S. Zhu, and L. Quan. Joint camera clustering and surface segmentation for large-scale multi-view stereo. In *ICCV*, 2015. 2

[54] R. Zhang, S. Zhu, T. Fang, and L. Quan. Distributed very large scale bundle adjustment by global camera consensus. In *ICCV*, 2017. 2, 6

[55] L. Zhou, S. Zhu, T. Shen, J. Wang, T. Fang, and L. Quan. Progressive large scale-invariant image matching in scale space. In *ICCV*, 2017. 2

[56] S. Zhu, T. Fang, J. Xiao, and L. Quan. Local readjustment for high-resolution 3d reconstruction. In *CVPR*, 2014. 2

[57] S. Zhu, T. Fang, R. Zhang, and L. Quan. Multi-view geometry compression. In *ACCV*, 2014. 2

[58] S. Zhu, T. Shen, L. Zhou, R. Zhang, J. Wang, T. Fang, and L. Quan. Parallel structure from motion from local increment to global averaging. In *arXiv:1702.08601*, 2017. 2, 5