Accurate, Scalable and Parallel Structure from Motion

Siyu Zhu Tianwei Shen Lei Zhou Runze Zhang Jinglu Wang Tian Fang Long Quan
The Hong Kong University of Science and Technology
{szhu,tshenaa,lzhouai,rzhangaj,jwangae,tianft,quan}@cse.ust.hk

Abstract
In this paper, we tackle the accurate Structure from Motion (SfM) problem, in particular camera registration, far exceeding the memory of a single compute node. Different from the previous methods which drastically simplify the parameters of SfM, we preserve as many cameras, tracks and their corresponding connectivity as possible for a highly consistent and accurate SfM. By means of a camera clustering algorithm, we divide all the cameras and associated images into clusters and leverage such formulation to process the subsequent track generation, local incremental SfM and final bundle adjustment in a scalable and parallel scheme. Taking the advantages of both incremental and global SfM methods, we apply the relative motions from local incremental SfM to the global motion averaging framework and obtain more accurate and robust global camera poses than the state-of-the-art methods. We intensively demonstrate the superior performance of our method on the benchmark, Internet and our own challenging city-scale data-sets.

1. Introduction
Thanks to the rapid development of portable cameras and the explosion of online image collections, large-scale Structure from Motion (SfM) has achieved extraordinary progress in the past few years. The state-of-the-art large-scale SfM methods [1, 16, 25, 28, 45, 49] have already provided ingenious designs in feature extraction [32, 56], overlapping image detection [1, 16, 25, 38], feature matching and verification [57], connected component merging [1, 25], and bundle adjustment [14, 32, 37, 58], while seldom deal with large-scale highly accurate camera registration, not to mention in a parallel fashion. In order to fit a whole camera registration problem into a single compute node, previous large-scale SfM works [1, 16, 25, 45, 49] generally build a compact skeletal geometry first and register the remaining cameras with respect to the skeletal reconstruction. As the problem size scales up, such works should drastically discard camera-to-camera and camera-to-track connectivity and suffer from a significant loss of geometry accuracy and completeness. Moreover, the degree of parallelism of such works is limited since they perform camera registration in terms of a whole connected component of cameras.

In this paper, we consider as many cameras, tracks and their corresponding connectivity as possible to guarantee a highly accurate and consistent reconstruction. To tackle the memory bottleneck of a single compute node and fully utilize the available parallel computation resources, we introduce a camera clustering algorithm to split the original problem into several smaller sub-problems in terms of clusters of cameras, individually solve the subsequent scalable SfM sub-problems including track generation and local incremental SfM in parallelism and finally merge them into an accurate and consistent reconstruction. We also embrace the advantages of both the robust incremental SfM method [1, 40, 41, 44, 47, 57], which is composed of RANSAC [6] based filters and repeated partial bundle adjustment [53] and can effectively remove outlier essential matrices and feature correspondences, and the global SfM method [2, 3, 4, 5, 17, 18, 22, 33, 39], which is extremely ef-
efficient and able to compensate for drifting errors, and apply the highly accurate relative motions from local incremental SfM in terms of partitioned camera clusters with overlapping into a global motion averaging framework. Consequently, we can obtain more accurate and robust global camera poses than the commonly used essential-matrix-based methods [2, 3, 17, 26, 39] and trifocal-tensor-based methods [7, 17, 26, 30].

The contributions of this paper are two-fold. First, we introduce a highly scalable framework by means of the proposed camera clustering algorithm to solve the SfM problem, in particular camera registration, far exceeding the memory of a single compute node in a parallel scheme. Second, we assimilate the advantages of both incremental and global SfM methods and leverage the more accurate and robust relative poses from local incremental SfM to the motion averaging framework and consequently obtain obviously improved global camera geometry.

2. Related Works

The previous state-of-the-art methods [15, 23, 30, 42, 52] have already endowed us with the hierarchical SfM ideology adopted in this paper but such works are limited to small-scale or sequential data-sets. Havlena et al. [23, 24] form the final 3D model by merging atomic 3D models reconstructed from camera triples together, while the merging process is not robust depending solely on common 3D points. Toldo et al. [52] propose a hierarchical camera cluster tree to merge the pairwise partial reconstruction results, the process of which however significantly depends on the tree structure. The work of Lhuillier [30] also suffers from the biased incremental motion averaging. To handle city-scale SfM in a highly scalable fashion, we arrange the sub-problems in terms of clusters of cameras and associated images divided by a camera clustering algorithm instead.

In this paper, we combine both advantages of incremental and global SfM methods and improve the state-of-the-art motion averaging method to merge partial camera poses into an unbiased global consistent camera geometry. The well-known incremental SfM method Photo Tourism [47] and its derivation methods [11, 40, 41, 44, 57] are able to robustly handle unstructured image data-sets, even the challenging ones harvested from the Internet. Based on an initial pair, such pipelines incrementally recover the “next-best-view” [13, 20] by carrying out perspective-three-point (P3P) [29] combined with RANSAC [6] and frequent non-linear bundle adjustment [53] to effectively remove outlier feature correspondences and essential matrices. However, the frequent intermediate bundle adjustment is at the expense of incredible time consumption and unbalanced optimization of geometry, namely drifted convergence, especially on large-scale data-sets.

In contrast, the global SfM methods [2, 3, 4, 5, 17, 18, 22, 33, 39] solves all the camera poses simultaneously from all the input pair-wise relative poses and can effectively handle significant drifting errors. Furthermore, the computation of relative poses combined with local bundle adjustment is highly parallel, and the global SfM methods are therefore superior in efficiency than the incremental methods. However, compared with incremental SfM methods which repeatedly adopt RANSAC based correspondence filtering when initializing new camera poses, global SfM methods are more sensitive to possible erroneous epipolar geometry despite the various delicate designs of epipolar geometry filters [19, 27, 43, 54, 55, 59, 60]. We therefore apply the accurate and robust relative motions obtained from incremental SfM in terms of small-scale camera clusters to the unbiased global motion averaging framework and consequently acquire superior final camera geometry in a highly efficient manner.

3. Scalable Formulation

3.1. Preliminary

We start with a given set of images $I = \{I_i\}$, their corresponding SIFT features $F = \{F_i\}$ and matching correspondences $M = \{M_{ij} \mid M_{ij} \subset F_i \times F_j, i \neq j\}$ where $M_{ij}$ is a set of inlier feature correspondences verified by epipolar geometry [21] between two images $I_i$ and $I_j$. Each image $I_i$ is associated with a camera $C_i \in C$. Our target of this paper is then to compute the global camera poses of such cameras $C = \{C_i\}$. On the other hand, all our SfM operations are performed in a distributed computing system.
consisted of three main components, namely a master, several compute nodes, and a distributed storage. The master is a logically centralized controller. Each of the compute nodes contains individual memory and is actually a process responsible for running tasks in parallel with other compute nodes. The distributed storage is a scalable network file system which is shared by the all the compute nodes and stores all the raw inputs, intermediate results, and final outputs.

3.2. Camera Clustering

As the problem of SfM, in particular camera registration, scales up, the following two problems emerge. First, the problem size gradually exceeds the memory of a single compute node. Second, the high degree parallelism of our distributed computing system can hardly be fully utilized. We therefore first introduce a camera clustering algorithm to split the original SfM problem into several smaller manageable sub-problems in terms of clusters of cameras and associated images. Specifically, our goal of camera clustering is to find camera clusters such that all the scalable SfM operations of each cluster can be fitted into a single compute node for efficient processing (size constraint) and that all the clusters have sufficient overlapping cameras with adjacent clusters to guarantee a complete reconstruction when their corresponding partial reconstructions are merged together in the motion averaging framework (completeness constraint).

3.2.1 Clustering Formulation

In order to encode the relationships between all the cameras and associated tracks, we introduce a camera graph \( G = \{V, E\} \), in which each node \( V_i \in V \) represents a camera \( C_i \in C \), each edge \( e_{ij} \in E \) with weight \( w(e_{ij}) \) connects two different cameras \( C_i \) and \( C_j \). In the subsequent scalable SfM, both local incremental SfM and bundle adjustment encourage cameras with great numbers of common features to be grouped together for a robust geometry estimation. We therefore define the edge weight \( w(e_{ij}) \) as the number of feature correspondences, namely \( w(e_{ij}) = |M_{ij}| \). Our target is then to partition the original camera graph \( G = \{V, E\} \) into a set of sub-graphs \( \{G_k|G_k = \{V_k, E_k\}\} \) with overlapping vertices and associated edges while satisfying the following size and completeness constraints.

**Size constraint** We encourage the number of cameras of each sub-problem \( G_k = \{V_k, E_k\} \) to be small and of similar size. First, each sub-problem should be small enough to be fit into a single compute node for efficient computation. Particularly for local incremental SfM, a comparatively small-scale sub-problem can effectively avoid redundant time-consuming intermediate bundle adjustment and possible drifting. Second, a balanced problem partition stimulates a fully utilization of the distributed computing system. The size constraint is therefore defined as

\[
\forall G_i \in \{G_k\}, |V_i| \leq \Delta_{up} \\
\forall G_i, G_j \in \{G_k\}, |V_i| \approx |V_j|
\]

(1)

where \( \Delta_{up} = \min(N_{\text{cam}}, \frac{|V_i|}{m}) \), \( N_{\text{cam}} \) is the upper bound of the number of cameras of a sub-problem and \( m \) is the number of compute nodes. We observe from Figure 3 that both the average relative rotation and translation errors computed from local incremental SfM in a cluster first remarkably decrease and then stabilize as the number of cameras in a cluster increases. The acceptable number of cameras in a cluster is therefore in a large range and we choose \( N_{\text{cam}} = 100 \) for the trade-off between accuracy and efficiency.

**Completeness constraint** The completeness constraint enforces that the individual partial reconstructions from sub-problems can be merged into a complete global reconstruction. More concretely, the vertices and edges of all the final sub-graphs \( \{G_k|G_k = \{V_k, E_k\}\} \) should be in a connected component of the original camera graph \( G = \{V, E\} \). To further encourage a robust merging of partial reconstructions from divided sub-problems, we increase the number of overlapping cameras between adjacent camera clusters and accordingly define \( \delta(G_i) = \frac{|\sum_{j \neq i} |V_i \cap V_j| |}{|V_i|} \) as the completeness ratio of a sub-problem \( G_i \), which quantifies the degree cameras covered in one sub-problem \( G_i \) are also covered by other sub-problems. We therefore enforce that \( \forall G_i \in \{G_k\}, \delta(G_i) = \delta_c \) where \( \delta_c = 0.7 \) in this paper.

3.2.2 Clustering Algorithm

Starting with the dominant connected component of the original camera graph \( G = \{E, V\} \), we design a three-step algorithm to solve the camera clustering problem. We illustrate a sample output of this algorithm in Figure 4.

1. **Graph simplification** We trim the edges of camera graph \( G = \{E, V\} \) to effectively reduce the complexity of the remaining two steps. More concretely, the simplified graph \( G' = \{E', V\} \) should satisfy that the degree of each vertex \( V_i \in V \) is below a threshold \( \lambda_d \) (\( \lambda_d = 80 \)). We then sort all the graph edges by edge weight \( w(e_{ij}) \) in ascending order and greedily discard the edge with the lowest weight and the degree of any of its connected two vertices larger than \( \lambda_d \). We iteratively perform the edge removal until that every graph node satisfies the degree constraint.
We enforce the completeness constraint by recursively splitting camera clusters. More specifically, a camera cluster violating the size constraint is divided into smaller components. Starting with the simplified graph $G'\prime$, we iteratively apply normalized-cut algorithm [10], which guarantees an unbiased vertex partition, to divide any sub-graph $G_c$ not satisfying the size constraint into two balanced sub-graphs $G_{c1}$ and $G_{c2}$, until that no sub-graphs violate the size constraint. Intuitively, camera pairs with great numbers of common features have high edge weights and are less likely to be cut.

3. Graph expansion We enforce the completeness constraint by introducing sufficient overlapping cameras between adjacent camera clusters. First, we guarantee that the vertices and edges of all the sub-graphs after graph expansion should belong to a single connected component of the original camera graph $G$. We sort $E_{\text{dis}}$ the edges discarded in the graph division step by edge weight $w(e_{ij})$ in descending order and iteratively add the edge $e_{ij}$ and associated vertices $V_i$ and $V_j$ randomly to one of its connected sub-graphs $G(V_i)$ and $G(V_j)$, if $G(V_i)$ and $G(V_j)$ have no common vertices, where $G(V_i)$ is the sub-graph containing vertex $V_i$. Such process is iteratively performed until no additional edges can be added to any of the sub-graph. Next, we introduce more overlapping cameras between adjacent sub-problems to guarantee a robust complete reconstruction. For each sub-problem $G_i$, we sort all its adjacent edges by edge weight $w(e_{ij})$ in descending order and iteratively add one edge $e_{ij}$ and associated vertex $V_j$ to $G_i$ if the completeness ratio of $G_i$ is lower than $\delta_c$. When all the adjacent edges are traversed and the completeness constraint is still not satisfied, we perform another iteration of graph expansion considering the adjacent edges of newly added vertices until all the sub-graphs satisfy the completeness constrain.

3.3. Camera Cluster Classification

The camera clusters from the clustering algorithm can be divided into two categories, namely independent and interdependent camera clusters. Before the graph expansion step, all the partitioned camera clusters are fully independent and we define them as independent camera clusters. Such independent camera clusters are vital to the following parallel 3D point triangulation and parallel bundle adjustment. We can also rely on the independent camera clusters to build the hierarchical camera cluster tree $T_c$, in which each leaf node corresponds to an independent camera cluster before graph expansion and each non-leaf node is associated with an intermediate camera cluster during the recursive binary graph division. The hierarchical camera cluster tree is a crucial structure in the subsequent parallel track generation. Accordingly, we define the camera clusters after graph expansion as interdependent camera clusters since they share overlapping cameras with adjacent clusters. Such interdependent camera clusters are applied into the process of parallel local incremental SfM.

4. Scalable Implementation

4.1. Track Generation

The first step of scalable SfM is to leverage the pairwise feature correspondences to generate globally consistent tracks across all the images, and the problem can be solved by a standard Union-Find [35] algorithm. However, as the size of the input image collections scales up, it gradually becomes impossible to concurrently load all the feature and associated match files into the memory of a single compute node for track generation. We therefore base on the hierarchical camera cluster tree $T_c$ to perform track generation. In detail, we define $N_k^i$ as the node in the $k$th level of a hierarchical camera cluster tree $T_c$, and $N_{k+1}^i$ and $N_{k+1}^i$
are respectively the left and right child of $N^k_i$. For two sub-problems of track generation associated with two sibling leaf nodes $N^k_{i_1}$ and $N^k_{i_2}$, we load their features and correspondences into memory, generate the tracks corresponding to $N^k_i$, reallocate the memory of features and correspondences, and save the tracks associated with $N^k_i$ into storage. As for the two sibling non-leaf nodes $N^{k+1}_{j_1}$ and $N^{k+1}_{j_2}$, we simply load their corresponding tracks, merge them, and save the tracks corresponding to $N^k_i$ into storage. Such processes are iteratively performed from the bottom up until the globally consistent tracks with respect to the root node of $T_c$ are obtained. All the track generation processes associated with each level of $T_c$ are handled in parallel.

4.2. Local Incremental SFM

For the cameras and corresponding tracks of each interdependent camera cluster denoted by the sub-graph $G_k = \{E_k, V_k\}$, we perform the subsequent local incremental SFM on each compute node in parallelism. Local incremental SFM considers $N$-view ($\geq 3$) in the pose estimation of each newly added camera \cite{22} and combines RANSAC \cite{6} based filters with repeated partial bundle adjustment \cite{57} to effectively discard erroneous feature correspondences and essential matrices. We can therefore acquire far more accurate and robust relative motions than the generally adopted pairwise essential-matrix-based methods \cite{2, 3, 17, 39} and the trifocal-tensor-based methods \cite{7, 17, 26, 36}, even for the relative poses with weak association, large angle of views, and great scale variation.

4.3. Motion Averaging

Given a set of $n$ images $I = \{I_i\}$, the motion averaging problem is to leverage the $C_n^2 = \frac{n(n-1)}{2}$ pairs of relative motions to compute the global camera poses $\{P_i\}$ of the corresponding cameras $C = \{C_i\}$. Although the global optimal solution to the least loss of relative motions compared with the complete camera graph cannot be obtained, we can still acquire more than adequate relative motions from local incremental SFM, in which $C_N^k = \frac{k(k-1)}{2}$; accurate and robust relative motions can be obtained from a cluster of $k$ cameras for a robust and accurate estimation of the global camera geometry. It is noteworthy that a single compute node can easily handle the whole process of motion averaging even considering tens of thousands of cameras and corresponding particularly sparse 3D points.

4.3.1 Preprocessing

First, we only leverage the relative motions associated with camera pairs with sufficient common tracks ($\geq 12$) for the following motion averaging and such preprocessing remarkably decreases the following computation complexity. Since a camera pair $(C_i, C_j)$ may exist in different interdependent camera clusters, there are multiple correspond-

![Figure 5: The comparison of the cumulative distribution functions (CDF) of camera position errors for the Union Square and Piccadilly data-sets between different translation averaging methods, and our method considering camera-to-point relative translations (C2P RT) or not.](image)

Ours with C2P RT
Ours without C2P RT
Sweeney
Moulon

Figure 5: The comparison of the cumulative distribution functions (CDF) of camera position errors for the Union Square and Piccadilly data-sets between different translation averaging methods, and our method considering camera-to-point relative translations (C2P RT) or not.

Ours with C2P RT
Ours without C2P RT
Sweeney
Moulon

We also follow the state-of-the-art work \cite{5} for efficient and robust rotation averaging.

4.3.2 Translation Averaging

We improve the highly efficient and principled translation averaging scheme \cite{55} by minimizing the translation averaging cost function $E_t$ defined by $E_t = E_c + \alpha E_p$, where $E_c$ and $E_p$ are camera-to-camera and camera-to-point translation averaging cost functions respectively, and $\alpha$ balances the initial values of $E_c$ and $\alpha E_p$ to be equal.

**Camera-to-camera relative translation** Given the input cameras $C = \{C_i\}$ with projection matrices denoted by $\{P_i|P_i = K_i[R_i - R_i c_i]\}$ and a set of camera-to-camera relative translations $T_c = \{t_{ij}\}$, the camera-to-camera translation averaging cost function $E_c$ can be formulated as:

$$E_c = \sum_{t_{ij} \in T_c} d_{\text{chord}}(\frac{R_i^T t_{ij}^c}{\|R_i^T t_{ij}\|}, \|c_i - c_j\|^2),$$

where $d_{\text{chord}}(\hat{t}_1, \hat{t}_2)$ is the chordal distance between two unit vectors $\hat{t}_1 \in \mathbb{R}^3$ and $\hat{t}_2 \in \mathbb{R}^3$.

**Camera-to-point relative translation** We also consider camera-to-point relative translations in translation averaging because the camera-to-point relative translation implicitly encodes the scale of a translation while the traditional translation averaging methods based on epipolar geometry \cite{2, 3, 17, 39} only encode the direction of a translation. It is also because camera-to-point relative translations enrich the constraints of less photographed scenes
and effectively eliminate the possibility that a consistent model from weakly associated images splits into disconnected sub-models. However, the state-of-the-art work [55] mentions that the augmentation of the translation averaging problem with camera-to-point relative translations generally introduces noisy structure and motion, since there exists enormous erroneous connectivity between cameras and tracks. Thanks to local incremental SFM which is consisted of RANSAC [6] based filters able to discard outlier camera-to-track correspondences and bundle adjustment [53] further optimizing the relative motions in a non-linear least square manner, we can directly apply the accurate and robust camera-to-point relative translations from the clusters of cameras and tracks recovered by local incremental SFM to our translation averaging formulation. More specifically, given the input cameras \( C = \{C_i\} \) with projection matrices denoted by \( \{P_i\}|P_i = K_i[R_i] [R, c_i] \), their visible sparse 3D points \( P = \{P_j\} \) and associated relative translations \( T_p = \{t^{(i)}_{ij}\} \) between cameras and visible 3D points with respect to the cameras, the camera-to-point translation averaging cost function \( E_p \) is defined as:

\[
E_p = \sum_{t^{(i)}_{ij} \in T_p} d_{\text{cheb}}(\frac{R_i t^{(i)}_{ij}}{||R_i t^{(i)}_{ij}||}, \frac{P_j - c_i}{||P_j - c_i||})^2 . \tag{3}
\]

In terms of implementations, we greedily select 3D points visible by most successfully recovered cameras until such cameras see \( k \) points, and we choose \( k = 10 \) for the trade-off between quality and efficiency. We still use random initialization for the positions of camera optical centers and 3D points and it works remarkably well [55].

As shown in Table 1 the averaged global camera geometry adopting the relative motions from local incremental SFM obviously outperforms those from the state-of-the-art methods [56, 51]. Figure 5 also demonstrates that the augmentation of camera-to-point relative translations further improves the final translation averaging results.

### 4.4. Bundle Adjustment

We follow the state-of-the-art algorithm proposed by Eriksson et al. [14] for distributed bundle adjustment. Since this work declares to have no restriction on the partitions of cameras, we directly refer to the independent camera clusters and associated cameras, tracks and projections as the sub-problems of the objective function of bundle adjustment. For each independent camera cluster, we triangulate [51] their corresponding 3D points with sufficient visible cameras (\( \geq 3 \)) from their feature correspondences validated by local incremental SFM based on the averaged global camera geometry. Next, the independent sub-problems including solving a total sum-of-squares problem which can be directly applicable to a standard bundle adjustment formulation and averaging multiple estimations of 3D points from different sub-problems are evaluated iteratively in parallel until convergence. The process of averaging multiple estimations of 3D points requires value exchanges between multiple camera clusters, and we perform a compromising non-distributed implementation for the small-scale data-sets (\( \leq 10000 \)) to avoid inefficient I/O. As for the city-scale bundle adjustment which is handled in a truly multi-computer distributed manner, such I/O between multiple compute nodes is inevitable.

## 5. Experiments

### Implementation and data-sets

We implement our approach in C++ and perform all the experiments on a distributed computing system consisting of 10 compute nodes each of which has 6-Core (12 threads) Intel 3.40 GHz processors and 128 GB memory. All the compute nodes are deployed on a scalable network file system similar to Hadoop File System [46]. We implement a multicore bundle adjustment solver similar to PBA [58] to solve the non-linear optimization problem, and utilize Graclus [11] to solve the normalized-cut problem. Three types of data-sets are introduced for testing, namely the benchmark data-sets [50] with absolute measurements of camera poses, publicly available landmark-scale Internet photo collections [55], and challenging city-scale data-sets containing up to hundreds of thousands of high resolution images.

### Benchmark data-sets

The statistics of the comparisons of the benchmark data-sets [50] between the state-of-the-art methods [8, 56, 51, 57] and our proposed method are shown in Table 1. To guarantee the success of camera clustering, we set \( N_{\text{cam}} \) = 7 for all the benchmark data-sets. For the data-sets with ground-truth camera intrinsics, our method generates obviously more accurate relative poses, including relative rotations and translations, and consequently guarantees almost an order of magnitude more accurate global camera geometry after motion averaging than the state-of-the-art methods [8, 56, 51, 57]. We therefore demonstrate

| Data-set | Input | Output | Ground truth intrusion | Approximate intrusion | Ours |
|----------|-------|--------|------------------------|----------------------|------|
| HerzJesuP8 | 204.1 | 25.6 | 320.1 | 61.9 | 81.2 |
| HerzJesuP25 | 204.1 | 25.6 | 320.1 | 61.9 | 81.2 |
| CastleP19 | 204.1 | 25.6 | 320.1 | 61.9 | 81.2 |
| EntryP10 | 204.1 | 25.6 | 320.1 | 61.9 | 81.2 |

Table 1: The comparison of the benchmark data-sets [50]. \( \delta R \) is the average relative rotation error measured in degrees, \( \delta t \) denotes the average relative translation error measured in degrees, \( x \) is the average position error in millimeters after motion averaging and before bundle adjustment, and \( x_{BA} \) denotes the average position error in millimeters after bundle adjustment.


that local SfM involving RANSAC [6] based pose estimation [29] and frequent local bundle adjustment [53] can effectively discard outlier feature correspondences and erroneous epipolar geometry and guarantee more accurate and robust relative poses for motion averaging. We also find that our superior performance is significantly more evident for the data-sets with approximate initial intrinsics, in which we regard robust relative poses for motion averaging. We also find that the running time overhead for each method, and introduce $T_{LS}$, $T_{BA}$ and $T_{BA}$ to denote the local incremental SfM time, final full bundle adjustment time, and total running time respectively.

Table 2: The comparison of the Internet data-sets [53]. We regard the SfM results of [47] as the reference model. $N_c$ denotes the number of recovered cameras in SfM, $\bar{x}$ is the median position error measured in meters after motion averaging and before bundle adjustment, and $\bar{x}_{BA}$ represents the median position error in meters after bundle adjustment. We also compare the time overhead for each method, and introduce $T_{LS}$, $T_{BA}$ and $T_{BA}$ to denote the local incremental SfM time, final full bundle adjustment time, and total running time respectively.

City-scale data-sets

The statistics of the input city-scale data-sets are shown in Table [3]. The resolution of the Internet images [1] [10] [45] [47] [48] is generally several megapixels and at most thousands of features are detected for each image, while we leverage the images with a resolution ranging from 24 megapixels to 50 megapixels for the city-scale SfM and the average number of detected features of each image ranges from 69.1K to 234.31. In Figure 10, all the input images of our four city-scale data-sets are reconstructed in a connected component without any simplification of connectivity. Obviously, the state-of-the-art methods [1] [10] [45] [47] [48] are unable to handle such large-scale SfM problems. Moreover, the statistics in Table [3] also show that the whole structure and motion generated by our pipeline are well constrained since the average number of connected cameras with sufficient tracks (≥ 12) of each camera ranges from 176.49 to 234.31 and each track is visible by at least 6.24 cameras in average. In Figure 10 we also provide the visual results of the city-scale data-sets containing both mesh and textured models with vivid and delicate reconstruction details to qualitatively demonstrate the high accuracy of the finally recovered camera geometry.

For the efficiency, we note in Table [3] that the running time of track generation and local incremental SfM grows much faster than the state-of-the-art methods [26] [39] [51] [55], which leverage auxiliary geographical information to separate the input data-sets into small-scale subsets of images for individual processing [1] [10] [25] and meanwhile extensively discard camera-to-camera and camera-to-track connectivity [48], all the input images of our four city-scale data-sets are reconstructed in a connected component without any simplification of connectivity. Obviously, the state-of-the-art methods [1] [10] [45] [47] [48] are unable to handle such large-scale SfM problems. Moreover, the statistics in Table [3] also show that the whole structure and motion generated by our pipeline are well constrained since the average number of connected cameras with sufficient tracks (≥ 12) of each camera ranges from 176.49 to 234.31 and each track is visible by at least 6.24 cameras in average. In Figure 10 we also provide the visual results of the city-scale data-sets containing both mesh and textured models with vivid and delicate reconstruction details to qualitatively demonstrate the high accuracy of the finally recovered camera geometry.

For the efficiency, we note in Table [3] the running time of track generation and local incremental SfM grows linearly as the number of images increases. We also find that the running time of bundle adjustment gradually dom-
City A: 138200 images of 24 megapixels, 138193 cameras, 100.2M sparse 3D points

City B: 36480 images of 36 megapixels, 36428 cameras, 27.8M sparse 3D points

City C: 91732 images of 50 megapixels, 91714 cameras, 76.2M sparse 3D points

City D: 48284 images of 24 megapixels, 47321 cameras, 36.3M sparse 3D points

Table 3: Statistics of the city-scale data-sets. \( N_f \) denotes the average number of features per image, \( N_k \) denotes the number of camera clusters, \( \bar{n}_{con} \) denotes the average number of connected cameras (\( \geq 15 \) common tracks) of each camera, \( \bar{x}_{proj} \) denotes the average reprojection error measured in pixels, and \( T_{TGC}, T_{LS}, T_{BA} \) and \( T_{\Sigma} \) denote the track generation, local incremental SfM, full bundle adjustment and total running time respectively.

| Data-set | \( N_f \) | \( N_k \) | \( \bar{n}_{con} \) | \( \bar{x}_{proj} \) | \( t_i \) | \( T_{TGC} \) | \( T_{LS} \) | \( T_{BA} \) | \( T_{\Sigma} \) |
|----------|----------|----------|----------------|-------------|-------|------------|------------|------------|------------|
| City A   | 73.0k    | 2478     | 216.14         | 0.81        | 7.34  | 5.65       | 3.56       | 7.13       | 6.14       |
| City B   | 96.4k    | 635      | 234.31         | 1.18        | 6.24  | 0.84       | 0.94       | 1.14       | 0.84       |
| City C   | 170.1k   | 1684     | 196.45         | 0.82        | 6.76  | 2.94       | 2.34       | 3.54       | 2.81       |
| City D   | 69.1k    | 871      | 176.49         | 1.01        | 3.52  | 1.24       | 1.72       | 1.79       | 1.97       |

Figure 7: The SfM visual results of the city-scale data-sets. In each row, the figures from left to right zoom successively closer to the representative buildings. The delicate and vivid 3D reconstruction model is shown in the last figure.

6. Conclusion

In this paper, we propose a parallel pipeline able to handle accurate SfM problems far exceeding the memory of a single compute node. A graph-based camera clustering algorithm is first introduced to facilitate the scalable and parallel SfM processing by dividing the original problem into sub-problems and a motion averaging method taking the advantages of both incremental and global SfM methods is subsequently proposed to merge partial reconstructions into a global consistent reconstruction. Our pipeline is able to handle the city-scale SfM problem, particularly camera registration, of far greater size than the state-of-the-art methods in a highly scalable and parallel manner and is finally able to generate highly accurate and consistent camera geometry for the subsequent dense stereo reconstruction.
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.

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

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

[4] 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.

[5] A. Chatterjee and V. M. Govindu. Efficient and robust large-scale rotation averaging. In *ICCV*, 2013.

[6] O. Chum and J. Matas. Randomized ransac with t-d,d test. *Image and Vision Computing*, pages 448–457, 2002.

[7] 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.

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

[9] Z. Cui and P. Tan. Global structure-from-motion by similarity averaging. In *ICCV*, 2015.

[10] I. S. Dhillon, Y. Guan, and B. Kulis. Kernel k-means: Spectral clustering and normalized cuts. In *SIGKDD*, 2004.

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

[12] X. Duan, H. Sun, and L. Peng. Riemannian means on special euclidean group and unipotent matrices group. The Scientific World Journal, page 292787, 2013.

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

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

[15] A. W. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In *ECCV*, 1998.

[16] 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.

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

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

[19] V. M. Govindu. Robustness in motion averaging. In *ACCV*, 2006.

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

[21] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

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

[23] M. Havlena, A. Torii, J. Knopp, and T. Pajdla. Randomized structure from motion based on atomic 3d models from camera triplets. In *CVPR*, 2009.

[24] M. Havlena, A. Torii, and T. Pajdla. Efficient structure from motion by graph optimization. In *ECCV*, 2010.

[25] J. Heinly, J. L. Schnberger, E. Dunn, and J. M. Frahm. Reconstructing the world* in six days. In *CVPR*, 2015.

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

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

[28] B. Klingner, D. Martin, and J. Roseborough. Street view motion-from-structure-from-motion. In *ICCV*, 2013.

[29] 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.

[30] M. Lhuillier and L. Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. *PAMI*, 27(3):418–433, 2005.

[31] P. Lindstrom. Triangulation made easy. In *CVPR*, 2010.

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

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

[34] K. Mitra and R. Chellappa. A scalable projective bundle adjustment algorithm using the l1 infinity norm. In *ICVGIP*, 2008.

[35] P. Moulon and P. Monasse. Unordered feature tracking made fast and easy. In *CVMP*, 2012.

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

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

[38] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, 2006.

[39] O. Özyesil and A. Singer. Robust camera location estimation by convex programming. *CoRR*, abs/1412.0165, 2014.

[40] M. Pollefeys, D. Nistér, J. M. Frahm, A. Akbarzadeh, P. Morohoi, 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.

[41] 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.

[42] B. Resch, H. P. Lensch, O. Wang, M. Pollefeys, and A. S. Hornung. Scalable structure from motion for densely sampled videos. In *CVPR*, 2015.

[43] R. Roberts, S. N. Sinha, R. Szeliski, and D. Steedly. Structure from motion for scenes with large duplicate structures. In *CVPR*, 2011.

[44] J. L. Schönhuber and J.-M. Frahm. Structure-from-motion revisited. In *CVPR*, 2016.
[45] J. L. Schönberger, F. Radenović, O. Chum, and J. M. Frahm. From single image query to detailed 3d reconstruction. In CVPR, 2015.

[46] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In MSST, 2010.

[47] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring image collections in 3d. SIGGRAPH, 2006.

[48] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the world from internet photo collections. IJCV, 80(2):189–210, 2008.

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

[50] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In CVPR, 2008.

[51] C. Sweeney, T. Sattler, T. Hollerer, M. Turk, and M. Pollefeys. Optimizing the viewing graph for structure-from-motion. In ICCV, 2016.

[52] R. Toldo, R. Gherardi, M. Farenzena, and A. Fusiello. Hierarchical structure-and-motion recovery from uncalibrated images. CoRR, abs/1506.00395, 2015.

[53] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In Proceedings of the International Workshop on Vision Algorithms: Theory and Practice, ICCV, pages 298–372, 2000.

[54] K. Wilson and N. Snavely. Network principles for sfm: Disambiguating repeated structures with local context. In ICCV, 2013.

[55] K. Wilson and N. Snavely. Robust global translations with 1dsfm. In ECCV, 2014.

[56] C. Wu. SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). http://cs.unc.edu/~ccwu/siftgpu, 2007.

[57] C. Wu. Towards linear-time incremental structure from motion. In 3DV, 2013.

[58] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In CVPR, 2011.

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

[60] C. Zach, M. Klopfenstein, and M. Pollefeys. Disambiguating visual relations using loop constraints. In CVPR, 2010.
7. Appendix

7.1. Discussion

completeness threshold $\delta_c$. In our approach, we have one important parameter, namely the camera cluster completeness ratio $\delta_c$, which quantifies the degree cameras covered in one sub-problem are also covered in other problems. A large cluster completeness threshold $\delta_c$ means more redundant cameras existed in multiple camera clusters. That is a large $\delta_c$ guarantees a more robust SfM while at the expense of some efficiency. We can see the statistics from Table 5 that when the completeness threshold $\delta_c$ is greater than or equal to 0.5, the accuracy of the final SfM results is almost the same in terms of both the number of recovered cameras and the camera median position error measured in meters. We therefore choose $\delta_c = 0.7$ in this paper for a robust SfM.

Number of relative motions The traditional essential-matrix-based methods [2, 3, 17, 39] and trifocal-tensor-based methods [7, 17, 26, 36] rely on camera pairs or triplets with sufficient feature correspondences to compute the epipolar geometry or trifocal tensor and require sufficient constraints of feature correspondences for a robust estimation. Since we obtain relative motions from a cluster of cameras reconstructed by local incremental SfM, and the relative motion of every pair of cameras can be obtained within a camera cluster even though such a camera pair has few or no feature correspondences. We can subsequently apply much more relative motions than the state-of-the-art methods to the motion averaging framework and subsequently obtain more robust global camera geometry. Table 4 shows the comparison of the number obtained relative motions between the state-of-the-art essential-matrix-based method [36] and our method. Our method can obtain much more redundant relative poses than the state-of-the-art method [36].

Cameras with Forward Motion We know that the traditional essential-matrix-based methods [2, 3, 17, 39] generally generate ill poses in col-linear camera motion, since essential matrices only encode the direction of a translation. Thanks to the accurate and robust relative motions from local incremental SfM, our proposed method can effectively handle the estimation of camera geometry from images in forward motion. The Street data-set is consisted of 1102 images and all the images capture the street site in both forward and back motion.

Fusion of Street-view and Aerial Images Finally, we provide an especially challenging data-set named University Campus. The University Campus data-set contains 12082 street-view and aerial images of 12 megapixels and the connections between aerial and street-view images are very weak with only a few matching correspondences. We find that the state-of-art methods [36, 51] both failed to recover the camera poses of aerial and street-view images as a whole. It is mainly because both the essential-matrix-based methods [2, 3, 17, 39] and trifocal-tensor-based methods [7, 17, 26, 36] tend to generate ill relative poses between images with large scale variance. While, our pipeline successfully reconstruct the camera geometry of both aerial and street-view images in a consistent manner and the experimental results further demonstrate the robustness of incremental SfM in pose estimation even for images with weak feature correspondences and large scale variance. The visual results are shown in Figure 11.

| Data-sets          | # images | # relative motions for motion averaging |
|--------------------|----------|----------------------------------------|
|                    |          | Essential-matrix-based method [36] | Our method |
| Alamo              | 577      | 6410                                   | 62357      |
| Ellis Island       | 227      | 6963                                   | 21178      |
| Metropolitan       | 341      | 5993                                   | 27822      |
| Montreal N.D.      | 450      | 14982                                  | 58869      |
| Notre Dame         | 353      | 4823                                   | 63211      |
| NYC Library        | 332      | 7561                                   | 30353      |
| Piazza del Popolo  | 328      | 8349                                   | 34261      |
| Piccadilly         | 2152     | 500445                                 | 343232     |
| Roman Forum        | 1084     | 21764                                  | 124728     |
| Tower of London    | 572      | 10135                                  | 38287      |
| Union Square       | 789      | 13088                                  | 51794      |
| Vienna Cathedral   | 836      | 22708                                  | 127411     |
| Yorkminster        | 437      | 8297                                   | 35406      |

Table 4: The comparison of the number obtained relative motions between the state-of-the-art essential-matrix-based method [36] and our method. Our method can obtain much more redundant relative poses than the state-of-the-art method [36].
Table 5: The comparison of the Internet data-sets [55] given different choices of the completeness threshold. We regard the SfM results of [47] as the reference model. $N_c$ denotes the number of recovered cameras in SfM, $\bar{x}$ is the median position error measured in meters after motion averaging and before bundle adjustment, and $\bar{x}_{BA}$ represents the median position error in meters after bundle adjustment.

| Data-sets          | $\delta_c$ | # images | $N_c$ | $\bar{x}$ | $\bar{x}_{BA}$ | $N_c$ | $\bar{x}$ | $\bar{x}_{BA}$ | $N_c$ | $\bar{x}$ | $\bar{x}_{BA}$ | $N_c$ | $\bar{x}$ | $\bar{x}_{BA}$ |
|--------------------|------------|----------|-------|-----------|----------------|-------|-----------|----------------|-------|-----------|----------------|-------|-----------|----------------|
| Alamo              | 0.1        | 577      | 550   | 0.91      | 0.21           | 549   | 0.93      | 0.20           | 548   | 0.96      | 0.24           | 541   | 1.12      | 0.21           |
| Ellis Island       | 0.3        | 227      | 221   | 2.67      | 0.46           | 221   | 2.66      | 0.46           | 220   | 2.69      | 0.49           | 220   | 2.42      | 0.47           | 202   | 4.12      | 2.18           |
| Metropolis         | 0.5        | 341      | 299   | 3.14      | 0.23           | 298   | 3.17      | 0.24           | 297   | 3.17      | 0.29           | 286   | 3.54      | 0.43           | 211   | 4.12      | 3.12           |
| Montreal N.D.      | 0.7        | 450      | 445   | 1.79      | 0.31           | 445   | 1.78      | 0.30           | 440   | 1.76      | 0.32           | 436   | 1.95      | 0.54           | 398   | 2.54      | 1.87           |
| Notre Dame         | 0.9        | 553      | 513   | 1.41      | 0.15           | 514   | 1.42      | 0.16           | 511   | 1.33      | 0.19           | 510   | 1.52      | 0.31           | 489   | 3.24      | 1.32           |
| NYC Library        | 0.3        | 332      | 290   | 2.75      | 0.26           | 290   | 2.78      | 0.26           | 287   | 2.89      | 0.29           | 286   | 2.75      | 0.54           | 267   | 3.05      | 1.89           |
| Piazza del Popolo  | 0.5        | 328      | 332   | 2.60      | 0.42           | 334   | 2.62      | 0.45           | 330   | 2.79      | 0.42           | 331   | 2.94      | 0.41           | 194   | 3.52      | 1.35           |
| Piccadilly         | 0.7        | 2152     | 2114  | 2.11      | 0.37           | 2114  | 2.10      | 0.38           | 2112  | 2.89      | 0.39           | 2107  | 2.91      | 0.49           | 1312  | 3.94      | 1.23           |
| Roman Forum        | 0.9        | 1084     | 1078  | 2.90      | 0.36           | 1079  | 2.91      | 0.38           | 1078  | 2.83      | 0.42           | 1071  | 3.21      | 0.92           | 988   | 4.22      | 0.78           |
| Tower of London    | 0.3        | 572      | 458   | 3.22      | 0.99           | 458   | 3.23      | 1.00           | 459   | 3.10      | 1.13           | 442   | 3.43      | 1.92           | 314   | 4.10      | 1.84           |
| Union Square       | 0.5        | 789      | 721   | 1.79      | 1.53           | 720   | 1.78      | 1.52           | 717   | 1.97      | 1.63           | 711   | 1.82      | 1.71           | 532   | 2.49      | 2.32           |
| Vienna Cathedral   | 0.7        | 836      | 793   | 1.30      | 0.50           | 793   | 1.32      | 0.51           | 790   | 1.39      | 0.64           | 786   | 1.42      | 0.71           | 591   | 3.91      | 1.98           |
| Yorkminster        | 0.9        | 437      | 408   | 0.56      | 0.24           | 407   | 0.55      | 0.25           | 406   | 0.61      | 0.29           | 400   | 0.85      | 0.52           | 390   | 2.14      | 0.78           |

Figure 8: The comparison of the Pittsburgh data-set. The visual results of the methods [9, 39, 57] are from the work [9]. We can see from the red ellipses that the works [39, 57] failed to recover the close loop structure of camera geometry.

Figure 9: The comparison of the Campus data-set. The visual results of the methods [9, 39, 57] are from the work [9]. We can see from the red ellipses that the works [39, 57] failed to recover the close loop structure of camera geometry. We also note from the green rectangles that the 3D sparse points of a road recovered by the work [9] is distorted while our method perfectly recover the sparse 3D points of the straight road.

Figure 10: The visual SfM results of the Street data-set from our pipeline.
Figure 11: The visual SfM results of the University Campus data-set from our pipeline.