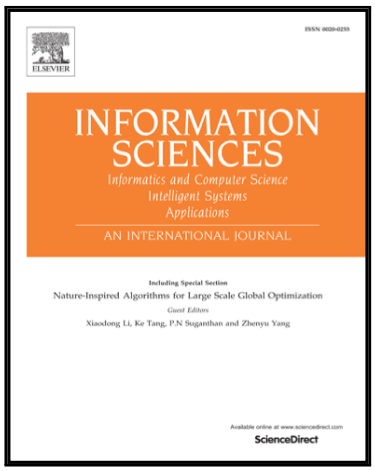

A Center-Driven Image Set Partition Algorithm for Efficient Structure from Motion

Kun Sun, Wenbing Tao

PII: S0020-0255(18)30942-3
DOI: https://doi.org/10.1016/j.ins.2018.11.055
Reference: INS 14099

To appear in: *Information Sciences*

Received date: 6 August 2018
Revised date: 23 November 2018
Accepted date: 25 November 2018

Please cite this article as: Kun Sun, Wenbing Tao, A Center-Driven Image Set Partition Algorithm for Efficient Structure from Motion, *Information Sciences* (2018), doi: https://doi.org/10.1016/j.ins.2018.11.055

A Center-Driven Image Set Partition Algorithm for Efficient Structure from Motion

Kun Sun[a], Wenbing Tao[b,*]

[a] *Hubei Key Laboratory of Intelligent Geo-Information Processing, School of Computer Science, China University of Geosciences, Wuhan 430074, China.*
[b] *National Key Laboratory of Science and Technology on Multi-spectral Information Processing, School of Automation, Huazhong University of Science and Technology, Wuhan 430074, China.*

Abstract

This paper proposes a novel center-driven image set partitioning method dedicated for efficient Structure from Motion (SfM) on unevenly distributed images. First, multiple base clusters are found at places with high image density. Instead of building a small initial model from two images, we build multiple initial base models from these base clusters. This promises that the scene is reconstructed from dense places to sparse areas, which can reduce error accumulation when images have weak overlap. Second, the whole image set is divided into several region clusters to decide which images should be reconstructed from the same base model. In this step, the base models are treated as centers and the affinity between an image with each of them is measured by the reconstruction path length. To enable faster speed, images in each region cluster are further divided into several sub-region clusters so that they could be added to the same base model simultaneously. Based

*Corresponding author
Email addresses: sunkun@cug.edu.cn (Kun Sun), wenbingtao@hust.edu.cn (Wenbing Tao)

on the above partitioning results, the partial 3D models are reconstructed in parallel and then merged. Experiments show that the proposed method achieves remarkable speedup and better completeness than state-of-the-art methods, without significant accuracy deterioration.

Keywords:
center-driven, image set partitioning, 3D reconstruction, Structure from Motion.

1. Introduction

Investigating 3D information assists many applications in computer vision [41, 40, 48, 37, 20, 11, 18]. Structure from Motion (SfM) is widely used in reconstructing 3D camera poses and sparse point cloud from unordered images. With the rapid development of the Internet, these images can be easily searched and downloaded through keywords. However, due to the large scale nature of such problems, accuracy and efficiency are still two most challenging issues.

Existing SfM methods can be divided into three classes: incremental [34, 32], global [42, 6, 7] and hybrid [5, 50]. This paper mainly focuses on the first type. A typical incremental SfM pipeline consists of three steps: 1) Constructing scene graph via image matching and geometry verification. 2) Selecting two starting images and build an initial model for the incremental process. 3) Adding new images to the existing model and run Bundle Adjustment (BA) [39] to refine parameters. The last step is repeated until no more images could be added.

Some of these methods perform in a top-down manner. A coarse model

which spans the whole scene is reconstructed as quickly as possible in the first stage and then it is enriched in the second stage. Snavely *et al.* extracted a skeletal graph [33] that covers the full scene with the minimum number of interior nodes. Leaf nodes can be added after the skeletal set is reconstructed. This method is further used in [2], which designed a system running on a distributed cluster to efficiently reconstruct a city in one day. The concept of using iconic scene graphs to capture the major aspects of the scene is proposed in [19] and [27]. After clustering images in the GIST [25] feature space, they selected an iconic image for each cluster. The viewing graph formed by iconic images is computed via vocabulary tree [24] indexing. Frahm *et al.* [13] improved the work of [2] by reconstructing a city on a single machine with multi-core CPUs and GPUs. When selecting iconic images the image descriptors were compressed to shorter binary codes so that it is memory efficient for GPU computation. Heinly *et al.* [16] advanced the state-of-the-art from city-scale modeling to world-scale modeling on a single computer. They also leveraged the idea of iconic images. The database-side feature augmentation is applied so that an iconic image can cover a broader set of views. For the ability to handle world scale images, their system stores an image in memory only when it is needed. COLMAP [28] improved several components of the state-of-the-art methods, such as geometry verification, view selection, triangulation and bundle adjustment to make a further step towards a robust, accurate, complete and scalable system. Havlena *et al.* [15] computed a minimal connected dominating set to reduce the input images. In order to get a good reconstruction in shorter computational time, the reconstruction pipeline follows a task queue ordered by priority. Shah

et al. [29] carried out hierarchical SfM without using iconic images. Motivated by the preemptive matching strategy [46], a coarse yet global model is quickly reconstructed using high scale SIFT feature correspondences in the first stage. This model offers useful geometric constraints for the second stage, in which the model is enriched by localizing unreconstructed images and triangulating remaining features.

Some other methods perform in a bottom-up manner. They firstly reconstruct several partial models and then gradually merges them. Fitzgibbon and Zisserman [12] used image triplets to build partial models. For denser images, Shum *et al.* [31] built a few local models from each video segment by applying long baseline two-frame SfM and motion interpolation. However, the triplets are always consecutive frames. Nistér [23] used a trifocal tensor tree to select appropriate baseline for the triplets to remove redundancy. Based on this work, Fang and Quan [9] used quasi-dense matches between consecutive frames. Instead of building local models from two or three consecutive views, some other methods do this within a larger set. Douterloigne *et al.* [8] directly divided the scene based on the order in which images were taken. Graph-based clustering methods such as spectral clustering [35] and k-way graph cut [22] are also widely used. Normalized Cuts [30] is used in [3] for efficient large scale SfM. All the partial models are merged via the cutting edges. Bottom-up reconstruction can be also performed on a tree. Farenzena *et al.* [10] built an aggregative clustering tree, whose leaves are images and internal nodes correspond to partial reconstructions. The reconstruction starts from sibling leaf pairs and gradually merges upwards. However, such a tree might be highly unbalanced. Toldo *et al.* [38] improved this work by

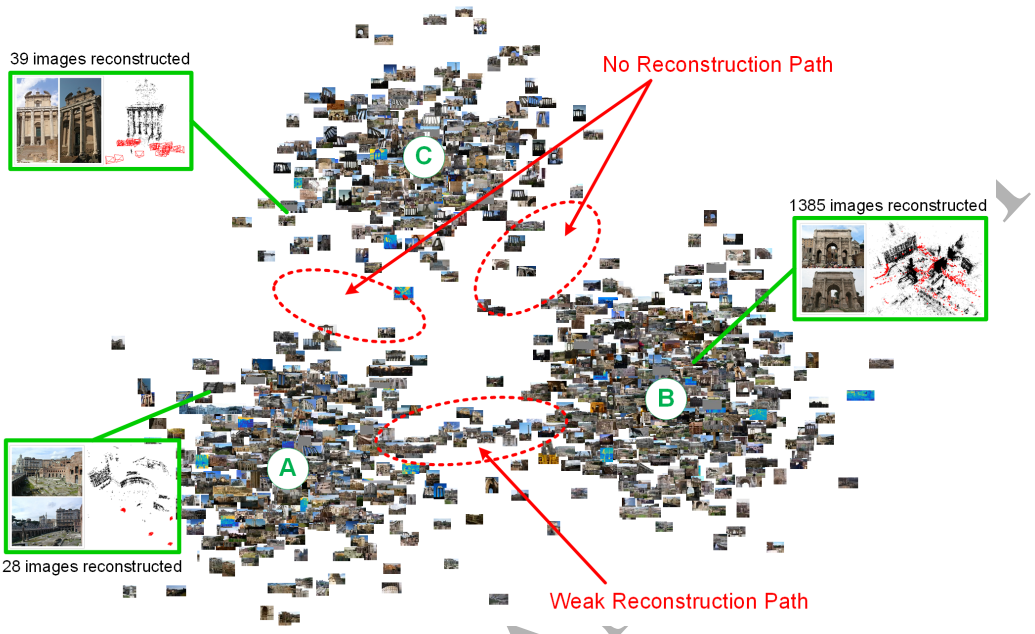

Figure 1: The distribution of Internet images used for SfM. A, B and C are three places where images are densely distributed. A and B are connected by a weak reconstruction path. But there is no sufficient overlap between AC and BC. The reconstruction result for traditional SfM will be quite different when starting from different places.

proposing a balanced hierarchical clustering tree. At each merging step, they selected top k nearest neighbors and then merged to the one with the smallest cardinality. Differently, Chen *et al.* [4] designed a tree whose leaves are atomic models reconstructed from pairwise geometry, and partial models are merged as a node in a higher layer. In [36], each partition obtained by Normalized Cuts is treated as a generalized camera [26, 49], whose pose will be estimated by the incremental SfM algorithm. It's worth to mention that top-down methods and bottom-up methods are sometimes used in combination with each other when the number of images is large.

Although great achievements have been made, the challenges brought by unevenly distributed images are somewhat ignored. Images downloaded from the Internet are taken freely by different consumers. They are crowded at some sites but sparse at other places in the scene. Fig. 1 is an example by mapping their positions on a 2D manifold. Images are dense at three places A, B and C, while they are sparse at other places. There is a weak reconstruction path formed by some images with small overlap between A and B. Traditional incremental methods will start to reconstruct the scene from a pair of images [34, 32]. However, this strategy is sometimes unstable since the relationship between the starting point and the other images are neglected. As shown in Fig. 1, if the reconstruction starts from different places, the results will be quite different. Besides, passing 3D structure between A and B along the weak reconstruction path will result in large accumulation error or even fail in the worst case. In terms of efficiency, when reconstructing from sparse areas the model will grow slower and more efforts will be paid in solving the optimization problem since the feature tracks may contain a high ratio of outliers. To address these problems, several proper starting points should be selected automatically according to the distributional properties of the dataset. Besides, more than two images should be used to suppress errors caused by inaccurate epipolar geometry when building the initial model. Next, which part of the scene should be reconstructed from the same starting point should be determined. This leads to a data partitioning problem. Aforementioned bottom-up methods usually partition the images by connected components, K-Means or Normalized-Cuts. The main drawback of these methods is that the starting point selected within each subset may not

be an optimal solution consistent with the global distribution. We call them "blind partitioning" because the starting points are not known in advance. Differently, our purpose is to divide the images by treating known starting points as centers. Compared with "blind partitioning", this "center-driven clustering" method promises that each sub-SfM procedure starts from a place with better global properties.

In this paper, a novel center-driven image set partitioning method dedicated for efficient SfM on unevenly distributed images is proposed. The whole image set is divided into three kinds of clusters: base clusters, region clusters and sub-region clusters. Base clusters are firstly extracted from the image set. A base cluster contains several images around the image density center. It will be used to build a base model of the scene, from which an incremental SfM process begins. Since images around the density center have sufficient overlap with each other, the base clusters have stronger ability to build reliable initial models and propagate 3D structure to nearby places. Next, to determine which part should be reconstructed from the same base model, the remaining images are partitioned into region clusters by treating the base models as centers. The affinities between an image with all the centers are measured by the reconstruction path length, and the winner-take-all strategy is used to group them. However, adding images in each region cluster to the base model sequentially might be still time-consuming. In order to further improve speed, each region cluster is divided into several sub-region clusters so that they can be added to the same base model in parallel. This performs as if the region is reconstructed from the same starting point to different directions. The relationship between these three kinds of clusters

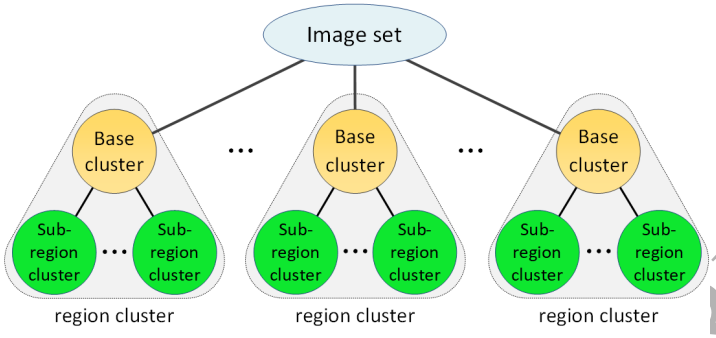

Figure 2: The tree representing the partitioning result. Base clusters act as the starting points. A region cluster contains one base cluster and several sub-region clusters. The scene is reconstructed in parallel not only between region clusters but also within each region cluster.

can be represented by a tree, which is shown in Fig. 2. Building a 3D model then needs two stages: reconstruction and merging. The reconstruction stage reconstructs the base model first and then adds the sub-region clusters to it. The merging stage performs inversely, which merges the sub-region models to a region model first and then merge different region models into a complete one.

The contribution of this work can be summarized in the following aspects. 1) We find multiple starting points according to the distribution property of the dataset. At each starting point we build a base model from a set of images with large overlap, which enforces global stability and reliability of the initial models. 2) A "center-driven" data partitioning method is proposed to divide the images into region clusters according to their reconstruction paths. Compared with "blind partitioning", it ensures that each partition could be reconstructed from a starting point that has better global properties. 3) A

balanced path tree partition method is proposed to further divide a region cluster into several sub-region clusters. Not only the region clusters but also the sub-region clusters could be reconstructed efficiently in parallel.

The remainder of this paper is organized as follows. In Section 2, the proposed center-driven data partitioning method is introduced. How to reconstruct 3D models from the partitioning results is presented in Section 3. Experiment results and analysis are given in Section 4. Finally this paper is concluded in Section 5.

2. The Proposed Center Driven Data Partitioning Method

2.1. Preparation and Overview

Suppose we have a set of unordered images $\boldsymbol{I} = \{I_i\}_{i=1}^{N}$. First of all, a fast GPU implementation [44] is used to extract SIFT [21] features and match them. Wrong matches are filtered out by estimating the epipolar geometry between two views using the RANSAC algorithm. Next, two kinds of matching graphs are constructed: a similarity graph $\boldsymbol{S}$ and a difference graph $\boldsymbol{D}$. The matching graph $\boldsymbol{G} < \boldsymbol{V}, \boldsymbol{E} >$ is an undirected weighted graph with a set of vertexes $\boldsymbol{V}$ and edges $\boldsymbol{E}$. A vertex v_i represents an image. If there is scene overlap between two images, an edge will be added between the corresponding vertexes. Both $\boldsymbol{S}$ and $\boldsymbol{D}$ have the same number of vertexes and edges, but the meaning of their edge weights are different. In the similarity graph $\boldsymbol{S}$, the edge weight s_{ij} reflects the content similarity between two images. An intuitive way is to measure this similarity with the number of matches between two images. However, it is sensitive to image resolution and texture. High resolution or textured images will have more matches than

low resolution or less textured images. In this paper, s_{ij} is computed from:

$$s_{ij} = \frac{n_{ij}}{n_i \cup n_j}, \tag{1}$$

in which n_{ij} is the number of matches between two images I_i and I_j, n_i and n_j are the number of feature points on image I_i and I_j that have corresponding points on the other images, respectively. Eq. (1) is also known as the Jaccard similarity coefficient. A larger s_{ij} indicates that I_i and I_j have larger scene overlap. The weights of the difference graph $\boldsymbol{D}$ are then computed from:

$$d_{ij} = 1 - s_{ij}. \tag{2}$$

The flowchart of the proposed method is shown in Fig. 3. In Fig. 3(a) three base clusters (black) are found as starting points first. Next, all the images are divided into three region clusters A (yellow), B (green) and C (cyan) according to their reconstruction path length to the starting points. Fig. 3(b) shows that images in a region cluster are split into several sub-region clusters to enable faster reconstruction speed. The reconstruction path from each image to the starting point should lie within the same sub-region cluster, so that the starting point could propagate 3D structure to each sub-region cluster independently without distinct accuracy deterioration. After finishing the above steps, the base clusters are first reconstructed in parallel to get several base models, which is shown in Fig. 3(c). Then several sub-region cluster models are reconstructed by adding them to the same base model simultaneously. As shown in Fig. 3(d), the base model is enriched from different directions. Since these partial models share the same base model, it's easy to merge them to get the model of each region cluster, which

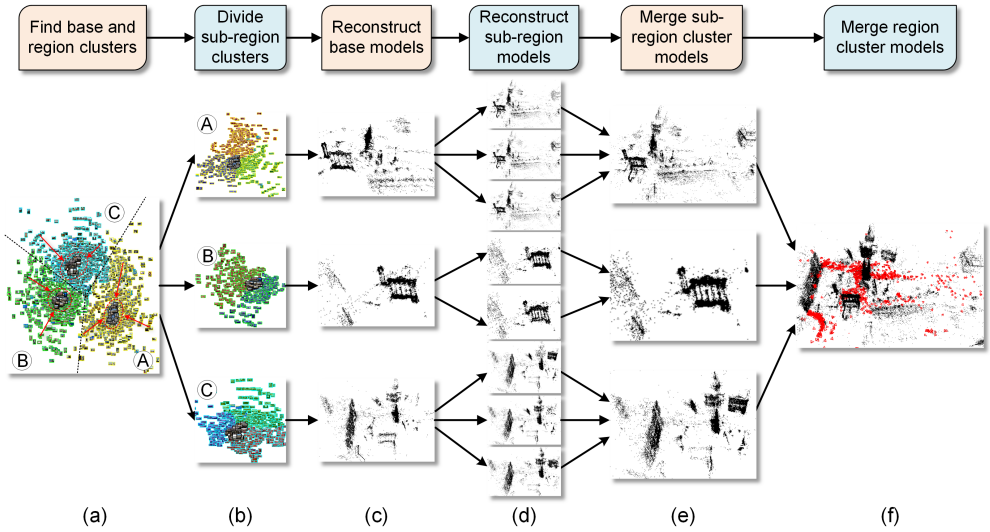

Figure 3: The flowchart of the proposed method. (a) Three base clusters and region clusters are found. (b) Each region cluster is divided into several sub-region clusters. (c) The base models reconstructed from the base clusters. (d) Reconstruction results after adding each sub-region cluster to the base models. (e) The model of each region cluster after merging the partial models in (d). (f) The final result after merging the models of different region clusters.

is shown in Fig. 3(e). Fig. 3(f) shows the final model acquired by merging different region cluster models.

2.2. Finding Base Clusters by Distribution Property Inferring

Base clusters are used to reconstruct base models of the scene. They should be found at places where images are densely distributed and are not expected to contain too many images. Images at such places have large scene overlap between each other, so the base models are accurate. In this part, we use a loose greedy manner to progressively find multiple base models.

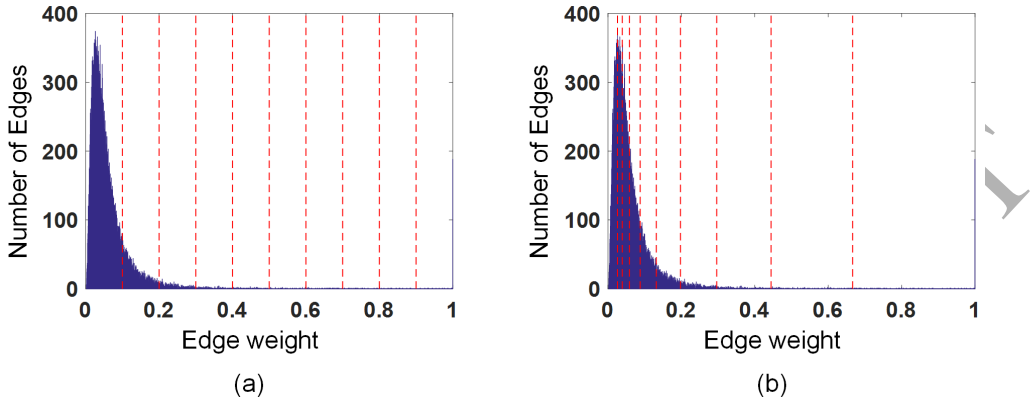

Figure 4: The distribution of edge weights in the similarity graph. (a) Uniform intervals and (b) Non-uniform intervals.

Suppose the ideal size of a base cluster lies within $[m, \alpha m]$, where m is a positive number and $\alpha \geq 1$ is an inflation factor. We divide the similarity graph $\boldsymbol{S}$ into multiple layers and find base clusters by traversing them. Given the number of layers k and a set of edge weight thresholds $\theta_i (i \in 1, 2, \ldots, k)$ satisfying $\theta_i > \theta_{i+1}$, the i^{th} layer contains edges whose weights are greater than θ_i. Then we find connected components in this layer. If none of them is larger than m, we move forward to the next layer, relaxing θ_i and adding more edges. If a connected component is larger than m but smaller than αm, the images in this component form a new base cluster and the corresponding vertexes are removed from the current graph. If a connected component is larger than αm, we will recursively divide the connected components into multiple layers until a base cluster with proper size is returned. In this way, each newly found base cluster is a set of images which have the strongest overlap between each other among the current remaining images.

Computing θ_i is a non-trivial task. Denote the minimum and maximum edge weights in the similarity graph as E_a and E_b, respectively. In practice, E_a is set to a truncation threshold $max(\varepsilon, min(s_{ij}))$ so that images having too weak overlap with others are not considered in this stage. The range $[E_a, E_b]$ is divided by a set of decreasing thresholds $\theta_i (i \in 1, 2, \ldots, k)$. Fig. 4 shows the distribution of all the edge weights in the similarity graph. It can be seen that there is a peak near 0.02. If the intervals are divided uniformly (Fig. 4(a)), lower intervals will contain more edges compared with higher intervals. As a result, it is difficult to find large enough connected components at the first few layers but will fall into deep recursion in higher layers because of rapid growing of the connected component. In this paper, θ_i is computed from the following formulation:

$$\theta_i = E_a + \frac{E_b - E_a}{1.5^{i-1}}, i \in 1, 2, \ldots, k. \tag{3}$$

As we can see from red lines in Fig. 4(b), such a division can keep the number of edges in each interval roughly equal.

The base clusters found will be used to reconstruct several base models via incremental SfM. Before that, we need to identify which image is the first one to be added in each base cluster. The Affinity Propagation (AP) clustering algorithm [14] is applied to images in each base cluster, and all the cluster centers are treated as the candidates of the first image. The affinity matrix required by the AP clustering algorithm is computed from Eq. (1). The reason for choosing AP clustering is two-fold. On the one hand, AP clustering algorithm can automatically determine the number of clusters. On the other hand, the center of a cluster is one data point instead of a virtual mean position. In practice, we expand the candidate set by adopting

Figure 5: An example of four base clusters selected from the Roman Forum dataset [43]. The image with red box is the first image to be added.

the adjacent neighbors of the centers on the similarity graph $\boldsymbol{S}$. For each candidate image, the following score is computed:

$$\delta(v) = h_{deg}(v) + \beta_1 \cdot h_{sim}(v) + \beta_2 \cdot h_{ndeg}(v). \tag{4}$$

The first term $h_{deg}(v)$ is the degree of the vertex v, which counts the number of images that overlap with it. It encourages v to overlap with as many images as possible. The second term $h_{dist}(v)$ is the average similarity from the vertex v to its neighbors, namely the mean adjacent edge weight on $\boldsymbol{S}$. This term encourages v to have large overlap with its neighbors. The last term $h_{ndeg}(v)$ is the average degree for the neighbors of v. That is to say, not only v itself should overlap with many images, but also the images overlapping with it

should also overlap with as many other images as possible. This strengthens the potential to spread 3D structure to nearby places. Finally, the image with the highest score is selected as the first image. Fig. 5 shows four base clusters found from the Roman Forum dataset [43]. The image with red box is the selected first image to be added when reconstructing base models. The second image to be added by the incremental SfM is selected by traversing the other images in this base cluster and finding the one who has the most matches as well as the widest baseline with the first image [34].

2.3. Region Cluster Partitioning according to Reconstruction Path

When reconstructing the whole scene from different base clusters in parallel, which part is to be reconstructed from the same base cluster should be decided in advance. However, it is not a general similarity-based classification problem. Indeed, this is a center-driven data partitioning problem, in which the partitioning strategy is more tightly coupled to the selected base clusters. In this part, a method that treats the base clusters as centers and groups the images according to their reconstruction path length to the centers is introduced. A reconstruction path between image A and B should meet two basic requirements: 1) It is connected, that is, 3D structure could propagate between A and B through this path; 2) It is compact, which means the overlap between two adjacent intermediate images should be as large and uniform as possible. This can be achieved by minimizing the maximum difference between adjacent images on such a path. As is shown in Fig. 6, usually there might be more than one "route" between A and B. Smaller edge weight indicates smaller scene difference and larger scene overlap. The red path has shorter length than the green path. However, it is not consid-

Figure 6: An illustration of the reconstruction path (gree path) in the proposed method.

ered as the reconstruction path because the overlap between adjacent images varies a lot. There is an edge whose weight (0.66) is much larger than the other two edges (0.14 and 0.18). This means that the 3D structure has to be propagated via relatively weak image overlap, which is unstable. Although the total length of the green path is longer than the red one, the edge weights on it are small and similar to each other. If the green path is selected as the reconstruction path, the risk of passing 3D structure via weak overlap will be reduced.

To fulfill the above task, a Multi-layer Shortest Path (MSP) algorithm is proposed to find the reconstruction path from each image to the base clusters. The MSP algorithm operates on the difference graph $\boldsymbol{D}$, in which the edge weight indicates the scene difference between images. At the beginning, each base cluster is initialized as a region cluster. The difference graph is divided into L layers by another set of weight thresholds $\phi_i (i \in 1, \ldots, L)$ satisfying $\phi_i < \phi_{i+1}$. More specifically, the range of edge weights $[\min(d_{ij}), \max(d_{ij})]$ on $\boldsymbol{D}$ is divided into L homogeneous intervals. For each interval the step length is $l = (\max(d_{ij}) - \min(d_{ij}))/L$ and ϕ_i is computed from

$$\phi_i = i * l + \min(d_{ij}), i = 1, \ldots, L. \tag{5}$$

Edges whose weights are smaller than ϕ_i are added to the i^{th} layer. For an image w in none of the base clusters, we find its shortest paths to all the images in a base cluster. The one whose length is the shortest is treated as the reconstruction path between w and this base cluster. In a certain layer, the reconstruction paths from w to all base clusters are computed in that way. If no reconstruction paths are found in this layer, we then add more edges by using a larger edge weight threshold and repeat the steps in the next layer. Otherwise, w will be put into the same region cluster with the base cluster who has the smallest reconstruction path length. Once an image w has been clustered, it will not be handled in the remaining layers. But the vertex corresponding to it is not removed from the graph because it may be on the reconstruction path of other unclustered images. In this way, the green path in Fig. 6 will be found before the red path. After all the images are clustered, we get a set of region clusters. Each region cluster will be reconstructed from the base model in it by sequentially adding the remaining images.

2.4. Dividing Sub-region Clusters via Balanced Path Tree Partitioning

So far all the region clusters can be reconstructed in parallel. However, sometimes the number of images in a region cluster might be still too large and adding them sequentially to the base model is time still consuming. In this section, a large region cluster will be further divided into several sub-region clusters so that each of them could be added to the same base model in parallel without distinct accuracy deterioration. Specifically, the splitting should satisfy three requirements. (1) Images within each sub-region cluster should have considerable overlap with each other, so that 3D structure

could propagate within this subset. (2) Each sub-region cluster should have strong overlap with the base cluster so that it can be added to the base model. (3) These sub-region clusters should be balanced in size to reduce the synchronization overhead between different threads.

Inspired by [17], a novel Balanced Path Tree Partition method is proposed in this part. For each image in a region cluster, its reconstruction path to the base cluster described in Section 2.3 is recorded. All these reconstruction paths start from different images but have the same end, *i.e.* the base cluster. A path graph formed by all the reconstruction paths in a region cluster is constructed. It is then converted to a tree $\boldsymbol{T}$ whose root is the base cluster by applying the Minimum Spanning Tree (MST) algorithm. In fact, it can be proved by means of reduction to absurdity that if the edges are not equally weighted, the MST is the path graph itself. This is because if there are two different paths P_1 and P_2 between image A and B on the path graph, then the one satisfying

$$\arg_P min(max(P_1), max(P_2), \tag{6}$$

will be found earlier than the other one in our MSP algorithm (Section 2.3). In Eq. (6) $max(P_1)$ and $max(P_2)$ are the largest edge weight in P_1 and P_2, respectively.

Denote T_p as the subtree rooted at node p, S_p as the sons of node p, W_p as the size of T_p and s as the ideal size of each sub-region cluster. The rooted $\boldsymbol{T}$ is partitioned by gradually removing a subtree. Specifically, the tree is traversed from bottom to top level by level. If $W_p > s$, it goes down to a lower level to check the subtrees rooted at the sons of node p. By solving a simplified knapsack problem, several subtrees whose total size is close to

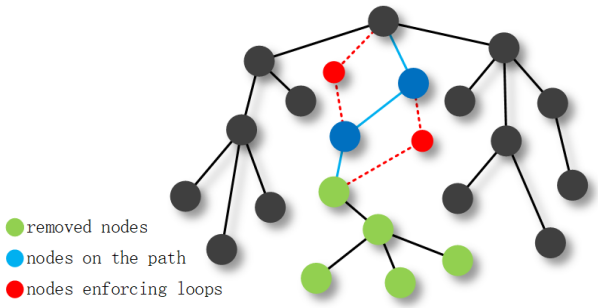

Figure 7: Two kinds of augmentation in our method. The nodes to be removed are in green. First augmentation: nodes on the path from the subtree to the root (in blue) are added. Second augmentation: nodes not in the current sub-region cluster (in red) are added to enforce loop consistency of the path.

but less than s are selected. They are pruned as a sub-region cluster. The remaining tree is pruned in the same way repeatedly until its size is no bigger than s. In this way, the sub-region clusters are nearly equal in size, except for the last one.

However, sub-region clusters found in the earlier stages might be at lower levels and far from the root. In this case, it's hard to directly propagate 3D structure from the base model to them. So we need to augment the sub-region clusters and enhance its linkage to the base cluster (or root). As is shown in Fig. 7, this step involves two kinds of augmentation. First, when removing a subtree from T_p, all the nodes on the path from this subtree to the root are added to its corresponding sub-region cluster. Second, if there is no loop between two nodes on the path from this subtree to the root, a new image not in the current sub-region cluster is added to enforce loop consistency. For efficiency, only 3-step and 4-step loops are considered here. The complete

Algorithm 1 Balanced Path Tree Partition

Input: A region cluster and a set of reconstruction paths

Output: A set of sub-region clusters

```
Build a path graph
Find the Minimum Spanning Tree (MST)
Root the MST at the base cluster in this region cluster
for i = max − level to root in T do
    for each node p in the i^th level do
        while W_p > s do
            Compute the sizes of the subtrees rooted at S_p
            Select a set of subtrees whose total weight w̃ satisfies w̃ → s^+
            Remove these subtrees as a sub-region cluster from T
            Augmentation 1: add nodes on the path from root to these subtrees
            Augmentation 2: add nodes not in this sub-region cluster to enforce loop
            consistency
            Update T_p, W_p and S_p
        end while
    end for
end for
Return all the sub-region clusters
```

Balanced Path Tree Partition algorithm is summarized in Algorithm 1.

3. Model Reconstruction and Merging

Given the partitioning results, 3D models are reconstructed from the base clusters to the sub-region clusters. Each base cluster is reconstructed with a standard incremental SfM pipeline on an independent thread. Afterwards, the sub-region clusters in the same region cluster will be added simultaneously to the base model reconstructed in the previous step, producing sev-

eral sub-region models. The pose of each newly added image is initialized by solving the PnP problem and then refined via bundle adjustment. In our implementation, BA is performed on multi-core CPUs using Ceres [1] when the number of reconstructed cameras is less than 20. Otherwise BA is carried out using [47] on a GPU card. This hybrid strategy is a trade off between accuracy and efficiency.

The reconstruction can be very fast if we have enough hardware resources such as CPU cores and GPU cards, so that all the base clusters and sub-region clusters can be truly reconstructed in parallel. The complexity of our method is relevant to the ideal base cluster size m and the ideal sub-region cluster size s, which is $O(m+s)$. A state-of-the-art linear-time SfM algorithm [46] has a complexity of $O(N)$, where N is the number of reconstructed cameras. Our method has a theoretical speedup factor of $\frac{N}{m+s}$. When m or s increases, the computational efficiency will decrease. However, small m may result in inaccurate base models and small s will lead to over segmentation of the scene. So in practice we recommend not to set them too small. For very large image sets whose difference between N and $m+s$ is large, the speedup will be more significant.

The reconstructed partial models are merged in the following steps. First, the sub-region models in the same region cluster are merged to a region model. This is not difficult because they share the same base model. Next, different region models are merged to a complete scene model. There have been several methods to fuse independent models into a global one. Although using high-level optimization [50] will lead to better merging result, we find that simply estimating a 3D similarity transformation from the common parts

can produce satisfactory results.

One of the difficulties when merging different region models is to detect the common parts between them. Since the same feature track reconstructed in different models may be inconsistent, directly finding common 3D points between models according to shared feature tracks will include a very large portion of outliers. In this paper we narrow down the number of suspicious common 3D points and then estimate the transformation in a RANSAC framework. Specifically, consider two models M_1 and M_2, our method first finds an image in M_2 who has the most feature tracks reconstructed in M_1. Then the feature tracks on this image who have also been reconstructed in M_2 are counted. If the number of such feature tracks is greater than a threshold τ, the corresponding points are used to estimate the transformation between M_1 and M_2. Otherwise nothing will be done. A practical trick is to merge smaller models to larger models, which can reduce numerical error accumulation. If a model overlaps with several other models, then it is merged to the one with the most overlapping 3D points in order to achieve a more accurate estimation of the transformation.

4. Experiment Results

4.1. Parameter Settings

This part introduces the settings for some key parameters in our method. When finding base clusters, edges in the similarity graph whose weights are less than $\varepsilon = 0.1$ are not considered. In Eq. (3), we divide the range of edge weights into $k = 15$ intervals. The size of a base cluster lies between m and αm. m and α are computed respectively by $m = min(60, 0.15 * Z)$ and

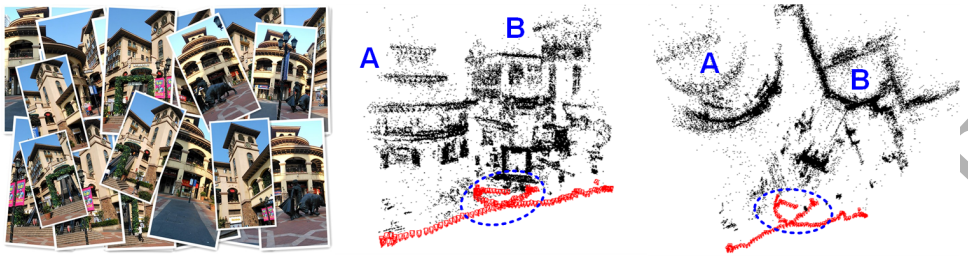

Figure 8: (a) Sample images of self-captured images. (b) Front view of its sparse 3D model. (c) Top view of its sparse 3D model.

$\alpha = 1.5$, where Z is the total number of images in that connected component. We set the coefficients $\beta_1 = 100$ and $\beta_2 = 1$ in Eq. (4) when selecting the first image in each base cluster. The number of layers in Eq. (5) is $L = 30$ when finding region clusters with the Multi-layer Shortest Path method. The ideal size of a sub-region cluster s is set to $3m$. In the merging step, the smallest count of shared feature tracks is $\tau = 4$.

Our algorithm is implemented using C++ on Ubuntu 14.10 operating system. The experiments are tested on a machine with two Intel Xeon CPU E5-2630 v3 2.40GHz, one NVIDIA GeForce GTX TitanX graphics card and 128GB RAM. This experimental platform is kept the same through all the experiments. All the compared methods are used with their default settings.

4.2. Results on Self-captured Images

In this part the results on a self-captured image set which contains 135 images are shown. These images are sampled from a short video captured by a handheld camera in front of two buildings A and B beside the street. Some of the example images and the sparse 3D scene structure are displayed

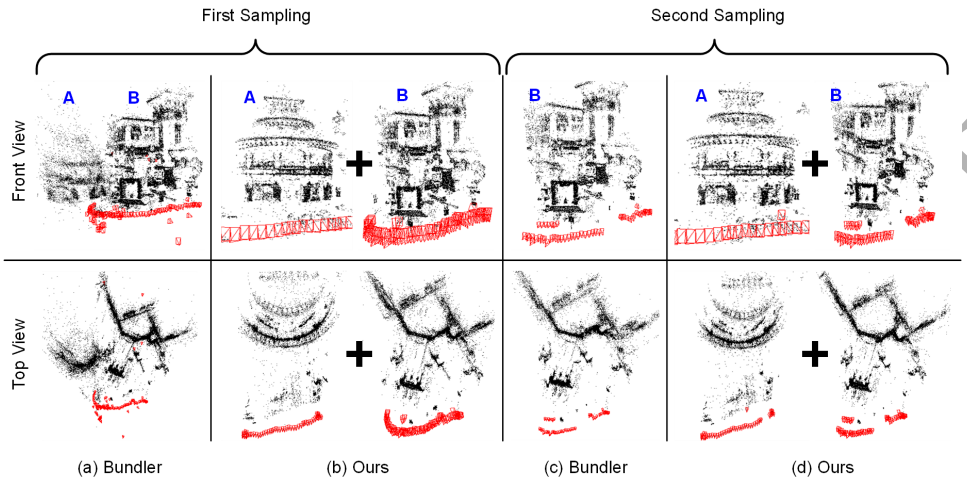

Figure 9: (a) and (b) are results of Bundler and our method after the first sampling. (c) and (d) are results of Bundler and our method after the second sampling. The fist row is the front view and the second row is the top view. See the text for details.

in Fig. 8.

We want to see what happens when the distribution of the image set changes. To achieve this goal, images within the blue region in Fig. 8 are sampled twice. After each sampling the remaining images are reconstructed by both Bundler [32] and the proposed method. The first sampling removes 14 images from the image set. Removing these images does not break the connectivity of the viewing graph, but the scene overlap between A and B is weakened and the reconstruction might be unreliable. The results of Bundler [32] and our method are shown in Fig. 9(a) and (b), respectively. Bundler starts the reconstruction from B and passes the structure to A. However, the structure of A and its camera poses are wrong, which is caused by passing 3D structure via weak scene overlap between them. The proposed method

reconstructs the whole scene from A and B simultaneously by assuming a boundary in the blue region. It avoids passing 3D structure via weak overlapping images and returns two good models. In the second sampling 21 more images are removed, which completely breaks the connection between A and B. As is shown in Fig. 9(c) and (d), Bundler can reconstruct only one part of the scene while our method can still reconstruct each of them correctly.

The goal of this experiment is to show that the proposed algorithm can reconstruct good models robustly, no matter how the distribution of images changes. When images are not strongly connected due to insufficient scene overlap, our method tries to build several good partial models rather than to build a wrong global model. If more images are added to reinforce the overlap, these partial models will be merged to a complete one without any problem.

4.3. Results on Dataset with One Model

Then the results on the Roman Forum dataset [43] are reported. It contains 2364 images in total and 1741 of them form a principal connected component. These images distribute unevenly in the scene and present several density centers. Compared with the previous dataset, this dataset is more challenging. There are many places where the overlap between images is weak. Because of large discrepancy such as viewpoint, scale and occlusion between these views, wrong feature correspondences are more likely to happen. Our method is compared with some state-of-the-art methods including: Bundler [32], VisualSFM [45] and COLMAP [28]. Bundler offers two BA alternatives: SBA with a single thread and Ceres with multiple

Table 1: Comparison results on the Roman Forum dataset.

Method	**#Cameras**	**#Points**	**MRE(pix)**	**Time**
Bundler-Ceres [32]	1441	498566	0.6568	6353.4
VisualSFM [45]	1267	343222	0.872	456
COLMAP [28]	1287	228876	0.5184	1398.76
Ours-MBMS	1312	343830	0.6512	186.065
Ours-MBSS	1366	521507	0.6368	276.924
Ours-SBSS	1381	502026	0.6974	1437.46
Ours-SBMS	1352	247313	0.6187	416.404
Ours-SBNC	794	96689	0.2897	7118.29

threads. Here we choose the faster Ceres with 12 threads. VisualSFM and COLMAP are more recent systems which use techniques such as GPU-based BA, Local-Global BA and so on. For the proposed method, different settings are tested. These settings differ in the number of base clusters and whether a region cluster is divided into several sub-region clusters. They are: (a) Use Multiple Base clusters and Multiple Sub-region clusters (MBMS). (b) Use Multiple Base clusters and a Single Sub-region cluster (MBSS). (c) Use a Single Base cluster and a Single Sub-region cluster (SBSS). (d) Use a Single Base cluster but Multiple Sub-region clusters (SBMS). In both MBSS and SBSS a region cluster is not divided so it is equivalent to have a single sub-regions cluster. Besides, we also try to use Normalized Cuts to partition the region cluster into multiple sub-region clusters when there is a Single Base cluster (SBNC).

We first compare our method MBMS with the other three methods. As we can see from the top five rows of Table 1, the number of reconstructed cameras and mean reprojection errors (MRE) for different methods do not

vary too much. MRE is conventionally used to measure the accuracy of the reconstruction result. It is a statistic of all the 3D points and their visible views on a certain dataset. The smaller MRE is, the more accurate the result will be. The number of 3D points for our method is much smaller than Bundler. This is mainly because data partitioning cuts many long tracks into short segments, which could not be reconstructed any more. The main difference between the four methods is the reconstruction speed. Bundler is the most time consuming because it only uses CPUs for computation. Although VisualSFM and COLMAP process the whole image set sequentially without partitioning the data, they are much faster than Bundler with GPU acceleration. The speedup also owns to the global-local bundle adjustment strategy used in both methods. Nevertheless, VisualSFM is about 3 times faster than COLMAP because COLMAP carried out additional steps such as re-triangulation after BA and iteratively outlier removing & refinement. Our method is about 2.5 and 7.5 times faster than VisualSFM and COLMAP, respectively. This speedup is mainly because data partitioning enables us to reconstruct different parts of the scene in parallel. Table 2 shows details about the base clusters and sub-region clusters in our method. There are four base clusters and region clusters. The number of sub-region clusters in each region cluster are 5, 3, 5 and 6, respectively. The sub-region clusters in each region cluster are nearly equal in size. An approximate ideal speedup ratio could be computed by $1300/(104+90) \approx 6.7$. However, when compared with VisualSFM the actual speedup ratio is only 2.4, which is far from the ideal value. The reason is that global BA used by our method will spend more time than local-global BA used by VisualSFM. This gap will be closed

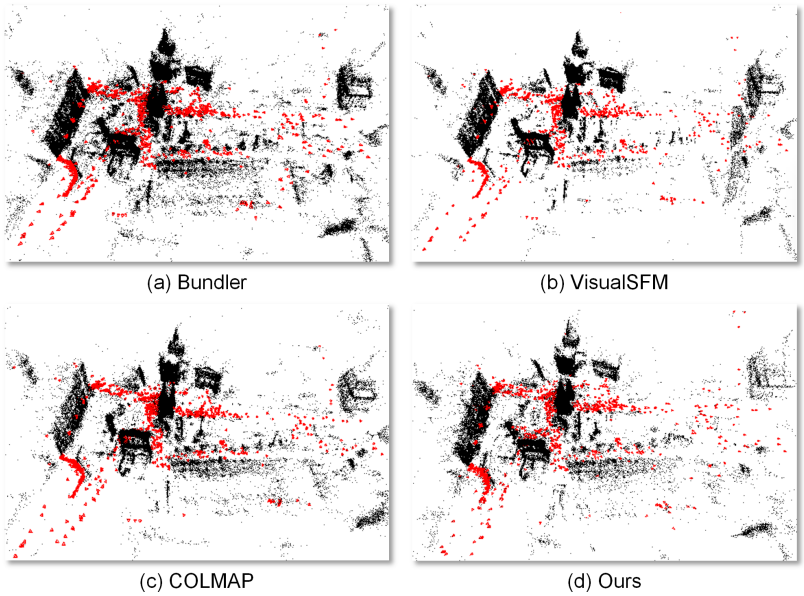

Figure 10: Results on the Roman Forum dataset [43]. From top-left to bottom-right are the results of Bundler [32], VisualSFM [45], COLMAP [28] and our method, respectively.

Table 2: The number of images in the partition result of the Roman Forum dataset.

Base Cluster	66					104			78					77					
Sub-region Cluster	76	87	76	81	43	66	48	62	85	61	82	79	39	90	83	70	69	77	63

by adopting local-global bundle adjustment into our framework, which is left as a future work. The reconstruction results are visualized in Fig. 10.

We then investigate our method when using different number of base clusters and sub-region clusters. When images in a region cluster are not partitioned (MBSS), the algorithm is parallel between region clusters but sequential within each of them. Its running time is 48% longer than MBMS. When using a single base cluster and a single sub-region cluster (SBSS), it is completely sequential without parallel mechanism and the running time

is the longest. If the region cluster is divided into several parts (SBMS), all the sub-region clusters could be processed in parallel and the running time is shortened by 3.4 times. The above tests show that either using multiple region clusters or multiple sub-region clusters will improve the reconstruction efficiency without significant accuracy loss. The last row of Table 1 is the result of SBNC, which uses Normalized Cuts to partition the single region cluster. Unfortunately, the number of both reconstructed cameras and 3D points decrease dramatically when compared with SBMS. This is because Normalized Cuts works in regardless of the starting point and consequently 3D structure is not able to propagate from the base model to some sub-region clusters. Due to bad partitioning result the reconstruction time rises rapidly as well.

4.4. Results on Dataset with Several Independent Models

In this part, three other public datasets including Montreal Notre Dame [43], Vienna Cathedral [43] and Yorkminster [43] are tested. The number of images in these datasets are 2298, 6288 and 3368, respectively. Different from the Roman Forum dataset which has one dominant connected component, these image sets contain several independent models. This experiment wants to see if the proposed method can automatically reconstruct all the models correctly.

Our method is compared with Bundler [32] (using Ceres for BA), VisualSFM [45] and COLMAP [28]. The number of reconstructed cameras, the mean reprojection errors and the running time for different methods are given in Table 3. The number of dominant partial models in each dataset is 3, 2 and 3, respectively. Since Bundler uses only a single starting point and runs

Table 3: Results on the Montreal Notre Dame, Vienna Cathedral and Yorkminster datasets. For each partial model, the number of reconstructed cameras and the mean reprojection error are given. The running time for bundle adjustment is in the last column.

Dataset	**Method**	**#Cam**			**MRE(pix)**			**Time(s)**
Montreal Notre Dame	Ours	385	355	97	0.6241	0.7286	0.5112	**231.1**
	COLMAP [28]	499	330	92	0.565	0.468	0.491	1489.7
	VisualSFM [45]	343	504	97	1.596	1.467	0.909	457
	Bundler [32]	399			1.5083			648.2
Vienna Cathedral	Ours	1000		292	0.6550		0.8684	**363.0**
	COLMAP [28]	962		278	0.511		0.612	2665.8
	VisualSFM [45]	929		275	1.901		1.519	1216
	Bundler [32]	1197			0.7106			12181.1
Yorkminster	Ours	593	333	121	0.6935	0.5451	0.5905	**281.1**
	COLMAP [28]	524	226	110	0.542	0.511	0.536	1358.2
	VisualSFM [45]	517	128	106	1.429	0.639	0.664	796
	Bundler [32]	122			0.6265			209.3

incremental SfM once, it can reconstruct only one of the primary models. What's more, it is still the most time-consuming. VisualSFM will find a new starting point in the remaining images and run another SfM procedure when the current process stops due to lack of overlapping images. So it manages to reconstruct several models from the whole image set by trial and error. However, most of these models are very small, containing few cameras. This means that the starting point selected is of strong locality and the initial model could not propagate 3D structure to further places. Usually a good model is returned after many failed attempts, which wastes a lot of time. On the Vienna Cathedral dataset, VisualSFM reconstructs similar number

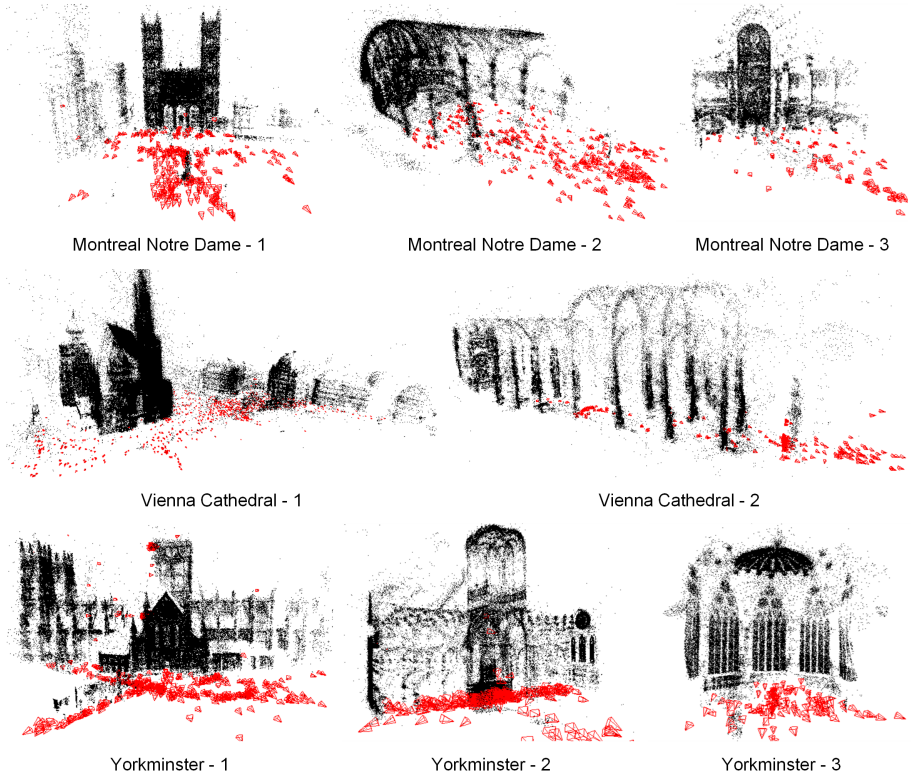

Figure 11: The reconstruction results of our method. From top to bottom are: 3 models in Montreal Notre Dame, 2 models in Vienna Cathedral and 3 models in Yorkminster, respectively.

of cameras with Bundler, but saves 90% of the time due to GPU acceleration. Since our method reconstructs the whole scene in parallel, it is 2-4 times faster than VisualSFM and 6-7 times faster than COLAMP. However, our method does not reconstruct multiple primary models in a trial and error manner. Instead, we investigate the data distribution and launch several starting points in possible sub-models before reconstruction, which is more effective and reliable. The reconstruction results of our method is visualized

in Fig. 11.

5. Conclusion

In this paper, an image set partitioning method is proposed for efficient Structure from Motion on unevenly distributed images. Different from existing "blind partitioning" methods which first divide the image set and the select a starting point in each part, what proposed here is a center driven method. It selects several base clusters at places with high image density, and then divide the remaining images into region clusters according to their reconstruction paths to the base clusters. To further improve reconstruction efficiency, images in each region cluster are further divided into several sub-region clusters so that they could be added to the base model reconstructed from the base cluster simultaneously. The proposed method reconstructs the scene in parallel not only between different region clusters, but also within each region cluster. This leads to significant speedup. Experiments show that the proposed data partitioning method achieves much faster speed than state-of-the-art methods without much precision deterioration.

SfM is an intermediate step in the full vision-based 3D model reconstruction. However, it estimates only camera poses and sparse point cloud. Its output can be fed into a subsequent Multi-view Stereo (MVS) process to build a dense model. If the result of SfM is accurate enough, the dense model is able to present rich details, which is very useful in many applications such as 3D printing, virtual reality and vision-based manipulation.

6. Acknowledgement

We would like to thank the editors and reviewers for their time. This work is supported by the Fundamental Research Funds for the Central Universities(CUG170675), by the NSFC (61802356, 61772213, 91748204), also in part by JCYJ20170818165917438, by Open Project Foundation of Intelligent Information Processing Key Laboratory of Shanxi Province (CICIP2018003) and the Open Research Project of The Hubei Key Laboratory of Intelligent Geo-Information Processing(KLIGIP-2017B04).

References

[1] S. Agarwal, K. Mierle, Others, Ceres solver, `http://ceres-solver.org`.

[2] S. Agarwal, N. Snavely, I. Simon, S. Seitz, R. Szeliski, Building rome in a day, in: Computer Vision, 2009 IEEE 12th International Conference on, 2009, pp. 72–79.

[3] B. Bhowmick, S. Patra, A. Chatterjee, V. M. Govindu, S. Banerjee, Divide and Conquer: Efficient Large-Scale Structure from Motion Using Graph Partitioning, Springer International Publishing, Cham, 2015, pp. 273–287.

[4] Y. Chen, A. B. Chan, Z. Lin, K. Suzuki, G. Wang, Efficient tree-structured sfm by ransac generalized procrustes analysis, Computer Vision and Image Understanding 157 (2017) 179 – 189, large-Scale 3D Modeling of Urban Indoor or Outdoor Scenes from Images and Range Scans.

[5] H. Cui, X. Gao, S. Shen, Z. Hu, Hsfm: Hybrid structure-from-motion, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[6] H. Cui, S. Shen, Z. Hu, Robust global translation averaging with feature tracks, in: 2016 23rd International Conference on Pattern Recognition (ICPR), 2016, pp. 3727–3732.

[7] Z. Cui, P. Tan, Global structure-from-motion by similarity averaging, in: 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 864–872.

[8] K. Douterloigne, S. Gautama, W. Philips, Speeding Up Structure from Motion on Large Scenes Using Parallelizable Partitions, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 13–21.

[9] T. Fang, L. Quan, Resampling Structure from Motion, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 1–14.

[10] M. Farenzena, A. Fusiello, R. Gherardi, Structure-and-motion pipeline on a hierarchical cluster tree, in: 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops, 2009, pp. 1489–1496.

[11] R. Feng, Y. Zhong, L. Wang, W. Lin, Rolling guidance based scale-aware spatial sparse unmixing for hyperspectral remote sensing imagery, Remote Sensing 9 (12).

[12] A. W. Fitzgibbon, A. Zisserman, Automatic camera recovery for closed

or open image sequences, Springer Berlin Heidelberg, Berlin, Heidelberg, 1998, pp. 311–326.

[13] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, M. Pollefeys, Building rome on a cloudless day, in: K. Daniilidis, P. Maragos, N. Paragios (eds.), Computer Vision ECCV 2010, vol. 6314 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2010, pp. 368–381.

[14] B. J. Frey, D. Dueck, Clustering by passing messages between data points, Science 315 (5814) (2007) 972–976.

[15] M. Havlena, A. Torii, T. Pajdla, Efficient Structure from Motion by Graph Optimization, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 100–113.

[16] J. Heinly, J. L. Schonberger, E. Dunn, J.-M. Frahm, Reconstructing the world* in six days, in: Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on, 2015, pp. 3287–3295.

[17] S. Kundu, J. Misra, A linear tree partitioning algorithm, SIAM Journal on Computing 6 (1) (1977) 151–154.

[18] B. Leng, Y. Liu, K. Yu, X. Zhang, Z. Xiong, 3d object understanding with 3d convolutional neural networks, Information Sciences 366 (2016) 188 – 201.

[19] X. Li, C. Wu, C. Zach, S. Lazebnik, J.-M. Frahm, Modeling and recognition of landmark image collections using iconic scene graphs, in:

D. Forsyth, P. Torr, A. Zisserman (eds.), Computer Vision ECCV 2008, vol. 5302 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2008, pp. 427–440.

[20] T. Liu, H. Liu, Z. Chen, A. M. Lesgold, Fast blind instrument function estimation method for industrial infrared spectrometers, IEEE Transactions on Industrial Informatics (2018) 1–1.

[21] D. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110.

[22] K. Ni, D. Steedly, F. Dellaert, Out-of-core bundle adjustment for large-scale 3d reconstruction, in: 2007 IEEE 11th International Conference on Computer Vision, 2007, pp. 1–8.

[23] D. Nistér, Reconstruction from Uncalibrated Sequences with a Hierarchy of Trifocal Tensors, Springer Berlin Heidelberg, Berlin, Heidelberg, 2000, pp. 649–663.

[24] D. Nister, H. Stewenius, Scalable recognition with a vocabulary tree, in: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, vol. 2, 2006, pp. 2161–2168.

[25] A. Oliva, A. Torralba, Modeling the shape of the scene: A holistic representation of the spatial envelope, International Journal of Computer Vision 42 (3) (2001) 145–175.

[26] R. Pless, Using many cameras as one, in: 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings., vol. 2, 2003, pp. II–587–93 vol.2.

[27] R. Raguram, C. Wu, J.-M. Frahm, S. Lazebnik, Modeling and recognition of landmark image collections using iconic scene graphs, International Journal of Computer Vision 95 (3) (2011) 213–239.

[28] J. L. Schönberger, J.-M. Frahm, Structure-from-motion revisited, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[29] R. Shah, A. Deshpande, P. J. Narayanan, Multistage sfm: A coarse-to-fine approach for 3d reconstruction (2015).

[30] J. Shi, J. Malik, Normalized cuts and image segmentation, Pattern Analysis and Machine Intelligence, IEEE Transactions on 22 (8) (2000) 888–905.

[31] H.-Y. Shum, Q. Ke, Z. Zhang, Efficient bundle adjustment with virtual key frames: a hierarchical approach to multi-frame structure from motion, in: Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149), vol. 2, 1999, p. 543 Vol. 2.

[32] N. Snavely, S. Seitz, R. Szeliski, Modeling the world from internet photo collections, International Journal of Computer Vision 80 (2) (2008) 189–210.

[33] N. Snavely, S. Seitz, R. Szeliski, Skeletal graphs for efficient structure from motion, in: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, 2008, pp. 1–8.

[34] N. Snavely, S. M. Seitz, R. Szeliski, Photo tourism: exploring photo collections in 3d, in: Proceedings of ACM SIGGRAPH, 2006, 2006, pp. 835–846.

[35] D. Steedly, I. Essa, F. Dellaert, Spectral partitioning for structure from motion, in: Proceedings Ninth IEEE International Conference on Computer Vision, 2003, pp. 996–1003 vol.2.

[36] C. Sweeney, V. Fragoso, T. Hllerer, M. Turk, Large scale sfm with the distributed camera model, in: 2016 Fourth International Conference on 3D Vision (3DV), 2016, pp. 230–238.

[37] C. Tang, W. Li, P. Wang, L. Wang, Online human action recognition based on incremental learning of weighted covariance descriptors, Information Sciences 467 (2018) 219 – 237.

[38] R. Toldo, R. Gherardi, M. Farenzena, A. Fusiello, Hierarchical structure-and-motion recovery from uncalibrated images, Computer Vision and Image Understanding 140 (2015) 127 – 143.

[39] B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon, Bundle adjustment a modern synthesis, in: B. Triggs, A. Zisserman, R. Szeliski (eds.), Vision Algorithms: Theory and Practice, vol. 1883 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2000, pp. 298–372.

[40] D. Valiente, A. Gil, L. Fernndez, scar Reinoso, A modified stochastic gradient descent algorithm for view-based slam using omnidirectional images, Information Sciences 279 (2014) 326 – 337.

[41] D. Viejo, J. Garcia-Rodriguez, M. Cazorla, Combining visual features and growing neural gas networks for robotic 3d slam, Information Sciences 276 (2014) 174 – 185.

[42] K. Wilson, D. Bindel, N. Snavely, When is rotations averaging hard?, in: B. Leibe, J. Matas, N. Sebe, M. Welling (eds.), Computer Vision – ECCV 2016, Springer International Publishing, Cham, 2016, pp. 255–270.

[43] K. Wilson, N. Snavely, Robust global translations with 1dsfm, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (eds.), Computer Vision – ECCV 2014, Springer International Publishing, Cham, 2014, pp. 61–75.

[44] C. Wu, Siftgpu: A gpu implementation of scale invariant feature transform (sift), `http://cs.unc.edu/ ccwu/siftgpu/` (2007).

[45] C. Wu, Visualsfm: A visual structure from motion system, `http://ccwu.me/vsfm/` (2011).

[46] C. Wu, Towards linear-time incremental structure from motion, in: 3D Vision - 3DV 2013, 2013 International Conference on, 2013, pp. 127–134.

[47] C. Wu, S. Agarwal, B. Curless, S. M. Seitz, Multicore bundle adjustment, in: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, 2011, pp. 3057–3064.

[48] L. Wu, Y. Wang, J. Gao, X. Li, Where-and-when to look: Deep siamese attention networks for video-based person re-identification, IEEE Transactions on Multimedia (2018) 1–1.

[49] E. Zheng, C. Wu, Structure from motion using structure-less resection, in: Proceedings of International Conference on Computer Vision (ICCV), 2015, pp. 2075–2083.

[50] S. Zhu, T. Shen, L. Zhou, R. Zhang, T. Fang, L. Quan, Accurate, scalable and parallel structure from motion, CoRR abs/1702.08601.
URL http://arxiv.org/abs/1702.08601