Progressive Structure from Motion

Alex Locher[1], Michal Havlena[2], and Luc Van Gool[1,3]

[1] Computer Vision Laboratory, ETH Zurich, Switzerland
[2] Vuforia, PTC, Vienna, Austria
[3] VISICS, KU Leuven, Belgium

Abstract. Structure from Motion or the sparse 3D reconstruction out of individual photos is a long studied topic in computer vision. Yet none of the existing reconstruction pipelines fully addresses a progressive scenario where images are only getting available during the reconstruction process and intermediate results are delivered to the user. Incremental pipelines are capable of growing a 3D model but often get stuck in local minima due to wrong (binding) decisions taken based on incomplete information. Global pipelines on the other hand need the access to the complete viewgraph and are not capable of delivering intermediate results. In this paper we propose a new reconstruction pipeline working in a progressive manner rather than in a batch processing scheme. The pipeline is able to recover from failed reconstructions in early stages, avoids to take binding decisions, delivers a progressive output and yet maintains the capabilities of existing pipelines. We demonstrate and evaluate our method on diverse challenging public and dedicated datasets including those with highly symmetric structures and compare to the state of the art.

1 Introduction

3D reconstruction from individual photographs is a long studied topic in computer vision [31, 1, 12]. The field of Structure from Motion (SfM) deals with the intrinsic and extrinsic calibration of sets of images and recovers a sparse 3D structure of the scene at the same time. Traditional methods are usually designed as batch processing algorithms, where image acquisition and image processing are separated into two independent steps. This contrasts with current demand, when one would like to be able to convert an object or a scene into a 3D model anytime and anywhere, just by using the mobile phone one is carrying in her pocket. Recent developments in mobile technology and the availability of 3D printers raised the need for 3D content even more and underlay the importance of 3D modeling. In a user-centric scenario, images are taken on the spot and processed by a 3D modeling pipeline on-the-fly [16, 19]. Any feedback which gets available to the user helps to guide her acquisition and, even more importantly, assures that the 3D model represents the real-world object in the desired quality. In a collaborative scenario, multiple users acquire pictures of the same object and images are gathered on the reconstruction server in the cloud. The 3D model is progressively built and intermediate reconstruction results are

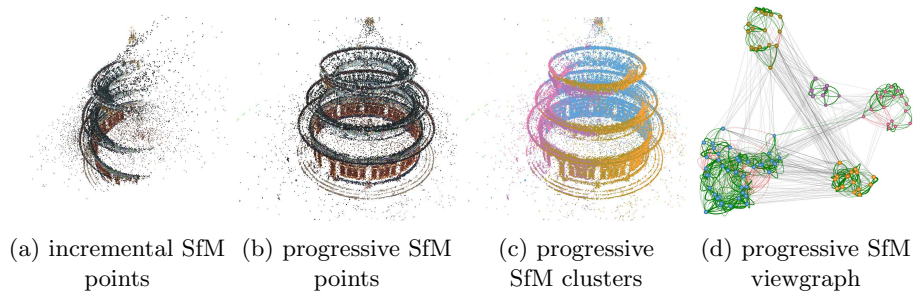|			
(a) incremental SfM points	(b) progressive SfM points	(c) progressive SfM clusters	(d) progressive SfM viewgraph

Fig. 1: Opposed to existing methods, a favorable image order is not crucial for the proposed progressive SfM pipeline. While the baseline method (a) fails to recover the structure of the scene our method successfully reconstructs the temple (b). Individual clusters are identified in the viewgraph (d) and merged (c) based on a lightweight optimization.

shown to the user. Images are getting available as they are taken and the SfM pipeline has never access to the complete set of information in the dataset. Moreover, the whole reconstruction process might have a starting, but no predefined end point. Users might always decide to add more images to an existing model.

In this work we therefore propose a progressive SfM pipeline which avoids taking (potentially fatal) binding decisions and therefore is as independent of the input image order as possible. Moreover, the proposed pipeline reuses already computed intermediate results in later steps and is suited for delivering progressive modeling results back to the user within seconds.

1.1   Related Work

Classical SfM pipelines are typically not suited to be used in such a progressive – multiuser-centric scenario. Global pipelines [22, 29] start by estimating poses of all cameras in the dataset and estimate the structure in a second step. Evidently, this relies on the access to the complete dataset which contradicts the idea of progressive 3D modeling. Sequential SfM pipelines, sometimes termed SLAM [7, 23], are inherently suitable for processing (potentially infinite) streams of images. Nevertheless, the underlying assumption often is that images neighboring in the sequence are spatially close in the scene, which is easily violated when streams from multiple users are combined together. Incremental SfM pipelines [31, 36, 28] build a 3D model by initializing the structure from a small seed and gradually growing it by adding additional cameras. This scheme is closer to the requested progressive scenario but, unfortunately, is strongly dependent on the order in which images are added to the model [21, 27]. View selection algorithms [34] carefully determine the image order usually by employing the global matching information which is not available in the progressive case. Hierarchical SfM pipelines [10, 8] try to overcome the problem of improper seed selection by starting from several seed locations at the same time and eventually merging the

partial 3D models into a single global model in later stages. Sweeney et al. [33] group multiple individual cameras and optimize them jointly as a distributed camera. However all hierarchical methods require the knowledge of all images in advance. Our proposed method is partially inspired by these approaches but in order to provide the progressive capability, the hierarchy is not fixed but rather re-defined every time new images are added to the reconstruction process.

Due to the lack of global information, an incremental pipeline would connect new images to the existing model based on incomplete information which often causes corrupted 3D models [14, 37, 26, 15, 11]. Even more importantly, incremental pipelines cannot recover from wrong decisions taken based on missing information and therefore can easily get stuck in only locally optimal reconstructions. The only reliable solution would be re-running an incremental or global pipeline from scratch when new images become available which leads to an impractical algorithm runtime. Heinly et al. [11] detected erroneous reconstructions of scenes with duplicate structure in a post processing step by evaluating different splits of the model into submodels and potentially merge them in the correct configuration by leveraging conflicting observations. Our method in contrary is a full fledged SfM pipeline which avoids getting trapped in a local minima in the first place. Faulty configurations are detected and corrected on-the-fly and not in a post processing step.

Our work bases on a lightweight representation of the complete scene as a viewgraph. Many existing approaches investigated robustification of the global view-graph by filtering out bad epipolar geometries [34, 6] and enforcing loop constraints [38]. Wilson and Snavely [35] are able to reconstruct scenes with repetitive structures by scoring repetitive features using local clustering. Recent work of Cui [5] takes a similar approach to our pipeline. The Hybrid SfM pipeline estimates all rotations of the viewgraph in a community based global algorithm. The estimated orientations are leveraged in the second phase of the pipeline by estimating the translations and structure of the scene in an incremental scheme with reduced dimensionality. While sharing the idea of combining global and incremental schemes, HSfM is a pure batch processing algorithm. The global rotation averaging needs access to the complete view graph in advance which is not available in a progressive scheme. In our work we combine a dynamic global view graph with a local clustering based on a connectivity score and combine the advantages of incremental and global structure from motion. In order to accommodate for the demands of a progressive pipeline, we allow for flexibility in already reconstructed parts of the model and constantly verify the local reconstructions against the globally optimized viewgraph.

1.2   Contributions

We propose a novel progressive SfM pipeline which enables 3D reconstruction in an anytime anywhere multiuser-centric scenario. Unlike traditional pipelines, the proposed approach avoids taking binding decisions, does not depend on the order of incoming images and is able to recover from wrong decisions taken due to the lack of information. Moreover, the computed intermediate results

are propagated along the reconstruction process resulting in an efficient use of computational resources. A C++ implementation of the proposed progressive pipeline is publicly available under: https://github.com/alexlocher/progsfm.

2 Progressive Reconstruction

In the following section we give an overview of our progressive SfM pipeline and detail individual components and key aspects later on.

2.1 Overview

Our progressive SfM pipeline takes an ordered sequence of images with its geometrically verified correspondences as the input and delivers a sparse pointcloud and calibrated camera poses as the output (see Figure 2). The resulting sparse configuration is updated with every image added to the scene. A viewgraph with nodes being images and edges connecting pairs of matched images is gradually built and serves as the global knowledge throughout the whole reconstruction. On every iteration of the algorithm, the viewgraph is clustered based on local connectivity and individual clusters are processed locally. In each of the local clusters a robust rotation averaging scheme filters out wrong two-view geometries and the 3D structure is estimated using either an incremental or a global SfM pipeline. The cluster configuration between two time steps is tracked and the already estimated parts of the 3D model are passed to the next stage. The global configuration of individual clusters is estimated in the last stage by robustly estimating 7 DoF similarity transforms between them using the remaining inter-cluster constraints. Generally, the local incremental method enables robust and efficient reconstructions while the viewgraph combined with the robust rotation averaging injects the global knowledge and allows for correction of corrupted 3D models.

2.2 Progressive Viewgraph

The algorithm takes a (randomly) ordered sequence of images $\mathbf{I} = (I_0, I_1, I_2, \dots)$ as the input. Every incoming image is matched against the most relevant images already present in the scene and geometrically verified pairwise correspondences are obtained. A viewgraph $\mathcal{G}(\mathcal{V}_t, \mathcal{E}_t)$ with images as vertices $\mathcal{V}_t = \{I_i | i \leq t\}$ is maintained at every time step t. Two vertices (V_i, V_j) are connected by an undirected edge $\mathcal{E}_{ij}$ iff there exists a minimum amount of correspondences $(\mathbf{M}_{ij} > \eta \wedge i < j \leq t)$ between them where $\mathbf{M} \in \mathbb{R}^{t \times t}$ is the matching matrix. Every edge $\mathcal{E}_{ij}$ has an associated relative rotation $\mathbf{R}_{ij}$ and translation direction $\mathbf{t}_{ij}$ which is obtained by decomposing either the estimated essential matrix in the calibrated case or the fundamental matrix when the focal length is not known.

The order in which images are fed to a reconstruction pipeline plays an important role and every snapshot of the viewgraph only captures the past information of the reconstruction process. This is why filtering of supposedly wrong
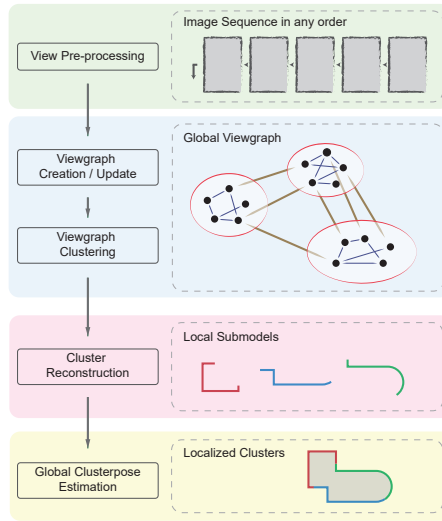
Fig. 2: An overview of the main steps of the progressive SfM pipeline and its involved components.
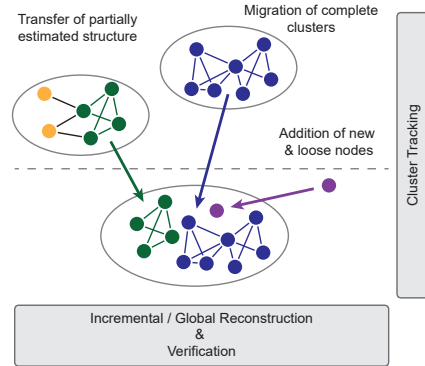
Fig. 3: Depending on the status of an individual cluster, its structure is estimated with different methods.

two-view geometries at this stage can be very dangerous. It might happen that a geometry is inconsistent with other local geometries in the neighborhood[4] at the current time-step but connecting images added in later steps may show that the two-view geometry was actually correct. As a wrong decision in the global viewgraph could lead to a local minimum in the reconstruction, i.e. a corrupted 3D model, we do not conduct any outlier rejection and defer robustification to a later stage.

2.3   Clustering

Motivated by the general observation that densely connected regions of the viewgraph are likely to form a 3D model worth reconstruction, the viewgraph is clustered in a second step. The distance d_{ij} between two vertices V_{ij} is based on the weighted Jaccard distance of the adjacent edges where connections to neighboring vertices are weighted by the number of verified correspondences.

$$d_{ij} = 1 - \frac{\sum\limits_{n \in \mathcal{N}} \mathbf{M}_{in} + \mathbf{M}_{nj}}{\sum\limits_{n=0}^{t} \mathbf{M}_{in} + \mathbf{M}_{nj}} \Bigg| \mathcal{N} = \{k | \mathbf{M}_{ik} \cdot \mathbf{M}_{kj} > 0\} \tag{1}$$

---

[4] This can, e.g., be checked by computing the cumulative rotation of loops in which the edge is participating [38]

A set of clusters $\mathcal{C}_t$ is hierarchically grown by single-linkage clustering until no single edge with $d_{ij} < \eta$ between two clusters exists. Single-linkage clustering tends to generate clusters with a chain-like topology where the two ending nodes might have a distance way larger than the defined threshold. While this might be a disadvantage in other applications it is actually beneficial in our application as local (incremental) reconstruction pipeline performs well for such graph structures.

Incremental Clustering Due to the single-linkage hierarchical clustering, a simplified incremental scheme can be used to update an existing cluster topology with a new node. While generally a single extra node can cause a complete change in topology of the clustered graph, the changes are limited to clusters which are connected to the new node. As a result only clusters with an edge connecting to the new node have to be re-clustered which leads to an efficient and scalable implementation. Note that in worst case the whole graph still might be updated – but in most cases a new image only connects to few clusters and therefore most of the existing clusters remain untouched.

2.4   Cluster Tracking and Recycling

Between the transition of two timesteps t_b and $t_a = t_b + 1$ the topology of clusters can undergo large changes but mostly will either stay the same or be extended by the new image. All changes in the clusters have to be propagated to the eventually estimated 3D structure. We therefore keep track of the nodes changing cluster between the two timesteps and add or remove the corresponding images in the local reconstruction. The recycling of intermediate (partial) 3D reconstructions can be realized by a merge and split scheme. If a group of interconnected nodes transfer together from one cluster to another, the corresponding cameras in the 3D model can be separated from the rest and merged into the potentially existing structure of the new cluster.

2.5   Cluster Reconstruction

Once individual clusters $\mathcal{C}_i$ have been identified by the hierarchical clustering, the 3D structure of the images and correspondences of every cluster C is estimated. The cluster reconstruction process is only triggered if its topology has changed meaning either some nodes were added or some nodes were removed from the cluster. If the topology of the cluster is unchanged between the two time steps t_b and t_a, the reconstruction step is skipped as a whole.

A sub-graph $\mathcal{V}_C$ capturing all vertices and edges of the cluster C is extracted in a first step. All following operations are restricted to the scope of the extracted graph $\mathcal{V}_C$.

Robust Rotation Averaging The unfiltered viewgraph is potentially corrupted by outliers and has to be cleaned up for further usage. As this step is performed in every iteration it is safe to reject outlier two-view geometries. As a result of the repetitive execution of the filtering stage, it is important to use a computationally efficient filtering scheme. As proposed by Chatterjee et al. [3] we estimate the global rotations of the subgraph with a robust ℓ_1 optimization. Global Rotations $\bar{\mathbf{R}}$ are initialized by concatenating the relative rotations of a Maximum Spanning Tree (MST) extracted from the viewgraph $\mathcal{V}_C$ using the number of verified inliers $\mathbf{M}_{ij}$ as edge weights. The global rotations are then optimized by minimizing the relative rotation errors ρ_{ij}.

$$\arg\min_{\bar{\mathbf{R}}} \sum_{(i,j)\in\mathcal{E}_C} \rho_{ij}\left(\mathbf{R}_{ij}, \bar{\mathbf{R}}_j\bar{\mathbf{R}}_i^{-1}\right) \tag{2}$$

By using the Lie-algebraic approximation of the relative rotation, the error can be expressed as a difference of the corresponding rotations ω. Where $\omega = \Theta\mathbf{n} \in \mathfrak{so}(3)$ denotes a rotation by angle Θ around the unit axis $\mathbf{n}$.

$$\omega_{ij} \approx \omega_j - \omega_i \tag{3}$$

The relative rotations can be encoded in a sparse matrix $\mathbf{A}$ where each row only has two nonzero $-1, +1$ entries. The robust ℓ_1 norm combined with an edge weighting ρ depending on the number of verified correspondences allows us to obtain the global rotations of the cluster C in an iterative scheme by optimizing Eqn. 4 in every step. For more details we refer the reader to the original publication [3].

$$\arg\min_{\omega_g} \|\mathbf{A}\,\Delta\omega_g - \rho\omega_{rel}\|_{\ell_1} \tag{4}$$

The obtained global rotations can optionally be refined by Iteratively Reweighted Least Squares (IRLS) method using a robustified ℓ_2 norm. As we are primarily interested in rejecting outliers, the results of the ℓ_1 optimization is accurate enough and we omit the refinement step.

Finally edges with a relative error above $\rho_{g\,\max}$ are removed from the local viewgraph.

Reconstruction Using the global rotations and the pairwise matches as the input, the structure of the individual clusters is estimated with different methods depending on the current status (Figure 3).

New: If we could not recover any structure from t_b and the cluster contains at least μ_{min} nodes but less than μ_{max} images a new reconstruction is starting using an incremental SfM pipeline similar to [31, 36]. If the number of images in the cluster is larger than μ_{max}, incremental pipelines get inefficient and a global pipeline is more suited. We therefore refine the global rotations and estimate global position using the 1DSfM method [35]. The structure is then obtained by triangulating consistent feature-tracks and the configuration is refined by a bundle adjustment step.

Extend: In most cases an existing local reconstruction can be extended by one or multiple images which are either new or transferred from another cluster. In order to extend an existing 3D reconstruction with a new image we extend existing feature tracks by the new correspondences and estimate the 3D position of the camera by the P3P algorithm [17]. New tracks are added and potential conflicting tracks are split up. A local bundle adjustment refines the structure and calibration of the newly added cameras. If the model has grown by more than η_{grow} percent, a bundle adjustment step over the local reconstruction refines the whole structure.

Transfer: If a large part (more than μ_{min} nodes) of an already estimated local model is transferred to another cluster, its already estimated structure is kept and transferred to the new cluster. Commonly estimated tracks are fused and a bundle adjustment step refines the structure. Potentially unestimated cameras are then added in an incremental scheme as described before.

Detection of Bad Configurations Due to the incremental reconstruction scheme for cluster reconstruction we can reuse most of the intermediate results from an earlier stage and propagate them to later stages. The incremental scheme is highly dependent on the actual image ordering and therefore some unfortunate decisions taken in early stages (e.g. wrongly connecting an image due to missing information) cannot be recovered in later stages. Global clustering usually solves this problem for us as the wrongly connected image or sub-model is likely to be transferred to another cluster at a later stage. But it might also happen that the image actually belongs to the same cluster and yet is wrongly connected. Without any additional counter measurements, we would end up with a corrupted reconstruction in such cases.

By using the relative global rotations of the local viewgraph (which is independent of the image order) we can evaluate an error measure between the rotations of the current local reconstruction $\hat{\mathbf{R}}$ and the global rotations $\bar{\mathbf{R}}$.

$$\rho_l = \left\| \left\{ \bar{\mathbf{R}}_{ij} \, \hat{\mathbf{R}}_{ij}^{-1} \, \middle| \, (i,j) \in \mathcal{E}_C \right\} \right\|_\infty \tag{5}$$

If the rotation error ρ_l is above a certain limit $\rho_{l\,\max}$ the reconstruction is likely to be erroneous. Hence it is reset and re-estimated in the next phase of the reconstruction.

2.6   Cluster Pose Estimation

The output of the pipeline so far are multiple locally highly consistent but disconnected model parts. In order to merge them to a final representation we make use of the remaining inter-cluster connections. As this step of merging multiple models is often rather fragile and can easily go wrong, several methods employed a rather conservative merging criteria as multiple cameras of overlap [10] or a high amount of common points [36]. Instead, we are estimating pairwise similarity transforms $\mathcal{T}_{ab} \in Sim(3)$ between clusters C_a and C_b and optimize the

global positions of the cluster centers. For robustification, we use a two stage verification scheme for individual inter-cluster constraints.

In the first stage, a similarity transform based on 3D-to-3D correspondences from every single edge $\mathcal{E}_{ij}$ between clusters C_a and C_b is estimated in a RANSAC scheme [13, 9]. The vertices (camera centers) of C_a are afterwards transformed using the estimated transformation and the camera configuration of the obtained combined model is compared to the individual configuration before. A neighborhood similarity measure $\mathfrak{s}$ evaluates how many of the cameras $\mathbf{p}_k$ in the merged cluster would retain their closest neighbors $\text{NN}_a(\mathbf{p}_k)$ and $\text{NN}_b(\mathbf{p}_k)$ for the cameras from clusters C_a and C_b, respectively. $\mathbf{p}_k$ denotes the camera center of the node k and the operator $\mathcal{T}_{ij} \circ \mathbf{p}_k$ transforms a point by the similarity transform.

$$\mathcal{T}_{ij} \; \forall \; i \in C_a \wedge j \in C_b \wedge (i,j) \in \mathcal{E}_C \tag{6}$$

$$\mathbf{P}_a = \mathbf{p}_k | k \in C_a \tag{7}$$

$$\mathbf{P}_b = \mathbf{p}_k | k \in C_b \tag{8}$$

$$\mathbf{P}_c = \mathbf{P}_b \cup \mathcal{T}_{ij} \circ \mathbf{p}_k | k \in C_a \tag{9}$$

$$\mathfrak{s} = \frac{1}{|\mathbf{P}_c|} \sum_{\mathbf{p}_k \in \mathbf{P}_c} s_k \tag{10}$$

$$s_k = \begin{cases} 1 & k \in C_a \wedge \mathcal{T}_{ij} \circ \text{NN}_a(\mathbf{p}_k) = \text{NN}_c(\mathcal{T}_{ij} \circ \mathbf{p}_k) \\ 1 & k \in C_b \wedge \text{NN}_b(\mathbf{p}_k) = \text{NN}_c(\mathbf{p}_k) \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

Edges with the neighborhood similarity measure below λ_c are rejected as outliers for the final constraint estimation. The measure is motivated by the observation that nearby cameras should already be clustered by the original algorithm and prevents merged models where the cameras are extensively interleaved.

In the second stage, a final transformation $\mathcal{T}_{ij}$ between the clusters is estimated using all correspondences of edges that survived the first filtering stage. All relative constraints are gathered and a cluster graph $\mathcal{G}_C$ with local reconstructions as nodes and pairwise inter-cluster constraints as edges is created.

$$\arg\min_{\mathcal{P}_C} \sum_{(a,b) \in \mathcal{G}_C} \left\| \delta \left(\mathcal{T}_{ab} \, \mathcal{P}_b \mathcal{P}_a^{-1} \right) \right\|^2 \tag{12}$$

The global poses $\mathcal{P}_C$ of the clusters are afterwards obtained by iteratively minimizing the pairwise relative pose constraints $\mathcal{T}_{ab}$, where $\delta(.)$ denotes the robust Huber error function.

While this in the optimal case leads to a single 3D model, the flexibility and efficiency of the global optimization allows us to choose a much less conservative threshold for cluster merging. In addition to its efficiency, the cluster graph representation can easily incorporate additional extrinsic constraints, e.g., the gravity vector for orientation with respect to the ground plane or coarse cluster-wise GPS locations for geo-referencing. Such additional constraints can help to put individual models in relative perspective even if pure vision based inter-cluster constraints are not (yet) available.

3   Experiments and Results

In the following section, we first give some implementation details of the proposed solution and introduce the conducted experiments.

3.1   Implementation

We implemented the proposed pipeline as a C++ software. SIFT [20] features are detected on all images and described with the RootSift descriptor [2]. The pipeline is setup for stream processing dedicated by the progressive reconstruction scheme. Every incoming image is indexed by a 1M word vocabulary trained on the independent Oxford5k [24] dataset and the top 100 nearest neighbors are retrieved using a fast tf-idf scoring [30]. Matches are geometrically verified by estimating the fundamental or essential matrix (depending on the availability of the focal length) as well as a homography matrix in a RANSAC scheme using the general USAC framework [25]. The relative rotation and translation direction are obtained by decomposing the fundamental, essential, or homography matrix (depending on the amount of inliers). The incremental and global reconstruction of cluster centers bases on the publicly available Theia library [32] and the final clustergraph optimization is realized in the g2o [18] framework. The following parameters were used within all the conducted experiments: $\mu_{min} = 5$, $\mu_{max} = 50$, $\lambda_s = 0.9$, $\rho_{l\,\max} = \rho_{g\,\max} = 10°$, $\eta = 0.2$, $\eta_{grow} = 0.15$.

3.2   Baselines

Throughout our expermiments we compare the proposed progressive pipeline against two linear pipelines (VisualSFM [36] and incremental Theia [32]) as well as to the recently published hybrid pipeline HSfM [5]. For an unbiased comparision, all geometrically verified matches were precomputed and imported into the various pipelines. The streaming part of the pipeline was skipped and matches directly loaded from disk.

Within the experiments we run the pipelines in different modes:

batch mode All images are added to the pipeline at once and the model is reconstructed without intermediate results. This is the classical SfM operation mode.

progressive mode Images are fed to the pipeline in a certain order and intermediate reconstructions are enforced. In VisualSFM this is realized by the "Add Image" option and in Theia we dictated the order in which views are added to an ongoing reconstruction by a slight code modification.

3.3   Randomized Image Order

One of the key contributions of the proposed pipeline is the ability of recovering from wrong connections between image pairs made in previous reconstruction
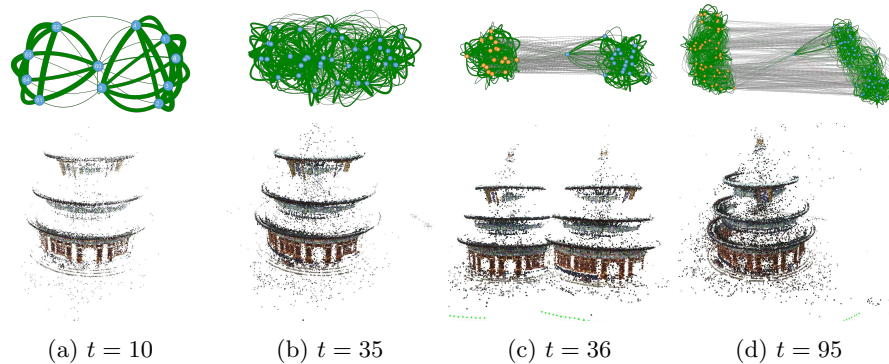
(a) $t = 10$ (b) $t = 35$ (c) $t = 36$ (d) $t = 95$

Fig. 4: Individual snapshots showing the viewgraph *(top)* and the sparse structure *(bottom)* of a reconstruction of the temple scene using a very unfortunate image ordering document the algorithm's recovery ability. Two separate parts of the temple are initially wrongly reconstructed in a single model (a-b) and are later successfully separated (c) and the real structure is recovered (d).

cycles. Wrong connections often occure in symmetric environments which is why we chose the publicly available TempleOfHeaven dataset [15] with a rotationaly symmetric structure for the first experiment. The dataset consists of 341 images taken in a regular spacing and perfectly demonstrates problems arising from unpleasant orderings. As a reference model, we reconstructed the scene using VisualSFM and restricted the matches to sequential matching in the original order. Figure 5 shows the development of the model as a function of the timestep t. We report the number of clusters before and after global cluster position estimation as well as the number of outlier cameras. A camera is considered to be an outlier if the position error is larger than half the minimum distance between two cameras in the reference model.

As expected both methods perform well in the "linear" case, where the images are fed to the pipeline in the original ordering. A single cluster is reconstructed and maximum of 8 cameras are classified as outliers. In a second experiment we randomly shuffled the order of images. After an initial set of 20 images, the baseline merges all clusters to a single one and as a result the number of outlier cameras increases linearly. The final 3D model is highly corrupted and only covers about a third of the temple (Figure 1a).

In contrast, the proposed pipeline creates up to 6 individual but locally consistent clusters. After 84 out of the 341 images the clusters are correctly localized into a single effective cluster by the cluster positioning (dashed line). Figure 1 shows the resulting reconstruction after 100 images as well as the corresponding clustered viewgraph.

3.4   Very Unfortunate Image Order

In order to demonstrate the recovery capabilities of our proposed pipeline we conducted an additional experiment on the TempleOfHeaven dataset. The temple
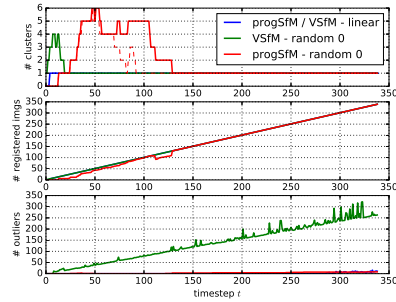
Fig. 5: Behavior of the incremental and progressive SfM pipelines on the TempleOfHeaven dataset. In the well behaving "linear" case *(blue)*, both pipelines reconstruct the temple. If the input order is shuffled the incremental pipeline gets stuck in a local minimum *(green)* whereas the proposed pipeline recovers and reconstructs the whole scene *(red)*.
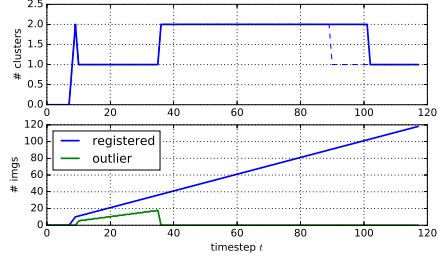
Fig. 6: Evolution of the clusters and images in a very unfortunately ordered image sequence. Despite the heavy rotational symmetry of the temple dataset, the pipeline is able to recover from a wrong configuration (timestep 36).
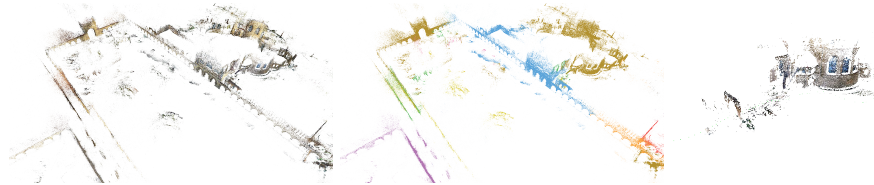

Fig. 7: Comparison of the proposed progressive pipeline *(left, center)* to the incremental SfM pipeline *(right)* on the Stanford dataset.

has a strong rotational symmetry which repeats with 60 degrees. One of the worst conditions for an algorithm would be if the image ordering had exactly this 60 degree periodicity. To test the algorithm, we created an artificial image sequence by feeding the images alternatively in the following order: $(I_0, I_{60}, I_1, I_{61}, ...)$. Figure 6 shows the number of clusters, images, and outliers in every timestep. Due to the high amount of matches between the symmetric part and the lack of images in between, the clustering algorithm sees enough evidence for putting all images into a single cluster (Figure 4a) and roughly every second image is an outlier. After the addition of the 36th image, the connectivity has sufficiently changed so that the two clusters are recognized and the 3D models are separated.

Due to the highly interleaved configuration the structure of the individual models cannot be recycled in this case and both clusters are reconstructed from scratch. At this stage there are no inter-cluster constraints available yet, which is

why the global cluster positioning does not succeed and the models are displayed as independent sub-models (Figure 4c). On the 95th image enough intra-cluster evidence is available s.t. the models can be placed into a common coordinate system (Figure 4d) and with the 101st image a single cluster is formed (this time the structure of the individual sub-models can be recovered and a simple merging is needed). The experiment shows that the proposed pipeline can recover from a wrong local minimum of the reconstruction.

3.5   Realworld Progressive Reconstruction

While the experiments so far demonstrated the capabilities of the progressive pipeline for unfortunate image orderings, its behavior in real world applications remains to be shown. Therefore we collected a series of 4516 images with three different mobile devices (*HTC Nexus 9*, *LG Nexus 5x*, and *Google Pixel*) consisting of 29 sequences on the main quad of the Stanford University. The sequences were then progressively reconstructed in an interleaved order (simulating multiple users collaboratively acquiring the pictures) as well as in the batch-processing mode. An additional similar experiment was conducted using the publicly available Quad dataset [4] consisting of 6514 images, mainly taken by *iPhone 3G*. Images were shuffled randomly and pushed to the reconstruction algorithms.

Table 1 shows the proposed pipeline compared to the incremental Theia [32] pipeline in batch and progressive modes. In addition we compare against the HSfM [5] pipeline purely in batch-processing mode. The proposed algorithm as the only pipeline can deal with the progressive scheme while the other get stuck in a local minimum and only reconstruct a very small subpart of the scene[5]. In the case of the progressive pipeline, we also report the number of failure recoveries during the whole reconstruction (the number of resets with a local cluster due to major topological changes). Figure 7 illustrates the final result of the proposed pipeline versus the local minimum of the incremental pipeline. The compute times of the proposed pipeline are comparable to existing pipelines despite the continuous flow of intermediate results. This is due to the fact that our pipeline practically never operates on the whole image sets, but only on the local clusters. This allows significantly faster execution times of the bundle adjustment.

We furthermore run our method on several of the datasets presented by [11]. We used random ordering of the crowd sourced data and pushed them to the progressive pipeline as done in the Quad [4] experiment. Figure 8 shows a sample result on the Radcliffe scene. While both HSfM and the incremental baseline method got confused by the duplicate structure, our pipeline successfully separated the front and rear views of Radcliffe. The two clusters were merged successfully into the correct configuration by the global cluster pose optimization. In contrary to [11], this is not done in a post processing step but erroneous connections are detected on-the-fly during the reconstruction. Table 1 shows the results on the resulting numbers on the dataset equally to the experiment

---

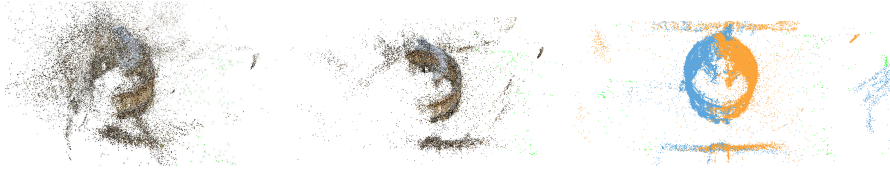[5] Experiments were repeated using multiple random orderings.

Fig. 8: Result on the Radcliffe dataset with HSfM (*left*), Theia (incr.)(*right*) and the proposed method (*right*).

			Stanford	Quad [4]	BigBen	Radcliffe Camera	Alexander Nevsky Cathedral	Branden-burg Gate	Arc De Triomphe
	total imgs [#]		4516	6514	402	282	448	175	434
batch	HSfM [5] imgs [#]		2,140	4,832	375	272	430	173	441
	time [s]		11,490	23,570	1,932	1,654	1,900	389	2,131
	Theia (incr.) [32] imgs [#]		3,298	5,462	394	278	443	173	410
	time [s]		34,749	158,853	2,385	1,307	3,687	1,018	3,348
progressive	Theia (incr.) [32] imgs [#]		51	527	394	280	418	173	416
	time [s]		7,605	215,064	2,026	1,473	2,268	1,141	3,516
	proposed imgs [#]		3,165	3,894	285	279	427	173	405
	time [s]		13,276	25,713	552	1,121	2,163	627	377
	clusters [#]		76	92	5	2	4	2	5
	recoveries [#]		14	29	1	0	5	1	1

Table 1: Evaluation of our pipeline versus the incremental Theia pipeline and HSfM. Only the proposed pipeline can successfully handle the progressive scenario.

before. Our method was able to separate duplicate structure in all datasets. As the reconstruction backbone of the presented pipeline bases on the well known incremental [32] and global [35] pipelines, the accuracy of the resulting structure is equals the accuracy of these pipelines.

4   Conclusions

We proposed a novel progressive SfM pipeline which addresses a multiuser-centric scenario, where a 3D model is simultaneously reconstructed from multiple image streams handled by a cloud-based reconstruction service. In contrary to existing work, our pipeline does not depend on the image order and does not require any a-priori global knowledge about image connectivity. The progressive pipeline avoids taking any binding decisions and is able to recover for erroneous configurations. A global viewgraph is incrementally built and maintained. The graph is clustered based on the local connectivity of cameras and individual clusters are reconstructed using either an incremental or a global reconstruction pipeline. In the last step, individual models are merged using a lightweight posegraph optimization just on the cluster centers. We demonstrated the effectiveness and efficiency of our pipeline on multiple dataset and compared it to existing solutions.

References

1. Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S.M., Szeliski, R.: Building rome in a day. Communications of the ACM **54**, 105–112 (2011)
2. Arandjelovic, R., Zisserman, A.: Three things everyone should know to improve object retrieval c. In: CVPR. pp. 2911–2918 (2012)
3. Chatterjee, A., Govindu, V.M.: Efficient and robust large-scale rotation averaging. In: ICCV. pp. 521–528 (2013)
4. Crandall, D., Owens, A., Snavely, N., Huttenlocher, D.: SfM with MRFs: Discrete-continuous optimization for large-scale structure from motion. PAMI **35**(12), 2841–2853 (December 2013)
5. Cui, H., Gao, X., Shen, S., Hu, Z.: Hsfm: Hybrid structure-from-motion. In: CVPR. pp. 1212–1221 (2017)
6. Cui, Z., Tan, P.: Global Structure-from-Motion by Similarity Averaging. In: CVPR. pp. 864–872 (2015)
7. Davison, A.J.: Real-time Simultaneous Localisation and Mapping with a Single Camera. In: ICCV. pp. 1403–1410 (2003)
8. Farenzena, M., Fusiello, A., Gherardi, R.: Structure-and-motion pipeline on a hierarchical cluster tree. In: ICCV Work. pp. 1489–1496 (2009)
9. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981)
10. Havlena, M., Torii, A., Pajdla, T.: Efficient Structure from Motion by Graph Optimization. In: ECCV. pp. 1–14 (2010)
11. Heinly, J., Dunn, E., Frahm, J.M.: Correcting for duplicate scene structure in sparse 3d reconstruction. In: ECCV. pp. 780–795. Springer (2014)
12. Heinly, J., Schönenberger, J.L., Dunn, E., Frahm, J.M.: Reconstructing the World * in Six Days. In: CVPR (2015)
13. Horn, B.K.P.: Closed-form solution of absolute orientation using unit quaternions. J. Opt. Soc. Am. A **4**(4), 629 (1987)
14. Irschara, A., Zach, C., Bischof, H.: Towards wiki-based dense city modeling. In: ICCV (2007)
15. Jiang, N., Tan, P., Cheong, L.F.: Seeing double without confusion: Structure-from-motion in highly ambiguous scenes. In: CVPR. pp. 1458–1465 (2012)
16. Kang, Z., Medioni, G.: Progressive 3D model acquisition with a commodity handheld camera. In: WACV. pp. 270–277 (2015)
17. Kneip, L., Scaramuzza, D., Siegwart, R.: A Novel Parametrization of the Perspective-Three-Point Problem for a Direct Computation of Absolute Camera Position and Orientation. In: CVPR (2011)
18. Kummerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W.: G2o: A general framework for graph optimization. In: ICRA. pp. 3607–3613 (may 2011)
19. Locher, A., Perdoch, M., Van Gool, L.: Progressive Prioritized Multi-view Stereo. In: CVPR. pp. 3244–3252 (2016)
20. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**(2), 91–110 (2004)
21. Martinec, D., Pajdla, T.: Robust rotation and translation estimation in multiview reconstruction. In: CVPR (2007)
22. Moulon, P., Monasse, P., Marlet, R.: Global fusion of relative motions for robust, accurate and scalable structure from motion. In: ICCV. pp. 3248–3255 (2013)

23. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: a versatile and accurate monocular slam system. IEEE Transactions on Robotics **31**(5), 1147–1163 (2015)
24. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR (2007)
25. Raguram, R., Chum, O., Pollefeys, M., Matas, J., Frahm, J.M.: Usac: a universal framework for random sample consensus. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 2022–2038 (2013)
26. Roberts, R., Sinha, S.N., Szeliski, R., Steedly, D.: Structure from motion for scenes with large duplicate structures. In: CVPR. pp. 3137–3144 (2011)
27. Schönberger, J.L., Radenovìc, F., Chum, O., Frahm, J.M.: From single image query to detailed 3d reconstruction. In: CVPR. pp. 5126–5134 (June 2015)
28. Schönberger, J.L., Frahm, J.M.: Structure-from-Motion Revisited. In: CVPR (2016)
29. Sinha, S.N., Steedly, D., Szeliski, R.: A multi-stage linear approach to structure from motion. In: ECCV. pp. 267–281. Springer (2010)
30. Sivic, J., Zisserman, A.: Video Google: a text retrieval approach to object matching in videos. In: ICCV (2003)
31. Snavely, N., Seitz, S.M., Szeliski, R.: Modeling the world from internet photo collections. International journal of computer vision **80**(2), 189–210 (2008)
32. Sweeney, C.: Theia multiview geometry library: Tutorial & reference. http://theia-sfm.org
33. Sweeney, C., Fragoso, V., Höllerer, T., Turk, M.: Large scale sfm with the distributed camera model. In: 3DV. pp. 230–238. IEEE (2016)
34. Sweeney, C., Sattler, T., Tobias, H., Turk, M., Pollefeys, M.: Optimizing the Viewing Graph for Structure-from-Motion. In: ICCV. pp. 801–809 (2015)
35. Wilson, K., Snavely, N.: Robust global translations with 1dsfm. In: European Conference on Computer Vision. pp. 61–75. Springer (2014)
36. Wu, C.: Towards linear-time incremental structure from motion. In: 3DV. pp. 127–134 (2013)
37. Zach, C., Irschara, A., Bischof, H.: What can missing correspondences tell us about 3D structure and motion? In: CVPR (2008)
38. Zach, C., Klopschitz, M., Pollefeys, M.: Disambiguating Visual Relations Using Loop Constraints. In: CVPR (2010)