Correcting for Duplicate Scene Structure in Sparse 3D Reconstruction

Jared Heinly, Enrique Dunn, and Jan-Michael Frahm

The University of North Carolina at Chapel Hill

Abstract. Structure from motion (SfM) is a common technique to recover 3D geometry and camera poses from sets of images of a common scene. In many urban environments, however, there are symmetric, repetitive, or duplicate structures that pose challenges for SfM pipelines. The result of these ambiguous structures is incorrectly placed cameras and points within the reconstruction. In this paper, we present a post-processing method that can not only detect these errors, but successfully resolve them. Our novel approach proposes the strong and informative measure of conflicting observations, and we demonstrate that it is robust to a large variety of scenes.

Keywords: Structure from motion, duplicate structure disambiguation

1 Introduction

In the last decade, structure from motion (SfM) has been taken out of the lab and into the real world. The achieved progress is impressive and has enabled large-scale scene reconstruction from thousands of images covering different scenes around the world [2, 9, 11, 27]. In crowd-sourced reconstructions of large-scale environments, SfM methods do not have any control over the acquisition of the images, leading to many new challenges. One major challenge that arises is the ambiguity resulting from duplicate structure, i.e. different structures with the same appearance. Fig. 1 shows an example of duplicate scene structure on Big Ben, where every side of the clock tower has the same appearance. SfM methods often erroneously register these duplicate structures as a single structure, yielding incorrect 3D camera registrations (see Fig. 1). We propose a method that can correct the misregistrations caused by the duplicate scene structure in the final SfM model (see Fig. 1 for the corrected reconstruction of Big Ben).

To correct the misregistration caused by duplicate structure, it is important to understand the nature of the ambiguity that causes the error. The most common SfM methods operate as an incremental reconstruction, i.e. they start from an initial pair or triplet and subsequently extend the reconstruction one-by-one for each remaining image. However, the decision of which image to add next to the reconstruction is not arbitrary. This choice is typically driven by an image similarity metric used to find images that are similar to the ones already registered [2, 4, 7, 11, 18, 20]. It is within this process that sometimes SfM algorithms

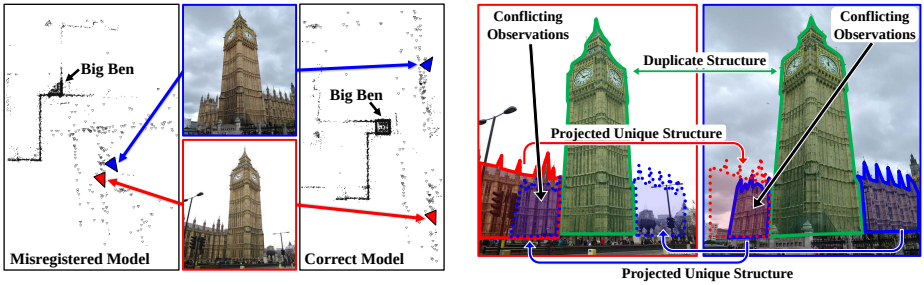

Fig. 1. Left: example camera placements in a misregistered and correct SfM model. Right: example illustration of conflicting observations between two images.

select images which do not actually overlap with the current reconstruction, but do overlap with a different instance of the duplicate structure. These images are then erroneously registered to the wrong instance of the duplicate structure. An indication for this erroneous registration is that only points on the duplicate structure register. Unfortunately, a priori knowledge of the duplicate structure is not available at registration time. Subsequent registrations extend the reconstruction further, but with the two copies of the duplicate structure combined into a single model. This erroneous reconstruction contains incorrectly placed unique structures due to the incorrect registration of the duplicate structure.

Fig. 2 shows an incremental SfM pipeline that results in erroneous geometry. The images are sequentially registered and added to the reconstruction, and upon reaching the fourth image in the set (which is taken from a different location than the first three), it registers, but only to the facades of the tower. This registration is incorrect, as the camera should have been rotated 90° around the tower. Now, when registering the remaining cameras (which should also be rotated 90°) they will correctly register to the fourth image and start to triangulate 3D structure. However, because of the fourth camera's mislocation, the new structure (and camera poses) will be incorrectly placed within the scene.

Given the difficulties of detecting erroneous registration during reconstruction, we propose a method which can correct the errors upon completion of SfM. Our method identifies incorrectly placed unique scene structures, and from this we infer the points belonging to the duplicate structure. Once our system identifies the duplicate structure, it attempts registration of cameras and points using only the distinct unique structures to obtain a correct model.

2 Related Work

Duplicate structure has been of recent interest in the research community and has motivated a variety of applications [3, 8, 14, 22, 23, 25, 28]. Generally, there are different types of duplicate scene structures, ranging from duplicate instances caused by 3D rotational symmetries, separate identical surfaces, or repetitive or mirrored structures often found on facades (a survey of symmetry is provided in

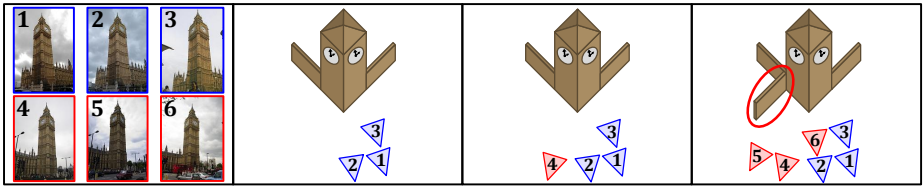

Fig. 2. Illustration of how duplicate structure causes incorrect reconstructions. Left to right: 1) Input images ordered for reconstruction, 2) Reconstruction after first three images, 3) Fourth camera registers, but only to duplicate structure on Big Ben facades, 4) Remaining images register, and an erroneous structure is created (circled in red).

[17]). Duplicate structures are prone to lead to misregistered scene reconstructions, though mirror symmetries do not typically contribute to these errors.

Symmetric and repetitive structures can generally be detected in images through techniques that detect symmetric or repetitive patterns [5, 6, 13, 15, 16, 31, 32]. Methods have leveraged these patterns for urban geolocalization [3, 22, 25] and reconstruction of a scene from only a single image [14, 23, 28]. Furthermore, there has been recent work on utilizing symmetry as a constraint in bundle adjustment to improve the accuracy of an SfM result [8].

The class of duplicate structures originating from 3D rotational symmetries and different identical copies of the same surface in the scene is typically not detectable by purely image-based measures. It is this class of duplicate structures that we target for correction. Next, we discuss several related approaches also aiming at mitigating the effects of this class of duplicate structures.

Zach *et al.* [29] introduce the concept of *missing correspondences*. The main idea is to identify image triplets that have consistent sets of three-view feature observations, while minimizing the number of features observed in only two of the images. The valid triplets are then combined into the correct reconstruction. The intuition is that if a substantial fraction of observations are missing from the third image, then that third image is incorrectly registered. They use a Bayesian framework that has a belief network for image triplets in order to verify their correctness and relation to each other. However, the authors enforce the very conservative assumption that a pair of images deemed to be incorrectly registered cannot exist within the same single reconstruction. This is conservative as the images could be viewing a duplicate instance within a single scene that could later be reconciled. In contrast, our method performs the inference over all cameras in the SfM model, and allows incorrectly matched images to correctly register to different parts of the same single reconstruction.

Roberts *et al.* [21] also utilize the idea of missing correspondences by exploiting them as a metric within an expectation-minimization framework to identify incorrectly registered image pairs. They improve over the original formulation of Zach *et al.* [29] by relying on image timestamps to provide approximate sequence information. Hence, they implicitly assume an association between the temporal and spatial order of the images. This allows their method to function in

potentially challenging scenarios, but prevents it being applied to unordered Internet photo collections. Our method, in contrast, does not require any temporal information for the images and does correctly handle Internet photo collections.

Loop constraints were exploited by Zach *et al.* [30] to detect incorrect camera registrations. Their algorithm analyzes loops (cycles) of connected cameras in a graph in which cameras are nodes and pairwise transformations between cameras are edges. When traversing a loop and chaining the transformations between the images, one should achieve the identity transform (with some natural drift) upon matching to the first image in that loop (loop closure). If this criterion is broken, then at least one of the transformations in the loop is assumed to be incorrect. Even if there is no drift, assuming an identity transformation over a loop is not sufficient for detecting all erroneous loops in the graph [12].

Jiang *et al.* [12] minimize the number of missing correspondences across the entire reconstruction instead of triplets [29]. To find the minimum they evaluate a set of possible reconstructions and make the key assumption that the set of images under consideration forms one connected model. This assumption fails when the images are actually from two separate scenes, or when there is insufficient overlap between views of a duplicate structure. Our method overcomes these shortcomings by correctly splitting and maintaining separate partial models of the scene (such as datasets 7–9 in Fig. 7).

Instead of using missing correspondences explicitly, Wilson and Snavely [26] utilize the bipartite local clustering coefficient over the visibility graph of an SfM model. This measure identifies tracks of 3D point observations that are suspected of incorrectly linking separate structures. They require prior knowledge of the desired number of splits as an input. Then, their method removes a large number of erroneous tracks and will split incorrectly generated models. Their primary mode of failure is over-segmentation (even splitting already correct models [26]). They do not propose a conclusive technique to merge the split components as they only use medoid SIFT features, matching, and estimating an alignment via RANSAC to fix oversplitting in one of their datasets. Additionally, the authors state that their method is not well suited for the "laboratory-style" datasets of previous papers (datasets limited to a few hundred images). Our method, in contrast, leverages an automatic merging technique, and circumvents oversplitting by detecting if an SfM model is already correct. Furthermore, we demonstrate successful results on both laboratory and large-scale real-world datasets (Fig. 7).

In summary, missing correspondences (an intuition used by many previous methods) report on structure that was expected. This implicitly assumes the repeatability of the correspondence mechanism, which can fail because of noise, occlusion, or changes in viewpoint [19]. This assumption severely limits the range and type of incorrect registrations that can be detected. Therefore, the remaining unsolved challenges for model correction include robustly handling duplicate instances without oversplitting, while at the same time being able to correctly recover one or more final models (depending on the configuration of the underlying scene). It is this challenge that our paper successfully addresses.

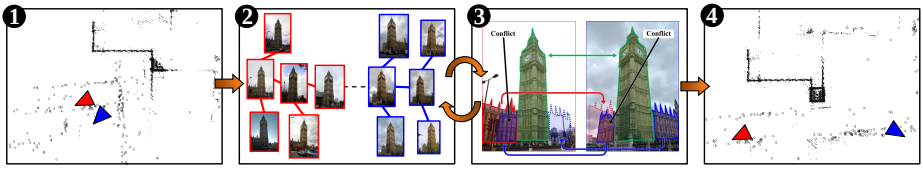

Fig. 3. Method overview: ❶ Input SfM model, ❷ Candidate camera graph splits, ❸ Conflicting observations, and ❹ Model merging.

3 Algorithm

We propose using *conflicting observations* to identify the incorrect registrations caused by duplicate structure as a more powerful alternative to using missing correspondences. Conflicting observations are 3D points that when projected, using the corresponding 3D reconstruction information, into and across pairwise registered images, conflict in their spatial location, i.e. there are observations of alternative structures in the same spatial location in the image plane. For instance, consider two separate views of duplicate structure (like the facades of Big Ben, shown in Fig. 1). Each image contains observations of the duplicate 3D points, but the observations of the disjoint secondary structures in the scene are unique. The unique structure in the first image, when projected into the second, overlaps with the second image's unique structure. It is this unique structure that we analyze for conflict, as it is separate from the duplicate object and provides distinct information about the layout of the scene.

3.1 Overview

Given the difficulty of detecting the duplicate structures during the initial registration, our method is a post-processing step to SfM, i.e. the input to our system is the output of a sparse 3D reconstruction pipeline. The registered cameras and the 3D points define a *camera graph* (CG) where nodes correspond to the cameras, and edges exist between nodes whenever the two cameras view a common set of 3D points. Our method uses a recursive processing procedure whose goal is to determine if there are any errors in the current reconstruction (step ❷ and ❸ in the method outline shown in Fig. 3). During this procedure, step ❷ proposes candidate CG splits, dividing the cameras in the current CG into two subgraphs, which are then evaluated for conflict in step ❸.

For step ❸, each 3D point of the model is assigned to one of three classes: points seen by cameras in both subgraphs (potentially duplicate structure), points seen by only the cameras in the first subgraph (unique structure for this subgraph), and points seen only by the cameras of the second subgraph (unique structure in the second subgraph). Then, the unique structures are used to test for conflict between the two subgraphs by counting the number of conflicting observations between camera pairs where both cameras observe common points but the cameras originate from different subgraphs. The number of conflicting

observations for such a camera pair provides a conflict score for the CG split. If there is considerable conflict between the subgraphs, the CG is permanently split and the two subgraphs are independently recursively evaluated for further conflict. If a CG has no considerable conflict it is accepted as valid.

After identifying the separate subgraphs of the model, our method (step ❹) attempts to merge the subgraphs to recover the correct model. It proposes candidate alignments between the split 3D models (subgraphs), and then evaluates the conflict for the merge (leveraging conflicting observations). If an alignment with sufficiently low conflict is found, then the two subgraphs are combined; otherwise, they are output as independent components of the reconstruction.

To review, we propose a conflict measure to detect overlap between structures that should be spatially unique. Applying this measure both to candidate CG splits and merges, we can successfully correct a misregistered SfM model.

3.2 Step ❶: Input Reconstruction

As our method is a post-processing step for SfM, we require as input typical outputs of such a system [2, 11, 24, 27]. In our results we used [27] and [24]. Our method assumes the availability of known 3D camera poses (positions and orientations), the original images (for step ❸ of our method), and the locations of the 3D points and their visibility with respect to each camera. To perform sub-model merging (step ❹), the original feature inliers are required, i.e. which features were verified geometric inliers between a pair of images. In the case that some of the above input information is missing, (e.g. SfM without correspondences [10]) it can always be computed from the images and camera poses.

3.3 Step ❷: Candidate Camera Graph Splits

We wish to generate candidate CG splits, where each split divides the cameras into two distinct subgraphs. These two subgraphs will then be passed to step ❸, which will evaluate the conflict between them.

Naïve potential splits can be proposed by enumerating all possible camera groupings. Given that the CG contains m cameras and is potentially densely connected, there would be at most $2^{m-1}-1$ possible ways to assign the cameras to two different groups where each is a connected component (subgraph). This is exponential in m and computationally prohibitive for most large-scale models.

Minimum Spanning Tree To reduce the number of candidate splits, we propose to leverage a minimum spanning tree (MST) representation similar to the one constructed in [12], and illustrated in Fig. 4, step 2. Jiang *et al.* [12] assigned to each edge a weight that was inversely proportional to the number of 3D point observations shared between the two cameras. We adopt a similar idea, but reformulate the edge cost to account for the fact that if duplicate structure is present, many of the images will have a large number of common points. Accordingly, this raw number of points should not be overemphasized, hence our edge cost

leverages the following ratio where e_{ij} is the edge cost between cameras i and j, and O_i, O_j are the sets of 3D points that are visible in each camera:

$$e_{ij} = 1 - \frac{|O_i \cap O_j|}{|O_i \cup O_j|} \tag{1}$$

Conceptually, an MST formed using this edge cost tends to link cameras together that share similar content, and in contrast to [12], avoids biasing the results for cameras with relatively few or many point observations.

If two cameras see the same set of 3D points then e_{ij} will be zero. Accordingly, two views seeing the same instance of the duplicate structure and the corresponding surrounding unique structure will have a low edge cost. We denote these edges as *desired edges*. Conversely, two cameras, which see two different instances of the duplicate structure but do not see a common unique structure, will have a significantly higher e_{ij} value and are denoted as *confusing edges*.

Intuitively, the MST prefers utilizing desired edges (low edge cost) to connect cameras seeing the same instance of the duplicate structure and will only retain confusing edges (high edge cost), when necessary, to connect cameras seeing different instances of the duplicate structure. Accordingly, the MST will group cameras together that see the same instance of the duplicate structure and the confusing edges will bridge between these groups. With this observation, we can now limit our CG splits to those that are defined by the MST, as removing a confusing edge creates two separate subgraphs defining a candidate split. Defining the search space of candidate model splits in terms of an MST representation reduces the number of potential candidate splits from one that is exponential in the number of cameras to $m - 1$, the edges in the MST. Refer to Fig. 4, steps 1-3 for an illustration of a split in the MST resulting in two subgraphs.

3.4 Step ❸: Conflicting Observations

After leveraging the MST, the next step evaluates the conflict between the two subgraphs produced by each split of the graph to obtain a reduced set of splits.

Common and Unique Structure First, our approach classifies each 3D point into one of three categories as defined below:

$$\begin{aligned} \mathbb{D} &= \{P_k : \ (\exists O_{ik} \in O) \wedge \ (\exists O_{jk} \in O)\} \\ \mathbb{U}_1 &= \{P_k : \ (\exists O_{ik} \in O) \wedge \neg(\exists O_{jk} \in O)\} \\ \mathbb{U}_2 &= \{P_k : \neg(\exists O_{ik} \in O) \wedge \ (\exists O_{jk} \in O)\} \end{aligned} \tag{2}$$

where $\{P\}$ is the set of 3D points, P_k is the k-th point in $\{P\}$, O represents the visibility of points in the cameras ($O_{ik} \in O$ if P_k is visible in camera i, otherwise it is false), and i, j referring to camera i in the first set of cameras and the j-th camera in the second set of cameras. The common points (the candidate duplicate structure) between the two camera groups are denoted by $\mathbb{D}$, with $\mathbb{U}_1, \mathbb{U}_2$ denoting the unique points to each subgraph.

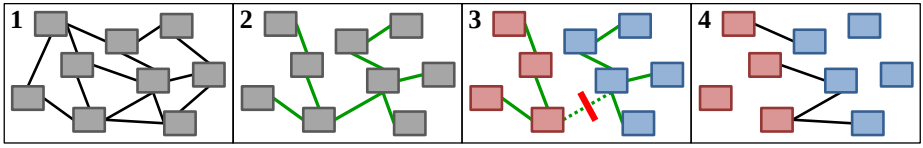

Fig. 4. 1. Full camera graph, 2. Minimum spanning tree, 3. Candidate minimum spanning tree split defining two camera groups, 4. Camera pairs from the full camera graph that were assigned to different groups

To improve our robustness to noisy scene geometry we enforce a minimum number of observations for each 3D point, where i is any arbitrary camera:

$$P = \left\{P_k : |\{i : O_{ik} \in O\}| \geq \rho\right\} \tag{3}$$

By setting $\rho = 3$ (as we did in all of our experiments), we maintain only those 3D points that are more likely to be stable and properly triangulated.

Split Camera Pairs To evaluate the conflict of a candidate split, we analyze the image pairs from the full CG that had originally matched but are now split due to the images being assigned to different subgraphs. For an example of such pairs, refer to Fig. 4, step 4. We require a minimum number γ of 3D points that the cameras of a pair must observe in common in order to avoid images that are weakly connected as they do not represent a reliable measure of conflict.

Next, we project the unique points observed in each image to the other image in the pair, and test for conflicting observations. To mitigate the effects of occlusion or large viewpoint change, only cameras observing the same points from a similar surface incidence angle are considered. The difference in surface incidence angle β between the cameras i and j with their centers (C_i, C_j) and the centroid of their common 3D points $\bar{p}$, where $\bar{p} = \texttt{mean}\big(\{P_k : (\exists O_{ik} \in O) \wedge (\exists O_{jk} \in O)\}\big)$ (also by construction, $\{P_k\} \subseteq \mathbb{D}$) is defined as:

$$\beta = \texttt{arccos}\left(\texttt{dot}\left(\frac{C_i - \bar{p}}{||C_i - \bar{p}||}, \frac{C_j - \bar{p}}{||C_j - \bar{p}||}\right)\right) \tag{4}$$

Given the limitations in matching over large viewpoint changes, our method disregards camera pairs with a difference in surface incidence angle β greater than a predefined threshold θ. The threshold θ is chosen according to the robustness of the SfM system's features with respect to viewpoint changes (for example, around 20° for SIFT features in our experiments).

Splitting at each of the $m - 1$ MST edges leads to a cubic complexity of the evaluated splits given that there is potentially a quadratic number of pairs of cameras (for a fully connected graph) for a given split. To boost efficiency, instead of evaluating all split camera pairs, we propose to inspect only those pairs with the smallest surface incidence angles β, which still allows us to detect the conflict. Specifically, we can inspect the s smallest pairs, giving quadratic

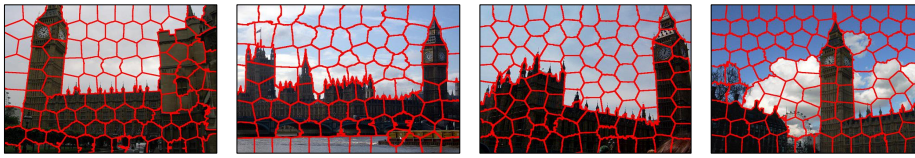

Fig. 5. Example SLICO [1] superpixel segmentations.

overall complexity when s is a function of m, or a fixed number of smallest pairs (e.g. $s = 100$) to achieve a linear overall complexity. In our experiments, we have found both strategies to be valid, and thus opt for the linear time approach.

As opposed to using an MST to determine the locations to evaluate conflict, one could imagine that we could instead use a clustering or graph-cutting approach on the full CG. We avoided these as they would necessitate computing conflict (t from the next step) between all (or a very large fraction) of the camera pairs which could quickly become computationally prohibitive for larger CGs.

Conflict Measure Our conflict measure leverages the unique points in the scene by projecting them into the other image of the camera pair, and expecting that they should not overlap with the unique points of that image. If there is substantial overlap, then we have reason to believe that there is an error in the current reconstruction (refer to Fig. 1 for an example). How does one measure overlap in two projected sparse point sets? Ideally, spatially nearby (in the image) unique points on the same structural surface should conflict, whereas nearby points on separate surfaces that lie at different depths should not conflict.

To establish the surface association of the sparse points, we leverage SLICO superpixels [1], whose only parameter is the desired number of superpixels. SLICO will automatically adapt to the texture in the image in order to maintain regular-sized superpixels (examples of which are shown in Fig. 5). To guard against arbitrary superpixel divisions along the same structural surface, we perform multiple (eight in our experiments) different segmentations of each image by providing mirrored and rotated versions of an image to SLICO. This proved to generate a different segmentation for each (there are only eight different combinations of 90° rotations and mirror operations). With these segmentations, we now define two points to be nearby if their projections lie within the same superpixel in any of the candidate segmentations.

By leveraging the current potential duplicate ($\mathbb{D}$) and unique point sets ($\mathbb{U}_1$, $\mathbb{U}_2$) for every single camera pair, we can evaluate the subsets of these points that are currently visible in the camera pair to identify the conflicting observations.

While we focus on conflicting unique points, the locations of the common points ($\mathbb{D}$) also provide useful information. Both [21] and [29] emphasized the usefulness of considering the spatial location of these observations. For instance, the presence of a matched point between two images would down-weigh the contribution of any nearby missing correspondences. Utilizing this concept, we

obtain reduced sets (U_1, U_2) by ignoring unique points (from $\mathbb{U}_1, \mathbb{U}_2$) that occupy the same superpixel as a common point (from $\mathbb{D}$) in any of the segmentations.

For a given pair of images, we define the conflict t between them to be the minimum number of points from U_1 or U_2 that conflict in both images. If $\texttt{proj}(U_1)$ is the projection of the points U_1 into the second image, and $\texttt{proj}(U_2)$ is the projection into the first, then the conflict t is defined as:

$$N = \texttt{near}\big(U_1, \texttt{proj}(U_2)\big) \cap \texttt{near}\big(U_2, \texttt{proj}(U_1)\big) \tag{5}$$

$$t = \texttt{min}\Big(\big|\{u_1 : u_1 \in U_1 \wedge u_1 \in N\}\big|, \big|\{u_2 : u_2 \in U_2 \wedge u_2 \in N\}\big|\Big) \tag{6}$$

where $\texttt{near}()$ returns the points that are nearby as defined by the superpixel segmentations. To provide further intuition, consider the case where one unique point from U_1 conflicts with many from U_2. This single point from U_1 could be an extraneous structure, and should not count as significant conflict even though it conflicts with many points from U_2. Therefore, we leverage the minimum of the two set sizes, as this enforces a stronger indication of the presence of conflict.

Given the conflict t for a single split camera pair, the conflict over the split CG is the average of the conflicts from all split camera pairs between the two subgraphs (step 4 in Fig. 4). This average is then independently computed for each split in the MST. If the MST split with the highest conflict is above a predefined threshold τ, we remove the corresponding edge from the MST to generate two separate subgraphs. Each of these subgraphs is then processed by reapplying steps ❷ through ❹ with the exception of not recomputing the MST. This is recursively repeated until we are left with a set of subgraphs that are free from conflicting observations, i.e. their conflict is below our threshold τ.

3.5 Step ❹: Model Merging

Once we have a set of camera subgraphs that are free from significant conflict, we now seek to merge them together and recover a correct reconstruction (if one is possible, as the images may come from entirely separate scenes).

The key concept that we leverage here is the idea of *disconnected inliers*. Disconnected inliers are pairs of 3D points whose 2D features had been identified as inliers during the two-view geometric verification of an image pair in the SfM processing. However, due to the duplicate structure in the scene (or potentially other factors, such as feature mismatches) the inlier ended up being triangulated as two separate 3D points. Therefore, to recover candidate merges, we estimate 3D similarities that would align and reconnect the disconnected inlier points.

To estimate the similarity between the split subgraphs (and their associated disconnected inliers), we leverage a RANSAC technique, once again enforcing that a candidate solution should be made up of at least γ points in order to be considered further. Note that when generating candidate similarities, we ignore any common points that are shared between two or more subgraphs (a union of the final $\mathbb{D}$ sets from each of the split subgraphs, which we denote $\mathbb{D}_{\text{Final}}$). These points are the final duplicate structure, and as such, are not reliable for

merging as they define the duplicate structure within the scene. However, once a candidate similarity has been proposed, we recompute the similarity inlier set using all disconnected inliers (even including duplicate points).

For each candidate solution, we transform the camera poses using the similarity T and update the unique 3D point structure of the subgraph. The points shared between subgraphs (the duplicate structure) are duplicated and transformed using T to correctly represent the duplicate scene structure. Then, the conflict between the two merged subgraphs is computed. In order to compute this conflict, inliers to T are identified as common structure $\mathbb{D}$. Furthermore, we load any existing 2D inliers for an image pair (from two-view geometric verification) and mark superpixels containing the 2D inlier locations as common structure. We do the latter to recover correspondences that would otherwise not have existed because the SfM algorithm ended up placing the cameras at separate locations within the scene (and choosing not to incorporate the relative pose initially computed between the images).

If the conflict is less than τ, the merge is considered correct. Otherwise, we ignore the points that were inliers to T, and attempt to estimate a different similarity. This continues until either a correct merge is found, or no solution can be computed with γ or more inliers. By repeating this process between all split camera groups, we merge all subgraphs that have valid overlapping geometry and recover a more complete and correct representation of the scene.

Now that we have correctly identified the duplicate structure within the scene ($\mathbb{D}_{\text{Final}}$), this information can be used to allow additional images to be registered to the reconstruction. For instance, when registering a new image, the image should not be allowed to register only to points contained within $\mathbb{D}_{\text{Final}}$, but should instead incorporate unique points not in $\mathbb{D}_{\text{Final}}$. In this manner, new images will not be incorrectly registered to the duplicate structure. Furthermore, this process could be embedded into an incremental SfM pipeline, so that disambiguation would occur at certain intervals to detect and mark as confusing any duplicate structure that is found. This would successfully addresses the source of the problem (the behavior of incremental SfM) as described in Section 1.

4 Results

In order to evaluate our method, we applied it to a wide variety of datasets (see Table 1 for a detailed overview of the datasets that led to misregistered models, not including those in Fig. 6 that were already correct). First, we evaluated our method on datasets from previous papers [12, 21, 26], using their qualitative evaluation metric for the correct camera and model arrangement. Fig. 7 (models 3, 4, 10–12) illustrates the output of our method on these existing benchmark datasets. Upon close inspection, we perform equally well or better than previous methods on their datasets as we split their models correctly (avoiding oversplits) and merge the ones that are mergeable (datasets 10–12). For instance, in dataset 4, we avoid oversplitting the front from the back of Notre Dame, as in [26], though our method did output a small nighttime model, as there were day and nighttime

Table 1. Statistics showing the number of cameras and points in the dataset, the time required for our method, and the software used to generate the initial reconstruction.

	Dataset Name	# Cams	# Points	Time	SfM
1	Big Ben (using iconics)	13590	167375	20.5 m	[27]
2	Berliner Dom	1618	245079	9.8 h	[27]
3	Sacre Coeur [26]	1112	378882	4.4 h	[24]
4	Notre Dame [26] (iconics)	885	176099	1.8 h	[24]
5	Alexander Nevsky Cathedral	448	92948	16.6 m	[27]
6	Arc de Triomphe	434	93452	16.3 m	[27]
7	Radcliffe Camera	282	71107	31.9 m	[27]
8	Church on Spilled Blood	277	76582	1.4 h	[27]
9	Brandenburg Gate	175	23933	3.0 m	[27]
10	Indoor [12]	152	69632	3.1 m	[27]
11	Cereal [21]	25	12194	36 s	[27]
12	Street [21]	19	7607	39 s	[27]

versions of local image features that corresponded to the same geometry, and thus generated conflict. We did leverage iconic selection [11] for this dataset, and for dataset 3, we used the set of images that viewed Sacre Coeur in the covering subgraph from [26]. In addition, we also ran our method on the Seville Cathedral and Louvre datasets from [26]. For the Seville Cathedral, our method split the model into three main components, whereas [26] had oversplit into four. For the Louvre, we had to set $\tau = 2.0$, and were able to split it into two main sub-models. As a note, [26] split the Louvre into three sub-models, the difference being that their method split two components that were correctly oriented with respect to each other but reconstructed at different scales.

To validate that our method only alters misregistered models, we tested it on reconstructions that were already correct (eight of which are shown in Fig. 6). In these cases, our method correctly identified them as having negligible conflict and did not attempt further processing.

Beyond benchmark comparisons, we evaluated our approach on seven novel datasets downloaded from Flickr (Fig. 7, models 1, 2, 5–9). These datasets contain several duplicate structures, and are common examples of the types of ambiguities found in urban scenes. They also represent the originally targeted (and previously unsolved) challenge of robustly disambiguating duplicate structure without making a priori assumptions on the number of correct final models to output. For datasets 1, 2, 5, 6, our method correctly split and merged the reconstructions into a single large model. It even handled the difficult challenge of Big Ben (dataset 1) where there were three split subgraphs in the reconstruction. For dataset 6, our method did output a small nighttime model. The remaining three novel datasets (7–9) successfully split and then remained as separate models, as we manually verified that there were insufficient overlapping views to support a merge. The primary reason for this lack of overlapping views is the layout of the scene itself, where photographers are limited in the number of accessible vantage points from which a desirable photo can be taken.

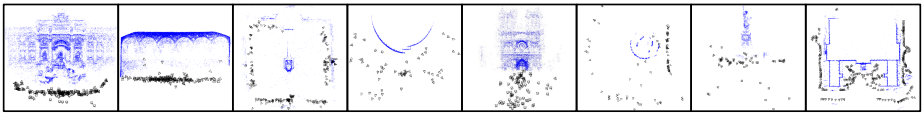

Fig. 6. Example error-free reconstructions correctly identified by our method. From left to right: Trevi Fountain, Sistine Chapel Ceiling, Harmandir Sahib, Colosseum, Notre Dame Facade, Stonehenge, Statue of Liberty, and CAB [8].

For further comparison, we ran the code from [26] on our novel datasets. For Big Ben, [26] split it into only two sub-models, failing to distinguish the front and back of the tower. The Berliner Dom split into five models, two of which failed to split the front of the building from the side. Alexander Nevsky Cathedral failed to split at all, but the Arc de Triomphe was correctly split into two components. Radcliffe Camera was oversplit into three models. The Church on Spilled Blood was split into two correct models, but the third smallest camera group was discarded as it was not included in the covering subgraph. Additionally, the Brandenburg Gate remained as one model, but all cameras from the back side of the structure had been discarded. Note that in the generation of these results, we extracted 2D tracks (the required input to [26]) from already triangulated 3D points. This should be a benefit to the system, as they are already cleaner than the tracks typically used as input in [26].

While our method exercises approximately linear computational complexity, for large datasets the corresponding overhead can still be reduced by leveraging the idea of iconic view selection from Frahm *et al.* [11]. Please note this reduction is not required but provides computational savings. For the Big Ben dataset (13,590 images) we extracted 402 iconics in approximately linear time. Then, we split and merged a reconstruction built from only iconic images and registered the remaining cluster images to the reconstruction by attaching them to their iconic, along with other nearby images of the same camera subgraph. By only registering to images within the same subgraph, we ignore the effect of multiple instances of duplicate structure in the scene and only register to the instance viewed in the current subgraph. Leveraging the iconics yields the desired corrected 3D model while boosting efficiency due to the significantly reduced number of images considered in the splitting and merging.

While our method performed well on the datasets that we tested, the key assumption enabling our method is the existence of unique structure. If, for instance, the set of images or resulting reconstruction consists only of duplicate structure, our method cannot identify that the images may have come from different instances of the duplicate structure. However, this is rarely the case for real-world datasets, thus making our approach a viable option for general use.

For all experiments (except where previously noted) the same set of parameters ($\gamma = 8$, $\theta = 20°$, 100 superpixels per image, $s = 100$, and $\tau = 7.0$) was used, underlining the robustness of our method. Execution times are from our MATLAB implementation on a 3.3 GHz Xeon processor with 48 GB of RAM.

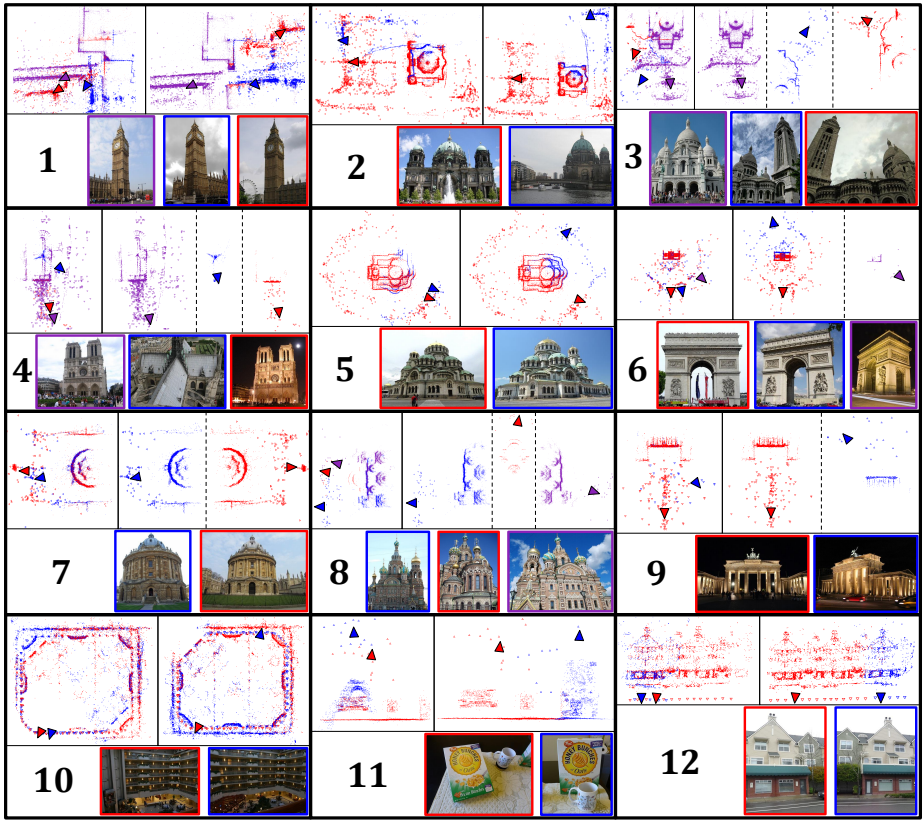

Fig. 7. Example results from our system. Within each dataset cell: top-left is the original reconstruction, top-right is the final merged or split result (split results are separated by a vertical dashed line), and the bottom shows example images from the different split camera subgraphs. Dataset ordering (1-12) corresponds to Table 1.

5 Conclusion

We have presented a novel post-processing method to detect and resolve reconstruction errors caused by duplicate structure (a common occurrence in urban environments). Our method is based on the strong and informative measure of conflicting observations. Our data-driven recursive formulation allows us to not only split an incorrect reconstruction, but to merge it back together (if possible) to recover an error-free result without making assumptions on the final number or configuration of distinct scene elements. In this regard, our experiments confirm that we outperform existing state-of-the-art methods.

Acknowledgments: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1252921 and No. IIS-1349074.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. TPAMI 34(11) (2012)
2. Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S., Szeliski, R.: Building Rome in a Day. Comm. ACM (2011)
3. Bansal, M., Daniilidis, K., Sawhney, H.: Ultra-wide Baseline Facade Matching for Geo-Localization. ECCV Workshops (2012)
4. Cao, S., Snavely, N.: Graph-based discriminative learning for location recognition. CVPR (2013)
5. Cho, M., Lee, K.M.: Bilateral Symmetry Detection via Symmetry-Growing. BMVC (2009)
6. Cho, M., Shin, Y.M., Lee, K.M.: Unsupervised Detection and Segmentation of Identical Objects. CVPR (2010)
7. Chum, O., Mikulík, A., Perdoch, M., Matas, J.: Total recall II: Query expansion revisited. CVPR (2011)
8. Cohen, A., Zach, C., Sinha, S.N., Pollefeys, M.: Discovering and Exploiting 3D Symmetries in Structure from Motion. CVPR (2012)
9. Crandall, D., Owens, A., Snavely, N., Huttenlocher, D.: Discrete-Continuous Optimization for Large-Scale Structure from Motion. CVPR (2011)
10. Dellaert, F., Seitz, S., Thorpe, C., Thrun, S.: Structure from Motion without Correspondence. CVPR (2000)
11. Frahm, J., Georgel, P., Gallup, D., Johnson, T., Raguram, R., Wu, C., Jen, Y., Dunn, E., Clipp, B., Lazebnik, S., Pollefeys, M.: Building rome on a cloudless day. ECCV (2010)
12. Jiang, N., Tan, P., Cheong, L.F.: Seeing Double Without Confusion: Structure-from-Motion in Highly Ambiguous Scenes. CVPR (2012)
13. Jiang, N., Tan, P., Cheong, L.: Multi-view Repetitive Structure Detection. ICCV (2011)
14. Köser, K., Zach, C., Pollefeys, M.: Dense 3D Reconstruction of Symmetric Scenes from a Single Image. DAGM (2011)
15. Lee, S., Liu, Y.: Curved Glide-Reflection Symmetry Detection. TPAMI 34(2) (2012)
16. Liu, J., Liu, Y.: GRASP Recurring Patterns from a Single View. CVPR (2013)
17. Liu, Y., Hel-Or, H., Kaplan, C.S., Gool, L.V.: Computational Symmetry in Computer Vision and Computer Graphics. Foundations and Trends in Computer Graphics and Vision 5(1-2) (2010)
18. Lou, Y., Snavely, N., Gehrke, J.: Matchminer: Efficiently mining spanning structures in large image collections. ECCV (2012)
19. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. TPAMI 27(10) (2005)
20. Raguram, R., Tighe, J., Frahm, J.: Improved geometric verification for large scale landmark image collections. BMVC (2012)
21. Roberts, R., Sinha, S.N., Szeliski, R., Steedly, D.: Structure from motion for scenes with large duplicate structures. CVPR (2011)
22. Schindler, G., Krishnamurthy, P., Lublinerman, R., Liu, Y., Dellaert, F.: Detecting and Matching Repeated Patterns for Automatic Geo-tagging in Urban Environments. CVPR (2008)
23. Sinha, S.N., Ramnath, K., Szeliski, R.: Detecting and Reconstructing 3D Mirror Symmetric Objects. ECCV (2012)

24. Snavely, N., Seitz, S., Szeliski, R.: Photo Tourism: Exploring image collections in 3D. SIGGRAPH (2006)
25. Torii, A., Sivic, J., Pajdla, T., Okutomi, M.: Visual Place Recognition with Repetitive Structures. CVPR (2013)
26. Wilson, K., Snavely, N.: Network principles for sfm: Disambiguating repeated structures with local context. ICCV (2013)
27. Wu, C.: VisualSFM: A Visual Structure from Motion System. http://ccwu.me/vsfm/ (2011)
28. Wu, C., Frahm, J., Pollefeys, M.: Repetition-based Dense Single-View Reconstruction. CVPR (2011)
29. Zach, C., Irschara, A., Bischof, H.: What Can Missing Correspondences Tell Us About 3D Structure and Motion. CVPR (2008)
30. Zach, C., Klopschitz, M., Pollefeys, M.: Disambiguating Visual Relations Using Loop Constraints. CVPR (2010)
31. Zhao, P., Quan, L.: Translation Symmetry Detection in a Fronto-Parallel View. CVPR (2011)
32. Zhao, P., Yang, L., Zhang, H., Quan, L.: Per-Pixel Translational Symmetry Detection, Optimization, and Segmentation. CVPR (2012)