Network Principles for SfM: Disambiguating Repeated Structures with Local Context

Kyle Wilson Noah Snavely
Cornell University
{klw229,snavely}@cornell.edu

Abstract

Repeated features are common in urban scenes. Many objects, such as clock towers with nearly identical sides, or domes with strong radial symmetries, pose challenges for structure from motion. When similar but distinct features are mistakenly equated, the resulting 3D reconstructions can have errors ranging from phantom walls and superimposed structures to a complete failure to reconstruct. We present a new approach to solving such problems by considering the local visibility structure *of such repeated features. Drawing upon network theory, we present a new way of scoring features using a measure of local clustering. Our model leads to a simple, fast, and highly scalable technique for disambiguating repeated features based on an analysis of an underlying visibility graph, without relying on explicit geometric reasoning. We demonstrate our method on several very large datasets drawn from Internet photo collections, and compare it to a more traditional geometry-based disambiguation technique.*

1. Introduction

The structure from motion (SfM) problem is to infer 3D camera poses and scene geometry from a set of 2D images with correspondence. While great strides have been made in automatic SfM methods, one continuing challenge is to handle scenes containing distinct objects that look similar.

For example, consider a symmetric four-sided tower with a similar appearance from each side (e.g., Big Ben). Incorrectly inferring correspondence between, say, the north and south faces can cause major problems for SfM methods. For instance, two or more distinct faces may be "glued" together, causing other objects in the surrounding scene to be incorrectly superimposed on top of each other in the reconstruction. In such cases, the resulting geometry can be hopelessly ensnarled, with no easy way to disentangle it. These problems commonly occur when local feature correspondence methods incorrectly associate distinct 3D points.

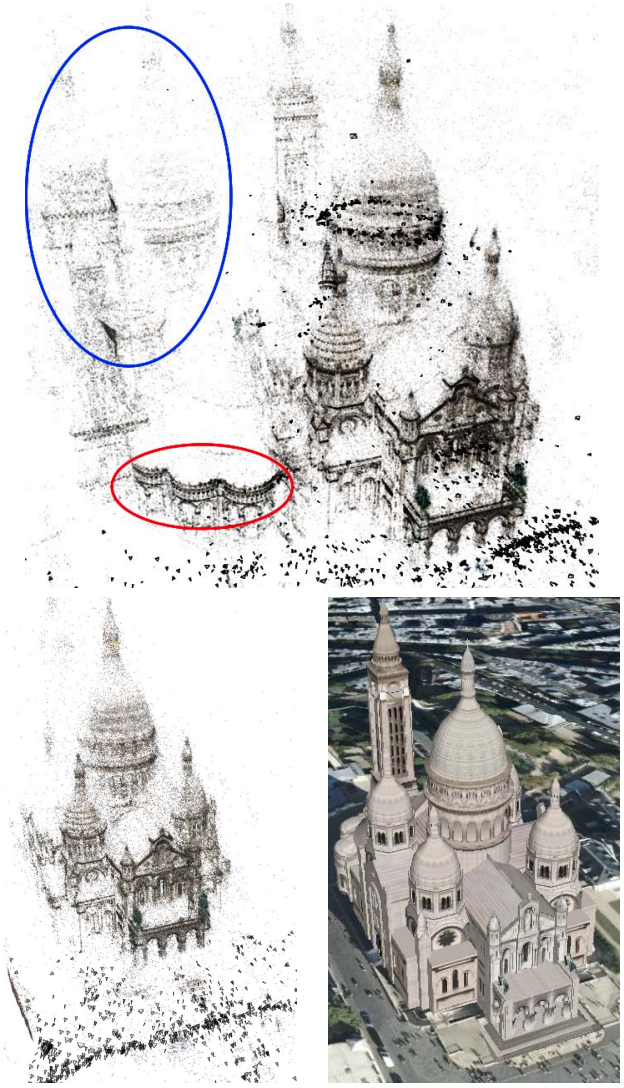

Figure 1. (a) A SfM model of the Sacre Coeur Basilica in Paris containing structural ambiguities. Prominent errors in the reconstruction, including repeated and phantom structures, are highlighted. (b) The same model, correctly disambiguated using our proposed method. (c) A Google Earth rendering of the cathedral.

For a more complex example, consider the reconstruction of the Sacre Coeur Basilica in Paris shown in Figure 1. The radially symmetric main dome and four nearly-identical surrounding spires form a complicated ambiguity structure. As parts of the model get reconstructed in the wrong position around the dome, sparse "phantom" domes and towers appear, as highlighted in blue. Furthermore, the four spires themselves look similar on each side, and the scalloped north and south sides of the rear apse also look the same, causing multiple copies of several more key features to appear. This problem is highlighted in red.

The *SfM disambiguation problem* seeks to produce a correct reconstruction in the presence of similar looking objects [12, 13, 8, 4]. Disambiguation methods often rely on additional assumptions. For instance, if the images are from a time sequence, then we have stronger priors on the camera configuration [8]. In scenes with relatively few occlusions, reasoning about unseen features can reveal differences between similar objects [4]. In this paper, we consider the problem of disambiguating large, unstructured Internet photo collections, which pose significant challenges. These collections rarely come in sequence and often capture scenes with a complicated occlusion structure. They also represent very uneven distributions of views—popular viewpoints will be heavily represented, while side streets may only be captured a few times. Finally, large datasets with thousands of images and millions of points require scalable methods.

We address this problem through the intuition that incorrect feature correspondences result in anomalous structures in a *visibility graph* representing images and the features they see. We describe a model of an incorrect, ambiguous feature track, based on the local structure of a visibility graph around such a track. Based on this model, we propose a simple, local, measure of track "goodness" inspired by local clustering coefficients used in social networks analysis. This measure is a graph-topological one distinct from geometric measures used in prior work, and is very efficient to compute. This new measure gives us a signal that we then use in scene disambiguation technique that is scalable and which works surprisingly well in practice. We demonstrate our technique on several large-scale Internet photo collections, and compare to an existing geometric method.

In summary, our contributions are (1) a simple, efficient new measure of anomalous behavior in feature correspondence, and (2) a scalable method for disambiguating and reconstructing scenes based on this measure. Our code and datasets are available are online at our project page, `www.cs.cornell.edu/projects/disambig`.

2. Related Work

Much of the recent work in scene disambiguation has focused on analysis of geometry. Zach *et al*. [12] use a Bayesian belief network based on positive belief for feature matches, negative belief for missing features, and consistency of triplets of epipolar geometry (EG) constraints between images. In [13], reasoning over triplets of images [5] is expanded to looking for consistent EGs over larger loops. In [3], EGs are added to a maximal spanning tree and discarded if their cycle error is high. Roberts *et al*. [8] find local reasoning insufficient, instead detecting bad EGs with global expectation maximization. Time-sequence information is used when available. Jiang *et al*. [4] propose a novel optimization function that reprojects 3D points into images to find missing correspondences. Given suitable background context, they provably find the global minimum. Finally, Cohen *et al*. [2] detect symmetries while reconstructing, exploiting them to improve reconstruction accuracy.

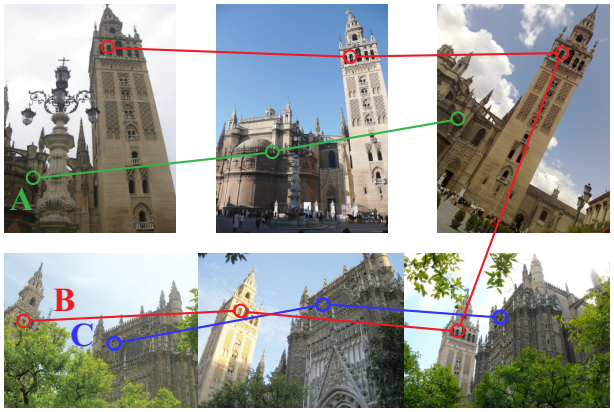

Figure 2. Six images from the **Seville** dataset. Tracks A and C, in green and blue respectively, each correctly correspond to a single 3D point, while track B (in red) mistakenly refers to both the front and back faces of a bell tower (making it a *bad track*).

These prior methods have been applied exclusively to relatively small datasets of 20-200 images, often laboratory scenes specifically designed to be difficult. In contrast, our method is designed for much larger, more unstructured collections of photos, such as those downloaded from the Internet. While our method generally cannot disambiguate prior laboratory datasets, we find it performs well for the ambiguities typical of large Internet photo collections, suggesting that the difficult constructions in prior work (for instance, moving objects between shots) are not necessarily representative of those in Internet datasets.

As with our method, many previous approaches are based on analysis of an underlying graph, including graphs encoding matches or EGs between images, performing explicit geometric reasoning over these graphs. Our work reasons about a different graph, a bipartite visibility graph encoding relations between cameras and points, and performs a purely topological, local analysis over this graph.

3. Overview

We first describe our problem setting. A typical structure from motion (SfM) pipeline begins by detecting local features in each input image (e.g., using SIFT [6]), then matching these features across images. Following [9], we refer to a set of matched features across several images—often found through transitive closure of pairwise feature matches—as a *track*. The set of tracks can be naturally described in terms of a bipartite graph, $G = (I, T, E)$, where nodes are images I and tracks T, and edges $(i, t) \in E$ exist when a feature in image i is a member of track t. We call G the *visibility graph*. G (along with locations of features in each image) is the input to an SfM reconstruction procedure.

Ideally, each track $t \in T$ represents a single 3D point, in which case G has a natural interpretation: the visibility graph encodes which 3D points are visible in which images. However, in the presence of structural ambiguity, some of these tracks (which we call *bad tracks*) refer to more than one 3D point. Figure 2 illustrates three tracks. A and C correctly correspond to a single 3D point, but B is a bad track comprised of points on both the north and south sides of a tower. While a single bad track may not break a subsequent SfM algorithm, an ambiguous scene can contain many such tracks that form geometrically consistent sets. We address the disambiguation problem by finding a new set of tracks that is as correct as possible; we realize this goal by attempting to identify and remove bad tracks.

Our method has two main elements: (1) a model and corresponding score function for bad tracks based on a local visibility analysis, and (2) an SfM procedure that uses this model and score to produce a disambiguated reconstruction. We now describe each of these elements in turn.

4. Modeling Bad Tracks using Visibility

Our model for a bad track is based on *background context*. Whether or not images see the same background objects is useful information for detecting bad tracks. In previous work, such as [4], such reasoning is geometric. In our case, we consider the weaker condition of visibility, as encoded topologically in the visibility graph G. We reason that even if objects look the same, they will often have different backgrounds. Figure 3(a) shows a simplified geometric view of the **Seville** scene in Figure 2, depicting cameras and the tracks they observe. Here B represents a feature on a window of the bell tower. Because the tower is symmetric, the windows on the left and right sides of the tower look the same, and so there are two distinct 3D points represented by B. Tracks A and C represent *neighbors* of B from the left or right views—other points in the world that are seen with B. These are our background context. Suppose for discussion that A and C are not close to each other (i.e., they cannot both be seen in a single photograph). In Figure 3(b) we see the visibility graph for this scene. In network terms, if A and C aren't seen together, then B is a *bridge* between them. This bridging property provides evidence that B is a bad track, as one would expect visibility to be, roughly speaking, "transitive" (e.g., if A is seen with B, and B with C, it would be surprising to never see A and C together).

This suspicion is heightened in Figure 3 (c), a more complex visibility graph where additional context is present. In this case, the neighboring tracks of B (i.e., other tracks seen in images that see B) form two clusters, and B bridges these two clusters. This leads to an intuition about local neighborhoods of tracks: bad tracks are those that are most like a bridge between two or more clusters of other "context" tracks in the visibility graph. To make this intuition quantifiable, we turn to network theory, which provides a useful measurement: the *bipartite local clustering coefficient*.

To explain this measure, we begin with single-mode (not necessarily bipartite) graphs. In [11], Strogatz introduces the *local clustering coefficient*. For a node v, this is defined as

$$\mathrm{lcc}(v) = \frac{\text{\# triangles centered at } v}{\text{\# 2-paths centered at } v}, \tag{1}$$

where a 2-path is any choice of two distinct neighbors of v. This can be phrased as the ratio of *closed* 2-paths (triangles) to possible 2-paths centered at v. The lcc score measures *local transitivity*: in social network parlance, it is the fraction of my pairs of friends (pairs of neighbors of v) who are themselves friends with each other. This can be computed per node; when a network divides roughly into clusters, this score will be low for vertices that bridge clusters.

In our case, G is a bipartite graph, over which the lcc score is identically zero, because there are no triangles by definition. However, Opsahl [7] proposes a natural extension of the lcc score to bipartite graphs, using four-paths instead of triangles. This is the *bipartite local clustering coefficient*:

$$\mathrm{blcc}(t) = \frac{\text{\# closed 4-paths centered at } t}{\text{\# 4-paths centered at } t} \tag{2}$$

This function is exactly the fraction of the time that local transitivity holds at track t. In the language of the visibility graph, it answers the question, "if I'm seen with neighbor tracks A and C, how often are they seen together, over all such neighbors?" A typical 4-path rooted at B is shown in bold red in Figure 3(c). For this path, local transitivity fails (i.e., the path is not closed), since tracks A and C are never both seen in the same image.

As a measure of local transitivity, we argue that the blcc tends to identify bad tracks that look like track B in Figure 3. By modeling bad tracks in this way, we assume that the different parts of a bad track have context and are not co-visible. Certainly scenes can be imagined where images tend to see all 3D points which comprise a bad track. For example, consider a sign with repeated letters and shapes;

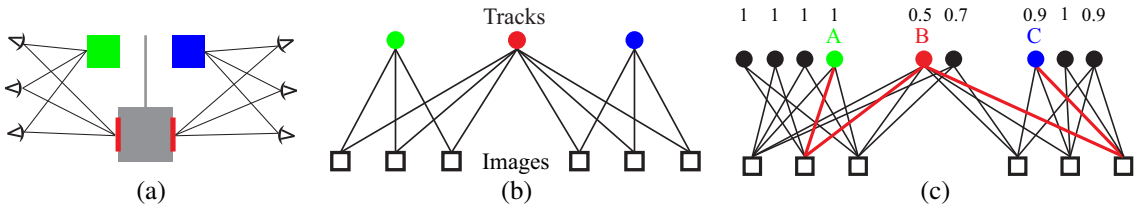

Figure 3. (a) A geometric interpretation of the **Seville** scene in Figure 2. The grey square represents the tower, while the blue and green objects are 3D points seen as tracks A and C. (b) The resultant bipartite image-track visibility graph. (c) A graph representing a larger problem. A sample 4-path rooted at bad track B, for which local transitivity fails, is shown in bold red. The blcc scores are above each track.

two points several meters apart might be confused, yet have nearly the same set of neighbor tracks. In practice, these bad tracks are not the sort of structural ambiguity which breaks reconstructions. The many correct neighbors are enough context for the bad track to be rejected on epipolar constraints earlier in a SfM pipeline. However, we acknowledge that this score is not a universal solution to SfM disambiguation; it relies on the separated context often present in natural scenes, which we find to be a very useful cue in practice.

5. Algorithm for Disambiguating a Model

We now have a score (the blcc) that captures our intuition about what makes a track good or bad. However, in real visibility graphs, an additional issue arises. Looking more closely at a bad track t_b that has n neighbor images in k clusters, $\text{blcc}(t_b)$ will be lowest when each cluster has $\frac{n}{k}$ images. However, if at our bad track some of these clusters are much larger than others, $\text{blcc}(t_b)$ will be inflated and thus less sensitive. Unfortunately, Internet photo collections often represent very non-uniform samplings of viewpoint, with a large disparity in the number of images that capture different parts of a scene. For example, in the **Sacre Coeur** dataset, over 75% of all images are taken on the front steps of the basilica. In **Notre Dame** (Figure 6), over 85% of the images are of the front facade. As a result, on real-world datasets the blcc computed on the full visibility graph G can be skewed depending on local density. We instead compute the blcc on a subgraph of G that spans G but is more uniform. Our SfM disambiguation procedure thus has two main steps: (1) compute a more uniform subgraph, then (2) remove bad tracks from that subgraph based on an analysis of blcc scores.

Step 1: Computing a covering subgraph. To address the issue of uneven sampling of views in Internet datasets, we propose to find a subgraph $G' = (I', T', E') \subset G$ by choosing a subset of images $I' \subset I$ and of tracks $T' \subset T$ that is better behaved. In particular, this subgraph G' should have the following properties:

Uniformity: Tracks should be seen similarly often.

Reconstructability: Most tracks should be seen often enough to be reconstructed.

Field of View Heuristic: I' should be made up of images with wide field of view, since these often see more context than telephoto pictures.

Our basic approach is to compute a subset of images I' such that each track is *covered* a certain minimum number of times by images in I'. We begin by restricting the subset of tracks we consider to *long tracks* (i.e., tracks visible in many images), based on the intuition that long tracks are the most important for connecting up the graph, and because the blcc score is less stable for tracks with few neighbors.

To formalize, let $\delta_G(t)$ denote the degree of a track node t in G. We define long tracks as the set $T_L = \{t \in T : \delta_G(t) \geq N_{long}\}$. We will compute a covering graph G' where almost all $t \in T_L$ have at least k neighbors: $|\{t \in T_L : \delta_{G'}(t) < k\}|/|T_L| < \epsilon$. We allow a fraction ϵ to not be covered based on our observation that real datasets often have a few outlier tracks that correspond to unusual objects not of interest to the reconstruction, and are difficult to cover.

We approach this as a graph covering problem. Our algorithm for selecting the subset of images I' greedily selects images that cover the most remaining not-yet-fully-covered tracks, according to a score function that also considers the field of view of that image. We define the score of an image i as $\text{fov}(i) \cdot \delta(i)^{\alpha}_{G_R}$, where $\text{fov}(i)$ is the field of view of image i, and $\delta(i)_{G_R}$ is the degree of i in the graph G restricted to uncovered tracks. This favors images with high degree and high field of view. This is summarized in Algorithm 1. The parameters k, ϵ, N_{long}, and α are discussed with our results. Note that we restrict the covering to images for which the field of view can be estimated from Exif metadata.

Figure 4 shows two ROC curves demonstrating the benefit of computing the blcc score on the covering subgraph rather than the full graph. For these plots, we manually created a ground truth classification of tracks as good or bad for the **Seville** dataset by identifying the structural ambiguities in the data, then manually classifying images into geographic groups. Any track spanning two disjoint groups of images is labeled as bad. Inspecting 1000 randomly sampled tracks confirmed that this labeling of bad tracks was nearly complete. Each ROC curve is created by thresholding tracks on

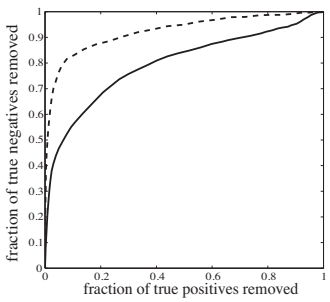

Figure 4. ROC curve comparing of effectiveness of the bad track score: computed over the whole visibility graph (solid line) or the covering subgraph (dashed line) of the **Seville** dataset.

Algorithm 1: $(1-\epsilon)$ complete k-cover

Input: visibility graph G, fields of view $\mathrm{fov}(i), i \in I$
Assemble $G' = (I', T', E)$, where
Initialize $T' = \{t \in T : \delta_G(t) \geq N_{long}\}$, $I' = \emptyset$
while $|\{t \in T' : \delta_{G'}(t) < k\}|/|T'| > \epsilon$: **do**
  define remaining tracks:
  $T_R = \{t \in T' : \delta_{G'}(t) < k\}$
  define remainder graph: $G_R = (I–I', T_R, E)$
  select $i \in I–I'$ that maximizes $\mathrm{fov}(i) \cdot \delta_{G_R}(i)^\alpha$
  add i to I'
Let E' be the restriction of E to vertices I', T'.
Output $G' = (I', T', E')$.

the blcc score. The scores computed on the subgraph G' (dotted curve) distinguish good and bad tracks more effectively than those computed on the full graph G (solid curve); in particular, we can remove 80% of the bad tracks while discarding only a small fraction of good ones. In all cases, we found that disambiguating tracks using the subgraph yields superior results to using the original graph.

Step 2: Disambiguation process. We now describe our full pipeline. We begin by running a standard feature matching procedure [1] to generate a set of tracks, with one modification. While generating matches, our pipeline finds pairwise epipolar geometries (EGs). We observed that these can sometimes include very wide-baseline image matches (e.g., where cameras are far apart, possibly even on opposite sides of the points they view). Since SIFT (our feature detector) is only stable for rotations of up to about 60 degrees [6], we discard all EGs with relative rotation larger than 90 degrees. We have observed that these pairs are almost always erroneous (but removing them was insufficient to fix our models).

Given the complete visibility graph G, we compute a covering subgraph G' as described above. To disambiguate the scene, we then compute $\mathrm{blcc}(t)$ for each long track in the covering subgraph G'. We delete all tracks with score less than a threshold τ from G and G'. Since this often results in G being nearly disconnected, we split G and G' into the components induced by the partition of all images in G which share at least N_{conn} tracks, for some parameter N_{conn}. Thus, our approach can break a scene up into several components. This is often desired behavior, as many of our scenes have physically separated pieces.

We have explored several ways to select τ. One was to set it directly; however, we found that the distribution of blcc values varies from scene to scene, and values that work for one scene may not transfer directly to others. We also considered parameterizing τ by removing some set percentage of low-scoring tracks. However, some datasets have just a few, egregiously bad tracks, while others (such as **Sacre Coeur**) are riddled with large numbers of bad tracks (though we always found some threshold to result in a correct model). Instead, we propose setting in advance an expected number of components that the model will separate into, and then choosing the lowest τ that achieves this number of components. We call this parameter N_{comp}. By choosing N_{comp} conservatively large we ensure that our primary failure mode is oversegmentation—splitting into too many components. From the standpoint of using SfM results, we believe this is a useful system design; it is much easier, on inspection, to see how components ought to fit together than to understand and tease apart a hopelessly broken model. We found this approach to give good results across all our datasets.

In order to validate that thresholding on our track scores is important to separating a graph into components, we also tried a simpler baseline approach of removing all EGs with fewer than a certain number of inliers. This did not improve the models—all ambiguities remained.

Given the new components and tracks, we reconstruct each component using an incremental SfM algorithm [1], first with the images and tracks in G', then adding the remaining images and tracks in G. By doing the reconstruction in this order, we first make a sparse skeleton reconstruction of the whole scene that ideally is disambiguated and error free. Then we the rest of the images, hanging them on this correct skeleton. Note that this skeleton is related to the one in [10], but is computed using our disambiguation process. (The broken reconstructions presented in the results section are computed with the prior skeletonization algorithm [10].)

6. Results

In this section we show results of our method on four large Internet datasets, **Seville**, **Louvre**, **Sacre Coeur**, and **Notre Dame**. (Reconstructions are shown in Figure 6.)

The disambiguation method defined in Section 5 has several parameters. In all cases, we found the values $k = 10$, $\alpha = 0.3$, $\epsilon = 0.02$, $N_{long} = 15$, $N_{conn} = 200$, and $N_{comp} = 4$ to produce correct and satisfactory results.

Table 1 summarizes our four datasets, and Figure 6 shows their original and disambiguated reconstructions. These datasets are large, containing as many as 8000 images and

Name	Images	Tracks	time (s)
Seville	916	299837	76
Louvre	839	229260	53
Sacre Coeur	4416	1296632	253
Notre Dame	8032	3702932	2718

Table 1. Results datasets: original size and time to disambiguate

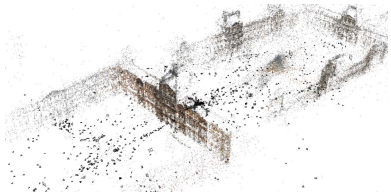

Figure 8. Reconstruction of **Louvre** using [13]

3.7 million tracks. Nevertheless, computing the score $\mathrm{blcc}(t)$ and disambiguating the model took under 45 minutes in all cases. The time shown is only the time required for the disambiguation; subsequent reconstruction takes no longer than reconstructing without disambiguation. Disambiguation was performed on 8 cores of a 2.40 GHz Xeon machine.

In each case the disambiguated models are in several components which correspond to distinct clusters of views. Often this is because the original model connected clusters of images that did not belong together. When false correspondences are removed by deleting the bad tracks the result is a model in several components. In other cases there may be enough information in principle to join two components, but the disambiguation process was over-conservative and did not connect them. We now discuss each dataset in detail.

In **Seville** (Figure 6 (a)), the original confusion was a symmetry around the tower. One of the two large (similar looking) doorways (3) floated in the middle of a plaza (circled in red). The disambiguation removes the tracks which incorrectly equate opposite faces of the tower. This splits the scene. However, enough good tracks were removed to also disconnect a doorway (1) from the rest (2). Finally, a few images (4) were from a back alley. These mostly see the back of the tower and also were originally reconstructed in the plaza. Disambiguation correctly removes the connection to the tower, which separates these into a fourth component.

The results for **Louvre** (Figure 6 (b)) are particularly interesting. The Louvre has a small inner courtyard (1) and a larger outer one (2,3). There are repeating walls and gateways around each. Originally the two courtyards are oddly attached—the scale is wrong and the outer courtyard faces the wrong direction. In fact, disambiguation finds a less obvious mistake: the images in the outer courtyard were originally reconstructed as facing the same direction, when in fact many of them face west (2), while several others face east (3). The result is in three components.

As described in the introduction, **Sacre Coeur** (Figure 6 (c)) is very challenging. The original model has extra towers and domes (circled in blue). Also, instead of a scalloped wall on each side of the rear of the cathedral, there is only one copy (circled in red) in the wrong place. Disambiguation separates each of these problems. There is a cluster of photos (4) that look outwards from the cathedral into Paris and see little in common with the other images. Note that very few pictures were taken in the narrow streets along the sides of the cathedral. This causes the cathedral to separate into three pieces: the front (2) and the two sides of the rear (1,3).

Notre Dame (Figure 6 (d) is our largest dataset. In the original reconstruction note the extra copy of the roof attached at an incorrect 45° angle to the rest of the model (circled in red). The central spire is the main feature that attaches the two together, and unfortunately it has an 8-way symmetry that causes the aerial views to get "glued on" at the wrong angle from the ground. These are correctly separated in (1). Again, we see that a sparsity of photos, combined with our removal of tracks, was enough to disconnect the rear of the cathedral (2) from the front (3)—a small point of failure. We note that (2) and (3) can be merged together by describing tracks by mediod SIFT features, matching, and then estimating an alignment with RANSAC.

Comparison with [13]. We ran the disambiguation procedure of [13] for comparison. This is a recent geometry-based method for which the authors have made code available. It examines cycles of epipolar geometries in the image graph. An inference step finds a blacklist of image pairs to remove.

We omit results for two of our datasets because we could not run them within the 64GB memory limitation of our machine. **Louvre** took 50.6 hours and removed 17% of all EGs. Figure 8 is a rendering of the results—all ambiguities remain. Similarly, **Seville** took 47.5 hours, removed 19% of all EGs, and also did not disambiguate the scene.

In each case the results still contain their original ambiguities. Interestingly, the same is true for running [13] on our covering subgraphs. In looking at these results, we speculate that our image graphs are much different—much denser and higher degree—than the graphs for which this technique was designed; the underlying inference engine, loopy belief propagation, is possibly more amenable to graphs that are more sparsely connected and more tree-like.

Unbroken Datasets. We also ran our method on datasets where the original reconstruction was correct. In each case it behaved reasonably, splitting the scenes in natural ways. **Roman Forum** (Figure 6) and **Acropolis** each split into two mergable components. For **Jefferson Memorial**, two clusters of images were connected by a few wide views. Our method returned these two components, but removed the sparse connecting tracks so that they couldn't be merged. Our method is slightly destructive by removing tracks, but

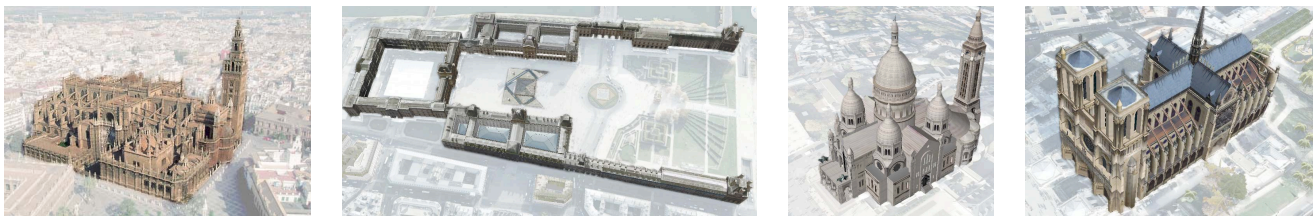

Figure 5. Reference views of each dataset: **Seville Cathedral**, **Louvre**, **Sacre Coeur**, and **Notre Dame**.

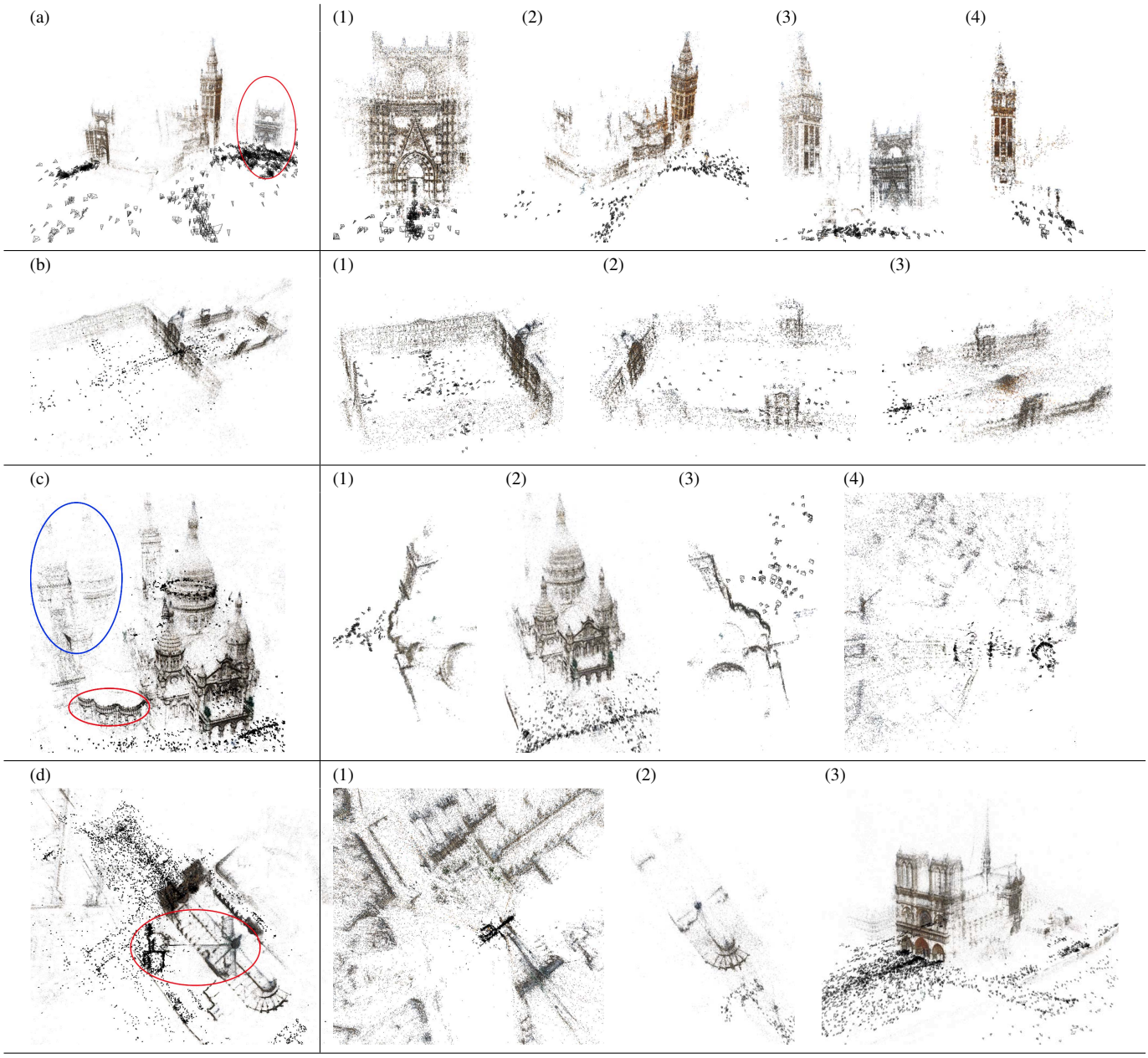

Figure 6. Original (left) and disambiguated (right) reconstructions of (a) **Seville Cathedral**, (b) **Louvre**, (c) **Sacre Coeur**, and (d) **Notre Dame**. In case, the disambiguated reconstruction is in more than one component. For the originals, the main mistakes are circled.

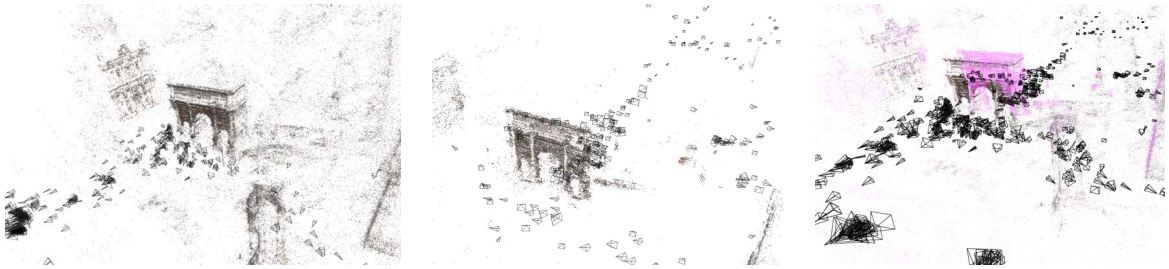

Figure 7. An unbroken dataset, **Roman Forum**, showing the two components we produce (left, middle) and the merged model (right). In the merged model, points from the second component are highlighted in purple. The merged model is very similar to the original.

for well-connected scenes the loss is barely perceptible.

Limitations. We have shown that our disambiguation method can successfully fix reconstructions broken by structural ambiguities in several large datasets. As discussed above, one mode of failure is *oversegmentation*. We see this in **Seville**, where an entryway is split off from a larger model, and in **Sacre Coeur**, where the outward-facing images were split away from the rest. These extra splits happen at places where the visibility graph is weakly connected. For instance, although parts of the **Seville** entrance are seen by cameras in components (1) and (2), there are very few matches between them because of the wide baseline between views. However, in such cases our approach substitutes a difficult problem (SfM disambiguation) with a more tractable one (model merging), with the added benefit that the output goes from an incorrect reconstruction to a set of correct but possibly partial models, which are potentially much more useful.

In addition, when we evaluated our method on the challenging laboratory datasets introduced by Roberts *et al*. [8] (and used in subsequent work, e.g. [4]), we saw that the blcc was weakly correlated with bad tracks, but we were only successful at disambiguating **Desk**. We believe this is due to both the small overlap within each dataset (yielding sparse visibility graphs), as well as the lack of the contextual cues which we found abundantly in our Internet datasets. Of these small datasets, **Desk** has the most background context.

7. Conclusion

Structure from motion models are often incorrect when there are similar-looking objects in the scene. We presented a new method for disambiguating such scenes. Bad tracks are identified as those which have similar local topology to a prototypical model in the visibility graph. We compute this via the bipartite local clustering coefficient, computed on a subgraph of the input set, and propose a reconstruction procedure for using this score to build a model. Our method, based on local context, is scalable to large datasets, and our results show that a visibility graph-based approach (without global geometric reasoning) is often sufficient to solve many problems of interest. Our code and data are available online at `www.cs.cornell.edu/projects/disambig`.

In the future we hope to explore ways of judiciously combining geometric reasoning with our pure visibility-based approach, in order to obtain more accurate disambiguations without sacrificing scalability.

Acknowledgements. This work was supported in part by the National Science Foundation under IIS-1149393, IIS-1111534, and IIS-0964027, and a grant from Intel Corporation. We would also like to thank Chun-Po Wang and Robert Kleinberg for their valuable discussions.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building Rome in a day. In *ICCV*, 2009.

[2] A. Cohen, C. Zach, S. N. Sinha, and M. Pollefeys. Discovering and exploiting 3d symmetries in structure from motion. In *CVPR*, 2012.

[3] O. Enqvist, F. Kahl, and C. Olsson. Non-sequential structure from motion. In *ICCV Workshops*, 2011.

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

[5] M. Klopschitz, A. Irschara, G. Reitmayr, and D. Schmalstieg. Robust incremental structure from motion. In *3DPVT*, 2010.

[6] D. G. Lowe. Distinctive image features from scale-invariant keypoints. In *IJCV*, 2005.

[7] T. Opsahl. Triadic closure in two-mode networks: Redefining the global and local clustering coefficients. *Social Networks*, 2011.

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

[9] N. Snavely, S. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH*, pages 835–846, 2006.

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

[11] S. H. Strogatz. Exploring complex networks. *Nature*, 410(6825):268–276, March 2001.

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

[13] C. Zach, M. Klopschitz, and M. Pollefeys. Disambiguating visual relationships using loop constraints. In *CVPR*, 2010.