SfM with MRFs: Discrete-Continuous Optimization for Large-Scale Structure from Motion

David J. Crandall, Andrew Owens, Noah Snavely, and Daniel P. Huttenlocher

Abstract—Recent work in structure from motion (SfM) has built 3D models from large collections of images downloaded from the Internet. Many approaches to this problem use incremental algorithms that solve progressively larger bundle adjustment problems. These incremental techniques scale poorly as the image collection grows, and can suffer from drift or local minima. We present an alternative framework for SfM based on finding a coarse initial solution using hybrid discrete-continuous optimization, and then improving that solution using bundle adjustment. The initial optimization step uses a discrete Markov random field (MRF) formulation, coupled with a continuous Levenberg-Marquardt refinement. The formulation naturally incorporates various sources of information about both the cameras and points, including noisy geotags and vanishing point estimates. We test our method on several large-scale photo collections, including one with measured camera positions, and show that it produces models that are similar to or better than those produced by incremental bundle adjustment, but more robustly and in a fraction of the time.

Index Terms—Structure from motion, 3D reconstruction, Markov random fields, belief propagation

1 Introduction

Structure from motion (SfM) techniques have recently been used to build 3D models from unstructured and unconstrained image collections, including photos from online social media sites such as Flickr [2], [3], [4], [5]. Many of these approaches operate incrementally, starting with a small seed reconstruction that is grown by repeatedly adding additional cameras and scene points. While such incremental approaches have been quite successful, they have two significant drawbacks. First, these methods tend to be computationally intensive, making repeated use of bundle adjustment [6] (a non-linear optimization that jointly refines camera parameters and scene structure) as well as outlier rejection to remove inconsistent measurements. Second, since these methods do not treat all images equally, they produce different results depending on the order in which photos are considered. This can lead to failures due to local minima or cascades of misestimated cameras. Such methods can also suffer from drift, especially in large scenes containing weak visual connections.

In this paper we propose a new SfM method for unstructured image collections that considers all photos at once, rather than building up a solution incrementally. This method is faster than current incremental bundle adjustment (IBA) approaches and more resilient against reconstruction failures. Our approach computes an initial estimate of the camera poses using all available photos, and then refines that estimate and solves for scene structure using bundle adjustment. This approach is reminiscent of earlier work in SfM, prior to recent work on unstructured collections, which first solved for a good initialization (e.g., using factorization methods [7]) and then applied bundle adjustment as a final nonlinear refinement step to obtain accurate camera parameters and scene structure. Similarly, our approach can be thought of as estimating an initialization for unstructured image sets which is readily refined using bundle adjustment.

In particular, we obtain such an initialization through a two-step process combining modern discrete and continuous optimization techniques. In the first step, discrete belief propagation (BP) is used to estimate camera parameters based on a Markov random field (MRF) formulation of constraints between pairs of cameras or between cameras and scene points; while such discrete methods are common in vision problems such as stereo or optical flow, we show they are also useful for SfM. The second step is a Levenberg-Marquardt nonlinear optimization related to bundle adjustment, but involving additional constraints. This hybrid discrete-continuous optimization allows for an efficient search over a very large parameter space of camera poses and 3D points. The method requires a fraction of the time of IBA, due to both its favorable asymptotic complexity and the fact that it is highly parallelizable on distributed-memory clusters (unlike IBA). In fact, most of the computation time of

- *David Crandall is with the School of Informatics and Computing, Indiana University. E-mail:* djcran@indiana.edu
- *Andrew Owens is with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology. E-mail:* andrewo@mit.edu
- *Noah Snavely and Daniel Huttenlocher are with the Computer Science Department, Cornell University. E-mail:* {snavely,dph}@cs.cornell.edu

our approach is spent in the final bundle adjustment.

A key benefit of our approach is robustness; unlike linear formulations of batch SfM, our approach can use robust objective functions, which are key for handling noisy, real-world data. An additional benefit is our formulation can naturally incorporate noisy prior information about camera poses. Such prior information is becoming common for online photos because many modern cameras (and smartphones) record GPS and compass heading information. These measurements are also highly noisy, however, presenting a major challenge. Geotag errors are routinely in the tens of meters, and geotags on the wrong continent are not uncommon. Our MRF-based formulation can naturally integrate this noisy information into the optimization using unary potentials with robustified distance functions. By using all of the available data at once (rather than incrementally), and by allowing additional types of constraints, we find that our approach is quite robust on large, challenging problems.

We evaluate our hybrid method on several large datasets downloaded from photo-sharing sites, and find that it produces comparable reconstructions—and in the case of a particularly challenging dataset, a much better reconstruction—to those produced by state-of-the-art IBA [2] in significantly less time. We also test it on a dataset of several thousand photos of a university campus, with some photos that have high precision, ground-truth camera positions (measured using survey-quality differential GPS). On this dataset our method and IBA have similar accuracy with respect to the ground truth, and thus our method not only can yield similar results to those of IBA, but the two achieve comparably accurate reconstructions.

2 Related work

Current techniques for large-scale SfM from unordered photo collections ([2], [4], [8], [5]) make heavy use of bundle adjustment to solve a non-linear optimization problem. Bundle adjustment is highly sensitive to initialization, so these systems are run iteratively by starting with a small set of photos, then repeatedly adding photos and refining 3D points and camera poses using bundle adjustment while discarding or downweighting outliers. While generally successful, incremental approaches are time-consuming for large image sets, with a worst-case running time of $O(n^4)$ in the number of images (though efficient implementations may avoid the worst case in practice).[1] One way to reduce the cost is by pruning the image set; recent work has used clustering or graph-based techniques to reduce the number of images that must be considered in SfM [4], [11], [12], [13]. For example, Li *et al.* [4] first cluster a set of images to find 'iconic images,' then compute 3D structure incrementally using an approach based on spanning trees. These techniques make SfM more tractable, but the graph algorithms themselves can be costly, the number of remaining images can be large, and the effects on solution robustness are not well understood.

Other approaches to SfM solve the problem in a single *batch* optimization. These include classical factorization methods [7], which in some cases can solve SfM in closed form. However, it is difficult to apply factorization to perspective cameras with significant outliers and missing data (which are the norm for Internet photo collections). Our work is most closely related to batch SfM methods that solve for a global set of camera poses given local estimates of geometry, such as pairwise relative camera poses. These include linear methods for solving for global camera orientations or translations [14], [15], [16], and L_∞ methods for solving for camera (and possibly point) positions given known rotations [17], [18]. While efficient, these methods do not have built-in robustness to outliers, which we have found can cause them to fail on the noisy, unstructured image collections that we consider; similarily, it can be difficult to integrate noisy prior pose information into the optimization. In contrast, our MRF formulation can easily incorporate both robust error functions and priors.

Other work has incorporated geotags and other prior information into SfM, as we do here. Sinha *et al* [19] propose a linear SfM method that incorporates vanishing points (but not geotags) in estimating camera orientations. They use only a small number of pairwise estimates of geometry (a spanning tree on an image graph) for initializing translations, while our method incorporates all available information. Prior information has also been used in pre- or post-processing for SfM, e.g., by applying vanishing point or map constraints to straighten out a model [20], [21], using sparse geotags to georegister an existing reconstruction [22], or using geotags and GIS data to register different connected components of a reconstruction [3], [23]. We incorporate geotag and vanishing point information into the optimization itself.

Work in other contexts has considered the problem of estimating camera poses. MRFs have been used to estimate camera pose in the Simultaneous Localization and Mapping (SLAM) literature [24], [25], but in SLAM there are strong sensor and motion models and the optimization is conducted using continuous techniques. In contrast, our approach handles large, unstructured collections of images with weak pose information utilizing both discrete and continuous methods. Researchers in sensor networks have investigated message passing techniques for calibrating distributed camera networks, including forms of distributed averaging [26] and belief propagation [27],

[1] If the problem is dense, so that all images see common features, then direct methods for solving the reduced camera matrix in bundle adjustment [6] take $O(n^3)$ time in the number of images. If a constant number of images is added in each round of incremental SfM, the total running time is $O(n^4)$; for some problems this can be alleviated with sparse or iterative methods [9], [10].

[28]. While efficient in their use of continuous optimization methods, these prior techniques are based on objective functions that are sensitive to outliers. Further, Devarajan and Radke [27] estimate a locally consistent reconstruction for each sensor, whereas our goal is to reconstruct a single, globally consistent set of camera poses.

Finally, other techniques for accelerating SfM have been proposed, including methods for hierarchical reconstruction and bundle adjustment [29], [30], [3]. These methods still depend on an incremental approach for initialization, but structure the computation more efficiently. We present an alternative that avoids incremental reconstruction altogether.

3 Estimating cameras and points

Our method computes camera poses for an entire image collection at once (or more precisely, for images corresponding to each visually connected component of the image set), allowing us to consider all available geometric constraints simultaneously, rather then utilizing incremental reconstruction techniques. At a high level we do this by first solving for consistent camera orientations and then solving for camera and 3D point positions. These subproblems are both formulated as MRFs, as we describe in this section.

Our approach represents a set of images as a graph that models geometric constraints between pairs of cameras or between cameras and scene points (as binary constraints), as well as single-camera pose information such as geotags (as unary constraints). This set of binary and unary constraints can be modeled as a Markov random field (MRF) with an associated energy function on configurations of cameras and points. A key contribution of our work is to use both discrete and continuous optimization to minimize this energy function; in particular, we use belief propagation (BP) on a discretized space of camera and point parameters to find a good initialization, and non-linear least squares (NLLS) to refine the estimate. Combining discrete and continuous optimization techniques has been found to work well on other vision problems, such as optical flow [31]. The power and generality of this combination of techniques allow us to efficiently optimize a more general class of energy functions than previous batch techniques (e.g., factorization). This class includes robust error functions, which are critical to obtaining good results in the presence of noisy observations.

Figure 1 illustrates a typical large-scale SfM pipeline and shows how our technique fits into it: our method is the red box in this figure, and can be thought of as taking geometric information including feature matches, pairwise relative pose information, and geotags as input, and producing a good initialization for bundle adjustment as output. The following sections first describe our formulation of the SfM problem, followed by the belief propagation (BP)-based discrete optimization and the continuous non-linear least squares optimization (NLLS).

3.1 Problem formulation

The input to our problem consists of (a) a set of images $\mathcal{I} = \{I_1, \ldots, I_n\}$, (b) relative pose estimates between some pairs of images (computed using two-frame SfM, described in Section 4), (c) noisy point correspondences between the images, and (d) noisy absolute pose estimates for a subset of images (derived from sources like geotags). Our goal is to estimate an *absolute* camera pose for each image, and a location for each scene point, consistent with all of the input measurements and in a geo-referenced coordinate system. We denote the absolute pose of image I_i's camera as a pair $(\mathbf{R}_i, \mathbf{t}_i)$, where $\mathbf{R}_i$ is a 3D rotation specifying the camera orientation and $\mathbf{t}_i$ is the camera position in a global coordinate frame. The 3D position of a scene point is denoted $\mathbf{X}_k$.

Each pairwise estimate of relative pose between two images I_i and I_j has the form $(\mathbf{R}_{ij}, \mathbf{t}_{ij})$ where $\mathbf{R}_{ij}$ is a relative orientation and $\mathbf{t}_{ij}$ is a translation direction (in the coordinate system of camera I_i). Given perfect pairwise pose estimates, the absolute poses $(\mathbf{R}_i, \mathbf{t}_i)$ and $(\mathbf{R}_j, \mathbf{t}_j)$ of the two cameras would satisfy

$$\begin{aligned} \mathbf{R}_{ij} &= \mathbf{R}_i^\top \mathbf{R}_j && (1) \\ \lambda_{ij}\mathbf{t}_{ij} &= \mathbf{R}_i^\top(\mathbf{t}_j - \mathbf{t}_i), && (2) \end{aligned}$$

where λ_{ij} is an unknown scaling factor (because only the direction of translation can be estimated from a pair of images due to the gauge ambiguity in SfM). We can also write constraints between cameras and scene points. For a scene point $\mathbf{X}_k$ visible to camera I_i, let $\mathbf{x}_{ik}$ denote the homogeneous 2D position of the point in I_i's image plane. Then we can relate the absolute pose of the camera and the 3D location of the point,

$$\mu_{ik}\mathbf{x}_{ik} = \mathbf{K}_i\mathbf{R}_i(\mathbf{X}_k - \mathbf{t}_i), \tag{3}$$

where $\mathbf{K}_i$ is the matrix of camera intrinsics for image I_i (assumed to be known from EXIF metadata as described in Section 4), and μ_{ik} is an unknown scale factor (the depth of the point). Equation (3) is the basis for the standard *reprojection error* used in bundle adjustment. The above three constraints can be defined on a *reconstruction graph* $G = (V, E_C \cup E_P)$ having a node for each camera and each point, a set E_C of edges between pairs of cameras with estimated relative pose, and a set E_P of edges linking each camera to its visible points (see Figure 2). Bundle adjustment typically only uses point-camera constraints (as relative poses between cameras are implicit from point correspondence), but in batch techniques constraints between cameras have proven useful. Figure 3 shows a sample reconstruction graph.

For real-world problems, Equations (1)–(3) cannot be satisfied exactly because of noise and outliers in

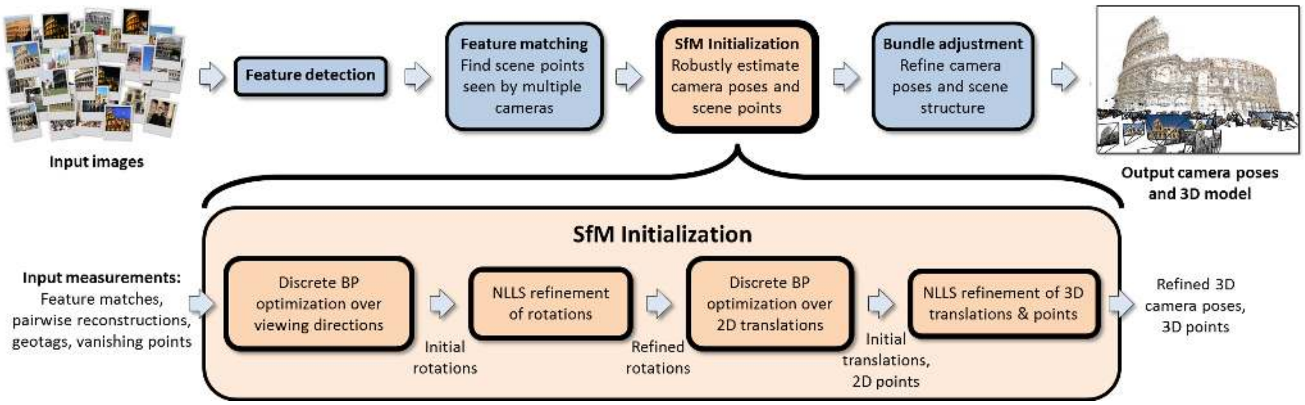

Input images
Feature detection
Feature matching
Find scene points seen by multiple cameras
SfM Initialization
Robustly estimate camera poses and scene points
Bundle adjustment
Refine camera poses and scene structure
Output camera poses and 3D model
SfM Initialization
Input measurements:
Feature matches, pairwise reconstructions, geotags, vanishing points
Discrete BP optimization over viewing directions
Initial rotations
NLLS refinement of rotations
Refined rotations
Discrete BP optimization over 2D translations
Initial translations, 2D points
NLLS refinement of 3D translations & points
Refined 3D camera poses, 3D points

Fig. 1. A *typical structure from motion pipeline* (top row) turns 2D images into 3D geometry. Features are detected and then matched across images, forming tracks. An initialization phase estimates camera and scene geometry, typically using repeated (incremental) rounds of bundle adjustment, followed by a final round of bundle adjustment to solve for camera and scene geometry. In contrast, we propose initializing with a multi-stage optimization (bottom row) that combines discrete belief propagation and continuous optimization.

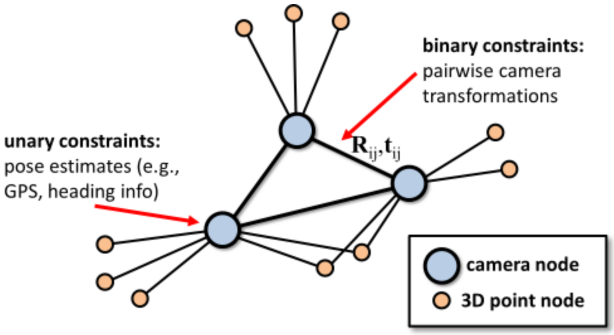


Fig. 2. A *reconstruction graph*, containing a node for each camera (blue), a node for each 3D point (red), some camera-camera edges (representing overlapping images having relative pose estimates) and some camera-node edges (representing visibility of points in images). Each camera-camera edge has estimates $\mathbf{R}_{ij}$ and $\mathbf{t}_{ij}$ of relative pose between the two cameras.

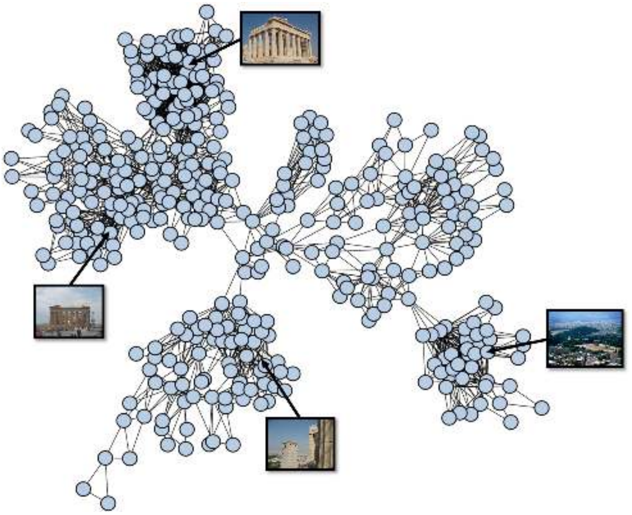


Fig. 3. *A reconstruction subgraph* from the **Acropolis** dataset, showing camera nodes and sample images.

relative pose estimates and point correspondences, so we pose the problem as an optimization which seeks absolute poses most consistent with the constraints according to a cost function. Ideally, one would optimize camera poses and points simultaneously, as in bundle adjustment, but in practice many batch techniques solve for camera rotations and translations separately to reduce computational cost [16], [17], [19]. We follow this custom and define an MRF for each of these two subproblems. A key concern will be to use cost functions that are robust to incorrect geotags, two-frame geometry, and point correspondence.

3.1.1 Rotations

We first solve for absolute 3D camera rotations $\mathbf{R}_i$, given the pairwise relative rotations $\mathbf{R}_{ij}$ as well as any available prior information about camera orientation. From Equation (1) we see that for neighboring images I_i and I_j in the reconstruction graph, we seek absolute camera poses $\mathbf{R}_i$ and $\mathbf{R}_j$ such that $d^{\mathbf{R}}(\mathbf{R}_{ij}, \mathbf{R}_i^\top \mathbf{R}_j)$ is small for some choice of distance function $d^{\mathbf{R}}$ between 3D rotations. This choice of distance function is tightly linked with the choice of parameterization of 3D rotations. Previous linear approaches to this problem have used a squared L_2 distance between 3×3 rotation matrices (i.e., the Frobenius norm) or between quaternions. Such methods relax the orthonormality constraints on these representations, which allows for an approximate least squares solution. In our case, we instead define $d^{\mathbf{R}}$ to be a robustified distance,

$$d^{\mathbf{R}}(\mathbf{R}_a, \mathbf{R}_b) = \rho_R(||\mathbf{R}_a - \mathbf{R}_b||), \tag{4}$$

for a rotation parameterization detailed below and robust function ρ_R (we use a truncated quadratic).

For some cameras we may have noisy orientation evidence from information such as vanishing point detection or electronic compass sensors in the camera.

To incorporate this 'unary' evidence into our optimization, we assume that for each image I_i there is a distance function $d_i^{\mathbf{O}}(\mathbf{R})$ that gives a cost for assigning any absolute orientation $\mathbf{R}$ to I_i's camera. This function can have any form, including uniform if no prior information is available; we propose a particular cost function in Section 4.

We combine the unary and binary distances into a total rotational error function $D^{\mathbf{R}}$,

$$D^{\mathbf{R}}(\mathcal{R}) = \sum_{e_{ij} \in E_C} d^{\mathbf{R}}\left(\mathbf{R}_{ij}, \mathbf{R}_i^\top \mathbf{R}_j\right) + \alpha_1 \sum_{I_i \in \mathcal{I}} d_i^{\mathbf{O}}(\mathbf{R}_i), \quad (5)$$

where $\mathcal{R}$ is an assignment of absolute rotations to the entire image collection, E_C is the set of camera-camera edges, and α_1 is a constant weighting the unary term with respect to the binary term.

3.1.2 *Camera and point positions*

Having solved for camera rotations, we fix them and estimate the positions of cameras and a subset of scene points by solving another MRF inference problem on the graph G. As with the rotations, we define an error function using both binary and unary terms, where binary terms correspond to the pairwise constraints in Equations (2) and (3), and unary terms correspond to prior pose information from geotags.

Equation (2) implies that for a pair of adjacent images I_i and I_j we seek absolute camera positions $\mathbf{t}_i$ and $\mathbf{t}_j$ such that the relative displacement induced by those absolute camera positions, $\mathbf{t}_j - \mathbf{t}_i$, is close to the relative translation estimate $\hat{\mathbf{t}}_{ij} = \mathbf{R}_i \mathbf{t}_{ij}$. Similarly, for a point $\mathbf{X}_k$ visible in image I_i, we want the displacement $\mathbf{X}_k - \mathbf{t}_i$ to be close to the 'ray direction' $\hat{\mathbf{x}}_{ik}$ derived from the 2D position of that point in the image (where $\hat{\mathbf{x}}_{ik} = \mathbf{R}_i^\top \mathbf{K}_i^{-1} \mathbf{x}_{ik}$ given observed position $\mathbf{x}_{ik}$ and known intrinsics $\mathbf{K}_i$). Thus we can utilize both *camera-camera* constraints (derived from two-view geometry) and *camera-point* constraints (derived from point correspondence).

Linear approaches in the literature have considered one or the other of these constraints by observing that the cross product $\hat{\mathbf{t}}_{ij} \times (\mathbf{t}_j - \mathbf{t}_i) = 0$ for camera-camera constraints [14], or that $\hat{\mathbf{x}}_{ik} \times (\mathbf{X}_k - \mathbf{t}_i) = 0$ for camera-point constraints [15]. Taken together over an image collection, these constraints form a homogeneous linear system, but the corresponding least squares problem minimizes a non-robust cost function and disproportionately weights distant points. Alternative formulations based on L_∞ have been defined [18], [17] but these too lack robustness. In contrast, we explicitly handle outliers by defining a robust distance on the angle between displacement vectors,

$$d^{\mathbf{T}}(\mathbf{v}_a, \mathbf{v}_b) = \rho(\mathrm{angleof}(\mathbf{v}_a, \mathbf{v}_b)), \quad (6)$$

where ρ again denotes a robust distance function.

We can also integrate geotags into the optimization. For now we simply assume that there is a cost function $d_i^{\mathbf{G}}(\mathbf{t}_i)$ for each image I_i over the space of translations, which may be uniform if no geotag is available; we propose a particular form for $d_i^{\mathbf{G}}$ in Section 4. We define the translational error of an assignment of absolute positions $\mathcal{T}$ to cameras and points as a combination of binary and unary terms,

$$D^{\mathbf{T}}(\mathcal{T}) = \alpha_2 \sum_{e_{ij} \in E_C} d^{\mathbf{T}}(\mathbf{t}_j - \mathbf{t}_i, \hat{\mathbf{t}}_{ij}) + d^{\mathbf{T}}(\mathbf{t}_i - \mathbf{t}_j, \hat{\mathbf{t}}_{ji}) \; + \\ \alpha_3 \sum_{e_{ik} \in E_P} d^{\mathbf{T}}(\mathbf{X}_k - \mathbf{t}_i, \hat{\mathbf{x}}_{ik}) + \sum_{I_i \in \mathcal{I}} d_i^{\mathbf{G}}(\mathbf{t}_i) \quad (7)$$

where E_C is the set of camera-camera edges in G, E_P is the set of camera-point edges, and α_2 and α_3 are constants. We could ignore either set by fixing α_2 or α_3 to 0; we evaluate these options in Section 5.

3.2 Initial poses and points via discrete BP

The objectives in Equations (5) and (7) can be minimized directly using Levenberg-Marquardt with reweighting for robustness, as we discuss in Section 3.3, but this algorithm requires a good initial estimate of the solution. We tried using raw geotags to initialize the camera positions, for example, but we have found that they alone are too noisy for this purpose. In this section we show how to compute a coarse initial estimate of camera poses and point positions using discrete belief propagation on an MRF.

The reconstruction graph G can be viewed as a first-order MRF with hidden variables corresponding to absolute camera orientations and camera and point positions, observable variables corresponding to prior camera pose information and constraints between pairs of cameras and between cameras and points. In discrete MRF inference one generally wishes to choose a label for each hidden variable, which in our problem corresponds to choosing a discretized rotation or position. The maximum a posteriori inference problem can be viewed as an energy minimization,

$$\min_{\mathcal{L}} \sum_{e_{ij}} d_{ij}(\mathbf{l}_i, \mathbf{l}_j) + \sum_{v_i} d_i(\mathbf{l}_i) \quad (8)$$

where for each edge there is a binary term d_{ij} that measures how well the labels $\mathbf{l}_i$ and $\mathbf{l}_j$ at vertices v_i and v_j are compatible with one another, and for each vertex v_i there is a unary term d_i measuring how well the label $\mathbf{l}_i$ fits the observation for that vertex.

Finding an optimal labeling of an MRF is NP-hard in general, but approximate methods work well on problems like stereo [33]. However, unlike the grid-structured graphs that arise in stereo, our MRF is highly non-uniform (dense in some places and sparse in others; see Figure 3) and the label space is very large. We use loopy discrete belief propagation (BP) to do approximate inference on this MRF efficiently [34]. BP is a message-passing technique in which in each iteration t each node v_i sends a message to each of its neighbors $j \in \mathcal{N}(i)$,

$$m^t_{i,j}(\mathbf{l}_j) = \min_{\mathbf{l}_i} d_{ij}(\mathbf{l}_i, \mathbf{l}_j) + d_i(\mathbf{l}_i) + \sum_{r \in \mathcal{N}(i) \setminus \{j\}} m^{t-1}_{r,i}(\mathbf{l}_i). \tag{9}$$

After running BP for T iterations, each node chooses a label based on its incoming messages,

$$l^*_i = \min_{\mathbf{l}_i} D_p(\mathbf{l}_i) + \sum_{r \in \mathcal{N}(i)} m^T_{r,i}(\mathbf{l}_i). \tag{10}$$

Belief propagation is not guaranteed to converge when applied to graphs with cycles, but it has been found to perform well on many loopy graphs in practice. The running time of BP is $O(mL^2)$ per iteration, where m is the number of edges in the MRF and L is the number of possible labels. However, for certain classes of pairwise distance functions, including the ones we propose here, the messages can be computed in $O(mL)$ time, as we describe in Section 4.

We first solve for absolute camera rotations $\mathcal{R}$ by minimizing Equation (5) using discrete BP. Instead of solving for full 3D rotations, we reduce the state space by assuming that most cameras have little twist (in-plane rotation) because most photos are close to landscape or portrait orientations and modern digital cameras automatically orient images correctly. (We estimate that about 80% of photos in our datasets have less than 5° twist, and 99% have less than 10° twist. The no-twist assumption is made only during the BP stage; in the later NLLS and bundle adjustment stages we allow twist angles to vary.) Under this assumption, camera orientations $\mathbf{R}_i$ can be represented as a single unit 3-vector $\mathbf{v}_i$ (the viewing direction). The distance function in Equation (4) then simplifies to

$$d^{\mathbf{R}_0}(\mathbf{v}_i, \mathbf{v}_j) = \rho_R(||\mathbf{v}_{ij} - \mathbf{R}_0(\mathbf{v}_i)^{-1}\mathbf{v}_j||), \tag{11}$$

where $\mathbf{v}_{ij}$ is the expected viewing direction of camera j in camera i's coordinate system (which can be computed from $\mathbf{R}_{ij}$) and $\mathbf{R}_0(\mathbf{v})$ is a 3D orientation with viewing direction $\mathbf{v}$ and no twist.[2] This distance function simply measures the difference between two viewing directions. For the robust error function we use a truncated quadratic, $\rho_R(x) = \min(x^2, K_R)$, where K_R is a constant (we use 1.0).

Having solved for absolute camera orientations, estimating camera and point positions involves minimizing equation (7). This minimization can also be formulated as an MRF and solved through BP since its form is the same as the MRF energy minimization in Equation (8), where $D^{\mathbf{T}}$ is the binary potential for each edge and $D^{\mathbf{G}}$ is the unary term for each vertex.

3.3 Refining poses using non-linear least squares

Using the coarse estimates of rotations or translations determined by BP, we apply continuous optimization to the objective functions in Equations (5) and (7), using the Levenberg-Marquardt (LM) algorithm for non-linear least squares [35]. Rather than using a robust objective for LM, we simply remove edges and geotags from the reconstruction graph that disagree with the BP estimates by more than a threshold, then run LM using squared residuals. These NLLS steps are related to bundle adjustment in that both minimize a non-linear objective by joint estimation of camera and point parameters. However, our NLLS stages separate rotation estimation from translation estimation, and integrate camera-camera constraints in addition to point-camera constraints. For the optimization over camera rotations, we use an outlier threshold of 20° and for the optimization over camera/point translations, we use an outlier threshold of 40°.

4 Large-scale reconstruction

We now show how to perform SfM on large image sets using the approach described in the previous section. Our method consists of the following steps:

1) Build the reconstruction graph G through image matching and two-view relative pose estimation;
2) Compute noisy priors for some images using geotags and vertical vanishing points;
3) Estimate camera orientations, $\mathcal{R}$, with discrete BP (Section 3.2) followed by continuous optimization (Section 3.3);
4) Estimate positions $\mathcal{T}$ of cameras and a subset of 3D points with BP and continuous optimization;
5) Solve for cameras and points with a single stage of bundle adjustment, initialized with pose estimates from steps 3 and 4.

4.1 Step 1: Producing pairwise transformations.

We use feature matching and two-frame SfM to estimate correspondence and pairwise poses between images. To avoid all-pairs matching, we use two heuristics to quickly find candidate matching image pairs. The first is similar to [2]. We use a vocabulary tree [37] to find, for each image, 80 similar images. For each such candidate image pair, we compute detailed SIFT matches [36] using approximate nearest neighbors [38]. For matching pairs, we estimate relative pose using the 5-point algorithm [32] followed by bundle adjustment. The second heuristic uses geotags to find nearby pairs of photos, as in [3]. For each photo p, we sample 80 photos with probability proportional to $\exp(-d^2/2\sigma^2)$, where d is the distance to p's geotag, and $\sigma = 40$. We match and reconstruct each candidate pair, adding an edge to the graph G if successful. We then alternate between (1) densifying G using query expansion [2], (2) sampling pairs from different connected components (CCs) of G with probability based on proximity, and (3) sampling pairs where exactly one image belongs to the largest CC of G.

[2] $\mathbf{R}_0(\mathbf{v})$ is unique unless $\mathbf{v}$ points straight up or down; in these cases we arbitrarily pick a rotation matrix consistent with $\mathbf{v}$, which may cause (11) to overestimate the error. We found that such cases were uncommon enough to not have an effect on the optimization.

During matching we attempt to remove images that are problematic for SfM. In particular, we discard images wthout EXIF focal length metadata since the 5-point algorithm requires camera intrinsics. We remove panoramas by filtering out photos with aspect ratios outside of $[0.5, 2.0]$. Finally, since our discrete BP assumes that cameras have nearly zero twist, we remove images for which the twist angle of most pairwise transformations in the match graph is above $20°$.

4.2 Step 2: Computing prior evidence

The estimation technique presented in Section 3 can make use of prior pose information on individual cameras, if available. We currently incorporate two sources of information: geotags for estimating positions, and a combination of vanishing point detection and geotags for estimating camera orientation.

4.2.1 *Prior on camera position*

For an image I_i with geotag g_i, we define the positional cost function $d_i^{\mathbf{G}}$ as a distance from the geotag,

$$d_i^{\mathbf{G}}(\mathbf{t}_i) = \rho_T(||\,\mathrm{en}(\mathbf{g}_i) - \pi(\mathbf{t}_i)||), \tag{12}$$

where ρ_T is a truncated quadratic (for robustness), π is a projection of 3D camera positions into a local Cartesian plane tangent to the surface of the earth, and en maps geotags in latitude-longitude coordinates to this plane.[3] Robust distances are essential because geotags are typically noisy and contaminated with outliers [23]. For images without geotags we use a uniform function for $d_i^{\mathbf{G}}$.

4.2.2 *Prior on camera orientation*

For rotations, we define a cost function for each image as a sum of distances over the three rotational axes,

$$d_i^{\mathbf{O}}(\mathbf{R}_i) = d_i^{\theta}(\mathbf{R}_i) + d_i^{\phi}(\mathbf{R}_i) + d_i^{\psi}(\mathbf{R}_i), \tag{13}$$

where d_i^{θ}, d_i^{ϕ}, and d_i^{ψ} measure the error between an absolute camera rotation $\mathbf{R}_i$ and prior pose information in pan, tilt, and twist, respectively. These prior estimates of camera orientation could come from a variety of sources, including the compass and gyroscopes that are common in modern smartphones; in our implementation, we use image analysis to estimate vertical vanishing points (VPs).

Many images of man-made scenes feature prominent vertical lines, which can be used to estimate camera tilt and twist angles. We roughly follow Sinha *et al*. [19], running Canny edge detection, edge linking, line fitting, and Hough voting, where each detected line segment longer than a threshold (40 pixels) votes for a set of VP hypotheses. We then find the three distinct VPs with the highest votes, and take the topmost one (in image coordinates) as the vertical VP (provided it has at least 10 supporting lines and corresponds to a reasonable tilt angle below 45 degrees). For a VP with vertical image coordinate y_i, we compute the tilt angle as $\phi_i = \arccos(\frac{y_i}{f_i})$, where f_i is the focal length (from EXIF metadata), and then define $d_i^{\phi}(\mathbf{R}_i)$ to penalize the tilt of $\mathbf{R}_i$ as a function of angular distance to ϕ_i. (If no vertical VP is found, we use a uniform function for d_i^{ϕ}.) This simple technique typically yields estimates within a few degrees of the true tilt, as we show in Section 5.5. We could similarly estimate twist angle to define a function $d_i^{\psi}(\mathbf{R}_i)$, but do not do this in our current implementation.

To estimate pan angle we observe that Equation (2) constrains the absolute orientation $\mathbf{R}_i$ of camera I_i, given absolute positions of cameras I_i and I_j and the relative translation between them. Using geotags as estimates of the camera positions, we obtain a weak cost distribution for camera pan (heading direction),

$$d_i^{\theta}(\mathbf{R}_i) = \sum_{j \in N(i)} w_i^g w_j^g \min(||\mathbf{R}_i \mathbf{t}_{ij} - \frac{g_{ij}}{||g_{ij}||}||, K_G)^2, \tag{14}$$

where $N(i)$ are the neighboring cameras of I_i, $g_{ij} = \mathrm{en}(\mathbf{g}_j) - \mathrm{en}(\mathbf{g}_i)$, w_i^g and w_j^g indicate whether I_i and I_j have geotags, and K_G is set empirically to 0.7.

4.3 Step 3: Solving for absolute rotations

We do inference on our MRFs using discrete loopy belief propagation (BP), as described in Section 3.2. To parameterize the rotations for BP, we assume zero twist angle and represent the viewing direction as a point on the unit sphere. We discretize the sphere into a 3D grid with 11 cells in each dimension, for a total of 11^3=1331 labels. This parameterization allows the distance in Equation (11) to separate into a sum over dimensions, so that messages can be computed in $O(mL^{\frac{4}{3}})$ time instead of $O(mL^2)$. Substituting the rotational prior and pairwise error terms from Equations (11) and (13) into Equation (9), the message update equation for the MRF in Equation (5) becomes,

$$m_{i,j}^t(\mathbf{l}_j) = \min_{\mathbf{l}_i} \rho_R(||\mu_{ij}(\mathbf{l}_j) - \mathbf{l}_i||) + D_i(\mathbf{l}_i), \tag{15}$$

where

$$\mu_{ij}(\mathbf{l}_j) = [\mu_{ij}^x(\mathbf{l}_j) \;\; \mu_{ij}^y(\mathbf{l}_j) \;\; \mu_{ij}^z(\mathbf{l}_j)]^T = \mathbf{R}_0(\mathbf{l}_j)\mathbf{v}_{ij} \text{ and,}$$

$$D_i(\mathbf{l}_i) = \alpha_1 d_i^{\mathbf{O}}(\mathbf{l}_i) + \sum_{r \in \mathcal{N}(i) \backslash \{j\}} m_{r,i}^{t-1}(\mathbf{l}_i),$$

and we set $\alpha_1 = 1.0$ in all our experiments reported here. Equation (15) can be written as nested minimizations over the viewing direction dimensions,

$$\begin{aligned} m_{i,j}^t(\mathbf{l}_j) = \min\,(K_R^2 + \min_{\mathbf{l}_i} D_i(\mathbf{l}_i), \\ \min_{\mathbf{l}_i^z}\,(\mu_{ij}^z(\mathbf{l}_j) - \mathbf{l}_j^z)^2 + (\min_{\mathbf{l}_i^y}\,(\mu_{ij}^y(\mathbf{l}_j) - \mathbf{l}_j^y)^2 \\ + (\min_{\mathbf{l}_i^x}\,(\mu_{ij}^x(\mathbf{l}_j) - \mathbf{l}_j^x)^2 + D_i(\mathbf{l}_i)))), \end{aligned}$$

[3] This frame is often called *local east-north-up*; we use only the 2D east and north coordinates since geotags do not include altitudes.

where $\mathbf{l}_i = [\mathbf{l}_i^x \ \mathbf{l}_i^y \ \mathbf{l}_i^z]$ [39]. Note that D_i can be computed for all labels in $O(L)$ time, each of the inner minimizations can be performed in $O(L^{\frac{4}{3}})$ time, and the outer minimization can be performed in $O(L)$ time, yielding an overall time of $O(mL^{\frac{4}{3}})$ to compute all messages along all edges of the MRF. Note that in this parameterization only the 482 (of 1331) cells intersecting the surface of the unit sphere are valid states, so all other states are clamped to infinite cost.

We then run non-linear least squares to optimize Equation (4) (using a squared distance), initializing viewing directions to the BP solutions and twist angles to 0. In this optimization, we represent displacement rotations using Rodrigues parameters and allow twist angles to vary. We used Matlab's sparse preconditioned conjugate gradients solver in `lsqnonlin`.

4.4 Step 4: Solving for translations and points

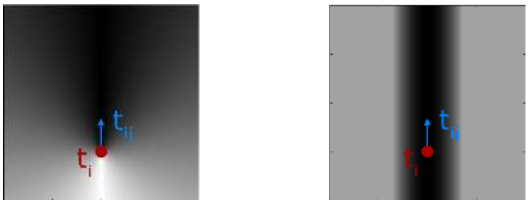
We next use discrete BP to estimate the positions of cameras and some scene points. We identify scene points by finding point *tracks* [2]—interest points across multiple images that have similar SIFT descriptors—and add some of these points as nodes in the MRF. To avoid adding too many nodes, we greedily select a subset of tracks that covers each camera-camera edge in the reconstruction graph at least k_1 times, and that covers each image at least $k_2 \geq k_1$ times (we use $k_1 = 5$ and $k_2 = 10$).

Applying BP to this problem is straightforward, but the large label space (the set of all possible 3D positions) makes a naive implementation costly. We use two strategies for reducing this cost. First, we reduce the label space by solving only for 2D positions, as in most scenes the camera and point positions vary predominantly over the ground plane. (We make this assumption only during discrete BP; the later non-linear least squares and bundle adjustment stages relax this constraint.) We discretize the space according to the geographic size of the region being reconstructed (estimated using geotags), typically using a 300×300 grid where each cell represents an area of about 1-4 meters square (so that $L = 90,000$).

Second, we compute messages efficiently using the generalized distance transform [39]. To allow this, we approximate the distance function in Equation (6) as,

$$d_{app}^{\mathbf{T}}(\mathbf{v}_a, \mathbf{v}_b) = \rho_T(||\mathbf{v}_a \times \mathbf{v}_b||) \qquad (16)$$
$$= \rho_T(||\mathbf{v}_a - (\mathbf{v}_a \cdot \mathbf{v}_b)\mathbf{v}_b||)$$

with robust distance function $\rho_T(x) = \min(x, K_T)^2$ with K_T set to about 10m. Recall that the parameters to this function are translation directions, not absolute displacements, due to the gauge ambiguity in pairwise SfM. This approximation penalizes distant cameras or points more than nearby cameras or points, even if their angular differences are the same (Figure 4). This approximation is related to the linear approach of [14], which uses a non-robust version of $d_{app}^{\mathbf{T}}$ and estimates translations by reweighted least squares; such approaches are sensitive to outliers, as without the truncation the error terms grow quadratically. Hence, we found the use of robust objective functions to be critical in practice.



$d^{\mathbf{T}}$ $\qquad$ $d_{app}^{\mathbf{T}}$

Fig. 4. *Visualization of Equations (6) and (16)*, in which $\mathbf{t}_i$ is at a fixed location and $\mathbf{t}_{ij} = (0, 1)$ (the expected translation is north), and grayscale intensity shows cost as a function of $\mathbf{t}_j$. $d^{\mathbf{T}}$ penalizes based on angular distance from $\mathbf{t}_{ij}$, while $d_{app}^{\mathbf{T}}$ penalizes distant choices of $\mathbf{t}_j$ more than nearby choices that do not lie along $\mathbf{t}_{ij}$.

We use discrete BP to minimize Equation (7) using this approximate distance function. We show how to efficiently compute the BP messages between cameras in detail, but the messages between points and cameras can be computed in a similar manner. The message update in Equation (9) that is implied by the pairwise cost in Equation (7) is difficult to compute efficiently, so we approximate it as

$$m_{i,j}^t(\mathbf{l}_j) = \min_{\mathbf{l}_i} 2\alpha_2 \rho_T(||(\mathbf{l}_j - \mathbf{l}_i) \times \hat{\mathbf{t}}_{ij}||) + D_i(\mathbf{l}_i). \quad (17)$$

Supposing that we rotate the coordinate system such that $\hat{\mathbf{t}}_{ij}{=}(0, 1)$, the messages simplify to

$$m_{i,j}^t(\mathbf{l}_j) = \min(2\alpha_2 K_T^2 + \min_{\mathbf{l}_i} D_i(\mathbf{l}_i),$$
$$\min_{\mathbf{l}_i^y}(\min_{\mathbf{l}_i^x} 2\alpha_2 \, (\mathbf{l}_j^x - \mathbf{l}_i^x)^2 + D_i(\mathbf{l}_i))), \quad (18)$$

where $\mathbf{l}_i = [\mathbf{l}_i^x \ \mathbf{l}_i^y]$ and $\mathbf{l}_j = [\mathbf{l}_j^x \ \mathbf{l}_i^y]$. We set $\alpha_2 = \alpha_3 = 0.5$, determined empirically. The innermost minimization can be performed for all $\mathbf{l}_j$ in linear time using the generalized distance transform [39], while the other minimizations and the computation of D_i can also be performed in linear time. Thus the overall running time for computing all messages is $O(mL)$, where m is the number of edges in the MRF, versus the $O(mL^2)$ time that would normally be required. Rotating the coordinate system such that $\hat{\mathbf{t}}_{ij} = (0, 1)$ means applying a rotation to the sampled (discrete) distribution D_i, performing the minimizations on those buffers, and then applying the opposite rotation to yield $m_{i,j}$ in the global coordinate frame.

Additionally, the approximate distance function lets us store each BP message in $O(\sqrt{L})$ space. Each message contains at most $O(\sqrt{L})$ distinct values, since $m_{i,j}(\mathbf{l}_j)$ in Equation (18) is a function only of l_j^x in the rotated space. Thus we can compress $m_{i,j}(\mathbf{l}_j)$ by storing a single row of the message in the rotated space along with the rotation angle.

We next apply non-linear least squares optimization with the solution found via BP as initialization, using `lsqnonlin` to minimize the squared residuals in Equation (7), allowing cameras and points to vary in height as well as ground position.

4.5 Step 5: Bundle adjustment

We use the estimates for the cameras and a sparse set of 3D points obtained in the last step as initialization to a global bundle adjustment stage in which all parameters including camera twist and height are refined simultaneously. We bundle adjust cameras and the subset of 3D points selected in the previous step, triangulate the remaining points with angular reprojection error below a threshold (in our implementation, we use 6°), and run a final bundle adjustment. For bundle adjustment, we use a preconditioned conjugate gradients bundle adjuster [10]. Because there are still outliers present in the correspondences, we use a robust Huber norm, with a parameter of 25 pixels, on the reprojection error.

5 Experimental Results

5.1 Datasets

We applied our reconstruction system to several large-scale image datasets, as summarized in Table 1. Three of our datasets were collected by downloading public images from Flickr, a popular photo-sharing website: **Acropolis** has 2,961 images geotagged within 100m of the Acropolis in Athens (37.9714°N, 23.7261°E), **Dubrovnik** has 12,092 photos tagged *dubrovnik* or geotagged within 500m of the center of Dubrovnik, Croatia (42.6415°N, 18.1084°E), and **CentralRome** has 74,394 images geotagged within 1km of the Roman Forum (41.8925°N, 12.4857°E). We used the public Flickr API to collect these datasets, downloading the highest-resolution image available for each photo.

We also use two datasets for which some ground truth is available. **Quad** consists of 6,514 images of the Arts Quad at Cornell University taken by the authors. For this dataset we recorded very accurate camera positions (error under 10cm) for a subset of 348 photos using differential GPS, in addition to geotags from a consumer GPS device (an iPhone 3G). We use the precise geotags as ground-truth and the iPhone geotags as priors in our optimization (as proxies for noisy geotags from photo-sharing sites). **SanFrancisco** consists of images of downtown San Francisco collected from NavTeq [40] and includes accurate camera position and orientation observations.

5.2 Qualitative results

For each dataset we ran our complete reconstruction system described in Section 4, including image matching and 2-frame SfM to build a match graph, discrete BP and continuous NLLS to solve for camera rotations and camera and point positions, and then a final round of bundle adjustment. Figure 5 shows sample views of our 3D models after applying a multiview stereo algorithm [41] to densify the point clouds.

Alternatively, we can visualize the reconstructions from above by overlaying estimated point and camera positions on a satellite map. Figure 6 shows reconstructions of CentralRome aligned to a map, comparing the results of our approach with those of the incremental technique of Agarwal *et al.* [2]. We see that our estimated scene points (in blue) align well with structures visible in the satellite map, and our estimated camera positions (in black) coincide well with sidewalks and roadways. In contrast, IBA produced a poor reconstruction; our approach likely performed better because the geotag priors helped to avoid problems with sparsely-connected components of the reconstruction graph. Figure 7 illustrates how the stages of our approach take noisy geotags and successively refine them into accurate camera poses.

5.3 Quantitative evaluations

Ideally, we would compare our reconstructions to dense, high-quality 3D ground truth (e.g., from a laser scanner), but it is difficult to collect this data for the scale of scenes that we consider here (and for this reason most papers do not attempt quantitative evaluations [3], [5], [10]). Instead, we evaluate our system using two imperfect techniques that nevertheless give some quantitative measurements. We first compare to reconstructions produced by state-of-the-art incremental bundle adjustment, showing that our approach produces similar point clouds and camera poses. Second, we measure the accuracy of the estimated camera poses on two datasets for which we have (incomplete) ground truth on camera pose.

5.3.1 Comparison to IBA

We compared our method to state-of-the-art IBA: a version of Bundler [5] that uses an efficient bundle adjuster with preconditioned conjugate gradients [10], then georegisters the model with the geotags using RANSAC. Table 2 summarizes the results of this comparison, including distances between corresponding camera positions and viewing directions. It is important to note that the IBA solution has errors and is thus not ground truth, but is a state-of-the-art SfM system and thus provides a useful comparison. Our results show that raw geotags are quite noisy, with a median translation error of over 100 meters for some datasets. The estimates from BP are significantly better, and results from the full process agree with the IBA solution within a meter for all datasets except CentralRome. The differences for CentralRome are large because IBA produced a low-quality reconstruction on this dataset, as discussed above. The median differences between

TABLE 1
Summary of datasets.

Dataset	Images matched	Largest component size ($\|V\|$)	Camera-camera edges ($\|E_C\|$)	Camera-point edges ($\|E_P\|$)	% images geotagged	Scene size (km^2)	Reconstructed images
Acropolis	2,961	463	22,842	42,255	100.0%	0.1×0.1	454
Quad	6,514	5,520	444,064	551,670	77.2%	0.4×0.3	5,233
Dubrovnik	12,092	6,854	1,000,178	835,310	56.7%	1.0×0.5	6,532
CentralRome	74,394	15,242	864,758	1,393,658	100.0%	1.5×0.8	14,754
SanFrancisco	17,357	7,866	203,024	515,100	100.0%	1.0×0.4	5,197

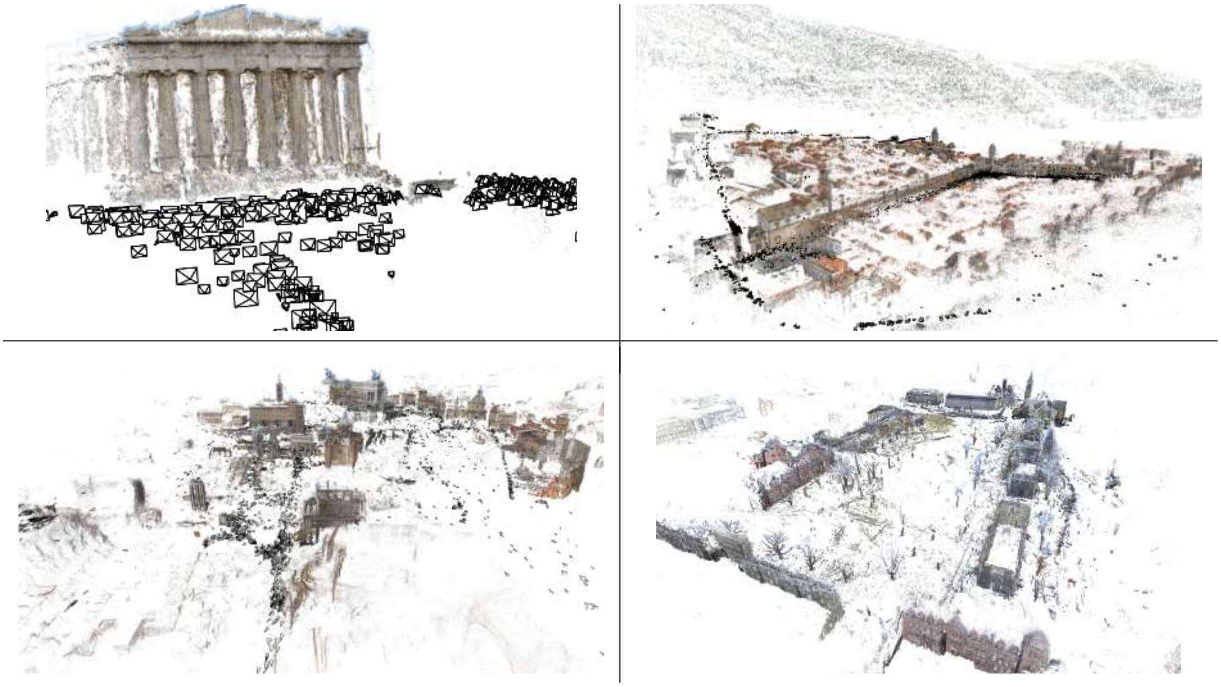


Fig. 5. *Sample reconstructions* for (clockwise from top left) Acropolis, Dubrovnik, Quad, and CentralRome.

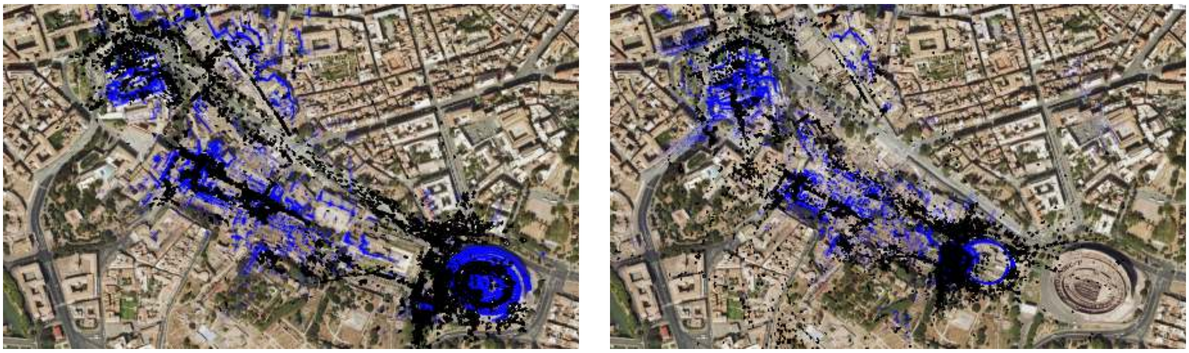
scene point positions for the two methods are also less than 1m for all datasets except CentralRome. The median angle between camera viewing directions for the IBA and BP solutions is between about 5° and 14°, with the continuous optimization reducing the difference below 5°, and the final BA further reducing it below 0.5° for all scenes except CentralRome. In terms of reconstruction size, our method produced a somewhat larger reconstruction for Quad than did IBA (5,233 versus 5,017 images), whereas IBA produced somewhat larger reconstructions for the other scenes (e.g. 462 vs. 454 images for Acropolis, and 6,844 vs. 6,532 images for Dubrovnik). Upon further inspection, we observed that our approach was better at bridging weak connections between two larger components, while IBA was better at connecting individual weakly connected images (e.g., blurry or low-resolution photographs), possibly due to our use of a subset of points for the initial bundle adjustment.

5.3.2 *Comparison to ground truth*

For a comparison with ground truth, we used two datasets for which we have good absolute pose information for some cameras: Quad, which has camera positions for several hundred images, and SanFrancisco, which has camera positions and orientations (see section 5.1). Table 3 presents median camera pose errors for these two datasets at various stages of our method. For Quad, IBA produces slightly better camera position estimates than our approach, but the difference is quite small (1.01m median error versus 1.16m). The table also studies the sensitivity of our approach to the fraction of photos having geotags. As the fraction of geotagged images decreases below about 10%, the accuracy starts to decrease. This seems to be due to less accurate global rotation estimates, indicating that weak orientation information is helpful.

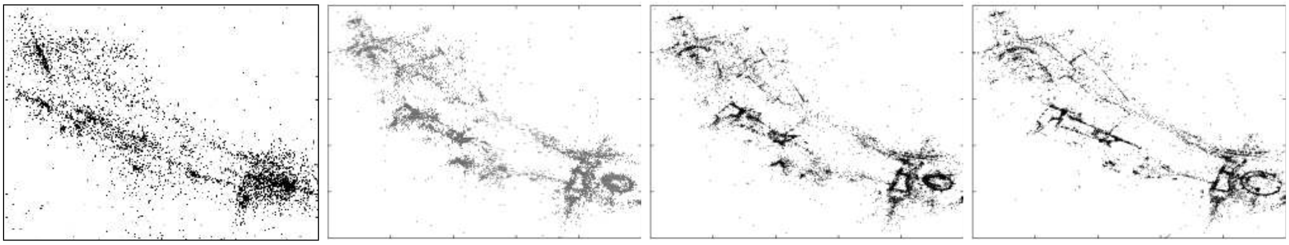
5.4 Running times

As shown in Table 4, our approach is significantly faster than incremental bundle adjustment on all of our datasets, but especially the larger ones: our approach is about six times faster than IBA on CentralRome, for example. We used a multi-threaded implementation of rotations BP on a single 16-core 3.0GHz machine, a single-threaded implementation of NLLS on the same machine, and a map-reduce implementation of translations BP on a 200-core 2.6GHz Hadoop



Our approach **Incremental bundle adjustment [2]**

Fig. 6. *CentralRome reconstruction*, using our approach (left) and incremental bundle adjustment (right), overlaid on a map. Black points are cameras while blue points are scene points. The IBA solution exhibits a large drift: the scale of the Colosseum (lower right) is too small given the scale of Il Vittoriano (upper left), while the inside and outside of the Colosseum do not align. The scale and alignment in our solution is much more consistent.



Raw geotags **After BP** **After NLLS** **After bundle adjustment**

Fig. 7. *Camera position estimates for CentralRome* at different stages of optimization. Raw geotags (left) are noisy with little visible structure. BP estimates camera positions on a discrete grid, providing initialization for non-linear least squares. These estimates in turn initialize a final round of bundle adjustment.

TABLE 2
Comparison with Incremental BA in terms of median differences for point positions and camera poses.

	Rotational difference					Translational difference				Point difference
	Our approach			Linear approach [14]		Our approach				Our approach
Dataset	BP	NLLS	Final BA	Linear	NLLS	Geotags	BP	NLLS	Final BA	Final BA
Acropolis	14.1°	1.5°	**0.2°**	1.6°	1.6°	12.9m	8.1m	2.4m	**0.1m**	**0.2m**
Quad	4.7°	4.6°	**0.2°**	41°	41°	15.5m	16.6m	14.2m	**0.6m**	**0.5m**
Dubrovnik	9.1°	4.9°	**0.1°**	11°	6°	127.6m	25.7m	15.1m	**1.0m**	**0.9m**
CentralRome	6.2°	3.3°	**1.3°**	27°	25°	413.0m	27.3m	27.7m	**25.0m**	**24.5m**

cluster. For BA and IBA we used the highly-optimized bundle adjuster of [2], which uses a multi-threaded BLAS library, on a single 16-core 3.0GHz machine. One of the reasons for our speed-up is that BP (unlike IBA) is easily parallelizable on a distributed memory cluster; if we had instead run BP on a single machine, our running times would have increased but would still be faster than IBA (about 0.4 hours versus 0.5 on Acropolis, 18.0 hours versus 62 hours on Quad, 11.1 hours versus 28 hours on Dubrovnik, and 48.9 hours versus 82 hours on CentralRome). The majority of our approach's running time is spent in the final bundle adjustment stage; this step can be omitted if one simply wants to infer coarse camera pose without reconstructing the scene.

The asymptotic running time of our approach also compares favorably to that of IBA. In contrast to the worst case $O(n^4)$ running time of IBA (using dense linear algebra and adding images at a constant rate), where n is the number of images, our approach is $O(n^3)$: each application of belief propagation takes time $O(n^2 L)$ per iteration, where L is the size of the label space, and the final bundle adjustment step takes $O(n^3)$ time in the worst case. Memory use is $O(n^2 L)$ for the rotations BP and $O(n^2 \sqrt{L})$ for the translations BP (assuming a dense reconstruction graph).

5.5 Discussion

Role of priors. Table 3 shows that the geotag-based priors are important in producing accurate reconstructions with our method. When pan angle priors are removed from the rotations estimation, the median error increases to 72.1° after BP and to 33.6° after NLLS and bundle adjustment on SanFrancisco. This is due

TABLE 3

Comparison with ground truth for Quad and SanFrancisco at various stages of reconstruction, in terms of median error, as the number of images with (noisy) geotags is varied. Median error of IBA on Quad was 1.01m.

% of images	Quad translations			SanFrancisco rotations			SanFrancisco translations		
geotagged	BP	NLLS	Final BA	BP	NLLS	Final BA	BP	NLLS	Final BA
100%	—	—	—	5.78°	3.91°	**3.30°**	4.26m	4.98m	**3.83m**
80%	7.50m	7.24m	**1.16m**	5.94°	3.84°	**3.39°**	5.48m	4.86m	**4.02m**
40%	7.67m	7.37m	**1.21m**	6.29°	3.96°	**3.79°**	6.58m	5.75m	**4.94m**
16%	7.66m	7.63m	**1.22m**	8.99°	4.30°	**3.18°**	5.95m	6.47m	**4.84m**
8%	8.27m	8.06m	**1.53m**	17.97°	4.44°	**4.15°**	8.19m	7.07m	**5.23m**
4%	18.25m	16.56m	**5.01m**	24.33°	5.98°	**5.24°**	9.70m	8.85m	**6.49m**

TABLE 4

Running times of our approach compared to incremental bundle adjustment.

	Our approach						Incremental
Dataset	Rot BP	Rot NLLS	Trans BP	Trans NLLS	Bund Adj	Total	BA
Acropolis	50s	16s	7m 24s	49s	5m 36s	**0.2 hours**	**0.5 hours**
Quad	40m 57s	8m 46s	53m 51s	40m 22s	5h 18m 00s	**7.7 hours**	**62 hours**
Dubrovnik	28m 19s	8m 28s	29m 27s	7m 22s	4h 15m 57s	**5.5 hours**	**28 hours**
CentralRome	1h 8m 24s	40m 0s	2h 56m 36s	1h 7m 51s	7h 20m 00s	**13.2 hours**	**82 hours**

TABLE 5

Effect of BP label space discretization on median camera errors for SanFrancisco with geotags for 40% of photos. Running times include both BP and NLLS and are on a single machine (not parallelized).

Rotations	Translations	Rotations	Translations	Rotational error			Translational error		
label space	label space	running time	running time	BP	NLLS	Final BA	BP	NLLS	Final BA
$3 \times 3 \times 3$	151×151	9m 49s	2h 17m 23s	24.98°	7.82°	6.76°	7.17m	7.13m	6.13m
$7 \times 7 \times 7$	151×151	14m 31s	2h 20m 16s	7.91°	3.69°	3.52°	6.53m	5.64m	4.67m
$11 \times 11 \times 11$	151×151	20m 47s	2h 28m 56s	6.29°	3.96°	3.79°	6.58m	5.75m	4.94m
$15 \times 15 \times 15$	151×151	28m 31s	2h 20m 19s	5.03°	4.01°	3.78°	6.72m	5.85m	4.82m
$19 \times 19 \times 19$	151×151	38m 33s	2h 22m 08s	4.30°	3.86°	3.59°	6.26m	5.39m	4.34m
$11 \times 11 \times 11$	51×51	20m 47s	33m 40s	6.29°	3.96°	3.79°	15.26m	13.69m	10.77m
$11 \times 11 \times 11$	101×101	20m 47s	1h 7m 39s	6.29°	3.96°	3.79°	8.67m	7.61m	5.28m
$11 \times 11 \times 11$	151×151	20m 47s	2h 28m 56s	6.29°	3.96°	3.79°	6.58m	5.75m	4.94m
$11 \times 11 \times 11$	201×201	20m 47s	7h 3m 36s	6.29°	3.96°	3.79°	5.03m	4.97m	4.50m
$11 \times 11 \times 11$	251×251	20m 47s	20h 5m 4s	6.29°	3.96°	3.79°	4.40m	4.92m	4.22m

in part to weak connections between tightly connected components in the graph, which can cause estimated camera orientations to be consistent with one another inside each component but not globally. Geotags help to enforce consistency across such weakly-connected parts of the graph. For the translations MRF, geotags play a similar role but also prevent the MRF from finding a trivial, zero-cost solution in which all cameras are placed at exactly the same 3D position.

The tilt priors from vertical vanishing point estimation (Section 4.2.2) produce a small but significant improvement in reconstruction accuracy: for Quad with 40% of images geotagged, removing these priors increases median camera position error after bundle adjustment from 1.21m to 1.25m, and for SanFrancisco increases the rotational error from 3.79° to 3.89° and the translational error from 4.94m to 5.17m. To measure the accuracy of the tilt priors, we compared the tilts estimated from vanishing points with those obtained from a reconstruction of the scene using IBA. The median angular difference was 3.7° for Acropolis, 2.1° for Quad, 4.3° for Dubrovnik, and 9.0° for CentralRome. In contrast, if we simply assumed that all cameras have no tilt, our errors would have been significantly higher: 15.8° for Acropolis, 10.5° for Quad, 6.8° for Dubrovnik, and 9.3° for CentralRome.

Role of label space discretization. The discretization of the label space during BP is important to both the running time and the quality of the solution, as shown in Table 5 for SanFrancisco. For rotations, a relatively coarse space of 7×7×7 produces almost the same reconstruction results as much finer discretizations at a fraction of the computational cost. The reconstructions begin to suffer when the label space drops to 3×3×3. For translations, we find that very coarse discretizations yield poor reconstructions while finer discretization produce better results, but that at some point the increase in computational cost outweighs the marginal improvement in reconstruction quality; for SanFrancisco the camera position error drops from 5.28m to 4.50m when the label space is increased from 101×101 to 201×201, but the running time increases by about a factor of 5.

Role of scene points. As discussed in Section 3, in the translations MRF we include pairwise constraints between pairs of cameras and between camera-point pairs. We have found that including both of these edge types improves the results: using only camera-

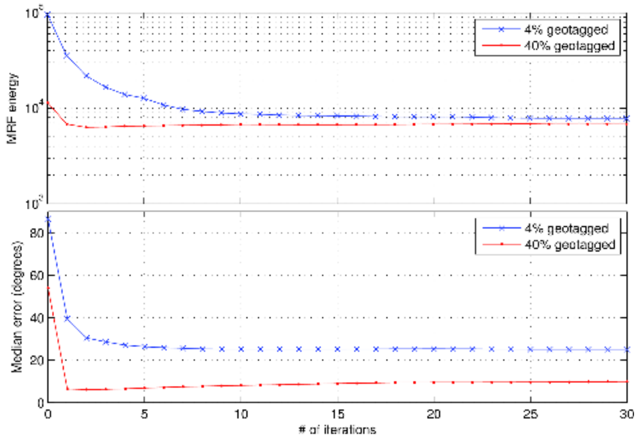


Fig. 8. Rotational MRF energy (top) and error with respect to ground truth after BP (bottom), as a function of number of BP iterations, for SanFrancisco.

point edges increases the final median camera position error after bundle adjustment by about 60% (from 1.21m to 1.9m) for Quad and only very slightly (4.94m to 4.97m) for SanFrancisco. Using only camera-camera edges increases the error by more than 300% for Quad (from 1.21m to 3.93m) and by almost 50% for SanFrancisco (from 4.94m to 7.14m).

BP energy minimization. The number of iterations required for BP to find a good solution is proportional to the diameter of the graph, since evidence only propagates a single hop in each iteration. Figure 8 plots the rotational MRF energy and angular error with respect to ground truth for SanFrancisco. When 40% of photos are geotagged, BP reaches a local minimum in 4 iterations; when only 4% of photos are geotagged, BP still reaches a local minimum within about 12 iterations. Since BP is not guaranteed to converge, we run BP for at least 30 iterations and use the iteration with minimum energy as the solution.

Importance of robustness. We tried the linear batch approach of Govindu [14] on these datasets, and found that it produced reasonable rotation estimates for the densely-connected Acropolis and Dubrovnik sets, but poor results for the others (see Table 2), even after running NLLS and bundle adjustment on its output. The translation estimates were very poor for all datasets, even after modifying it to use geotag priors. This suggests that robust optimization is important for large, noisy datasets (as most evaluations of linear approaches are on much simpler datasets [14], [19]). We also tried simpler initializations to BA and NLLS, including random initialization of camera pose and point locations as well as initializing translations using geotags, but both resulted in poor reconstructions, suggesting that good initialization is also critical.

6 Conclusion

We have presented a SfM approach that avoids solving sequences of incremental bundle adjustment problems by initializing all cameras at once using hybrid discrete-continuous optimization on an MRF and integrating prior evidence from geotags and vanishing points. Our approach is faster than incremental SfM in practice and asymptotically, and it gives better reconstructions on scenes with weakly-connected match graphs. These results demonstrate the utility of discrete optimization, long used in other vision problems such as stereo, in the domain of SfM, when used in conjunction with continuous optimization. In future work we plan to study the performance and tradeoffs of our algorithm on even larger datasets. We also plan to study improvements to our approach, including solving for rotations and translations in a single optimization step and exploring optimization schemes tailored to our MRFs (which are much more complex than the simple grid graphs with 1D label spaces that arise in low-level vision). For example, the distribution of photos across space is highly non-uniform, so we might use hierarchical or adaptive discretizations of the label space.

Acknowledgments

We thank Jon Kleinberg for the idea of applying MRFs to SfM, and Cornell Facilities for measuring survey points for the Quad dataset. This work was supported by the National Science Foundation (IIS-0705774 and IIS-0964027), the Indiana University Data to Insight Center, the Lilly Endowment, Quanta Computer, MIT Lincoln Labs, and Intel Corp., and used compute resources of the Cornell Center for Advanced Computing and IU (funded by NSF EIA-0202048 and IBM).

References

[1] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher, "Discrete continuous optimization for large-scale structure from motion," in *Proc. CVPR*, 2011.

[2] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski, "Building Rome in a day," in *Proc. ICCV*, 2009.

[3] J.-M. Frahm, P. Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, and S. Lazebnik, "Building Rome on a cloudless day," in *Proc. ECCV*, 2010.

[4] X. Li, C. Wu, C. Zach, S. Lazebnik, and J.-M. Frahm, "Modeling and recognition of landmark image collections using iconic scene graphs," in *Proc. ECCV*, 2008, pp. 427–440.

[5] N. Snavely, S. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3D," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 835–846, 2006.

[6] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle adjustment: a modern synthesis," in *Vision Algorithms: Theory and Practice*. Springer, 2000.

[7] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: a factorization method," *IJCV*, vol. 9, no. 2, pp. 137–154, 1992.

[8] F. Schaffalitzky and A. Zisserman, "Multi-view matching for unordered image sets, or "How do I organize my holiday snaps?"," in *Proc. ECCV*, 2002, pp. 414–431.

[9] M. Byrod and K. Astrom, "Conjugate gradient bundle adjustment," in *Proc. ECCV*, 2010, pp. 114–127.

[10] S. Agarwal, N. Snavely, S. Seitz, and R. Szeliski, "Bundle adjustment in the large," in *Proc. ECCV*, 2010.

[11] F. Bajramovic and J. Denzler, "Global uncertainty-based selection of relative poses for multi camera calibration," in *Proc. BMVC*, 2008.

[12] N. Snavely, S. Seitz, and R. Szeliski, “Skeletal graphs for efficient structure from motion,” in *Proc. CVPR*, 2008.

[13] J. Vergés-Llahí, D. Moldovan, and T. Wada, “A new reliability measure for essential matrices suitable in multiple view calibration,” in *Proc. VISAPP*, 2008, pp. 114–121.

[14] V. M. Govindu, “Combining two-view constraints for motion estimation,” in *Proc. CVPR*, 2001, pp. 218–225.

[15] C. Rother, “Linear multi-view reconstruction of points, lines, planes and cameras using a reference plane,” in *Proc. ICCV*, 2003, pp. 1210–1217.

[16] D. Martinec and T. Pajdla, “Robust rotation and translation estimation in multiview reconstruction,” in *Proc. CVPR*, 2007.

[17] K. Sim and R. Hartley, “Recovering camera motion using l∞ minimization,” in *Proc. CVPR*, 2006, pp. 1230–1237.

[18] F. Kahl and R. Hartley, “Multiple-view geometry under the l-infinity-norm,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 9, pp. 1603–1617, Sep. 2008.

[19] S. Sinha, D. Steedly, and R. Szeliski, “A multi-stage linear approach to structure from motion,” in *Proc. ECCV*, 2010.

[20] P. Lothe, S. Bourgeois, F. Dekeyser, E. Royer, and M. Dhome, “Towards geographical referencing of monocular SLAM reconstruction using 3D city models: Application to real-time accurate vision-based localization,” in *Proc. CVPR*, 2009.

[21] S. Sinha, D. Steedly, R. Szeliski, M. Agrawala, and M. Pollefeys, “Interactive 3D architectural modeling from unordered photo collections,” *ACM Trans. Graph.*, vol. 27, no. 5, p. 159, 2008.

[22] R. Kaminsky, N. Snavely, S. Seitz, and R. Szeliski, “Alignment of 3D point clouds to overhead images,” in *CVPR Workshop on Internet Vision*, 2009.

[23] C. Strecha, T. Pylvänäinen, and P. Fua, “Dynamic and scalable large scale image reconstruction,” in *Proc. CVPR*, 2010.

[24] F. Dellaert and M. Kaess, “Square root SAM: Simultaneous localization and mapping via square root information smoothing,” *I. J. Robotic Res.*, vol. 25, no. 12, pp. 1181–1203, 2006.

[25] A. Ranganathan, M. Kaess, and F. Dellaert, “Loopy SAM,” in *IJCAI*, 2007, pp. 2191–2196.

[26] R. Tron and R. Vidal, “Distributed image-based 3-D localization of camera sensor networks,” in *Proc. IEEE Conference on Decision and Control*, 2009.

[27] D. Devarajan and R. J. Radke, “Calibrating distributed camera networks using belief propagation,” *EURASIP J. Adv. Sig. Proc.*, vol. 2007, 2007.

[28] A. Ihler, J. Fisher, R. Moses, and A. Willsky, “Nonparametric belief propagation for self-localization of sensor networks,” *IEEE J. Sel. Areas. Commun.*, vol. 23, no. 4, pp. 809–819, 2005.

[29] K. Ni, D. Steedly, and F. Dellaert, “Out-of-core bundle adjustment for large-scale 3d reconstruction,” in *Proc. ICCV*, 2007.

[30] R. Gherardi, M. Farenzena, and A. Fusiello, “Improving the efficiency of hierarchical structure-and-motion,” in *Proc. CVPR*, 2010, pp. 1594–1600.

[31] V. Lempitsky, S. Roth, and C. Rother, “FusionFlow: Discrete-continuous optimization for optical flow estimation,” in *Proc. CVPR*, 2008.

[32] D. Nistér, “An efficient solution to the five-point relative pose problem,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 6, pp. 756–777, 2004.

[33] Y. Boykov, O. Veksler, and R. Zabih, “Fast approximate energy minimization via graph cuts,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, 2001.

[34] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*. San Mateo: Morgan Kaufmann, 1988.

[35] J. Nocedal and S. J. Wright, *Numerical optimization*, 2nd ed. New York: Springer, 2006.

[36] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *IJCV*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[37] D. Nistér and H. Stewénius, “Scalable recognition with a vocabulary tree,” in *Proc. CVPR*, 2006, pp. 2161–2168.

[38] S. Arya and D. Mount, “Approximate nearest neighbor queries in fixed dimensions,” in *Proc. SODA*, 1993.

[39] P. Felzenszwalb and D. Huttenlocher, “Efficient belief propagation for early vision,” *IJCV*, vol. 70, no. 1, pp. 41–54, 2006.

[40] D. M. Chen, G. Baatz, K. Köser, S. S. Tsai, R. Vedantham, T. Pylvänäinen, K. Roimela, X. Chen, J. Bach, M. Pollefeys, B. Girod, and R. Grzeszczuk, “City-scale landmark identification on mobile devices,” in *Proc. CVPR*, 2011.

[41] Y. Furukawa and J. Ponce, “Accurate, dense, and robust multiview steropsis,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 8, pp. 1362–1376, 2010.

David Crandall is an assistant professor in the School of Informatics and Computing at Indiana University in Bloomington. He received the Ph.D. in Computer Science from Cornell University in 2008 and the M.S. and B.S. degrees in Computer Science and Engineering from the Pennsylvania State University in 2001. He was a Postdoctoral Research Associate at Cornell from 2008–2010, and a Senior Research Scientist with Eastman Kodak Company from 2001–2003. His research interests are computer vision and data mining, with a focus on object recognition, large-scale visual mining, applied machine learning, and modeling complex networks and systems.

Andrew Owens is a Ph.D. student in the Computer Science and Artificial Intelligence Laboratory at the Massachusetts Institute of Technology. He received the B.A. in Computer Science from Cornell University in 2010. He is a recipient of the National Defense Science and Engineering Fellowship and the National Science Foundation Graduate Research Fellowship. His research interests include computer vision and machine learning.

Noah Snavely is an assistant professor of Computer Science at Cornell University, where he has been on the faculty since 2009. He received a B.S. in Computer Science from the University of Arizona in 2003, and a Ph.D. in Computer Science and Engineering from the University of Washington in 2008. Noah works in computer graphics and computer vision, with a particular interest in using vast amounts of imagery from the Internet to reconstruct and visualize our world in 3D. His thesis work was the basis for Microsoft’s Photosynth, a tool for building 3D visualizations from photo collections that has been used by many thousands of people. Noah is the recipient of a Microsoft New Faculty Fellowship and an NSF CAREER Award.

Daniel Huttenlocher is Vice Provost, Dean of the Faculty of Computing and Information Science, and the John P. and Rilla Neafsey Professor of Computer Science and Business at Cornell University. He has been the chief technology officer of Intelligent Markets and a member of senior management at Xerox PARC. His research interests include computer recognition of visual information, studies of online social networks, the role of technology in transforming financial markets, and management of high-performance software development teams. He is a fellow of the Association for Computing Machinery, holds 24 U.S. patents, and has published extensively in computer vision and artificial intelligence. He has also received a number of distinguished teaching awards and, in 1996, was named a Stephen H. Weiss Presidential Fellow in recognition of teaching excellence.