Large Scale SfM with the Distributed Camera Model

Chris Sweeney
University of Washington
Seattle, Washington, USA
csweeney@cs.washington.edu

Victor Fragoso
West Virginia University
Morgantown, West Virginia, USA
victor.fragoso@mail.wvu.edu

Tobias Höllerer
University of California Santa Barbara
Santa Barbara, California, USA
holl@cs.ucsb.edu

Matthew Turk
University of California Santa Barbara
Santa Barbara, California, USA
mturk@cs.ucsb.edu

Abstract

We introduce the distributed camera model, a novel model for Structure-from-Motion (SfM). This model describes image observations in terms of light rays with ray origins and directions rather than pixels. As such, the proposed model is capable of describing a single camera or multiple cameras simultaneously as the collection of all light rays observed. We show how the distributed camera model is a generalization of the standard camera model and describe a general formulation and solution to the absolute camera pose problem that works for standard or distributed cameras. The proposed method computes a solution that is up to 8 times more efficient and robust to rotation singularities in comparison with gDLS[20]. Finally, this method is used in an novel large-scale incremental SfM pipeline where distributed cameras are accurately and robustly merged together. This pipeline is a direct generalization of traditional incremental SfM; however, instead of incrementally adding one camera at a time to grow the reconstruction the reconstruction is grown by adding a distributed camera. Our pipeline produces highly accurate reconstructions efficiently by avoiding the need for many bundle adjustment iterations and is capable of computing a 3D model of Rome from over 15,000 images in just 22 minutes.

1. Introduction

The problem of determining camera position and orientation given a set of correspondences between image observations and known 3D points is a fundamental problem in computer vision. This set of problems has a wide range of applications in computer vision, including camera calibration, object tracking, simultaneous localization and mapping (SLAM), augmented reality, and structure-from-motion (SfM). Incremental SfM is commonly used to create a 3D model from a set of images by sequentially adding images to an initial coarse model thereby "growing" the model incrementally [18]. This incremental process is extremely robust at computing a high quality 3D model due to many opportunities to remove outliers via RANSAC and repeated use of bundle adjustment to reduce errors from noise.

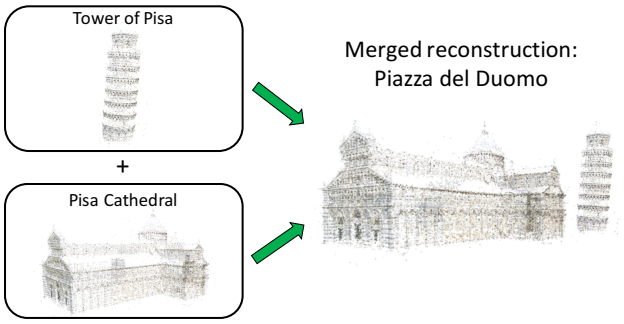

Figure 1. We are able to merge individual reconstructions by representing multiple cameras as a single distributed camera. The proposed merging process localizes the distributed camera to the current 3D model by solving the generalized absolute pose-and-scale problem.

A core limitation of incremental SfM is its poor scalability. At each step in incremental SfM, the number of cameras in a model is increased by Δ. For standard incremental SfM pipelines $\Delta = 1$ because only one camera is added to the model at a time. In this paper, we propose to increase the size of Δ, thereby increasing the rate at which we can grow models, by introducing a novel camera parameterization called the *distributed camera model*. The distributed camera model encapsulates image and geometric information from one or multiple cameras by describing pixels as a collection of light rays. As such, observations from multiple cameras may be described as a single distributed camera.

Definition 1 *A distributed camera is a collection of observed light rays coming from one or more cameras, parameterized by the ray origins c_i, directions $\bar{x}_i$, and a single scale parameter for the distributed camera.*

The distributed camera model is similar to the generalized camera model [16] with the important distinction that the scale of a distributed camera is unknown and must be recovered[1]. It is also a direct generalization of the standard camera model which occurs when all c_i are equal (i.e. all light rays have the same origin). If we can use distributed cameras in incremental SfM, we can effectively increase the size of Δ. This is because we can add multiple cameras (represented as a single distributed camera) in a single step. This dramatically improves the scalability of the incremental SfM pipeline since it grows models at a faster rate.

In order to use the distributed camera model for incremental SfM, we must determine how to add distributed cameras to the current model. While standard incremental SfM pipelines adds a camera at a time by solving for its absolute pose from 2D-3D correspondences, the proposed method adds a distributed camera by solving the generalized pose-and-scale problem from 2D-3D correspondences. As part of this work, we show that the generalized pose-and-scale problem is a generalization of the P*n*P problem to multiple cameras which are represented by a distributed camera; we recover the position and orientation as well as the internal scale of the distributed camera with respect to known 3D points. Our solution method improves on previous work [20] by using the Gröbner basis technique to compute the solution more accurately and efficiently. Experiments on synthetic and real data show that our method is more accurate and scalable than other alignment methods.

We show the applicability of the distributed camera model for incremental SfM in a novel incremental pipeline that utilizes the generalized pose-and-scale method for model-merging. We call our method the generalized Direct Least Squares+++ (gDLS+++) solution, as it is a generalization of the DLS PnP algorithm presented in [9] and an extension of the gDLS algorithm [20][2]. Our pipeline achieves state-of-the-art results on large scale datasets and reduces the need for expensive bundle adjustment iterations.

In this paper, we make the following contributions:

1. A new camera model for 3D reconstruction: the *distributed camera model*.

2. Theoretical insight to the absolute pose problem. We show that the gPnP+s problem [20] is in fact a generalization of the standard absolute pose problem and show that its solution is capable of solving the absolute pose problem.

3. An improved solution method compared to [20]. By using techniques from UPnP [11], we are able to achieve a much faster and more efficient solver than [20].

4. A novel incremental SfM pipeline that uses the distributed camera model to achieve large improvements in efficiency and scalability. This pipeline is capable of reconstructing Rome from over 15,000 images in just 22 minutes on a standard desktop computer.

2. Related Work

Since the seminal work of Phototourism [18], incremental SfM has been a popular pipeline for 3D reconstruction of unorganized photo collections. Robust open source solutions such as Bundler [18] and VisualSfM exist [24] using largely similar strategies to grow 3D models by successively adding one image at a time and carefully performing filtering and refinement. These pipelines have limited scalability but display excellent robustness and accuracy. In contrast, global SfM techniques [8, 10, 23] are able to compute all camera poses simultaneously, leading to extremely efficient solvers for large-scale problems; however, these methods lack robustness and are typically less accurate than incremental SfM. Alternatively, hierarchical SfM methods compute 3D reconstructions by first breaking up the input images into clusters that are individually reconstructed then merged together into a common 3D coordinate system [2]. Typically, bundle adjustment is run each time clusters are merged. This optimization ensures good accuracy and robustness but still an more scalable overall pipeline since fewer instances of bundle adjustment are performed compared to incremental SfM.

Computing camera pose is a fundamental step in 3D reconstruction. There is much recent work on solving for the camera pose of calibrated cameras [12, 3, 7, 13, 14]. Extending the single-camera absolute pose problem to multi-camera rigs is the Non-Perspective-*n*-Point (NP*n*P) problem. Minimal solutions to the NP3P problem were proposed by Níster and Stéwenius [15] and Chen and Chang [4] for generalized cameras. The UP*n*P algorithm [11] is an extension of the DLS P*n*P algorithm [9] that is capable of solving the single-camera or multi-camera absolute pose problem; however, it does not estimate scale and therefore cannot be used with distributed cameras.

To additionally recover scale transformations in multi-camera rigs, Ventura *et al*. [22] presented the first minimal solution to the generalized pose-and-scale problem. They use the generalized camera model and employ Gröbner basis computations to efficiently solve for scale, rotation, and translation using 4 2D-3D correspondences. Sweeney *et*

[1] While a generalized camera [16] may not explicitly include scale information, the camera model and accompanying pose methods [4, 11, 15] assume that the scale is known.

[2] The "+++" is commonly used in literature to denote a non-degenerate rotation parameterization [9]

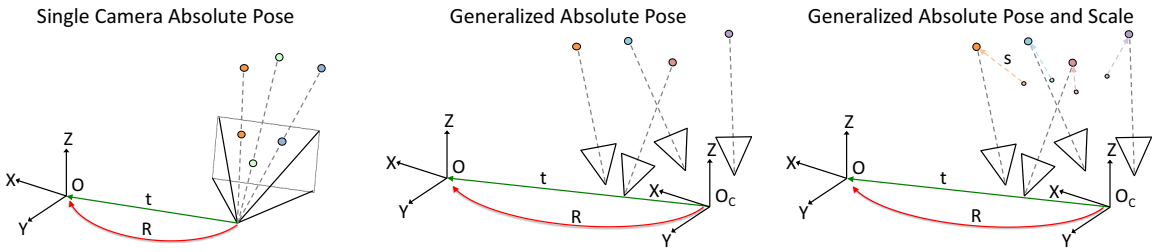

Figure 2. The absolute camera pose problem determines a camera's position and orientation with respect to a coordinate system with an origin O from correspondences between 2D image points and 3D points. In this paper, we show how solving the generalized absolute pose-and-scale (right) is a direct generalization of solving the single-camera (left) and multi-camera absolute pose problems.

al. [20] presented the first nonminimal solver, gDLS, for the generalized pose and scale problem. By extending the Direct Least Squares (DLS) P*n*P algorithm of Hesch and Roumeliotis [9], rotation, translation, and scale can be solved for in a fixed size polynomial system. While this method is very scalable the large elimination template of the gDLS method makes it very inefficient and the Cayley rotation parameterization results in singularities. In this paper, we extend the gDLS solution method to a non-singular representation and show that using the Gröbner basis technique achieves an $8\times$ speedup.

3. The Absolute Camera Pose Problem

In this section we review the absolute camera pose problem and demonstrate how to generalize the standard formulation to distributed cameras.

The fundamental problem of determining camera position and orientation given a set of correspondences between 2D image observations and known 3D points is called the *absolute camera pose problem* or Perspective n-Point (PnP) problem (*c.f*. Figure 2 left). In the minimal case, only three 2D-3D correspondences are required to compute the absolute camera pose [7, 9, 12]. These methods utilize the reprojection constraint such that 3D points X_i align with unit-norm pixel rays $\bar{x}_i$ when rotated and translated:

$$\alpha_i \bar{x}_i = RX_i + t, \qquad i = 1,2,3 \tag{1}$$

where R and t rotate and translate 3D points into the camera coordinate system. The scalar α_i stretches the unit-norm ray x_i such that $\alpha_i = ||RX_i + t||$. In order to determine the camera's pose, we would like to solve for the unknown rotation R and translation t that minimize the reprojection error:

$$C(R,t) = \sum_{i=1}^{3} ||\bar{x}_i - \frac{1}{\alpha_i}(RX_i + t)||^2 \tag{2}$$

The cost function $C(R,t)$ is the sum of squared reprojection errors and is the basis for solving the absolute camera pose problem.

3.1. Generalization to 7 d.o.f.

When information from multiple images is available, P*n*P methods are no longer suitable and few methods exist that are able to jointly utilize information from many cameras simultaneously. As illustrated in Figure 2 (center), multiple cameras (or multiple images from a single moving camera) can be described with the generalized camera model [16]. The generalized camera model represents a set of observations by the viewing ray origins and directions. For multi-camera systems, these values may be determined from the positions and orientations of cameras within the rig. The generalized camera model considers the viewing rays as static with respect to a common coordinate system (*c.f*. O_C in Figure 2 center, right). Using the generalized camera model, we may extend the reprojection constraint of Eq. (1) to multiple cameras:

$$c_i + \alpha_i \bar{x}_i = RX_i + t, \qquad i = 1,\ldots,n \tag{3}$$

where c_i is the origin of the feature ray $\bar{x}_i$ within the generalized camera model. This representation assumes that the scale of the generalized camera is equal to the scale of the 3D points (*e.g*., that both have metric scale). In general, the internal scale of each generalized camera is not guaranteed to be consistent with the scale of the 3D points. Consider a multi-camera system on a car that we want to localize to a point cloud created from Google Street View images. While we may know the metric scale of the car's multi-camera rig, it is unlikely we have accurate scale calibration for the Google Street View point cloud, and so we must recover the scale transformation between the rig and the point cloud in addition to the rotation and translation. When the scale is unknown then we have a distributed camera (*c.f*. Definition 1). This leads to the following reprojection constraint for distributed cameras that accounts for scale:

$$sc_i + \alpha_i \bar{x}_i = RX_i + t, \qquad i = 1,\ldots,n \tag{4}$$

where α_i is a scalar which stretches the image ray such that it meets the world point X_i such that $\alpha_i = ||RX_i + t - sc_i||$. Clearly the generalized absolute pose problem occurs when

$s = 1$ and the single-camera absolute pose problem occurs when $c_i = \mathbf{0}\ \forall i$.

By extending Eq (1) to multi-camera systems and scale transformations, we have generalized the PnP problem to the generalized pose-and-scale (gPnP+s) problem in Eq (4) shown in Figure 2 (right). The goal of the gPnP+s problem is to determine the pose and internal scale of a distributed camera with respect to n known 3D points. This is equivalent to aligning the two coordinate systems that define the distributed camera and the 3D points. Thus the solution to the gPnP+s problem is a 7 d.o.f. similarity transformation.

4. An L_2 Optimal Solution

To solve the generalized pose-and-scale problem we build on the method of Sweeney *et al*. [20], making modifications to the solver based on the UPnP method [11]. We extend the method of [20] with the following ideas from UPnP [11] to achieve a faster and more accurate solver:

1. Using the quaternion representation for rotations. This avoids the singularity of the Cayley-Gibbs-Rodrigues parameterization and improves accuracy.

2. Solve the system of polynomials using the Gröbner basis technique instead of the Macaulay Matrix method.

3. Take advantage of p-fold symmetry to reduce the size of the polynomial system [1].

We briefly review the solution method. When considering all n 2D-3D correspondences, there exists $8+n$ unknown variables (4 for quaternion rotation, 3 for translation, 1 for scale, and 1 unknown depth per observation). The gPnP+s problem can be formulated from Eq. (4) as a non-linear least-squares minimization of the measurement errors. Thus, we aim to minimize the cost function:

$$C(R,t,s) = \sum_{i=1}^{n} ||\bar{x_i} - \frac{1}{\alpha_i}(RX_i + t - sc_i)||^2. \tag{5}$$

The cost function shown in Eq. (5) can be minimized by a least-square solver. However, we can rewrite the problem in terms of fewer unknowns. Specifically, we can rewrite this equation solely in terms of the unknown rotation, R. When we relax the constraint that $\alpha_i = ||RX_i + t - sc_i||$ and treat each α_i as a free variable, α_i, s, and t appear linearly and can be easily reduced from Eq. (5). This relaxation is reasonable since solving the optimality conditions results in $\alpha_i^* = z_i^\top (RX_i + t - sc_i)$ where z_i is $\bar{x_i}$ corrupted by measurement noise.

We begin by rewriting our system of equations from Eq. (4) in matrix-vector form:

$$\underbrace{\begin{bmatrix} \bar{x_1} & & & c_1 & -I \\ & \ddots & & \vdots & \vdots \\ & & \bar{x_n} & c_n & -I \end{bmatrix}}_{\text{A}} \underbrace{\begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \\ s \\ t \end{bmatrix}}_{\text{x}} = \underbrace{\begin{bmatrix} R & & \\ & \ddots & \\ & & R \end{bmatrix}}_{\text{W}} \underbrace{\begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix}}_{\text{b}} \tag{6}$$

$$\Leftrightarrow Ax = Wb, \tag{7}$$

where A and b consist of known and observed values, x is the vector of unknown variables we will eliminate from the system of equations, and W is the block-diagonal matrix of the unknown rotation matrix. From Eq. (6), we can create a simple expression for x:

$$x = (A^\top A)^{-1} A^\top W b = \begin{bmatrix} U \\ S \\ V \end{bmatrix} W b. \tag{8}$$

We have partitioned $(A^\top A)^{-1} A^\top$ into constant matrices U, S, and V such that the depth, scale, and translation parameters are functions of U, S, and V respectively. Matrices U, S, and V can be efficiently computed in closed form by exploiting the sparse structure of the block matrices (see Appendix A from [20] for the full derivation). Note that α_i, s, and t may now be written concisely as linear functions of the rotation:

$$\alpha_i = u_i^\top W b \tag{9}$$

$$s = SWb \tag{10}$$

$$t = VWb, \tag{11}$$

where $u_i^\top$ is the i-th row of U. Through substitution, the geometric constraint equation (4) can be rewritten as:

$$\underbrace{SWb}_{s} c_i + \underbrace{u_i^\top W b}_{\alpha_i} \bar{x_i} = RX_i + \underbrace{VWb}_{t}. \tag{12}$$

This new constraint is quadratic in the four rotation unknowns given by the unit-norm quaternion representation.

4.1. A Least Squares Cost Function

The geometric constraint equation (12) assumes noise-free observations. We assume noisy observations $\bar{z_i} = \bar{x_i} + \eta_i$, where eta_i is zero mean noise. Eq. (12) may be rewritten in terms of our noisy observation:

$$SWbc_i + u_i^\top W b(\bar{z_i} - \eta_i) = RX_i + VWb \tag{13}$$

$$\Rightarrow \eta_i' = SWbc_i + u_i^\top W b \bar{z_i} - RX_i - VWb, \tag{14}$$

where η_i' is a zero-mean noise term that is a function of η_i (but whose covariance depends on the system parameters, as noted by Hesch and Roumeliotis [9]). We evaluate u_i, S, and V at $\bar{x}_i = \bar{z}_i$ without loss of generality. Observe that u_i can be eliminated from Eq. 14 by noting that:

$$UWb = \begin{bmatrix} \bar{z}_i^\top & & \\ & \ddots & \\ & & \bar{z}_n^\top \end{bmatrix} Wb - \begin{bmatrix} \bar{z}_1^\top c_1 \\ \vdots \\ \bar{z}_n^\top c_n \end{bmatrix} SWb + \begin{bmatrix} \bar{z}_1^\top \\ \vdots \\ \bar{z}_n^\top \end{bmatrix} VWb \tag{15}$$

$$\Rightarrow u_i^\top Wb = \bar{z}_i^\top RX_i - \bar{z}_i^\top c_i SWbc_i + \bar{z}_i^\top VWb. \tag{16}$$

Through substitution, Eq. (14) can be refactored such that:

$$\eta_i' = (\bar{z}_i\bar{z}_i^\top - I_3)(RX_i - SWbc_i + VWb). \tag{17}$$

Eq. (17) allows the gPnP+s problem to be formulated as an unconstrained least-squares minimization in 4 unknown rotation parameters. We formulate the least squares cost function, C', as the sum of the squared constraint errors from Eq. (17):

$$C'(R) = \sum_{i=1}^{n} ||(\bar{z}_i\bar{z}_i^\top - I_3)(RX_i - SWbc_i + VWb)||^2 \tag{18}$$

$$= \sum_{i=1}^{n} \eta_i'^\top \eta_i'. \tag{19}$$

Thus, the number of unknowns in the system has been reduced from $8+n$ to 4. This is an important part of the formulation, as it allows the size of the system to be independent of the number of observations and thus scalable. To enforce a solution with a valid rotation we must additionally enforce that the quaternion is unit-norm: $q^\top q = 1$.

4.2. Gröbner Basis Solution

An alternative method for solving polynomial systems is the Gröbner basis technique [13]. We created a Gröbner basis solver with an automatic generator similar to [13][3] while taking advantage of several additional forms of improvement. Following the solver of Kneip *et al.* [11], the size of the Gröbner basis is reduced by only choosing random values in $\mathbb{Z}_p$ that correspond to valid configurations for the generalized pose-and-scale problem. Next, double-roots are eliminated by exploiting the 2-fold symmetry technique used in [1, 11, 25]. This technique requires that all polynomials contain only even or only odd-degree monomials. The first order optimality constraints (formed from the partial derivitives of C') contain only uneven monomials; however, the unit-norm quaternion constraint contains even monomials. By modifying the unit-norm quaternion constraint to the squared norm:

$$(q^\top q - 1)^2 = 0 \tag{20}$$

[3]See [13] for more details about Gröbner basis techniques.

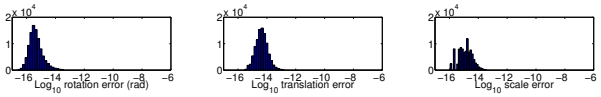

Figure 3. Histograms of numerical errors in the computed similarity transforms based on 10^5 random trials with the minimal 4 correspondences. Our algorithm is extremely stable, leading to high accuracy even in the presence of noise.

we obtain equations whose first order derivatives contain only odd monomials. Our final polynomial system is then:

$$\frac{\partial C'}{\partial q_i} = 0 \qquad i = 0,1,2,3 \tag{21}$$

$$(q^\top q - 1)q_i = 0 \qquad i = 0,1,2,3. \tag{22}$$

These eight third-order polynomials contain only uneven degree monomials, and so we can apply the 2-fold symmetry technique proposed by Ask *et al.* [1]. As with the UPnP method [11], applying these techniques to our Gröbner basis solver creates a 141×149 elimination template with an action matrix that is 8×8. Both the elimination template and the action matrix are dramatically smaller than with the Macaulay Matrix solution of [20], leading to a total execution time of just 151 μs.

5. Results

5.1. Numerical Stability

We tested the numerical stability of our solution over 10^5 random trials. We generated uniformly random camera configurations that placed cameras (*i.e.*, ray origins) in the cube $[-1,1] \times [-1,1] \times [-1,1]$ around the origin. The 3D points were randomly placed in the volume $[-1,1] \times [-1,1] \times [2,4]$. Ray directions were computed as unit vectors from camera origins to 3D points. An identity similarity transformation was used (*i.e.*, $R = I$, $t = 0$, $s = 1$). For each trial, we computed solutions using the minimal 4 correspondences. We calculated the angular rotation, translation, and scale errors for each trial, and plot the results in Figure 3. The errors are very stable, with 98% of all errors less than 10^{-12}.

5.2. Simulations With Noisy Synthetic Data

We performed two experiments with synthetic data to analyze the performance of our algorithm as the amount of image noise increases and as the number of correspondences increases. For both experiments we use the same configuration as the numerical stability experiments with six total 2D-3D observations. Using the known 2D-3D correspondences, we apply a similarity transformation with a random rotation in the range of $[-30,30]$ degrees about each of the x, y, and z axes, a random translation with a distance between 0.5 and 10, and a random scale change between 0.1

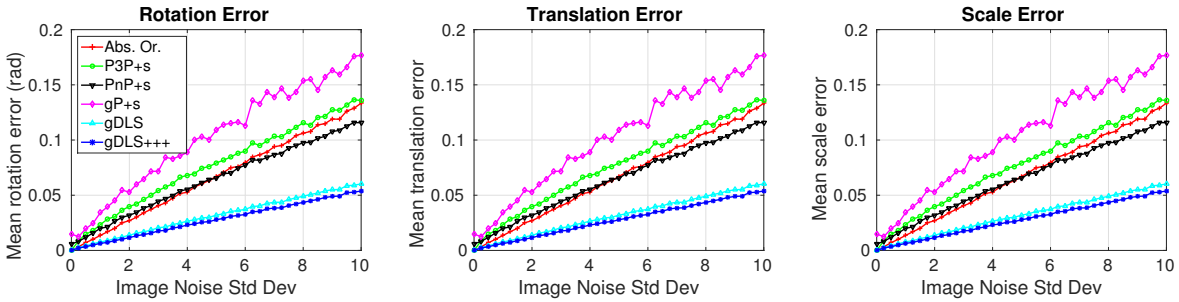

Figure 4. We compared similarity transform algorithms with increasing levels of image noise to measure the pose error performance: the absolute orientation algorithm of Umeyama [21], P3P+s, P*n*P+s, gP+s[22], and our algorithm, gDLS. Each algorithm was run with the same camera and point configuration for 1000 trials per noise level. Our algorithm has mean better rotation, translation, and scale errors for all levels of image noise.

and 10. We measure the performance of the following similarity transform algorithms:

- **Absolute Orientation:** The absolute orientation method of Umeyama [21] is used to align the known 3D points to 3D points triangulated from 2D correspondences. This algorithm is only an alignment method and does not utilize any 2D correspondences.

- **P3P+s, P*n*P+s:** First, the scale is estimated from the median scale of triangulated points in each set of cameras. Then, P3P *et al*. [12] or P*n*P [9] is used to determine the rotation and translation. This process is repeated for all cameras, and the camera localization and scale estimation that yields the largest number of inliers is used as the similarity transformation.

- **gP+s:** The minimal solver of Ventura *et al*. [22] is used with 2D-3D correspondences from all cameras. While the algorithm is intended for the minimal case of $n = 4$ correspondences, it can compute an overdetermined solution for $n \geq 4$ correspondences.

- **gDLS:** The algorithm presented in [20], which uses $n \geq 4$ 2D-3D correspondences from all cameras.

- **gDLS+++:** The algorithm presented in this paper, which is an extension of the gDLS algorithm [20]. This method uses $n \geq 4$ 2D-3D correspondences from all cameras.

After running each algorithm on the same testbed, we calculate the rotation, translation, and scale errors with respect to the known similarity transformation.

Image noise experiment: For our first experiment, we evaluated the similarity transformation algorithms under increased levels of image noise. Using the configuration described above, we increased the image noise from 0 to 10 pixels standard deviation, and ran 1000 trials at each level. Our algorithm outperforms each of the other similarity transformation algorithms for all levels of image noise, as shown in Figure 4. The fact that our algorithm returns all minima of our modified cost function is advantageous under high levels of noise, as we are not susceptible to getting stuck in a bad local minimum. This allows our algorithm to be very robust to image noise as compared to other algorithms.

Scalability experiment: For the second experiment, we evaluate the similarity transformation error as the number of 2D-3D correspondences increases. We use the same camera configuration described above, but vary the number of 3D points used to compute the similarity transformation from 4 to 1000. We ran 1000 trials for each number of correspondences used with a Gaussian noise level of 0.5 pixels standard deviation for all trials. We did not use the P3P+s algorithm for this experiment since P3P is a minimal solver and cannot utilize the additional correspondences. The accuracy of each similarity transformation algorithm as the number of correspondences increases is shown in Figure 5. Our algorithm performs very well as the number of correspondences increases, and is more accurate than alternative algorithms for all numbers of correspondences tested. Further, our algorithm is $O(n)$ so the performance cost of using additional correspondences is favorable compared to the alternative algorithms.

5.3. SLAM Registration With Real Images

We tested our new solver for registration of a SLAM reconstruction with respect to an existing SfM reconstruction using the indoor dataset from [22]. This dataset consists of an indoor reconstruction with precise 3D and camera position data obtained with an ART-2 optical tracker. All algorithms are used in a PROSAC [5] loop to estimate similarity transformations from 2D-3D correspondences. We compare these algorithms to our gDLS+++ algorithm.

We compute the average position error of all keyframes with respect to the ground truth data. The position errors, reported in centimeters, are shown in Table 1. Our gDLS+++ solver gives higher accuracy results for every image sequence tested compared to alternative algorithms. By

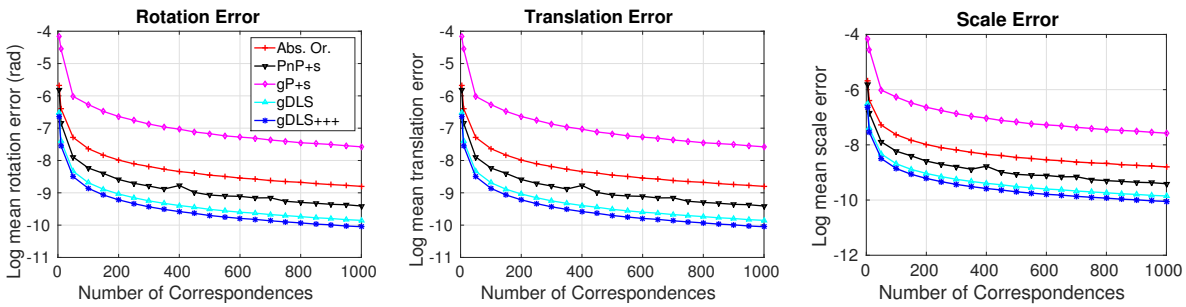

Figure 5. We measured the accuracy of similarity transformation estimations as the number of correspondences increased. The mean of the log rotation, translation, and scale errors are plotted from 1000 trials at each level of correspondences used. A Gaussian image noise of 0.5 pixels was used for all trials. We did not use P3P+s in this experiment because P3P only uses 3 correspondences. Our algorithm has better accuracy for all number of correspondences used and a runtime complexity of $O(n)$, making it ideal for use at scale.

Table 1. Average position error in centimeters for aligning a SLAM sequence to a pre-existing SfM reconstruction. An ART-2 tracker was used to provide highly accurate ground truth measurements for error analysis. Camera positions were computed using the respective similarity transformations and the mean camera position error of each sequence is listed below. Our method, gDLS+++, outperforms all other methods and is extremely close to the solution after BA.

Sequence	# Images	Abs. Ori. [21]	P3P+s	P*n*P+s	gP+s[22]	gDLS [20]	gDLS+++	After BA
office1	9	6.37	6.14	4.38	6.12	3.97	3.68	3.61
office2	9	8.09	7.81	6.90	9.32	5.89	5.59	5.57
office3	33	8.29	9.31	8.89	6.78	6.08	4.91	4.86
office4	9	4.76	4.48	3.98	4.00	3.81	3.09	3.04
office5	15	3.63	3.42	3.39	4.75	3.39	3.17	3.14
office6	24	5.15	5.23	5.01	5.91	4.51	4.35	4.31
office7	9	6.33	7.08	7.16	7.07	4.65	2.99	2.72
office8	11	4.72	4.85	3.62	4.59	2.85	2.30	2.12
office9	7	8.41	8.44	4.08	6.65	3.19	2.25	2.25
office10	23	5.88	6.60	5.73	5.88	4.94	4.68	4.61
office11	58	5.19	4.85	4.80	6.74	4.77	4.66	4.57
office12	67	5.53	5.20	4.97	4.86	4.81	4.45	4.44

using the generalized camera model, we are able to exploit 2D-3D constraints from multiple cameras at the same time as opposed to considering only one camera (such as P3P+s and P*n*P+s). This allows the similarity transformation to be optimized for all cameras and observations simultaneously, leading to high-accuracy results.

We additionally show the results of gDLS+++ with bundle adjustment applied after estimation (labeled "After BA" in Table 1). In all datasets, our results are very close to the optimal results after bundle adjustment, and typically bundle adjustment converges after only one or two iterations. This indicates that the gDLS+++ algorithm is very close to the geometrically optimal solution in terms of reprojection error. Further, our singularity-free rotation parameterization prevents numerical instabilities that arise as the computed rotation approaches a singularity, leading to more accurate results than the gDLS [20] algorithm.

6. Incremental SfM with Distributed Cameras

We demonstrate the viability of our method for SfM model-merging in a novel incremental SfM pipeline. Our pipeline uses distributed cameras to rapidly grow the model by adding many cameras at once. We use the gDLS+++ method to localize a distributed camera to our current model in the same way that traditional incremental pipelines use P3P to localize individual cameras. This allows our pipeline to be extremely scalable and efficient because we can accurately localize many cameras to our model in a single step. Note that a 3D reconstruction of points and cameras may be alternatively described as a distributed camera where ray origins are the camera origins and the ray directions (and depths) correspond to observations of 3D points. In our incremental SfM procedure we treat reconstructions as distributed cameras, allowing reconstructions to be merged efficiently and accurately with the gDLS+++ algorithm.

We begin by partitioning the input dataset into subsets using normalized graph cuts [17] similar to Bhowmick *et al*. [2]. The size of the subsets depends on the size of the

Table 2. Results of our Hierarchical SfM pipeline on several large scale datasets. Our method is extremely efficient and is able to reconstruct more cameras than the Divide-and-Conquer method of Bhowmick *et al*. [2]. Time is provided in minutes.

Dataset	N_{cam}	Visual SfM [24]		DISCO [6]		Bhowmick *et al*. [2]		Ours		
		N_{cam}	Time	N_{cam}	Time	N_{cam}	Time	N_{cam}	BA Its	Time
Colosseum	1164	1157	9.85	N/A	N/A	1032	3	1097	1	2.6
St Peter's Basilica	1275	1267	9.71	N/A	N/A	1236	4	1256	1	3.7
Dubrovnik	6845	6781	16.8	6532	275	N/A	N/A	6677	2	8.9
Rome	15242	15065	100.17	14754	792	10534	27	12329	2	22

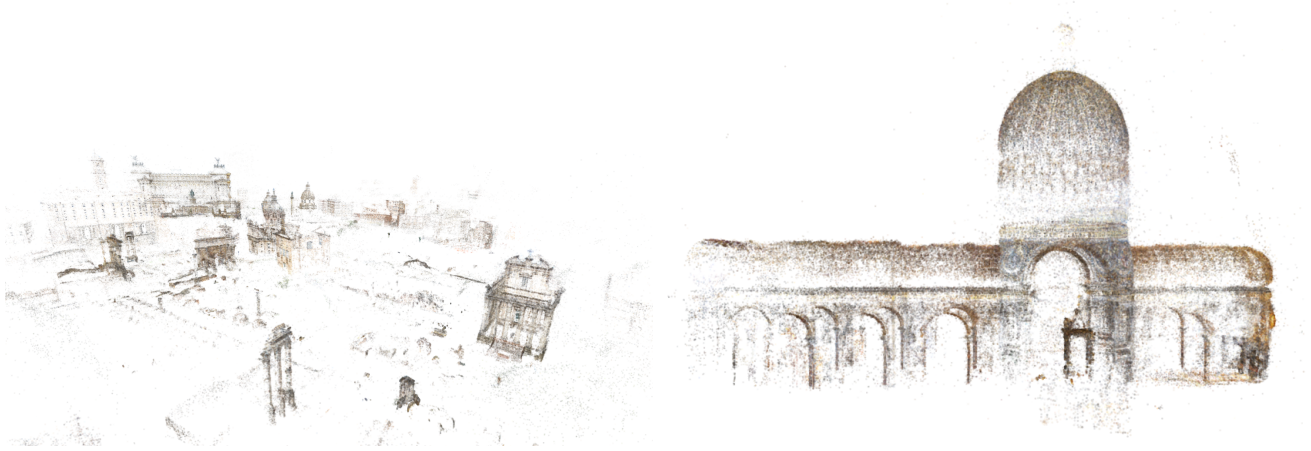

Figure 6. Final reconstructions of the Central Rome (left) and St Peters (right) datasets computed with our hierarchical SfM pipeline. Our pipeline produces high quality visual results at state-of-the-art efficiency (*c.f*. Table 2).

particular dataset, but typically the subsets contain between 50 and 250 cameras. Each of the subsets is individually reconstructed in parallel with the "standard" incremental SfM pipeline of the Theia SfM library [19]. Each reconstructed subset may be viewed as a distributed camera. The remaining task is then to localize all distributed cameras into a common coordinate system in the same way that traditional incremental SfM localizes individual cameras as it grows the reconstruction.

The localization step operates in a similar manner to the first step. Distributed cameras are split into subsets with normalized graph cuts [17]. Within each subset of distributed cameras, the distributed camera with the most 3D points is chosen as the "base" and all other distributed cameras are localized to the "base" distributed camera with the gDLS+++ algorithm. Note that when two distributed cameras are merged (with gDLS or another algorithm) the result is another distributed camera which contains all observation rays from the distributed cameras that were merged. As such, each merged subset forms a new distributed camera that contains contains all imagery and geometric information of the cameras in that subset. This process is repeated until only a single distributed camera remains (or no more distributed cameras can be localized). We do not run bundle adjustment as subsets are merged and only run a single global bundle adjustment on the entire reconstruction as a final refinement step. Avoiding the use of costly bundle adjustment is a driving reason for why our method is very efficient and scalable.

We compare our SfM pipeline to Incremental SfM (VisualSfM [24]), the DISCO pipeline of Crandall *et al*. [6], and the hierarchical SfM pipeline of Bhowmick *et al*. [2] run on several large-scale SfM datasets and show the results in Table 2. All methods were run on a desktop machine with a 3.4GHz i7 processor and 16GB of RAM. Our method is the most efficient on all datasets, though we typically reconstruct fewer cameras than [24]. Using gDLS+++ for model-merging allows our method produces high quality models by avoiding repeated use of bundle adjustment. As shown in Table 2, the final bundle adjustment for our pipeline requires no more than 2 iterations, indicating that the gDLS+++ method is able to merge reconstructions extremely accurately. We show the high quality visual results of our SfM pipeline in Figure 6.

7. Conclusion

In this paper, we introduced a new camera model for SfM: the distributed camera model. This model describes image observations in terms of light rays with ray origins and directions rather than pixels. As such, the distributed camera model is able to describe a single camera or multiple cameras in a unified expression as the collection of all light rays observed. We showed how the gDLS method [20] is in fact a generalization of the standard absolute pose problem

and derive an improved solution method, gDLS+++, that is able to solve the absolute pose problem for standard or distributed cameras. Finally, we showed how gDLS+++ can be used in a scalable incremental SfM pipeline that uses the distributed camera model to accurately localize many cameras to a reconstruction in a single step. As a result, fewer bundle adjustments are performed and the resulting efficient pipeline is able to reconstruct a 3D model of Rome from more than 15,000 images in just 22 minutes. We believe that the distributed camera model is a useful way to parameterize cameras and can provide great benefits for SfM. For future work, we plan to explore the use of distributed cameras for global SfM, as well as structure-less SfM by merging distributed cameras from 2D-2D ray correspondences without the need for 3D points.

References

[1] E. Ask, K. Yubin, and K. Astrom. Exploiting p-fold symmetries for faster polynomial equation solving. In *Proc. of the International Conference on Pattern Recognition (ICPR)*. IEEE, 2012. 4, 5

[2] B. Bhowmick, S. Patra, A. Chatterjee, V. Govindu, and S. Banerjee. Divide and conquer: Efficient large-scale structure from motion using graph partitioning. In *The Asian Conference on Computer Vision*, pages 273–287. Springer International Publishing, 2015. 2, 7, 8

[3] M. Bujnak, Z. Kukelova, and T. Pajdla. A general solution to the p4p problem for camera with unknown focal length. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. 2

[4] C.-S. Chen and W.-Y. Chang. On pose recovery for generalized visual sensors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7):848–861, 2004. 2

[5] O. Chum and J. Matas. Matching with prosac-progressive sample consensus. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 220–226. IEEE, 2005. 6

[6] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. SfM with MRFs: Discrete-continuous optimization for large-scale structure from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 35(12):2841–2853, December 2013. 8

[7] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2, 3

[8] V. M. Govindu. Combining two-view constraints for motion estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–218. IEEE, 2001. 2

[9] J. Hesch and S. Roumeliotis. A direct least-squares (dls) solution for pnp. In *Proc. of the International Conference on Computer Vision*. IEEE, 2011. 2, 3, 5, 6

[10] N. Jiang, Z. Cui, and P. Tan. A global linear method for camera pose registration. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 481–488. IEEE, 2013. 2

[11] L. Kneip, H. Li, and Y. Seo. Upnp: An optimal o (n) solution to the absolute pose problem with universal applicability. In *The European Conference on Computer Vision (ECCV)*, pages 127–142. Springer International Publishing, 2014. 2, 4, 5

[12] L. Kneip, D. Scaramuzza, and R. Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 2969–2976. IEEE, 2011. 2, 3, 6

[13] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. In *European Conference on Computer Vision*, pages 302–315. Springer, 2008. 2, 5

[14] Z. Kukelova, M. Bujnak, and T. Pajdla. Polynomial eigenvalue solutions to minimal problems in computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1381–1393, 2012. 2

[15] D. Nistér and H. Stewénius. A minimal solution to the generalised 3-point pose problem. *Journal of Mathematical Imaging and Vision*, 27(1):67–79, 2007. 2

[16] R. Pless. Using many cameras as one. In *Proc. IEEE Conference on Conference on Computer Vision and Pattern Recognition*, volume 2, pages II–587. IEEE, 2003. 2, 3

[17] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(8):888–905, 2000. 7, 8

[18] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM transactions on graphics (TOG)*, volume 25, pages 835–846. ACM, 2006. 1, 2

[19] C. Sweeney. Theia multiview geometry library: Tutorial & reference. http://theia-sfm.org. 8

[20] C. Sweeney, V. Fragoso, T. Höllerer, and M. Turk. gdls: A scalable solution to the generalized pose and scale problem. In *The European Conference on Computer Vision (ECCV)*, pages 16–31. Springer International Publishing, 2014. 1, 2, 4, 5, 6, 7, 8

[21] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991. 6, 7

[22] J. Ventura, C. Arth, G. Reitmayr, and D. Schmalstieg. A minimal solution to the generalized pose-and-scale problem. *Accepted to: IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 2, 6, 7

[23] K. Wilson and N. Snavely. Robust global translations with 1dsfm. In *Proceedings of the European Conference on Computer Vision*, pages 61–75. Springer, 2014. 2

[24] C. Wu. Towards linear-time incremental structure from motion. In *The International Conference on 3D Vision-3DV*, pages 127–134. IEEE, 2013. 2, 8

[25] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi. Revisiting the pnp problem: A fast, general and optimal solution. In *Proc. of the International Conference on Computer Vision*. IEEE, Dec 2013. 5