Structure from Motion on a Sphere

Jonathan Ventura(✉)

Department of Computer Science,
University of Colorado Colorado Springs, Colorado Springs, USA
jventura@uccs.edu

Abstract. We describe a special case of structure from motion where the camera rotates on a sphere. The camera's optical axis lies perpendicular to the sphere's surface. In this case, the camera's pose is minimally represented by three rotation parameters. From analysis of the epipolar geometry we derive a novel and efficient solution for the essential matrix relating two images, requiring only three point correspondences in the minimal case. We apply this solver in a structure-from-motion pipeline that aggregates pairwise relations by rotation averaging followed by bundle adjustment with an inverse depth parameterization. Our methods enable scene modeling with an outward-facing camera and object scanning with an inward-facing camera.

1 Introduction

Accurate visual 3D reconstruction is highly dependent on establishing sufficient baseline between images so that the translation between them can be reliably estimated and 3D points can be accurately triangulated. However, we have found that, in practice, it is difficult for an untrained user to capture image sequences with sufficient baseline; typically, the natural inclination is to rotate the camera instead of translating it, which causes the structure-from-motion system to fail.

In this work, we instead specifically target camera rotation as the basis for structure from motion. The critical assumption here is that the camera rotates at some fixed distance from the origin, with its optical axis aligned with the ray between the origin and the camera center. We call this "spherical motion."

The camera could be pointing inward or outward. An example of an inward-facing camera would be object scanning setups such as a turntable or spherical gantry. An example of an outward-facing camera would be a typical user capturing a panorama – the user holds the camera away from their body at a fixed distance while rotating.

In either case, the global scale of the 3d reconstruction is unknown, as is always the case in pure monocular structure-from-motion. The global scale is determined by the radius of the sphere, which we arbitrarily set to unit length. However, what is interesting about this particular case of camera motion is that the relative scale between camera pairs is known, because the radius of the sphere is fixed. This is a distinct advantage over general monocular camera motion estimation, where the relative scale of the translation between camera

B. Leibe et al. (Eds.): ECCV 2016, Part III, LNCS 9907, pp. 53–68, 2016.
DOI: 10.1007/978-3-319-46487-9_4

pairs must be determined by point triangulation and scale propagation, which is highly susceptible to scale drift. With spherical camera motion, we can directly compose relative pose estimates to determine the complete camera trajectory without needing to propagate scale. A second advantage is that the relative pose between cameras is fully determined by three rotational degrees of freedom and so can be estimated from three point correspondences as opposed to the five correspondences needed in the general case.

In this paper, after a survey of related work (Sect. 2), we analyze the geometry of spherical camera motion (Sect. 3) and derive efficient solvers for the essential matrix (Sect. 4). We integrate these solvers into a complete structure-from-motion pipeline (Sect. 5) and present an evaluation of our methods on synthetic and real data (Sect. 6) followed by conclusions and future work (Sect. 7).

2 Related Work

A particular problem of interest in geometric computer vision is inferring the essential matrix relating two images from point correspondences, especially from a minimal set of correspondences [14]. Minimal solutions are useful for application in a random sample consensus (RANSAC) [6] loop to robustly estimate the motion parameters and separate inliers from outliers. Nistér [17] derived an efficient minimal solution from five point correspondences and Stewénius et al. [26] later improved the accuracy of this method. In this work, we derive a solution for the essential matrix from at least three correspondences which applies when the camera undergoes spherical motion.

Several previous works have considered solutions for monocular relative pose given circular motion or single-axis rotation as observed with a turntable [7,11,16] or a non-holonomic vehicle [23]. In this work we derive a spherical motion solver which allows three rotational degrees of freedom and thus requires three point correspondences in the minimal case.

Also closely related are the works by Peleg and Ben-Ezra [19] and Shum and Szeliski [24] on stereo or multi-perspective panoramas. In these works, an outward-facing camera is spun on a circular path and images are captured at regular intervals. They demonstrated that by careful sampling of the images, stereo cylindrical panoramas can be created and used for either surround-view stereo viewing or 3D stereo reconstruction. These works use either controlled capture on a turntable [24] or manifold mosaicing [20] to obtain the positions of the images in the sequence, whereas we develop an automatic, accurate structure-from-motion pipeline which applies to both circular and spherical motion image sequences.

Structure-from-motion refers to simultaneously estimating camera poses and 3D scene geometry from an image sequence or collection. For example, Pollefeys et al. describe a pipeline for visual modeling with a handheld camera [21], and Snavely et al. developed a structure-from-motion system from unstructured internet image collections [25]. Most related to the present work is the 1DSfM approach of Wilson and Snavely [28], where the camera orientations are first estimated using a robust global rotation averaging approach [2] and then the camera translations are estimated separately. In our approach, the camera's position is directly determined by its orientation, so the second translation estimation step is not needed.

3 The Geometry of Spherical Camera Motion

In this section we give expressions for the absolute and relative pose matrices induced by spherical motion and derive the form of the essential matrix.

3.1 Camera Extrinsics

Inward-Facing Camera. For an inward-facing camera, the 3×4 camera extrinsics matrix P can be expressed using a 3×3 rotation matrix R and a 3×1 vector $\mathbf{z}$:

$$\mathsf{P}_{\text{in}} = [\mathsf{R} \mid \mathbf{z}], \tag{1}$$

where $\mathbf{z} = [0\ 0\ 1]^\top$.

Outward-Facing Camera. For an outward-facing camera, the translation direction is reversed:

$$\mathsf{P}_{\text{out}} = [\mathsf{R} \mid -\mathbf{z}]. \tag{2}$$

3.2 Relative Pose

Given two inward-facing cameras with extrinsics $\mathsf{P}_1 = [\mathsf{R}_1 \mid \mathbf{z}]$ and $\mathsf{P}_2 = [\mathsf{R}_2 \mid \mathbf{z}]$, we can now derive the relative pose $[\mathsf{R} \mid \mathbf{t}_{\text{in}}]$ between them. The relative rotation is

$$\mathsf{R} = \mathsf{R}_2\mathsf{R}_1^\top. \tag{3}$$

and the relative translation is

$$\mathbf{t}_{\text{in}} = \mathbf{z} - \mathbf{r}_3 \tag{4}$$

where $\mathbf{r}_3$ denotes the third column of R.

For outward-facing cameras with relative pose $[\mathsf{R} \mid \mathbf{t}_{\text{out}}]$, the rotation is the same and the translation direction is reversed:

$$\mathbf{t}_{\text{out}} = \mathbf{r}_3 - \mathbf{z}. \tag{5}$$

3.3 Essential Matrix

The essential matrix E relates corresponding camera normalized (i.e. calibrated) homogeneous points $\mathbf{u}$ and $\mathbf{v}$ in two images such that

$$\mathbf{v}^\top\mathsf{E}\mathbf{u} = 0. \tag{6}$$

If the two images have relative pose $[\mathsf{R}|\mathbf{t}]$ then

$$\mathsf{E} = [\mathbf{t}]_\times \mathsf{R}, \tag{7}$$

where $[\mathbf{a}]_\times$ is the skew-symmetric matrix such that $[\mathbf{a}]_\times \mathbf{b} = \mathbf{a} \times \mathbf{b} \; \forall \; \mathbf{b}$.

Plugging in the relative pose expressions given above, the essential matrices for inward- and outward-facing cameras are

$$\mathsf{E}_{\text{in}} = [\mathbf{z} - \mathbf{r}_3]_\times \mathsf{R} \tag{8}$$

and

$$\mathsf{E}_{\text{out}} = [\mathbf{r}_3 - \mathbf{z}]_\times \mathsf{R}. \tag{9}$$

Note that $\mathsf{E}_{\text{in}} = -\mathsf{E}_{\text{out}}$. Since the essential matrix is only defined up to scale, the essential matrix for inward- and outward-facing cameras underoing the same relative rotation is equivalent.

4 Solving for the Essential Matrix

Here we characterize the essential matrix relating cameras undergoing spherical motion and derive a solution for all possible essential matrices arising from at least three correspondences between two images.

In general, the essential matrix has five degrees of freedom [9], corresponding to three rotational degrees of freedom and two translational, since in the general case the translation is only defined up to scale. However, the special form of essential matrix for spherical motion derived above is determined completely by three rotational degrees of freedom. This implies that we can solve for the essential matrix using only three correspondences, as opposed to the five correspondences needed in the general case [17].

The essential matrix for spherical motion has a special form and can be fully described by six parameters $e_1, \ldots, e_6$:

$$\mathsf{E} = \begin{bmatrix} e_1 & e_2 & e_3 \\ e_2 & -e_1 & e_4 \\ e_5 & e_6 & 0 \end{bmatrix} \tag{10}$$

This can be derived using the fact that, since R is orthonormal, each column can be expressed as a cross product of the other two.

4.1 Finding the Nullspace

Given $n \geq 3$ corresponding image points $\mathbf{u}_1, \ldots, \mathbf{u}_n$ and $\mathbf{v}_1, \ldots, \mathbf{v}_n$, we have n epipolar constraint equations of the form

$$\mathbf{v}_i^\top \mathsf{E} \mathbf{u}_i = 0. \tag{11}$$

We re-arrange and stack the epipolar constraints to form a linear system on the parameters:

$$\begin{bmatrix} u_{11}v_{11} - u_{12}v_{12} & u_{11}v_{12} + u_{12}v_{11} & u_{13}v_{11} & u_{13}v_{12} & u_{11}v_{13} & u_{12}v_{13} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_{n1}v_{n1} - u_{n2}v_{n2} & u_{n1}v_{n2} + u_{n2}v_{n1} & u_{n3}v_{n1} & u_{n3}v_{n2} & u_{n1}v_{n3} & u_{n2}v_{n3} \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \end{bmatrix} = \mathbf{0} \tag{12}$$

where u_{ij} denotes the j-th element of $\mathbf{u}_i$.

We now find three 6×1 vectors $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ spanning the right nullspace of the $n \times 6$ matrix on the left-hand side of Eq. 12. The essential matrix must be of the form

$$\mathsf{E} = \begin{bmatrix} b_{11}x + b_{21}y + b_{31}z & b_{12}x + b_{22}y + b_{32}z & b_{13}x + b_{23}y + b_{33}z \\ b_{12}x + b_{22}y + b_{32}z & -b_{11}x - b_{21}y - b_{31}z & b_{14}x + b_{24}y + b_{34}z \\ b_{15}x + b_{25}y + b_{35}z & b_{16}x + b_{26}y + b_{36}z & 0 \end{bmatrix} \tag{13}$$

for some scalars x, y, z. Here b_{ij} denotes the j-th element of vector b_i.

Any choice of scalars x, y, z will produce a solution for E which satisfies the epipolar constraints. However, a second requirement is that the matrix must satisfy the properties of an essential matrix; namely, that it is rank two and that both non-zero singular values are equal [5]. These properties lead to non-linear constraints which are solved in the following subsection.

4.2 Applying Non-linear Constraints

The requirements on the singular values of the essential matrix are enforced by the following cubic constraints [5]:

$$\mathsf{E}\mathsf{E}^\top\mathsf{E} - \frac{1}{2}\text{trace}(\mathsf{E}\mathsf{E}^\top)\mathsf{E} = 0. \tag{14}$$

This 3×3 matrix equation gives a system of nine cubic constraints in x, y, z. Since the essential matrix is only determined up to scale, we let $z = 1$.

Using a symbolic math toolbox, we found that this system has rank six, and that the second and third rows of the system form a linearly independent set of six equations.

We separate these six equations into a 6×10 matrix A of coefficients and a vector $\mathbf{m}$ of 10 monomials such that

$$\mathsf{A}\mathbf{m} = \mathbf{0} \tag{15}$$

where

$$\mathbf{m} = \begin{bmatrix} x^3 & x^2y & xy^2 & y^3 & x^2 & xy & y^2 & x & y & 1 \end{bmatrix}^\top. \tag{16}$$

4.3 Solution Using the Action Matrix Method

The action matrix method has been established as a general tool to solve systems of polynomial equations arising from geometric computer vision problems [13]. Briefly, once we have found a Gröbner basis [3,4] for the system of polynomial equations, we can derive a transformation from the coefficient matrix to an action matrix whose eigenvalues and eigenvectors contain the solutions.

Using the Macaulay2 algebraic geometry software system, we determined that Eq. 15 has at most four solutions. By ordering the monomials in $\mathbf{m}$ using graded reverse lexicographic ordering and running Gauss-Jordan elimination on A, we immediately arrive at a Gröbner basis for the ideal I generated by the six polynomial equations, since this leaves only four monomials that are not divisible by any of the leading monomials in the equations. These monomials form a basis for the quotient ring $\mathbb{C}[x, y]/I$ and are the same basis monomials reported by Macaulay2.

Let G be the 6×4 matrix such that $\left[\mathsf{I}_6 \ \mathsf{G}\right]$ is the result of running Gauss-Jordan elimination on A. Now we have

$$\left[\mathsf{I}_6 \ \mathsf{G}\right] \mathbf{m} = \mathbf{0} \tag{17}$$

which implies that

$$x^3 + G_{11}y^2 + G_{12}x + G_{13}y + G_{14} = 0 \tag{18}$$
$$x^2y + G_{21}y^2 + G_{22}x + G_{23}y + G_{24} = 0 \tag{19}$$
$$xy^2 + G_{31}y^2 + G_{32}x + G_{33}y + G_{34} = 0 \tag{20}$$
$$y^3 + G_{41}y^2 + G_{42}x + G_{43}y + G_{44} = 0 \tag{21}$$
$$x^2 + G_{51}y^2 + G_{52}x + G_{53}y + G_{54} = 0 \tag{22}$$
$$xy + G_{61}y^2 + G_{62}x + G_{63}y + G_{64} = 0. \tag{23}$$

Using Eqs. 20, 22 and 23 we can define a 4×4 matrix A_x as

$$\mathsf{A}_x = \begin{bmatrix} -G_{31} & -G_{32} & -G_{33} & -G_{34} \\ -G_{51} & -G_{52} & -G_{53} & -G_{54} \\ -G_{61} & -G_{62} & -G_{63} & -G_{64} \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{24}$$

so that

$$\mathsf{A}_x \begin{bmatrix} y^2 \\ x \\ y \\ 1 \end{bmatrix} = x \begin{bmatrix} y^2 \\ x \\ y \\ 1 \end{bmatrix}. \tag{25}$$

Thus the eigenvalues of A_x are solutions for x, and the eigenvectors contain corresponding solutions for y. A_x is the "action matrix" for x and $y^2, x, y, 1$ are the basis monomials.

Once we have found up to four real-valued solutions for x and y by eigen-decomposition of A_x, we apply them in Eq. 13 to produce four solutions for the essential matrix E.

4.4 Solution by Reduction to Single Polynomial

A possibly faster method to find solutions for x and y would be to use the characteristic polynomial of A_x to find its eigenvalues:

$$|\mathsf{A}_x - x\mathsf{I}_3| = 0. \tag{26}$$

This involves computing the determinant of a 4×4 matrix of polynomials in y. We found that a slight speedup is possible by transforming the problem to instead use a 3×3 symbolic determinant.

First, we define $\mathbf{m}'$ which is a reordering the monomials in $\mathbf{m}$:

$$\mathbf{m}' = \left[x^3 \; x^2y \; xy^2 \; y^3 \; y^2 \; y \; x^2 \; xy \; x \; 1\right]^\top. \tag{27}$$

The system of equations $\mathsf{A}\mathbf{m} = \mathbf{0}$ from Eq. 15 is rewritten using this new ordering. We form a reordered matrix of coefficients A' such that

$$\mathsf{A}'\mathbf{m}' = \mathbf{0}. \tag{28}$$

Let G' be the 6×4 matrix such that $\left[\mathsf{I}_6 \; \mathsf{G}'\right]$ is the result of running Gauss-Jordan elimination on A'. Now we have

$$\left[\mathsf{I}_6 \; \mathsf{G}'\right]\mathbf{m}' = \mathbf{0} \tag{29}$$

which implies that

$$x^3 + G'_{11}x^2 + G'_{12}xy + G'_{13}x + G'_{14} = 0 \tag{30}$$
$$x^2y + G'_{21}x^2 + G'_{22}xy + G'_{23}x + G'_{24} = 0 \tag{31}$$
$$xy^2 + G'_{31}x^2 + G'_{32}xy + G'_{33}x + G'_{34} = 0 \tag{32}$$
$$y^3 + G'_{41}x^2 + G'_{42}xy + G'_{43}x + G'_{44} = 0 \tag{33}$$
$$y^2 + G'_{51}x^2 + G'_{52}xy + G'_{53}x + G'_{54} = 0 \tag{34}$$
$$y + G'_{61}x^2 + G'_{62}xy + G'_{63}x + G'_{64} = 0. \tag{35}$$

Using Eqs. 33, 34 and 35 we can define a 3×3 matrix $\mathsf{B}(y)$ as

$$\mathsf{B}(y) = \begin{bmatrix} G'_{41} & G'_{43} + G'_{42}y & G'_{44} + y^3 \\ G'_{51} & G'_{53} + G'_{52}y & G'_{54} + y^2 \\ G'_{61} & G'_{63} + G'_{62}y & G'_{64} + y \end{bmatrix} \tag{36}$$

so that

$$\mathsf{B}(y)\begin{bmatrix} x^2 \\ x \\ 1 \end{bmatrix} = \mathbf{0}. \tag{37}$$

Because $\mathsf{B}(y)$ has a null vector, its determinant must be equal to zero, leading to a quartic polynomial $\langle n \rangle$ in y:

$$\langle n \rangle \equiv |\mathsf{B}(y)| = 0. \tag{38}$$

The quartic polynomial $\langle n \rangle$ can be solved in closed-form using Ferrari's method. Once we have four solutions for y, the corresponding solutions for x are found by finding a null vector of $\mathsf{B}(y)$. Then the solutions for x and y are used to produce solutions for the essential matrix using Eq. 13.

4.5 Decomposition of the Essential Matrix

Once we have a solution for the essential matrix, we need to decompose it into a rotation and translation to find the relative pose. The decomposition follows the normal procedure for extracting a "twisted pair" of solutions [9], giving two solutions for the rotation, R_a and R_b, and one solution for the translation direction $\hat{\mathbf{t}}$ which is only determined up to scale.

Let $\mathsf{E} \sim \mathsf{USV}^\top$ be the singular value decomposition of E where U and V are chosen such that $|\mathsf{U}| > 0$ and $|\mathsf{V}| > 0$. Define matrix D as

$$\mathsf{D} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{39}$$

Then $\mathsf{R}_a = \mathsf{UDV}^\top$ and $\mathsf{R}_b = \mathsf{UD}^\top\mathsf{V}^\top$. The solution for the translation direction is $\hat{\mathbf{t}} = [U_{13}\ U_{23}\ U_{33}]^\top$.

Only one of the rotations is consistent with spherical motion. Let $\mathbf{t}_a$ and $\mathbf{t}_b$ be corresponding translation vectors for rotation solutions R_a and R_b, respectively, determined by Eq. 4 if the cameras are inward-facing or Eq. 5 if the cameras are outward-facing. We can choose the correct relative pose solution by choosing the rotation whose corresponding translation is closest to the translation solution $\mathbf{t}$.

Specifically, we define scores s_a and s_b according to the absolute value of the normalized dot product between $\mathbf{t}_a$ or $\mathbf{t}_b$ and $\hat{\mathbf{t}}$:

$$s_a = \frac{|\mathbf{t}_a \cdot \hat{\mathbf{t}}|}{||\mathbf{t}_a||}, s_b = \frac{|\mathbf{t}_b \cdot \hat{\mathbf{t}}|}{||\mathbf{t}_b||}. \tag{40}$$

The solution with higher score is chosen as the correct relative pose:

$$[\mathsf{R}|\ \mathbf{t}] = \begin{cases} [\mathsf{R}_a|\ \mathbf{t}_a] & \text{if } s_a > s_b, \\ [\mathsf{R}_b|\ \mathbf{t}_b] & \text{otherwise.} \end{cases} \tag{41}$$

5 An Integrated Structure from Motion Pipeline

The algorithms described in Sect. 4 enable us to solve for the relative pose between two cameras from a minimal or overdetermined set of image correspondences. The minimal solver is useful for random sample consensus (RANSAC) [6] where we compute relative pose hypotheses from randomly sampled minimal sets of correspondences and accept the hypothesis with the highest number of inliers.

In this section we describe an integrated structure-from-motion pipeline which uses our novel solvers to recover the camera trajectory from a spherical motion image sequence by aggregating pairwise relationships and produce a 3D point cloud reconstruction of the scene.

The input to the pipeline is a sequence of images captured by an inward- or outward-facing camera undergoing spherical motion. We assume the camera is pre-calibrated so that its intrinsic parameters including focal length, principal point, and radial and tangential distortion coefficients are known.

5.1 Feature Tracking and Relative Pose Estimation

We first use feature detection and tracking to establish image correspondences between neighboring images in the sequence. Between each successive pair of images in the sequence, we apply one of our spherical motion solvers from Sect. 4 in a Preemptive RANSAC loop [18] in order to robustly estimate the essential matrix and find a consensus set of inliers. To test inliers we threshold the Sampson error [9] between the epipolar line and the image point in the second image. Outlier feature tracks are removed from consideration in successive frames.

Either minimal solver gives at most four solutions for the essential matrix from three correspondences. We choose the essential matrix with lowest error on a fourth randomly sampled correspondence. Finally, the decomposition of the essential matrix gives two possible rotations R_a and R_b which are disambiguated using the score function described in Sect. 4.5.

5.2 Loop Closure

After processing the images in sequence, we detect loop closures by matching features between non-neighboring images. Each feature in image i is matched to its nearest neighbor in image j. The set of putative matches between images i and j are then filtered using Preemptive RANSAC with one of our minimal solvers, and the resulting relative pose is recorded if the number of inliers exceeds a threshold.

5.3 Global Pose Estimation by Rotation Averaging

At this point, we have a set of estimated rotations R_{ij} between images i and j in the sequence. Now we can apply rotation averaging [8] to produce a global estimate of all camera orientations. Specifically, we use the robust L1 rotation averaging method of Chatterjee and Madhav Govindu [2]. Since the translation of each camera is fully determined by the camera's rotation, this effectively produces an estimate of all camera poses.

5.4 Inverse Depth Bundle Adjustment

Finally, we refine the camera pose estimates using bundle adjustment. The output of the previous steps is an estimated pose $[\mathsf{R}_i|\mathbf{t_i}]$ for each camera and features matches between images. The feature matches are aggregated into feature tracks across multiple images.

The rotation R_i of each camera is parameterized by a 3×1 vector $\mathbf{r}_i$ where $\mathsf{R}_i = \exp_{SO(3)}(\mathbf{r}_i)$. Since the translation is vector is fixed, we do not need to explicitly parameterize it.

We found that, especially with an outward-facing camera, the traditional methods of algebraic triangulation and bundle adjustment over 3D point locations are unstable because of the small baselines involved. Instead, we use an inverse depth parameterization which extends the work of Yu and Gallup [29].

Each 3D point $\mathbf{x}_j$ has a designated reference camera with index n_j so that

$$\mathbf{x}_j = \mathsf{R}_{n_j}^\top (w_j \mathbf{u}_{n_j,j} - \mathbf{t}_{n_j}) \tag{42}$$

where $\mathbf{u}_{n_j,j}$ is the observation of point j in camera n_j and w_j is the inverse depth of point j. We set the reference camera of each point to be the first camera which observed it in the image sequence.

The output of global rotation averaging gives an initialization for the camera rotations. To initialize the 3D points, we first linearly solve for w_j using all observations of the point. Then we use non-linear optimization over all rotation and inverse depth parameters to minimize the total robustified re-projection error

$$\sum_{(i,j)\in\mathcal{V}} h(||\pi(\mathbf{u}_{i,j}) - \pi(\mathsf{R}_i \mathbf{x}_j + \mathbf{t}_i)||^2) \tag{43}$$

where $\pi(\mathbf{u})$ is the perspective projection function $\pi([x\ y\ z]^\top) = [x/z\ y/z]^\top$ and $(i,j) \in \mathcal{V}$ if camera i observes point j. $h(\cdot)$ is the Huber cost function which robustifies the minimization against outlier measurements [10].

6 Evaluation

6.1 Essential Matrix Solvers

In this section we evaluate the speed and accuracy of our novel essential matrix solvers and compare them against the state-of-the-art five-point solution by Stewénius et al. [26] for general camera motion. We will refer to our action matrix solution from Sect. 4.3 as **Spherical** and our polynomial solution from Sect. 4.4 as **SphericalPoly**, and we will refer to the five-point solution as **Stewénius**. We use the implementation of **Stewénius** provided in the OpenGV library [12].

Random Problem Generation. To make synthetic data for our tests, we generate random spherical motion problems using the following scheme. First we generate a random rotation of the desired magnitude θ and calculate the first and second camera poses according to this relative rotation, so that both cameras lie on the unit sphere. Then we randomly generate 3D points within a range of distances from the first camera; we use a distance range of [0.25 0.75] for inward-facing cameras and [4 8] for outward-facing cameras. Each 3D point is projected into both cameras using a focal length of 600 and Gaussian noise is added to the point observations with standard deviation of σ pixels.

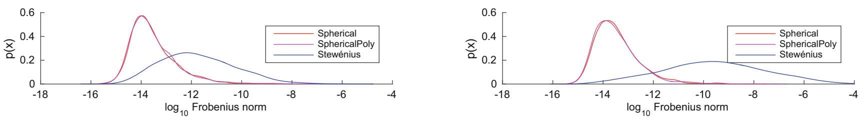

Fig. 1. Kernel density plots for numerical error of minimal solvers with ideal observations and inward-facing cameras (*top*) and outward-facing cameras (*bottom*).

Timing. We calculated the average computation for our solvers over 10 000 randomly generated problems. The testing was performed on a 2.6 GHz Intel Core i5 with optimized code written in C++. **Spherical** and **SphericalPoly** takes 6.9 µs and 6.4 µs on average, respectively. **Stewénius** takes 98 µs; however, the implementation in OpenGV is not optimized for speed.

Numerical Accuracy. We tested the numerical accuracy of the solvers with ideal, zero-noise observations. We generated 1000 random problems with a rotation of $\theta = 1$ degrees and $\sigma = 0$ pixels using both the inward- and outward-facing configuration. We then ran each solver on the problem sets and calculated the Frobenius norm of the error between estimated and true essential matrix. To test the minimal configuration, our solvers used three correspondences for estimation with a fourth for disambiguation and **Stewénius** used five corresponces for estimation with a sixth for disambiguation. The results are plotted in Fig. 1.

Our spherical solvers are almost equivalent to each other in numerical accuracy and are two to four orders of magnitude more accurate than **Stewénius** for spherical camera motion estimation.

Noise. We tested the robustness of the solvers to varying levels of added noise in the image observations. We generated 1000 random problems with a rotation of $\theta = 1$ degrees and $\sigma = 0, 1, \ldots, 10$ pixels using both the inward- and outward-facing configuration. We then ran each solver on the problem sets and calculated the angular error $||\log_{SO(3)}(\mathsf{R}_{\text{true}} \cdot \mathsf{R}_{\text{estimate}}^{\top})||$. For a fair comparison on noisy data, each solver used five correspondences for estimation. The results are plotted in Fig. 2.

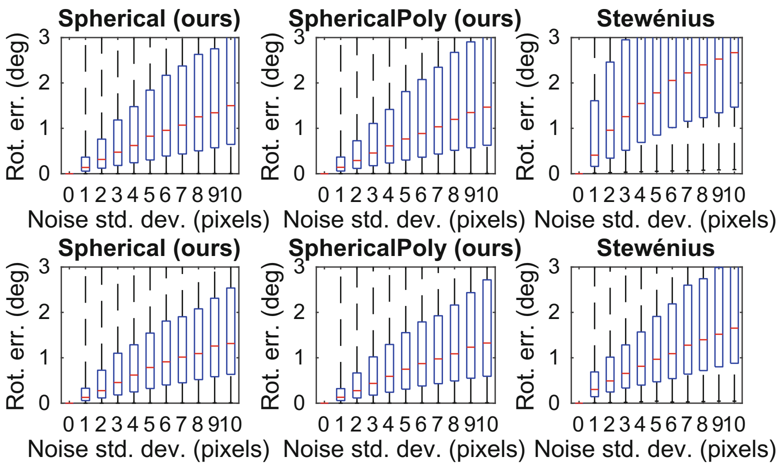

Fig. 2. Box plots for angular error of minimal solvers with noisy observations and inward-facing cameras (*top*) and outward-facing cameras (*bottom*).

Again, both of our solvers are about equivalent in terms of accuracy and outperform **Stewénius** for spherical motion estimation with noisy correspondences.

6.2 Structure-from-Motion Pipeline

We tested the entire proposed structure-from-motion pipeline on several image sequences captured with both inward- and outward-facing configurations. We describe here the details of our implementation and show the resulting 3D reconstructions.

Implementation Details. In our experiments we apply the Oriented FAST and Robust BRIEF (ORB) feature detector [22] and Kanade-Lucas-Tomasi (KLT) feature tracker [15,27]. The KLT tracker uses sub-pixel refinement which is especially helpful for a handheld, outward-facing camera where baseline the between images might be small relative to the scene depth. We use an threshold of 2 pixels for both RANSAC inlier testing and the Huber cost function.

For efficiency, in our experiments we only detect loop closures with the first frame in the sequence. We detect loop closures by iterating through the sequence backward from the last frame and stop when the number of inliers is below the threshold. Feature matches are chosen as the nearest neighbor in Hamming distance between ORB descriptors. A loop closure is accepted if it has at least 100 inliers.

The pipeline was implemented in C++ using OpenCV for image processing functions and Ceres [1] for non-linear optimization. We set a minimum inverse depth of 0.01; points at this distance are essentially points at infinity.

Video Tests. We tested our system on several image sequences captured both indoors and outside with inward- and outward-facing camera configurations. While the general motion of the test videos is circular, they were captured with a handheld camera and thus inevitably exhibit deviations from the circular path. A circular motion solver similar to [23] failed on these sequences in our tests.

For the *street* and *bookshelf* sequences, we used a Sony α5100 camera with 16 mm lens. For the *face* sequence we used an Apple iPhone 5s. Both devices were set to record 1080p video.

The *street* sequence was captured in the middle of a neighborhood street corner. We spun in a circle while holding the camera in our outstretched hands. This sequence has a complete loop which is successfully detected by our system. Figure 3 shows the reduction in drift after loop closure and a view of the complete 3D reconstruction.

The *bookshelf* sequence was captured in an indoor office. This sequence was also capture with outward-facing configuration but does not complete a full loop and has much closer objects than the *street* sequence. Figure 3 shows a view of the 3D reconstruction.

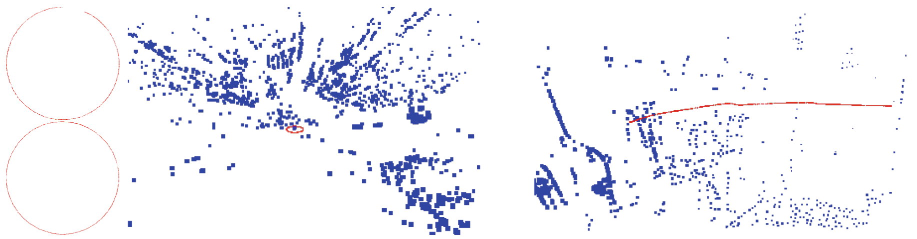

Fig. 3. *Top left*: Estimated camera centers from the *street* sequence before loop closure. *Bottom left*: Camera centers after rotation avergaging and bundle adjustment. *Middle*: 3D reconstruction of the *street* sequence. The red dots are camera centers and the blue dots are reconstructed scene points. *Right*: 3D reconstruction of the *bookshelf* sequence. (Color figure online)

The *face* sequence was captured by holding the iPhone in an outstretched hand with the lens pointed at the user's shoulder. Rotating the arm produces inward-facing spherical motion which was used to capture a scan of the user's face. Figure 4 shows a sample image and the 3D reconstruction.

To further illustrate the accuracy of these reconstructions, we selected image pairs from each sequence and performed stereo rectification using the recovered relative pose. We used block matching to produce disparity maps as shown in Figs. 4 and 5.

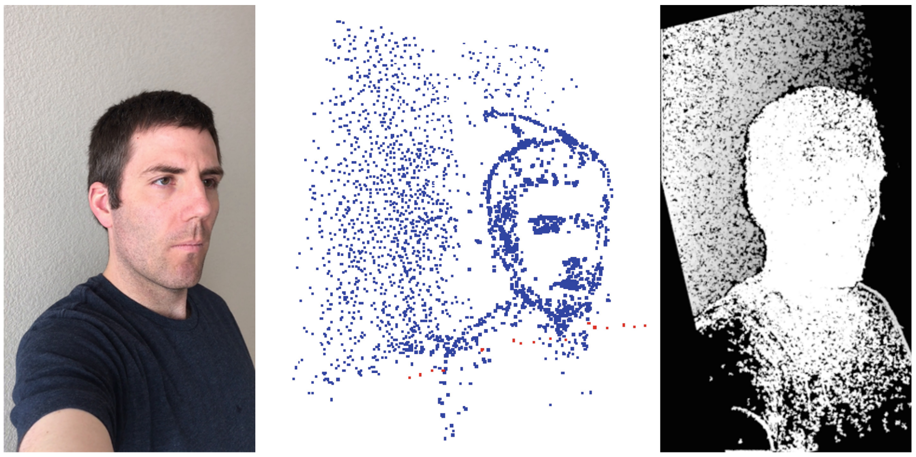

Fig. 4. *Left*: Image from the *face* sequence. *Middle*: 3D reconstruction of the *face* sequence. The red dots are camera centers and the blue dots are reconstructed scene points. *Right*: Disparity map from a rectified stereo pair from the *face* sequence. (Color figure online)

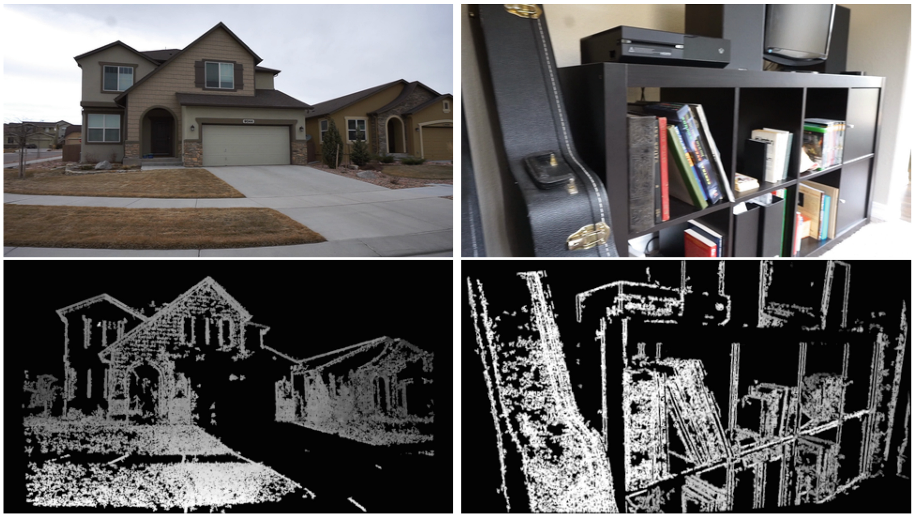

Fig. 5. Disparity maps from from the *street* and *bookshelf* sequences.

7 Conclusions and Future Work

In this work we analyzed the geometry of spherical camera motion. We introduced two solvers for the essential matrix arising from spherical motion. The solvers require three point correspondences in the minimal case as opposed to the five needed for general motion, which reduces the number of hypothesese needed for random sample consensus. The solvers are fast and exhibit better numerical accuracy and robustness to noise than the state-of-the-art.

By integrating these solvers into a structure-from-motion pipeline, we demonstrated that spherical motion greatly simplifies the problem by eliminating the need for translation estimation. Despite the small baselines captured by a hand-held camera, we found that accurate and large-scale reconstruction is possible using spherical structure-from-motion.

One limitation of the approach is the rigidness of the spherical motion constraint; deviations from spherical motion will cause the structure-from-motion pipeline to fail. The system is less sensitive to deviations from the spherical constraint when the sphere's radius is small relative to scene depth; however, the precision of the 3D reconstruction also degrades as the radius-to-depth ratio decreases. One possible way to alleviate this problem would be to increase the image resolution to make the parallax detectable again.

Future work includes more exploration of the potential applications of spherical structure-from-motion for user-friendly scene modeling and object scanning.

Acknowledgments. This material is based upon work supported by the National Science Foundation under Grant No. 1464420.

References

1. Agarwal, S., Mierle, K., et al.: Ceres solver. http://ceres-solver.org
2. Chatterjee, A., Madhav Govindu, V.: Efficient and robust large-scale rotation averaging. In: IEEE International Conference on Computer Vision (ICCV) (2013)
3. Cox, D.A., Little, J., O'shea, D.: Using Algebraic Geometry, 2nd edn. Springer, New York (2005)
4. Cox, D.A., Little, J., O'shea, D.: Ideals, Varieties, and Algorithms, 3rd edn. Springer, New York (2007)
5. Faugeras, O.: Three-Dimensional Computer Vision: A Geometric Viewpoint. MIT Press, Cambridge (1993)
6. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981)
7. Fitzgibbon, A.W., Cross, G., Zisserman, A.: Automatic 3D model construction for turn-table sequences. In: Koch, R., Van Gool, L. (eds.) SMILE 1998. LNCS, vol. 1506, pp. 155–170. Springer, Heidelberg (1998)
8. Hartley, R., Trumpf, J., Dai, Y., Li, H.: Rotation averaging. Int. J. Comput. Vis. **103**(3), 267–305 (2013)
9. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2003)
10. Huber, P.J.: Robust estimation of a location parameter. Ann. Math. Stat. **35**(1), 73–101 (1964)
11. Jiang, G., Quan, L., Tsui, H.T.: Circular motion geometry using minimal data. IEEE Trans. Pattern Anal. Mach. Intell. **26**(6), 721–731 (2004)
12. Kneip, L., Furgale, P.: OpenGV: a unified and generalized approach to real-time calibrated geometric vision. In: 2014 IEEE International Conference on Robotics and Automation (ICRA) (2014)
13. Kukelova, Z., Bujnak, M., Pajdla, T.: Automatic generator of minimal problem solvers. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part III. LNCS, vol. 5304, pp. 302–315. Springer, Heidelberg (2008)
14. Longuet-Higgins, H.C.: A computer algorithm for reconstructing a scene from two projections. In: Readings in Computer Vision: Issues, Problems, Principles, and Paradigms, pp. 61–62 (1987)
15. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI, pp. 674–679 (1981)
16. Mendonça, P.R.S., Wong, K.Y.K., Cipolla, R.: Epipolar geometry from profiles under circular motion. IEEE Trans. Pattern Anal. Mach. Intell. **23**(6), 604–616 (2001)
17. Nister, D.: An efficient solution to the five-point relative pose problem. IEEE Trans. Pattern Anal. Mach. Intell. **26**(6), 756–770 (2004)
18. Nister, D.: Preemptive RANSAC for live structure and motion estimation. In: Ninth IEEE International Conference on Computer Vision, Proceedings (2003)
19. Peleg, S., Ben-Ezra, M.: Stereo panorama with a single camera. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (1999)
20. Peleg, S., Herman, J.: Panoramic mosaics by manifold projection. In: 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Proceedings (1997)
21. Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., Koch, R.: Visual modeling with a hand-held camera. Int. J. Comput. Vision **59**(3), 207–232 (2004)

22. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: an efficient alternative to SIFT or SURF. In: 2011 IEEE International Conference on Computer Vision (ICCV) (2011)
23. Scaramuzza, D., Fraundorfer, F., Siegwart, R.: Real-time monocular visual odometry for on-road vehicles with 1-point ransac. In: IEEE International Conference on Robotics and Automation, ICRA 2009, pp. 4293–4299 (2009)
24. Shum, H.Y., Szeliski, R.: Stereo reconstruction from multiperspective panoramas. In: The Proceedings of the Seventh IEEE International Conference on Computer Vision (1999)
25. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3D. ACM Trans. Graph. (TOG) **25**, 835–846 (2006). ACM
26. Stewénius, H., Engels, C., Nistér, D.: Recent developments on direct relative orientation. ISPRS J. Photogramm. Remote Sens. **60**(4), 284–294 (2006)
27. Tomasi, C., Kanade, T.: Detection and tracking of point features. Technical report, CMU-CS-91-132, School of Computer Science, Carnegie Mellon University, Pittsburgh (1991)
28. Wilson, K., Snavely, N.: Robust global translations with 1DSfM. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014, Part III. LNCS, vol. 8691, pp. 61–75. Springer, Heidelberg (2014)
29. Yu, F., Gallup, D.: 3D reconstruction from accidental motion. In: 27th IEEE Conference on Computer Vision and Pattern Recognition (2014)