Spherical Approximation for Multiple Cameras in Motion Estimation: its Applicability and Advantages

Jun-Sik Kim[a,*], Myung Hwangbo[a], Takeo Kanade[a]

[a]*Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA*

Abstract

Estimating motions of a multi-camera system which may not have overlapping fields of view is generally complex and computationally expensive because of the non-zero offset between each camera's center. It is conceivable that if we can assume that multiple cameras share a single optical center, and thus can be modeled as a spherical imaging system, motion estimation and calibration of this system would become simpler and more efficient.

In this paper, we analytically and empirically derive the conditions under which a multi-camera system can be modeled as a single spherical camera. Various analyses and experiments using simulated and real images show that spherical approximation is applicable to a surprisingly larger extent than currently expected. Moreover, we show that, when applicable, this approximation even results in improvements in accuracy and stability of estimated motion over the exact algorithm.

Keywords: Camera Motion Estimation, Multiple Cameras, Spherical Approximation, Camera Calibration, Structure from Motion

1. INTRODUCTION

Assume that vehicles such as cars and small sized unmanned aerial vehicles (UAV) flying in urban environments have multiple cameras whose fields of view (FOV) may not overlap. Two major current approaches exist for estimating ego-motions of such a vehicle using a multiple camera system. One approach uses the linear 17-point algorithm based on the generalized camera model [1][2]. With this algorithm, all six degrees of freedom are linearly recovered for motions that include a scale. The other approach is based on the assumption that each camera shares a single optical center, which makes it possible to approximate the imaging system as a spherical camera. Motion estimation algorithms for a spherical camera recover the camera's motions only up to scale. It is generally expected that spherical approximation, while resulting in a simpler and more efficient algorithm, causes systematic errors in estimated motions compared with the generalized 17-point algorithm.

To test this understanding, we applied both methods to real image sequences taken by a set of cameras, as shown in Fig. 1. The three cameras can view forward, left and right, and the distance between the camera centers is about 100 mm. Fig. 2 shows examples of 320×240 synchronized images captured by the camera set. We manually chose correspondences in the sequences to remove outliers.

The generalized 17-point algorithm gave the ego-motion estimation results shown in Fig. 3(a). This *exact* algorithm was unable to obtain accurate results for both rotation and translation. On the other hand, the spherical approximation method, which assumes all the cameras share a single optical center, also viewed as aggressive approximation, provided the more accurate results shown in Fig. 3(b). This method can not determine an absolute scale of the estimated motion, but does obtain more accurate and stable estimations for both rotation and translation direction.

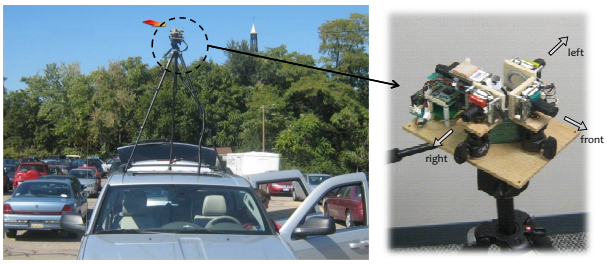


Figure 1: An outdoor vehicle carrying a set of multiple cameras on its roof: Three NTSC cameras look in three different directions (front, left and right) with no overlapping fields of view.

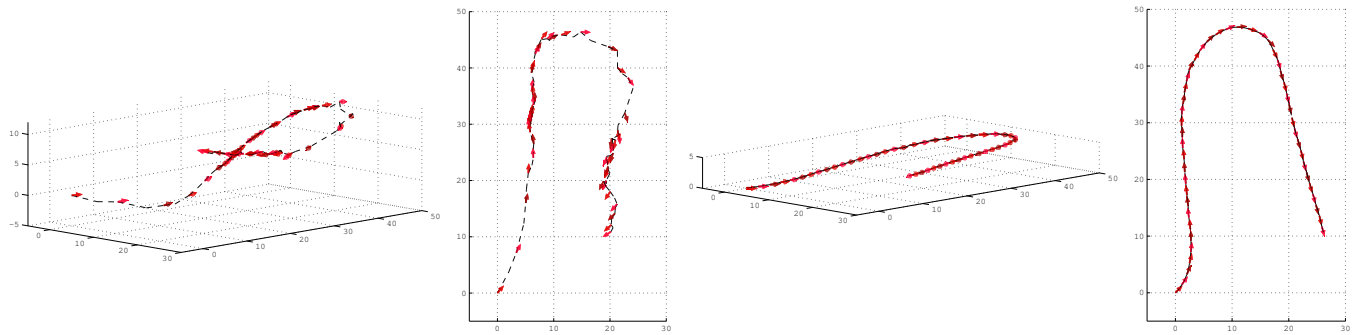
We wondered why the *exact* algorithm was outperformed by the *aggressive approximation*. Because it can be speculated that a poorer performance may be caused by inaccurate camera calibration or incorrect selection of features, we conducted another experiment using simulated data which should be free of these potential inaccuracies. To make the experiment as similar to the real situation as possible, we replicated the path and scene points obtained by real experiment data. With noiseless feature data, the generalized 17-point algorithm provided the results shown in Fig. 4(a), which represents the exact given trajectory. However, Fig. 4(b) shows, when 1-pixel Gaussian

*Corresponding author.

Email addresses: kimjs@cs.cmu.edu (Jun-Sik Kim), myung@cs.cmu.edu (Myung Hwangbo), tk@cs.cmu.edu (Takeo Kanade)

Published in Computer Vision and Image Understanding 114 (2010) 1068-1083, available at www.elsevier.com/locate/cviu.

Figure 2: An example of the time-synchronized input image set obtained from three cameras.



(a) 17-point algorithm using a generalized camera model

(b) 5-point algorithm using a spherical camera model

Figure 3: Outdoor experiment using a real image sequence: comparison between two different algorithms in regard to camera models in estimated camera trajectory

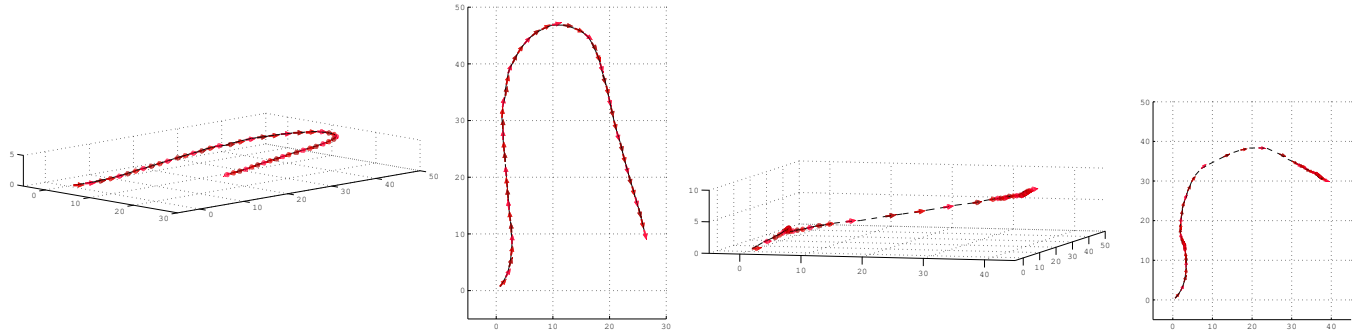
tracking errors were added to the features, an inaccurate trajectory was obtained.

We also tested the spherical approximation method using the same feature data. With the noiseless data, the *aggressive* approximation method provided the results shown in Fig. 5(a), which suffered from systematic errors caused by this approximation. With the addition of 1-pixel Gaussian feature tracking error, we obtained the results shown in Fig. 5(b), which were more accurate than the results obtained by the generalized 17-point algorithm in Fig. 4(b). This suggests that feature tracking error affects the approximation algorithm much less than it affects the exact algorithm in this simulation. If scene is very distant compared to the distance between camera centers, an exact algorithm can be less robust than an approximation algorithm which suffers from errors induced by the approximation.

These results can be explained if we consider the principles involved in depth estimation using a stereo system. In stereo, it is not possible to obtain accurate depth estimation of very distant points. If system's baseline is too short compared to distances from the system to scene points, the cameras in a stereo system seem to have an identical projection center for the points. This principle applies to a multiple camera system whose centers do not coincide, and explains why the motions estimated by the generalized 17-point algorithm, especially traveling distances, are so different from the real ones. If the scene points are too far, the distance between camera centers becomes negligible. In this situation, trying to estimate the scale, which can not be accurate, makes the entire motion estimation inaccurate and unstable.

These phenomena can also serve as an example to show that a simpler and more restricted model (spherical camera model) may tend to be more stable than a more general model (generalized camera model). Still, to the best of our knowledge, there has been no systematic analysis done to this point to understand which algorithm best fits the problem of the ego-motion estimation with multiple cameras.

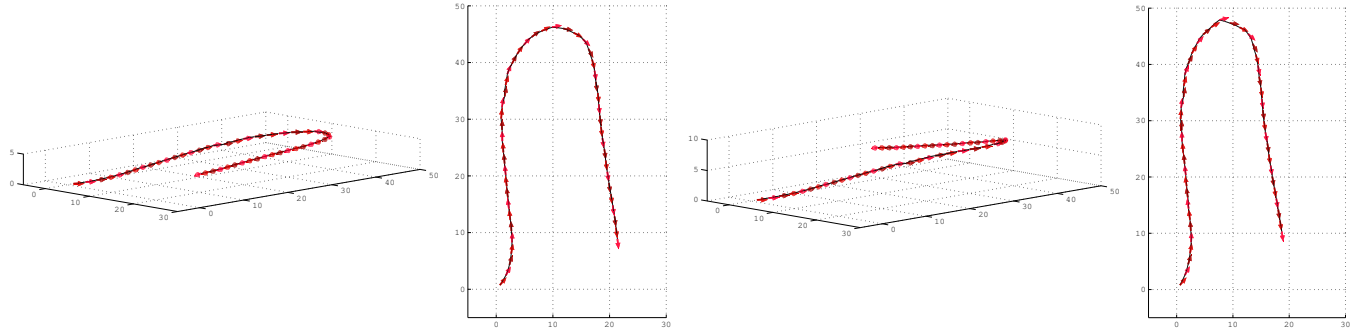
In this paper, we will show the conditions under which the simpler spherical assumption is more applicable both theoret-



(a) Results with no addition of noise in feature tracking

(b) Results after addition of 1-pixel noise in feature tracking

Figure 4: Simulation of the generalized 17-point algorithm: fragile with respect to feature tracking noise in camera motion estimation



(a) Results with no addition of noise in feature tracking

(b) Results after addition of 1-pixel noise in feature tracking

Figure 5: Simulation of the spherical approximation algorithm: robust in camera motion estimation in respect to feature tracking noise

ically and empirically. Performance analyses will be given in regard to various aspects such as feature tracking error, distance to scene points, calibration error, configurations of cameras, and image resolutions. Surprisingly, the range in which spherical approximation outperforms the exact 17-point algorithm is larger than we tend to expect. In addition, we will show that spherical approximation makes camera calibration simpler than previous methods.

2. RELATED WORKS

One of the fundamental difficulties in using single-camera structure from motion (SFM) is *translation-rotation ambiguity*. The apparent motion or optical flows between two frames of, for example, small sideways translational motion and small panning rotational motion are hard to distinguish. Similarly, it is hard to accurately estimate these two motion components based on observed optical flows. Baker et al. [3] showed that this difficulty in decoupling the two components is mainly a result of insufficient fields of view (FOV) of an imaging system.

By expanding the FOV, this ambiguity can be reduced. To do this, one can use a single camera with a wider-FOV optical system, such as an omnidirectional mirror or a fisheye lens. For an optical system which has a larger FOV, however, the camera should have a much higher resolution in order to obtain a sufficient angular resolution per pixel. Since a high-resolution camera is expensive and heavy, and requires more bandwidth to transfer captured image sequences in realtime, it is impractical to use for applications which have cost, weight or bandwidth limitations such as the limitations of a small-sized UAV.

Alternatively, one can use multiple cameras whose FOV may not overlap to obtain a larger FOV as a whole. One can use a panoramic camera system in which each camera physically shares a single focal point [4]. For a general setup, the problem of estimating motions of multiple non-overlapping cameras is most conveniently cast in the form of a generalized camera model [5], for which several algorithms have been proposed. Chen and Chang, and Frahm et al. proposed pose estimation methods for a generalized camera [6–8]. To use these methods to estimate motions, some known 3D points are required in advance. Another option is to use initial motions obtained using accurate odometry rather than known 3D points [9]. Ego-motion estimation of multiple cameras without the use of prior scene point or motion knowledge has been also investigated. Chen et al. [10] presented an algorithm based on nonlinear optimization. Another algorithm, proposed by Baker et al. [3], decomposes rotation and translation estimations using their *Argus eye* system. Pless [1] observed that a linear constraint of correspondences exists in a generalized camera, just as in a projective camera, and that motion parameters can be extracted from this constraint. This algorithm explicitly use the distance between camera centers, so the case of the multi-camera systems in outdoor environment, for which we will use a spherical camera model, becomes a degenerate one [2], when features

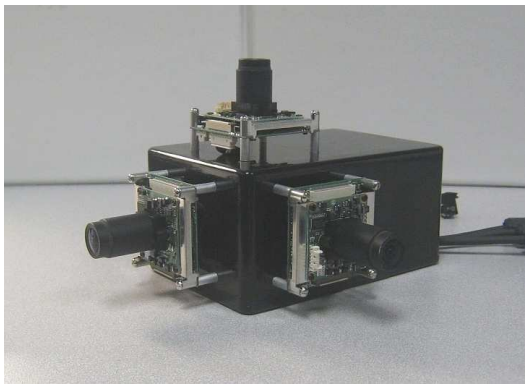


Figure 6: An example of a multiple non-overlapping view camera system looking in top, front and side directions

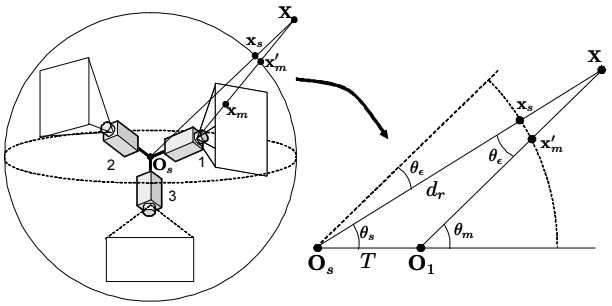


Figure 7: Single spherical camera model. Spherical mapping of features onto a unit sphere, and error induced by this spherical approximation.

are too far from the system. In addition, the algorithm requires 17 points to estimate a motion, and this makes it harder to use RANSAC based algorithms to exclude outliers. Recently, linear and nonlinear solutions for geometric cost minimization have been proposed for some degenerate cases [2, 11].

We will show that spherical approximation of multiple cameras outperforms the exact generalized camera model when the centers of the cameras are *sufficiently close* to each other compared to the distance of the cameras to the scene. We find conditions for this spherical approximation both analytically and empirically. Compared to the conventional motion estimation method, which uses a single camera or the generalized camera model, we show that spherical approximation for multiple cameras is effective for motion estimation in an outdoor environment. Interestingly, not only does the spherical approximation simplify the ego-motion estimation and calibration procedures, but also motion estimation based on this assumption becomes more accurate and stable.

3. MULTI-CAMERA SYSTEM AS A SPHERICAL CAMERA

Fig. 6 shows one of our multi-camera systems, consisting of three low-weight cameras placed in three orthogonal directions with non-overlapping fields of view. The distance between each camera center is approximately 100 mm.

We will treat this set of cameras as a single spherical camera, or equivalently, we will assume that the three cameras share a common projection center. Applying a spherical approximation to non-spherical cameras induces positional errors when mapping to the spherical image the feature positions located in the individual camera image as shown in Fig. 7. Spherical approximation is applicable under the conditions in which the induced errors are less than the feature tracking errors.

Fig. 7 depicts how spherical approximation induces the errors. There are a camera center $\mathbf{O}_1$ and a spherical camera center $\mathbf{O}_s$ at a distance T. Suppose that this camera system sees a point $\mathbf{X}$ whose distance from the center $\mathbf{O}_s$ is d_r. Because an intrinsically calibrated camera is an angle sensor, a pixel on an image is expressed as an angle from the camera center. The angle measured with the camera $\mathbf{O}_1$ is θ_m which would be θ_s if $\mathbf{O}_1$ coincided with $\mathbf{O}_s$. Thus, the induced error obtained under this assumption is $\theta_\epsilon = \theta_m - \theta_s$.

On the triangle $\mathbf{X}\mathbf{O}_1\mathbf{O}_s$,

$$\sin\theta_\epsilon = -\frac{T}{d_r}\sin\theta_m \tag{1}$$

according to the law of sines. Because the feature tracking error $|\theta'_\epsilon|$ should be very small and larger than the induced error $|\theta_\epsilon|$, we can derive a condition

$$|\theta'_\epsilon| > |\theta_\epsilon| = \frac{T}{d_r}\sin|\theta_m| \tag{2}$$

which gives

$$T < \frac{|\theta'_\epsilon|}{\sin|\theta_m|}d_r. \tag{3}$$

This inequality in (2) and (3) shows that under the given feature tracking error $|\theta'_\epsilon|$, spherical approximation is applicable when 1) the maximum $|\theta_m|$ is small, which means the FOV of each camera is narrow, and 2) the distance to the scene point d_r is much larger than the distance between camera centers T, or equivalently, T is as small as possible.

For example, suppose that the expected uncertainty in feature tracking is one pixel and the cameras have 90° FOVs with 300 pixels, seeing in radial directions. In this case, feature tracking error is 0.3° maximum and $\max|\theta_m| = 45°$, and the condition (3) that safely assumes a spherical camera is $T < 0.0074d_r$ or equivalently, $d_r/T > 135.05$. If the distance to scene points d_r is 10 m, then the distance between cameras should be less than 74 mm in order to safely assume a spherical camera.

Note that this is a very strict sufficient condition for multi-camera images to be spherical, not for motion estimation of a multi-camera system. We will empirically find the necessary condition for motion estimation in Section 5.1.2.

4. MOTION ESTIMATION AND CALIBRATION METHOD UNDER SPHERICAL CAMERA APPROXIMATION

By assuming that all the cameras share a single projection center, any motion estimation or SFM algorithm for a single focal point camera can be used. We follow a conventional process to build SFM: motion estimation between two frames, integration of motions, and optional bundle adjustment [12].

4.1. Motion Estimation Between Two Frames of Spherical Cameras

To estimate a motion of a spherical camera between frames, we follow three steps: mapping points on a unit sphere, estimating motions using a random sample consensus (RANSAC), and finally, optimizing motion parameters.

4.1.1. Mapping points on a unit sphere

In the first step, image correspondences of each camera are normalized using the intrinsic and distortion parameters of the camera. The normalized points are mapped on a unit sphere using camera rotations in the rig coordinate system. To represent a point on a unit sphere, we use a non-homogeneous 3-vector which is a directional unit vector originated from the sphere center. A point on a unit sphere $\mathbf{x}'_{kn}$ is given as

$$\mathbf{x}'_{kn} = \mathbf{R}_{k:1}\mathbf{x}_{kn} \text{ s.t. } |\mathbf{x}_{kn}| = 1 \tag{4}$$

where $\mathbf{x}_{kn}$ is a n-th normalized point observed by camera k, and $\mathbf{R}_{k:1}$ represents a rotation from the local coordinate of camera k to the coordinate of camera 1, selected as the rig coordinate system.

4.1.2. Estimating an essential matrix with RANSAC

It can not be expected that all the correspondences are tracked and matched correctly, thus in order to eliminate false matches, a RANSAC algorithm is applied. Considerations in implementing RANSAC include an estimator with a small number of correspondences and a verification function for a hypothesis. We use the five-point algorithm [13] as an estimator of an essential matrix.

Given a hypothesis $\mathbf{E}$, we use the verification function for correspondences $\mathbf{x}'_1$ and $\mathbf{x}'_2$ on the unit sphere as

$$d(\mathbf{E}, \mathbf{x}'_1, \mathbf{x}'_2) = \frac{|\mathbf{x}_2'^{\top}\mathbf{E}\mathbf{x}'_1|}{\sqrt{a^2+b^2}} + \frac{|\mathbf{x}_1'^{\top}\mathbf{E}^{\top}\mathbf{x}'_2|}{\sqrt{d^2+e^2}} \tag{5}$$

where a, b, d and e are epipolar line coefficients defined as $(a, b, c)^{\top} = \mathbf{E}\mathbf{x}_1$ and $(d, e, f)^{\top} = \mathbf{E}^{\top}\mathbf{x}_2$. This function does not actually compute a geodesic distance on the unit sphere, but a distance on an infinitely large image plane. Though this is not ideal, we still use (5) because it makes it easy to use spherical approximation on the existing implementation for single camera SFM.

At this point, there is another advantage of applying spherical approximation rather than the generalized 17-point algorithm. Note that the 17-point algorithm requires 17 points to make one motion hypothesis, so a RANSAC based method would require more trials to select a set of inliers. Motion estimation using spherical approximation uses just 5 points, and thus, it requires fewer trials to select inlier sets.

4.1.3. Optimizing motion parameters

After getting an initial estimate of an essential matrix $\mathbf{E}$ and a set of inliers P, the essential matrix is estimated more accurately by minimizing

$$C(\mathbf{E}; \mathbf{x}_1, \mathbf{x}_2) = \sum_{\{\mathbf{x}_1, \mathbf{x}_2 \in P\}} d(\mathbf{E}, \mathbf{x}_1, \mathbf{x}_2). \tag{6}$$

We use the Levenberg-Marquardt algorithm implemented by Lourakis [14]. After optimization, physically meaningful motion parameters are retrieved from the estimated essential matrix using cheirality [12].

4.2. Trajectory Estimation by Motion Integration between Two Views

The trajectory of the camera rig is estimated by incrementally integrating the estimated motions between frames. One integration from frame i to frame j is done by

$$\begin{bmatrix} \mathbf{R}(j) & \mathbf{t}(j) \\ \mathbf{0}^{\top} & 1 \end{bmatrix} = \begin{bmatrix} \Delta\mathbf{R}(i,j) & \Delta\mathbf{t}(i,j) \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}(i) & \mathbf{t}(i) \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \tag{7}$$

where $\Delta\mathbf{R}(i,j)$ and $\Delta\mathbf{t}(i,j)$ are a rotation matrix and a translation vector estimated between frame i and j, respectively.

Note that it is impossible to get a Euclidean traveling distance, because there is no absolute distance available. We can only expect to estimate a trajectory defined up to scale. Therefore, one problem in integrating motions is to match scales of the translational motions, i.e. $\mathbf{t}(i)$ and $\Delta\mathbf{t}(i,j)$ in (7). We match the scales using the conventional method by solving

$$\arg_s \min \sum_{\text{all}\mathbf{X}} |s(\Delta\mathbf{R}(i,j)\mathbf{X}_i + \Delta\mathbf{t}(i,j)) - \mathbf{X}_j|^2 \tag{8}$$

where $\mathbf{X}_i$ and $\mathbf{X}_j$ are corresponding 3D points recovered in i and j frames, respectively.

4.3. Optional Bundle Adjustment

The trajectory obtained by integration is achieved by minimizing the cost functions defined between two views. As time goes by, estimation errors of the trajectory accumulate gradually. To reduce accumulation of errors, bundle adjustment (BA) can be used [12]. We use the efficient implementation of the sparse BA by Lourakis and Argyros [15].

The cost function to be minimized is a distance c between a measured feature $\mathbf{x}$ and a predicted feature from a given motion $\{\mathbf{R}, \mathbf{t}\}$, and the recovered scene point $\mathbf{X}$ in camera i:

$$c\left(\mathbf{x}, \{\mathbf{R}, \mathbf{t}\}, \mathbf{X}\right) = \|\mathbf{x} - h(\mathbf{R}_{1:k}(\mathbf{R}\mathbf{X} + \mathbf{t}))\|^2 \tag{9}$$

where $\mathbf{R}_{1:k}$ is a rotation matrix of the camera k in the rig coordinate system, and $h(\cdot)$ is a projection function used to make a 2D prediction. This cost function is defined on the image plane of each camera, although the motion estimation between frames is formulated on a unit sphere. We use this formulation to use the raw measurements $\mathbf{x}$ obtained by feature tracking. Because implementation parameterizes a motion of a camera rig with seven parameters, for m frames and n 3D features, it requires l image projections such that $2l > 7m + 3n$, which is easily achievable in this case.

4.4. Compensation of Errors in Intrinsic Parameters

If the intrinsic parameters of cameras in a rig are not accurate, the accuracy and stability of estimated motions become worse. The experimental results for effects of errors in intrinsic parameters are shown in Section 5.1.7, and in Fig. 17.

To compensate for the errors in the intrinsic parameters, we assume that the intrinsic parameters of the cameras do not change throughout the whole sequence, and that rough estimates of the intrinsic parameters are available. Based on these assumptions, we apply Mendonca and Cipolla's autocalibration algorithm [16] to compensate for the errors of the intrinsic parameters. By normalizing features using the roughly estimated intrinsic parameters, the initial estimate of the intrinsic parameters can be set to an identity matrix. Although all the five parameters have errors, we use a simpler model for the errors $\Delta\mathbf{K}$

$$\Delta\mathbf{K} \triangleq \mathrm{diag}(\alpha, \beta, 1) \tag{10}$$

which means that only the focal length and pixel aspect ratio are erroneous. The autocalibration algorithm minimizes the cost function

$$C(\Delta\mathbf{K}) = \sum \frac{{}^1\sigma(i,j) - {}^2\sigma(i,j)}{{}^2\sigma(i,j)} \tag{11}$$

for each camera, where ${}^1\sigma(i,j)$ and ${}^2\sigma(i,j)$ are two singular values of the essential matrix induced by the erroneous camera matrix $\Delta\mathbf{K}$ between frame i and j. For the simple model given in (10), which has only two unknowns, this algorithm requires at least two views. In practice, we use more views, as much as 20 or 30 to achieve stability.

4.5. Calibration of Extrinsic Parameters

The cameras in the multi-camera system are fixed in a camera rig. Thus, each camera motion in a local camera coordinate system is related to the motion of the rig and the pose of the camera in the rig coordinate system. Various methods have been proposed to estimate the relationships between each camera in a rig. Recently, Kumar et al. presented a method which uses mirrors to calibrate non-overlapping cameras [17]. Kim et al. [18] and Esquivel et al. [19] developed similar calibration algorithms based on motion constraints. In this section, we propose another method based on spherical approximation as a direct extension of these works.

Let us start with the method based on motion constraints, and derive the proposed method from it. Without loss of generality, the coordinate system of the first camera is set as the rig coordinate system. Suppose that the second camera coordinate is expressed with rotation $\mathbf{R}_{1:2}$ and translation $\mathbf{t}_{1:2}$ in the rig coordinate system. If the rig moves by rotation $\Delta\mathbf{R}_1(i,j)$ and translation $\Delta\mathbf{t}_1(i,j)$ between frames i and j, the relative rotation $\Delta\mathbf{R}_2(i,j)$ and translation $\Delta\mathbf{t}_2(i,j)$ of the second camera are expressed as

$$\begin{bmatrix} \Delta\mathbf{R}_2(i,j) & \Delta\mathbf{t}_2(i,j) \\ \mathbf{0}^\top & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{1:2} & \mathbf{t}_{1:2} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{R}_1(i,j) & \Delta\mathbf{t}_1(i,j) \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{1:2} & \mathbf{t}_{1:2} \\ \mathbf{0}^\top & 1 \end{bmatrix}^{-1}. \tag{12}$$

This equation is a form of $\mathbf{AX} = \mathbf{XB}$ on the Euclidean group. The left illustration in Fig. 8 shows this motion constraint using two cameras. This problem is known as hand-eye calibration for a robotic manipulator [20]. This equation can be solved by measuring motions of each camera using planar patterns. This method still requires a pattern for each camera even though the pose relations between patterns do not need to be either known or constrained.

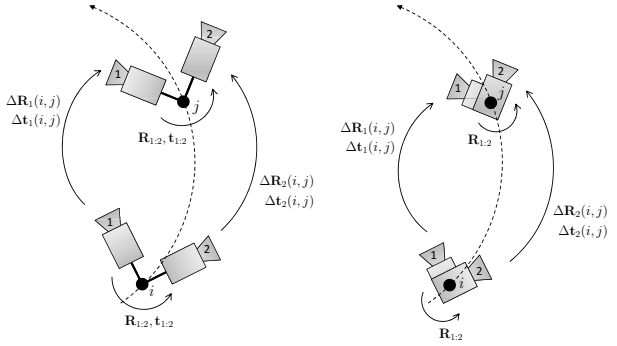


Figure 8: Multi-camera calibration. (Left) Motion constraint of a multi-camera system. The motion of a camera should be determined using the motion of the camera rig. (Right) Proposed calibration method. By ignoring the discrepancy $\mathbf{t}_{1:2}$ between cameras, rotational relation $\mathbf{R}_{1:2}$ between cameras can be estimated using rotation measurements $\Delta\mathbf{R}_1(i,j)$ and $\Delta\mathbf{R}_2(i,j)$ obtained from single camera SFM.

When using spherical approximation, we only need to know the rotations between cameras, not the discrepancies between camera centers. Although the method based on the multiple patterns works well, it is possible to estimate rotations between cameras without any pattern, by estimating motions of each camera directly from image sequences.

Eq. (12) consists of two parts: rotation and translation. The rotation part can be written as

$$\Delta\mathbf{R}_2(i,j)\mathbf{R}_{1:2} = \mathbf{R}_{1:2}\Delta\mathbf{R}_1(i,j) \tag{13}$$

which is also a form of $\mathbf{AX} = \mathbf{XB}$ on the rotation group. We can measure the relative rotations $\Delta\mathbf{R}_1(i,j)$ and $\Delta\mathbf{R}_2(i,j)$ of intrinsically calibrated cameras between two frames i and j using a conventional single camera SFM algorithm as depicted in the right illustration in Fig. 8. Park and Martin's method [20] is used to solve (13) in a least square manner. Note that the absolute translation can not be estimated, because the estimated translations of cameras are all defined up to scale. This proposed method does not require any patterns to calibrate a set of cameras, although it needs intrinsic parameters of each camera. This requires at least two motions, while we use more motions for accuracy and stability, indicated in Fig. 18(a).

5. EXPERIMENTS

We conducted a series of experiments to analyze the performance of spherical approximation in various aspects, including both the motion estimation and the online calibration method.

5.1. Performance of Motion Estimation

We analyzed the performance of motion estimation based on spherical approximation using simulated data. In generating simulated data, we assumed a small aerial vehicle flying in a hallway with a width, height, and depth of 10 m, 20 m and

20 m, respectively. Randomly generated 300 points were on each wall. The distances from the points to the wall were determined randomly with a Gaussian distribution whose standard deviation is 1 m.

In the following experiments, the camera system was modeled on the real camera setup shown in Fig. 6. To replicate this setup, we used real calibration data that was obtained using static planar patterns [18]. The distances between camera centers were approximately 100 mm. Each camera had a resolution of 300×300, and each focal length was set to 300 pixels. Feature tracking errors were assumed to follow a 1-pixel Gaussian distribution. For each experiment, we performed 300 trials and generated the motions of the camera rig randomly. Unless explicitly mentioned, each of these parameters remained constant in the following experiments.

5.1.1. Feature tracking error

We compared the performance of spherical approximation to that of single camera SFM [13], that of a purely spherical camera and that of the generalized 17-point algorithm. Because the original 17-point algorithm is degenerate for non-overlapping FOV cameras, we tested the modified algorithm which was proposed by Li et al. [2] The purely spherical camera had the same number of cameras with coinciding focal points, and thus modeled a panoramic camera.

Fig. 9(a) shows the absolute errors of the estimated rotations. One can see that spherical approximation resulted in significant improvements in both accuracy and stability for estimating rotations between views. Compared to the purely spherical camera, which did not suffered from errors induced by approximation, the spherical approximation had slightly lower performance, but was comparable when the feature tracking error was large.

Fig. 9(b) is an enlarged version of Fig. 9(a). When the feature tracking error was very small, a larger estimation error occurred with spherical approximation. This represents the error induced by spherical approximation. Spherical approximation achieved better results than the single-camera SFM and the generalized 17-point algorithm when the feature tracking error was larger than 0.05 and 0.2 pixels, respectively.

We also compared the accuracy of estimated translation directions in Fig. 9(c). As these results indicate, more accurate rotation estimation tends to induce more accurate estimation of translation direction under the same feature tracking errors. Similar to the rotation estimation, spherical approximation showed more accurate and stable results than the single-camera SFM and the generalized 17-point algorithm when the tracking error was large.

In Fig. 9(d), the performance of translation estimation methods using an external gyroscope sensor [21] were compared. For this experiment, we fixed the feature tracking error at 1 pixel and thus all the methods which did not use the gyroscope required consistency. We contaminated the measurements of the gyroscope sensor with Gaussian noise. The spherical approximation estimated more accurate translational directions than the methods using a gyroscope when the gyroscope error was larger than 0.2°. Even when the gyroscope provided accurate results, the spherical approximation model worked comparably. The stability achieved by the spherical camera assumption was also better than the other tested methods.

5.1.2. Distance to scene points

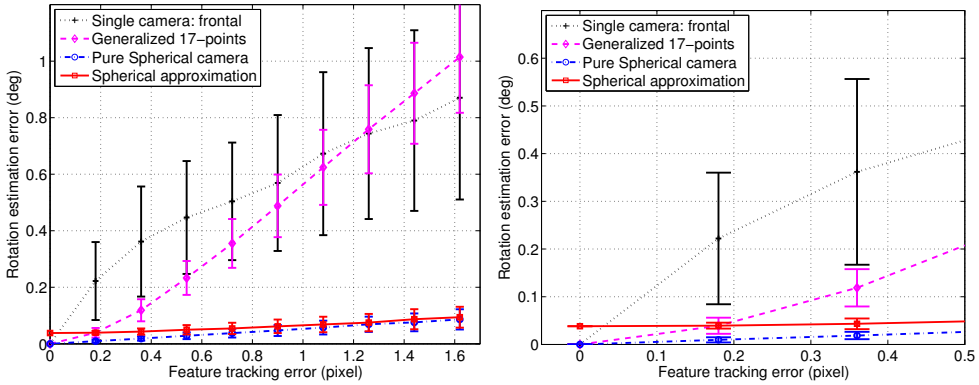
At the beginning of this paper, we claimed that better performance is achieved when using an approximation algorithm with less DOFs than using an exact algorithm with more DOFs, if the distances T between camera centers are small compared with the distance to scene points d_r. In this experiment, we empirically investigated the condition of d_r and T. The distances between cameras were fixed, while the scene points increased in distance. Thus, the single-camera SFM and the purely spherical cameras in this case were tested only for distance to the scene. Fig. 10 shows the performance comparison under the feature tracking error following a 1-pixel Gaussian distribution.

Under the same feature tracking errors, the results of the spherical approximation became more accurate and stable, when the distance to the scene points increased. We made three additional interesting observations from the result. First, the translation direction estimation obtained by spherical approximation maintained accuracy and stability when the scene points became more distant, only degrading slowly, while the accuracy and stability obtained by the generalized 17-point algorithm degraded faster. Second, the error in rotation estimation provided by the generalized 17-point algorithm became larger when the scene points gained distance. We tend to think that the rotation estimation should become more accurate in this case, as the single camera SFM does in Fig. 10(a). However, traveling distance obtained by the generalized 17-point algorithm could not be accurate due to the relatively short baseline, and this error in estimated distance propagated to the rotation estimation. Third, the working range of spherical approximation was much larger than we expected. The theoretical d_r/T ratio of a spherical image is approximately 135 as derived in Section 3, but the spherical approximation outperformed the exact algorithm even when d_r/T was just 10 for motion estimation.

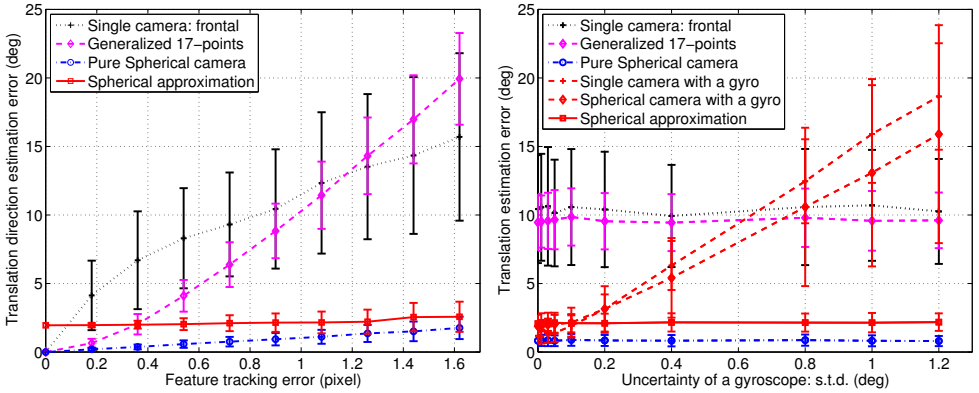
Fig. 11 shows the minimum feature tracking errors under which the spherical approximation outperformed the other exact algorithms in motion estimation. In other words, it shows the point at which the feature tracking error starts to dominate the approximation error under the given d_r/T ratio for motion estimation. Because the spherical approximation can not outperform the pure spherical camera, we only compared it to the single camera SFM and the generalized 17-point algorithm. The minimum feature tracking errors were obtained by checking motion estimation errors under various feature tracking noise, as shown in Fig. 9(b). For a given camera system whose 1-pixel feature tracking error corresponds to about 0.3 degrees, spherical approximation outperformed the other exact methods when the d_r/T ratio was larger than 10, which is the same as the result of the previous experiment under 1-pixel feature tracking error.

5.1.3. Image resolution

The uncertainty in feature tracking is a result of the digitization of input images. If we use higher resolution cameras, the



(a) Rotation (b) Enlarged view of (a)



(c) Translation direction (d) Comparison with a gyro-based method

Figure 9: Comparisons of motion estimation accuracy: (a) Errors of estimated rotations under various feature tracking errors, and (b) Enlarged view of the graph (a). (c) Errors of estimated translational directions under various feature tracking errors. (d) Comparison of translation direction estimation algorithms under the same feature tracking errors. The proposed "spherical" approximation is more robust than other methods which use an external gyroscope when their uncertainty is larger than 0.1 degrees.

angular uncertainty of features is reduced, and the accuracy of the estimated motions shows better performance. Fig. 12 shows the effects of changing the image resolutions.

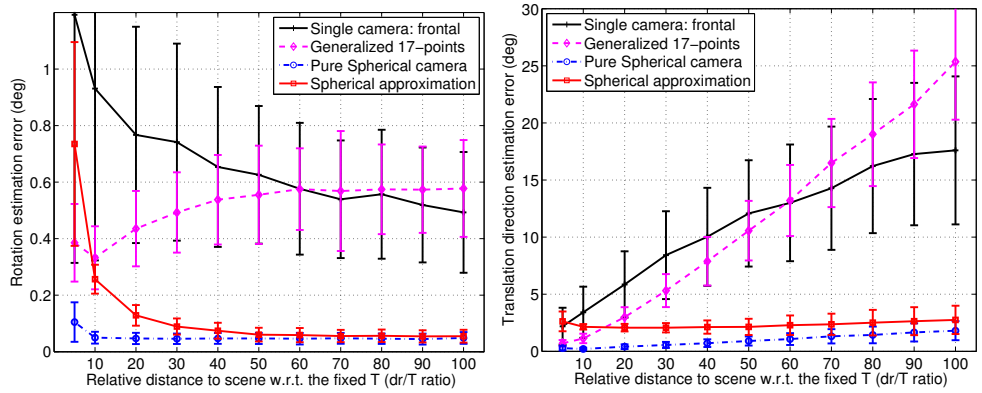
Fig. 12(a) shows the results of increasing resolution for one of three cameras. As expected, the performance of the single-camera SFM improved quickly when the image resolution was increased, but the other methods did not make significant improvements. Note that spherical approximation worked better than the single camera whose resolution was 10-times-higher. This implies that a larger FOV is more helpful for estimating motions than accurate feature tracking, but this depends on the estimation method. The performance of the generalized 17-point algorithm did not improve when the image resolution was increased for only one of three cameras.

In the experiment illustrated in Fig. 12(b), the resolution of all three cameras was increased. The performance of the generalized 17-point algorithm improved very rapidly, while that of the spherical approximation did not. When the resolution was 5-times higher, the generalized 17-point algorithm started to outperform spherical approximation. This shows that the generalized 17-point algorithm can be more useful with very high resolution cameras than spherical approximation.

5.1.4. Covered field of view

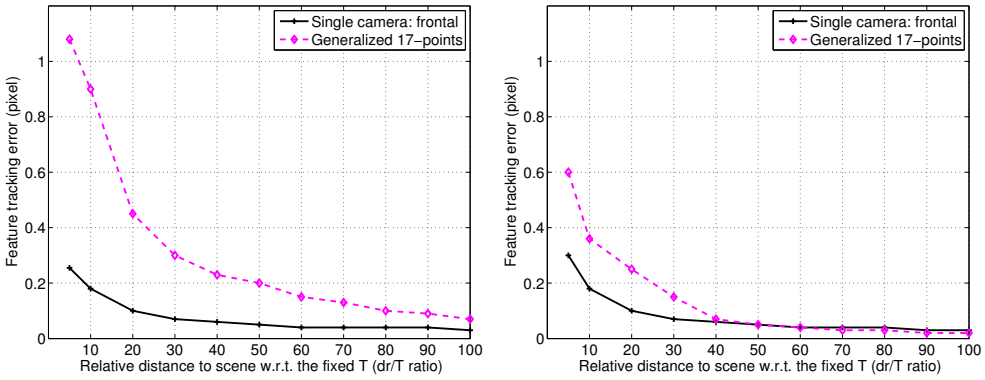
To investigate the effects of covered FOVs on motion estimation, we conducted experiments using a different number of cameras with the same FOVs. In this experiment, the FOV of each camera was $4\pi/6$ steradian on a viewing sphere. The cameras are positioned as symmetrically as possibly when covering the viewing sphere. Fig. 13 shows the results of two cases in which the number of cameras was increased; in one case, the total number of pixels in the system was increased, and in the other case, the number of pixels remained constant. Under 1-pixel Gaussian error in feature tracking, increasing the FOVs improved the performance of the motion estimation as shown in Fig. 13(a). Note that increasing the number of cameras from one to two resulted in significant improvement, and the system continued to show steady improvement as more cameras were added. This is expected because the two symmetrical cameras can resolve most of the translation-rotation ambiguity.

Fig. 13(b) shows the performance of motion estimation for multiple cameras in the case where the image resolution on the whole FOV remains same. For example, the image resolution of a single camera setting is six times higher than that of each camera in the six-camera setting. Introducing a second camera in this case also improved the performance a great deal. This



(a) Rotation (b) Translation direction

Figure 10: Error of estimated motions when increasing the distance to the scene points d_r while fixing the distance between camera centers T under a 1-pixel Gaussian feature tracking error. The "spherical" assumption dominates the other methods even for the system whose d_r/T ratio is small.



(a) Rotation (b) Translation direction

Figure 11: Minimum feature tracking errors under which "spherical" approximation outperforms the single camera SFM and the generalized 17-point algorithm given ratio between the distance to the scene points d_r and the distance between centers T.

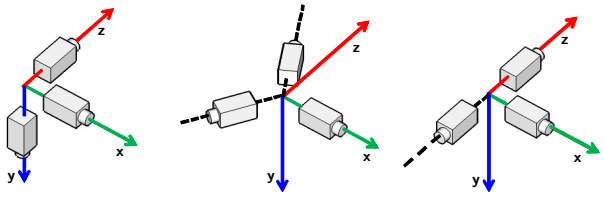
shows that increasing the FOV is more beneficial for motion estimation than increasing camera resolution.

One can notice that the generalized 17-point algorithm worked better with three cameras than with four cameras. This is because its performance depends on camera configuration even when the cameras have the same FOVs. We will study the performance under different configurations in the next section.

5.1.5. Camera configurations

When using multiple cameras, one should consider how to arrange the cameras in the rig. For motion estimation using a generalized camera model, Pless [1] provided a theoretical analysis based on the Fisher information matrix of various configurations of cameras. In this section, we showed the result of the performance test using different configurations of three cameras.

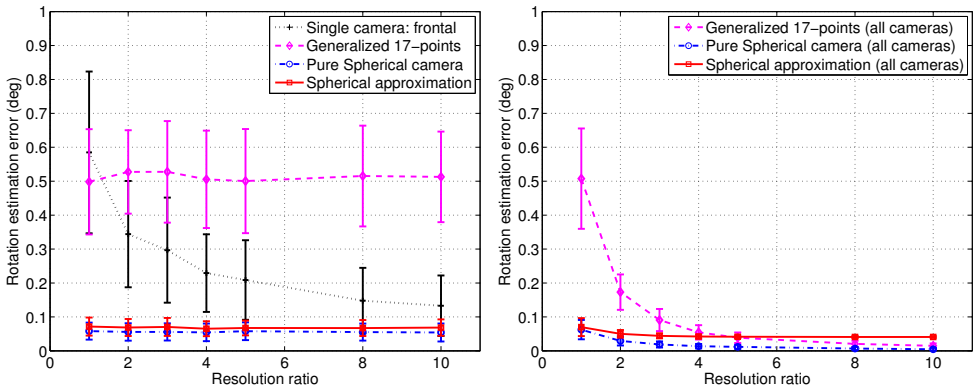
Fig. 14 shows three typical configurations of three cameras. In the first configuration, three cameras see in directions which are orthogonal to each other. Thus, the covered FOV is not uniformly distributed on a viewing sphere. In the second configuration, the centers of three cameras are on a plane and their viewing directions are 120° to each other, which is the most symmetrical setup achievable with three cameras. In the third configuration, all three cameras are on a plane, but see in orthogonal directions. The covered FOV remains the same in all three configurations.



(a) Config 1 (b) Config 2 (c) Config 3

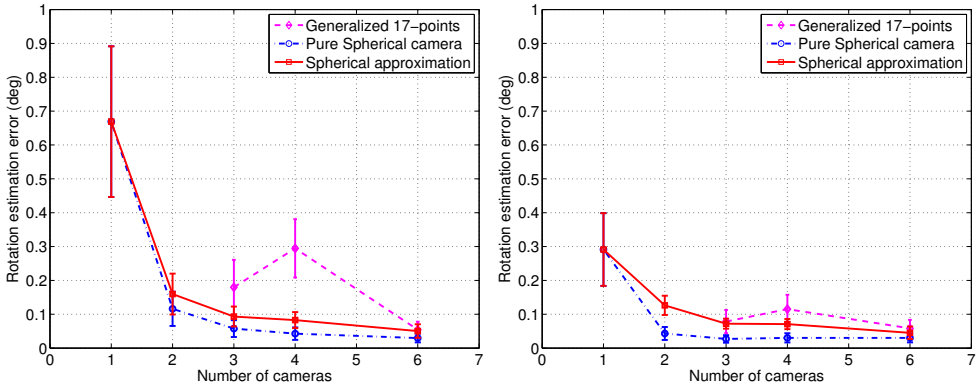
Figure 14: Three configurations of the three cameras used in this analysis; (a) Orthogonally aligned in an X-Y-Z axis, (b) Symmetrically aligned on an X-Z plane (the angle between each camera is 120°) and (c) Orthogonally aligned on the X-Z plane.

Fig. 15(a) shows the performance comparison between a single camera motion estimation and spherical approximation using the three configurations provided in Fig. 14. Though the third configuration works slightly better, there was no signif-



(a) Increasing resolution of one camera (b) Increasing resolution of all cameras

Figure 12: Effects of image resolution change (a) when the resolution of one of three cameras is increased, (b) when the resolution of all three cameras is increased. Note that the proposed method works better than single camera SFM with 10-times-higher camera resolution. The generalized 17-point algorithm does not improve when only one of the cameras has a high resolution, but it achieves better results when all three cameras have high-resolution. This analysis shows that enlarging the field of view using the spherical assumption is more important for estimating motions than increasing image resolution of a single camera.



(a) Constant resolution of each camera (b) Decreasing resolution of each camera

Figure 13: Effects of the field of view of a system. The FOV of each camera is $4\pi/6$ steradian, and thus six cameras can cover the whole viewing sphere. (a) When the resolution of each camera is the same, the number of pixels in the system increases by the number of the cameras. (b) When the resolution of each camera reduces with respect to the covered FOVs, the number of pixels in the system remains the same. In the case of a full FOV (using 6 cameras), the cameras in (a) and (b) have the same number of pixels.

icant difference of performances achieved by selecting a specific configuration. For comparison, we tested the generalized 17-point algorithm using the same camera configurations. As shown in Fig. 15(b), the overall performance of spherical approximation was better for all the configurations than this exact algorithm. One can notice that the performance of the generalized 17-point algorithm for the first configuration was much poorer than the others, while the spherical approximation method resulted in more stability for all configurations.

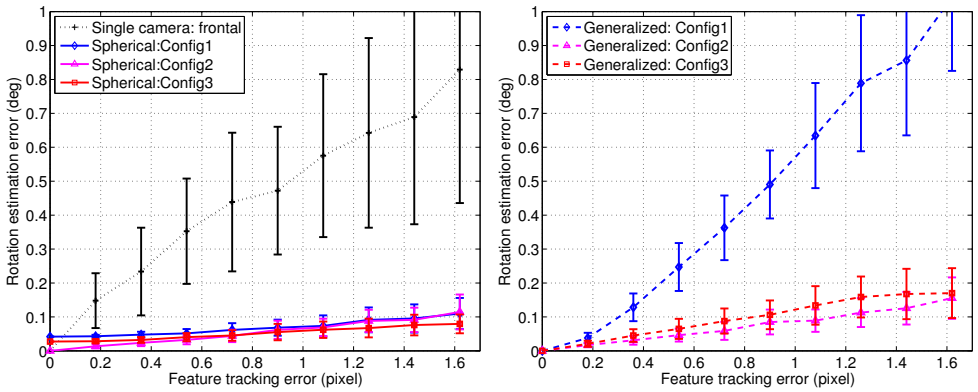
5.1.6. Erroneous rotations between cameras

Next, we analyzed the effect of the erroneous rotations between cameras. Fig. 16 shows the effects of errors of rotations between cameras for estimation of motions. In up to 2° of calibration error, spherical approximation performed better than the single-camera SFM and the generalized 17-point algorithm. With a larger error than 2°, the performance of spherical approximation got significantly worse. Interestingly, the generalized 17-point algorithm had much more stability under larger calibration error. From the analysis in Fig. 16, we can conclude that less than 2° of errors in rotation between cameras are sufficient for obtaining sufficient results using spherical approximation.

5.1.7. Errors in intrinsic parameters

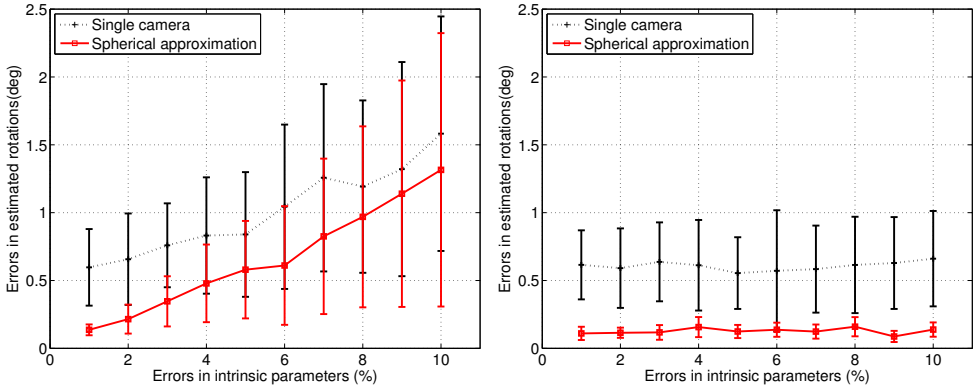
We tested the performance of motion estimation under erroneous intrinsic parameters of cameras. For this experiment, we added perturbation in all the intrinsic parameters for all cameras with features contaminated by 1-pixel Gaussian noise. In Fig. 17(a), the performance of spherical approximation was poorer than that of the single-camera SFM. This shows that accuracy of the intrinsic parameters is important in the spherical approximation method.

In Section 4.4, an algorithm used to compensate errors of the intrinsic parameters of cameras is presented. Fig. 17(b) shows the effects of error compensation of intrinsic parameters. Adopting the simple compensation of errors in intrinsic parameters improved the stability of motion estimation. Even though



(a) Spherical approximation (b) Generalized 17-point algorithm

Figure 15: Effects of configurations of multiple cameras; (a) Performance comparison between configurations using spherical approximation, and (b) using the generalized 17-point algorithm. The performance of motion estimation depends on both the camera configuration and the estimation method. (See text.)



(a) Before focal length compensation (b) After focal length compensation

Figure 17: Effects of errors in intrinsic parameters; Errors in estimated rotations (a) when the intrinsic parameters have errors, and (b) when the errors in focal lengths are compensated by the algorithm shown in Section 4.4. Before error compensation, spherical approximation is more sensitive than the single camera SFM to the errors of intrinsic parameters. The simple compensation makes it stable.

we used a simple model with only two parameters, the performance improved greatly. This shows that the accuracy of focal lengths is critical in motion estimation using spherical approximation.

In summary, the applicable conditions of spherical approximation identified in the experiments are given in Table 1.

Table 1: Summary of applicable conditions of spherical approximation.

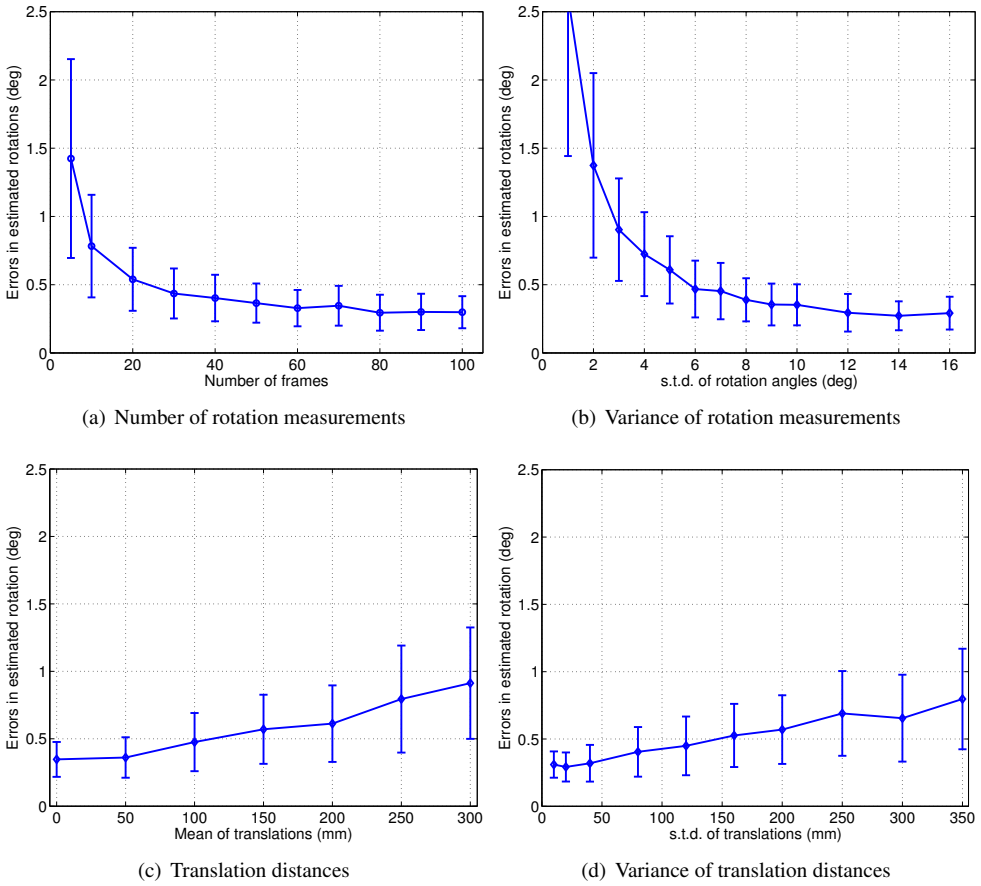
Feature tracking error	> 0.4 pixels in 300×300 images.
Distance to scene points	$> 10 \times$ distance between centers
Image resolution	Not sensitive. Spherical approximation works better than a single camera whose resolution is 10 times higher.
Covered field of view	Not sensitive when the number of camera is two or more.
Camera configuration	Not sensitive while the generalized 17-point algorithm is more sensitive.
Error in calibration	< 2 degrees in rotations. $< 4\%$ in intrinsic parameters.

5.2. Performance of the Proposed Online Calibration Algorithm

We analyzed the performance of the proposed calibration algorithm using simulated data. To generate the simulated data, we used the same configuration as in Section 5.1.

5.2.1. Simulation aspects

Because the proposed calibration algorithm uses the estimated motions from single-camera SFM, the rotation measurements would be more erroneous than those of the pattern based calibration method [18]. To find conditions under which calibration can be achieved up to the required accuracy shown in Section 5.1.6, we tested the algorithm in four aspects: the number of rotation measurements, the variance of rotation measurements, the distance of translations, and the variance of translation distances. In all cases, tracked features were perturbed with Gaussian noise of $\sigma = 1$ pixel.



(a) Number of rotation measurements

(b) Variance of rotation measurements

(c) Translation distances

(d) Variance of translation distances

Figure 18: Accuracy analysis of the proposed calibration algorithm: (a) under various numbers of rotation measurements, (b) under variance of the applied rotation measurements, (c) under various translation distances, and (d) under variance of the translation distances. In most cases, the estimation error is less than 2 degrees.

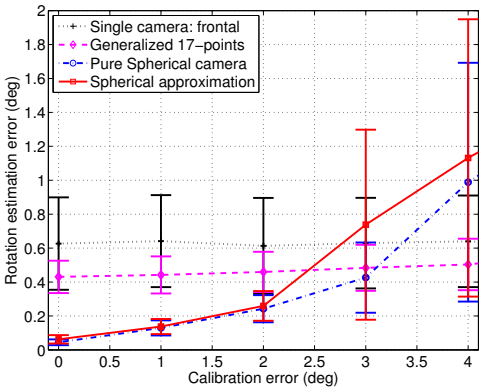


Figure 16: Accuracy analysis of the motion estimation algorithms under inaccurate rotation calibration between cameras. Results from a single camera and the generalized 17-points algorithm are given for comparison. In up to 2 degrees of calibration errors of the rotations between cameras, the spherical approximation outperforms the other methods.

5.2.2. *Number of rotation measurements*

To test the effect of the number of rotation measurements, we generated random motions of the camera rig. Each motion gave one rotation measurement for each camera. As in Fig. 18(a), the estimated rotations became stable using more than 40 motions. When analyzing the other aspects, we used 100 motions.

5.2.3. *Variance of rotations*

Fig. 18(b) shows the effect of variance of rotation measurements. In this experiment, we randomly rotated the rig using random translations with a deviation of 50 mm. When standard deviation of the rig rotations was larger than 10°, the proposed calibration worked well. The calibration errors increase, when the camera rig rotates in a constant speed.

5.2.4. *Mean of traveling distances*

Fig. 18(c) shows the effect of distances of translations. In this case, the rotation of the rig was randomly generated in the range of [-10,10]° and standard deviation of its translation was 50 mm with the given amount of translation in a fixed direction. If the rig moves in a constant speed in a fixed direction, the calibration errors increase.

5.2.5. *Variance of traveling distances*

Fig. 18(d) shows the effect of variance of rig translations. In this experiment, the mean of translations was set to zero. Larger translation produces more errors in rotation estimations between cameras because of the translation-rotation ambiguity. Rotations with smaller translations appear more dominant than those with larger translations.

Note that in every experiment, the estimation error of the rotation calibration was less than the required accuracy 2° of calibration given in Section 5.1.6.

5.3. Experiments using Real Images

In addition to the comparison experiment using real image sequences given in Fig. 3, we have performed experiments using longer sequences. In these experiments, we used the camera set in Fig. 1, attaching the set of three cameras on the top of a van. Driving the van through the Morewood Parking Lot in Carnegie Mellon University campus, we collected a set of synchronized input images shown in Fig. 2. The cameras saw in the front, left and right sides, so that the FOVs of the cameras were rarely overlapped. Because the cameras were placed higher than the parked cars, almost every feature was tracked in the lower half of the images. We used the affine-photometric KLT feature tracker [22] to find correspondences between frames. In each camera, about 200 features were tracked. The vehicle traveled about 400 m in 80 seconds.

By registering all the cameras in sequence, we recovered the whole trajectory as shown in Fig. 19. Throughout the whole sequence, we did not perform any optimization including local or global bundle adjustment. The spherical approximation method still generated a reasonably accurate trajectory of the camera set. As shown in Fig. 3, the generalized 17-point algorithm was too unstable to estimate the whole trajectory using only incremental update.

Fig. 20 shows the trajectory refined by sparse BA for the whole sequence. This refinement did not greatly improve the results, though some scene structures were refined. In this case, the intrinsic parameter refinement did not change the camera calibration at all, because the intrinsic parameters were sufficiently accurate.

To verify this result, we overlaid the recovered trajectory on a satellite image of the parking lot as shown in Fig. 21. Because there was no way to determine the absolute scale, we chose the scale manually. Fig. 21 illustrates that the whole trajectory was estimated well. The trajectory should not be closed because the starting and end locations denoted by circles in the figure were different.

In order to analyze effects of local bundle adjustment, let us examine a few parts of the trajectory more closely. The 'A' and 'B' parts in Fig. 21 were used to look into two typical motions, sharp turn and straight motion, respectively. When a vehicle turns sharply, camera systems with insufficient FOVs tend to fail in accurately estimating motions. Fig. 22 shows the estimated trajectory in the case of turning sharply. Black dots on the figure represent 3D positions of features from all three cameras. In this case, bundle adjustment barely changed the camera trajectory and the scene points, because the initial estimation was accurate. Fig. 23 shows the case of straight motion. The bundle adjustment refined both the motions and scene points in this case.

In the current implementation, the motion estimation algorithm runs in about 10 frames/sec on an Intel Zeon 3.0 GHz CPU. The major bottleneck in implementation is the RANSAC based robust algorithm for estimating motions between frames, and the speed depends on the inlier ratio of the tracked features, and efficient implementation of RANSAC.

6. CONCLUSION

One of the fundamental difficulties of single-camera SFM is translation-rotation ambiguity due to limited FOVs. To resolve this problem, it is beneficial to use multiple cameras to get a larger FOV as a whole. However, the discrepancy between camera centers makes motion estimation nonlinear and challenging. Though a linear algorithm using 17 point correspondences has been proposed, it is not sufficiently accurate or stable in outdoor applications. It is conceivable that if we can assume that all the cameras in an imaging system share a single optical center, then motion estimation and calibration for the system would become simpler and more efficient.

We derive the conditions under which spherical approximation is applicable both analytically and empirically. As expected, spherical approximation simplifies motion estimation of multiple non-overlapping cameras to a form of single camera SFM. In addition, it also simplifies the problem of finding relationships between cameras in the fixed rig.

We have analyzed the effect of approximation in various aspects such as errors in feature tracking, errors of intrinsic and extrinsic camera parameters, image resolutions, covered field of view, and camera configurations. Our experiments show that spherical approximation results in improvements in accuracy and stability compared to the other exact algorithms in some ranges of parameters and that the range is much larger than expected.

References

[1] R. Pless, "Using many cameras as one," in *IEEE Computer Vision and Pattern Recognition or CVPR*, 2003, pp. II: 587–593.

[2] H. Li, R. Hartley, and J. Kim, "A linear approach to motion estimation using generalized camera models," in *IEEE Computer Vision and Pattern Recognition or CVPR*, 2008, pp. 1–8.

[3] P. Baker, C. Fermuller, Y. Aloimonos, and R. Pless, "A spherical eye from multiple cameras (makes better models of the world)," in *IEEE Computer Vision and Pattern Recognition or CVPR*, 2001, pp. 576–583.

[4] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis, "Monocular visual odometry in urban environments using an omnidirectional camera," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, September 22-26, 2008, Acropolis Convention Center, Nice, France*, 2008, pp. 2531–2538.

[5] M. D. Grossberg and S. K. Nayar, "A general imaging model and a method for finding its parameters." in *International Conference on Computer Vision*, 2001, pp. 108–115.

[6] W. Chang and C. Chen, "Pose estimation for multiple camera systems," in *International Conference on Pattern Recognition*, 2004, pp. III: 262–265.

[7] C. Chen and W. Chang, "On pose recovery for generalized visual sensors," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 7, 2004, pp. 848–861.

[8] J. Frahm, K. Koser, and R. Koch, "Pose estimation for multi-camera systems," in *26th Symposium of the German Association for Pattern Recognition (DAGM)*, 2004, pp. 286–293.

[9] M. Kaess and F. Dellaert, "Visual SLAM with a multi-camera rig," Georgia Institute of Technology, Tech. Rep. GIT-GVU-06-06, February 2006.

[10] Y. Chen, L. Liou, Y. Hung, and C. Fuh, "Three-dimensional ego-motion estimation from motion fields observed with multiple cameras," *Pattern Recognition*, vol. 34, no. 8, 2001, pp. 1573–1583.

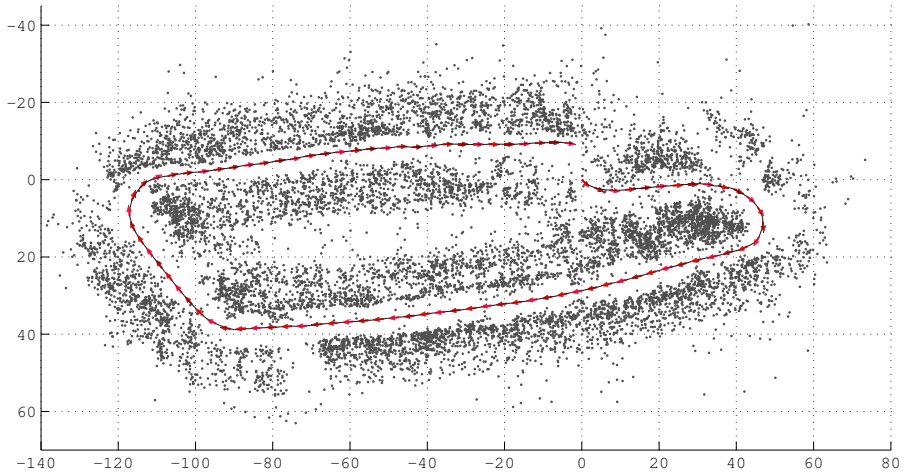


Figure 19: The spherical camera model used in the experimental data: the recovered camera trajectory and 3D feature points over a whole loop path

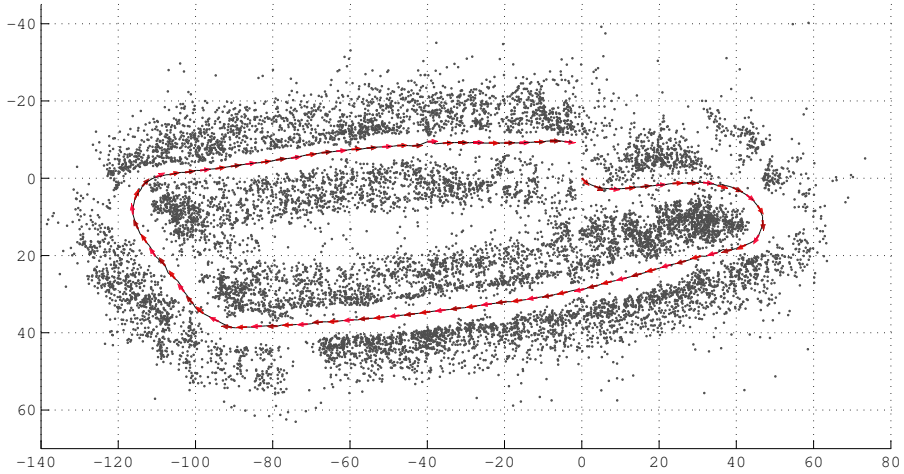


Figure 20: The spherical camera model on the experimental data: after the sparse bundle adjustment over a whole loop path

[11] J.-H. Kim, H. Li, and R. Hartley, "Motion estimation for non-overlapping multi-camera rigs: Linear algebraic and l∞ geometric solutions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, 2009, 1044–1059.

[12] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, 2003.

[13] D. Nister, "An efficient solution to the five-point relative pose problem," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, 2004, pp. 756–777.

[14] M. Lourakis, "levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++," [web page] `http://www.ics.forth.gr/~lourakis/levmar/`, Jul. 2004.

[15] M. Lourakis and A. Argyros, "The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm," Institute of Computer Science - FORTH, Heraklion, Crete, Greece, Tech. Rep. 340, Aug. 2004.

[16] P. Mendonca and R. Cipolla, "A simple technique for self-calibration," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 1, 1999, pp. 500–505.

[17] R. Kumar, A. Ilie, J. Frahm, and M. Pollefeys, "Simple calibration of non-overlapping cameras with a mirror," in *IEEE Computer Vision and Pattern Recognition or CVPR*, 2008, pp. 1–7.

[18] H. Kim, J.-S. Kim, and I. S. Kweon, "Motion estimation using centers of non-overlapping cameras," in *Proceedings of the 13th Japan-Korea Joint Workshop on Frontiers of Computer Vision*, 2007, pp. 387–393.

[19] S. Esquivel, F. Woelk, and R. Koch, "Calibration of a multi-camera rig from non-overlapping views," in *German Pattern Recognition Symposium or DAGM*, 2007, pp. 82–91.

[20] F. Park and B. Martin, "Robot sensor calibration: Solving $\mathsf{AX} = \mathsf{XB}$ on the euclidean group," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 5, 1994, pp. 717–721.

[21] T. Mukai and N. Ohnishi, "The recovery of object shape and camera motion using a sensing system with a video camera and a gyro sensor," in *International Conference on Computer Vision*, 1999, pp. 411–417.

[22] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, *An Invitation to 3-D Vision: From Images to Geometric Models*. SpringerVerlag, 2003.

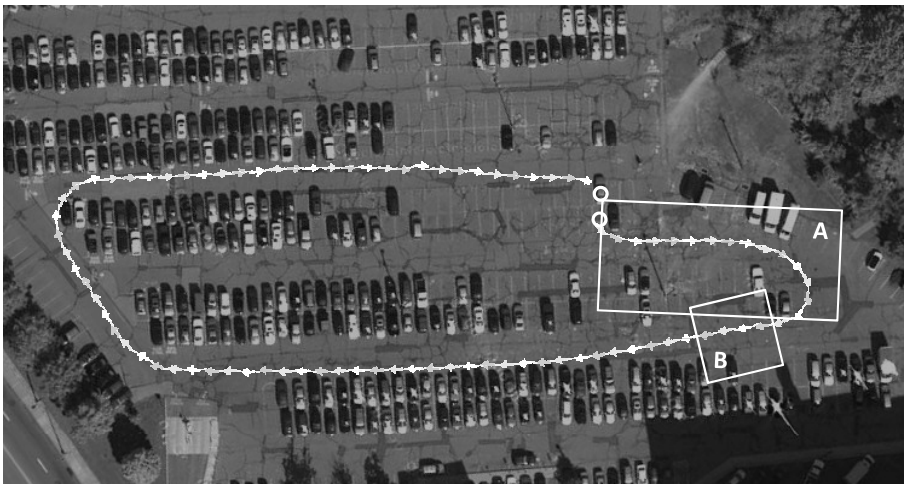
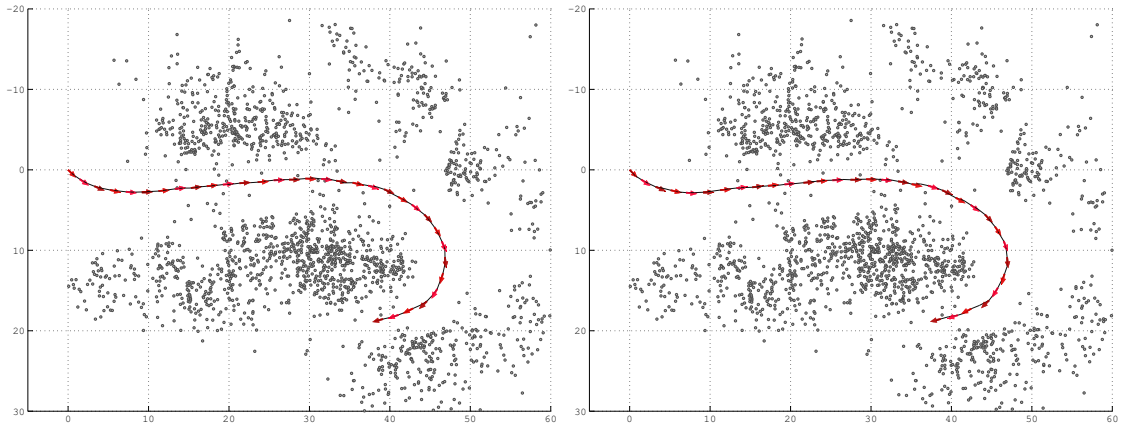
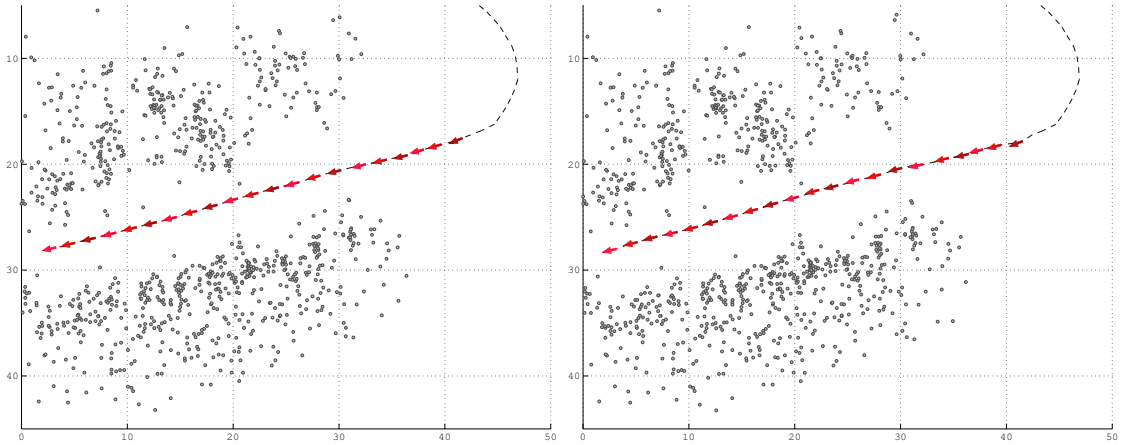


Figure 21: Estimated camera motions overlaid on a Google Map. The scale is selected manually. Two circles denote the start and end locations of the trajectory.



(a) Before BA (b) After BA

Figure 22: Estimated sharp-turn motions using spherical approximation (a) before bundle adjustment, and (b) after bundle adjustment. Black dots denote 3D positions of feature points. In this case, it is hard to find differences between them.



(a) Before BA (b) After BA

Figure 23: Estimated straight motions using spherical approximation (a) before bundle adjustment, and (b) after bundle adjustment. Black dots denote 3D positions of feature points.