Motion Estimation for Multi-Camera Systems using Global Optimization

Jae-Hak Kim, Hongdong Li, Richard Hartley

The Australian National University and NICTA*

{Jae-Hak.Kim, Hongdong.Li, Richard.Hartley}@anu.edu.au

Abstract

We present a motion estimation algorithm for multi-camera systems consisting of more than one calibrated camera securely attached on a moving object. So, they move all together, but do not require to have overlapping views across the cameras. The geometrically optimal solution of the motion for the multi-camera systems under L_∞ norm is provided in this paper using a global optimization technique which has been introduced recently in the computer vision research field. Taking advantage of an optimal estimate of the essential matrix through searching rotation space, we provide the optimal solution for translation by using linear programming and Branch & Bound algorithm. Synthetic and real data experiments are conducted, and they show more robust and improved performance than the previous methods.

1. Introduction

Relative pose estimation of a single camera in two views can be obtained from essential matrix estimation. Then, how about more than one camera? Motion estimation of multiple cameras has been studied in [9] as a generalized camera model. This problem is particularly relevant in the urban-mapping application where many non-overlapping cameras are attached to a vehicle and used for large-scale data capture. Many estimation algorithms for this problem have been developed in [2, 11, 8, 6]. However, most methods provide a solution for 5 DOF (Degrees of Freedom) of motion, three for rotation and two for translation, so the scale of the translation could not be recovered. In addition, their methods do not present a geometrically optimal solution to the problem, either.

In this paper, we would like to present a geometrically optimal L_∞ solution for 6 DOF motion for multi-camera systems from image point correspondences without any 3D point reconstruction. Hartley and Kahl recently showed that it is possible to find an optimal solution of the essential matrix for a single camera under L_∞ using a Branch and Bound algorithm, by searching for the optimal rotation over the rotation space [5]. Here we extend that algorithm to make it solve the 6 DOF motion for multiple cameras as well.

The method relies on the observation that if the rotation of the rigid multi-camera setup is known, then the optimal rotation may be found using Second-Order Cone Programming (SOCP), as shown in [6]. As in [5], we use a branch-and-bound search over rotation space to find the optimal rotation. This allows the optimal translation to be computed at the same time. Instead of using SOCP, we improve the speed of computation by using Linear Programming (LP) which speeds up the computation enormously. In addition, a preemptive feasibility test allows us to speed up the branch-and-bound computation. In our experiments, the LP method with the feasibility-test showed 90 times faster convergence of errors than the pure LP method.

Multi-Camera Systems. A multi-camera system is a set of cameras which are placed rigidly on a moving object with possibly non-overlapping views. Let us suppose that there are m cameras in the multi-camera system. We assume that the complete calibration of the camera system is known. The system of m cameras is moved rigidly and point correspondences are obtained between two points seen before and after the motion. Given this camera and motion configuration, we would like to estimate the 6 DOF motion, namely the rotation and translation with scale, of the multi-camera system. Examples of such multi-camera systems are a vehicle with multiple cameras mounted, an omnidirectional camera that consists of multiple pinhole cameras, and a generalized camera model [4].

For multi-camera systems, there is an algorithm to estimate motion of the multi-camera systems using SOCP [6]. In that paper, it is shown that the motion problem is the same as a triangulation problem, once the rotation is known. SOCP was applied to obtain an optimal solution for transla-

*This research was partly supported by NICTA, a research centre funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

tion in the multiple camera system. However, their method uses an unstable initial estimate of rotation which is extracted from an essential matrix from a single camera. Although, their method tries to obtain good initial estimates by averaging the selected rotations, the initial estimates come from each camera not from all cameras. Therefore, the rotation that they estimated from a single camera is still not an optimal solution for the whole system in terms of global optimization. Surely, it can be improved if we could estimate the initial rotation from all cameras.

In this paper, we introduce a way of using all cameras to estimate the motion – rotation and translation – from the optimal essential matrix for the multi-camera system.

2. Branch and Bound algorithm

As given by Hartley and Kahl in [5], the Branch and Bound algorithm for essential matrix estimation finds the optimal rotation by dividing the space of all rotation into several blocks and testing them one by one to find which one gives the best solution. Rotation space is represented as a 3-dimensional space using the angle-axis representation of a rotation. As the algorithm progresses, the blocks may need to be subdivided into smaller blocks in order to get a more accurate answer. Ultimately after a finite number of steps, one can find the optimal rotation, and hence translation within any required degree of accuracy.

The key to the branch-and-bound technique is a method of bounding the cost associated with the rotations within a block. Let $\hat{\mathtt{R}}_0$ be the rotation represented by the centre of a block in rotation space, and let r represent the maximum radius of the block (measured in radians). Since the translational part of the motion may be computed optimally (in L_∞ norm) once the rotation is known, we might find this optimal solution assuming the rotation $\hat{\mathtt{R}}_0$, and compute the best residual δ (namely the maximum reprojection error, also measured in radians) over all possible choices of translation. Now the key point is that for all other rotations $\mathtt{R}$ in the rotation block of radius r, the best residual is bounded below by $\delta + r$ (see [5]).

Now, suppose that $\delta_{\min}$ is a best residual found so far in the search, we ask the following question. Is it possible to find a solution with rotation assumed equal to $\hat{\mathtt{R}}_0$ that has residual less than $\delta_{\min} + r$. If the answer is no, it means that no rotation inside the current rotation block can beat the best residual $\delta_{\min}$. In this case, we do not need to consider the current block any further. If on the other hand the answer is yes, or possibly, then the result is inconclusive. In this case, we subdivide the rotation block by dividing into 8 subblocks, and keep them for future consideration. This method is guaranteed to find the optimal rotation, and hence translation within any desired bound within a finite number of steps.

The main computation in the method just described is, for each block we need to answer a feasibility question: is it possible with rotation $\hat{\mathtt{R}}_0$ to find a solution with residual less than $\epsilon = \delta_{\min} + r$? We will see that this feasibility problem can be answered very efficiently using LP.

This LP problem arises in the following way. It will be shown that each point correspondence (before and after the motion) must constrain the translation vector of the motion to lie in a wedge of space bounded by a pair of planes. The placement and angle of this wedge depends on the value of ϵ just defined. The feasibility problem has a positive answer if the set of all these wedges (one wedge for every point correspondence) has a common intersection. This is a standard LP problem, and may be solved quickly and efficiently.

3. Theory

We now give more details of the method given above. We assume a rotation $\hat{\mathtt{R}}$ is given, and our task is to find whether there exists a solution to the motion problem with residual less than a given value ϵ.

Single camera constraints. Let $\mathbf{x} \leftrightarrow \mathbf{x}'$ be a pair of matched points observed in one of the cameras. These represent direction vectors expressed in a coordinate frame attached to the camera rig. Knowing (or rather hypothesizing) the rotation, we may transform one of the vectors so that they are both in the same coordinate system. Therefore, define $\mathbf{v} = \hat{\mathtt{R}}\mathbf{x}$ and $\mathbf{v}' = \mathbf{x}'$. These two vectors and the translation vector must now satisfy the coplanarity condition $\mathbf{t}^\top(\mathbf{v} \times \mathbf{v}') = 0$ which specifies that the three vectors involved are coplanar. This obviously places a constraint on the vector $\mathbf{t}$.

However, we do not expect this constraint to be satisfied exactly for all point correspondences. Rather, we wish to know if it may be satisfied within a given error bound ϵ. A technical detail discussed in [5] allows us to specify different bounds ϵ and ϵ' on the two points. This is not necessary to follow the argument further, but we will assume that $\mathbf{v}$ and $\mathbf{v}'$ are allowed different error bounds ϵ and ϵ'. If we allow $\mathbf{v}$ and $\mathbf{v}'$ to be perturbed in this way, then this means they must lie inside cones of radius ϵ and ϵ' respectively as shown in Fig 1(a).

The translation direction $\mathbf{t}$ must lies inside a wedge bounded by planes tangent to the two cones. The two normals of these planes are shown in Fig 1(b). For several matched points, the translation direction must lie inside all such wedges.

To solve the feasibility problem, we need to express the normals to the planes in terms of $(\mathbf{v}, \epsilon)$, and $(\mathbf{v}', \epsilon')$. Then answering the feasibility problem is equivalent to solving the LP problem. We give the formulas for the normals below, without full details.

As shown in Fig 2, let us assume that angles α, β and ϵ

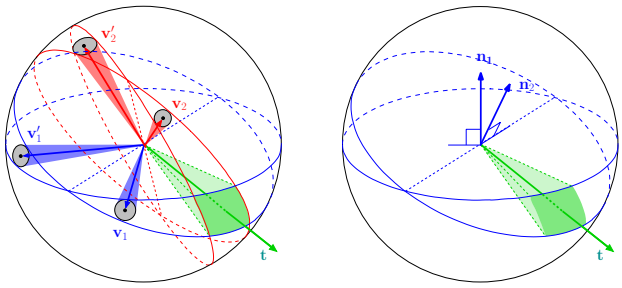


Figure 1. *(a) Translation direction* $\mathbf{t}$ *exists in a region of intersections (shaded as green) of half-spaces bounded by planes which are tangent to two cones having axes* $\mathbf{v}_i$ *and* $\mathbf{v}'_i$*. Two matched pairs of points* $\mathbf{v}_1 \leftrightarrow \mathbf{v}'_1$ *and* $\mathbf{v}_2 \leftrightarrow \mathbf{v}'_2$ *give the two intersections of two wedges. The intersection of the two wedges is a polyhedron containing the translation direction* $\mathbf{t}$*. (b) The two normals of the two half-spaces.*

are the angle between two axes of cones, the angle between bi-tangent planes and the cones, and radius error of matched points, respectively. Let $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ be vectors given by two cones $\mathbf{v}$ and $\mathbf{v}'$ as shown in Fig 2.

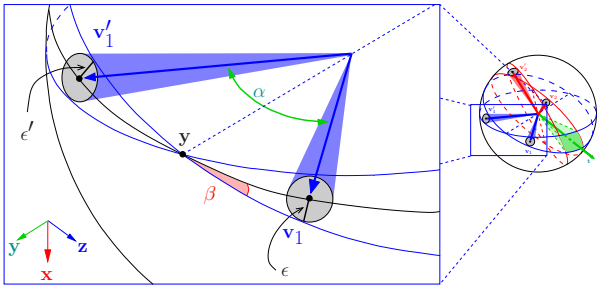


Figure 2. *The angle* β*, between the planes which are bi-tangent to two cones and the plane containing the axes* $\mathbf{v}_1$ *and* $\mathbf{v}'_1$ *of the two cones, is determined by the angle* α*,* ϵ *and* ϵ' *where* α *is the angle between* $\mathbf{v}_1$ *and* $\mathbf{v}'_1$*, and both* ϵ *and* ϵ' *are the angle errors at measured image point coordinates of matched points. The vectors* $\mathbf{x}$ *and* $\mathbf{z}$ *are given by* $\mathbf{v}_i \times \mathbf{v}'_i$*, and* $\mathbf{y} \times \mathbf{x}$*, respectively, and the vectors* $\mathbf{x}$*,* $\mathbf{y}$ *and* $\mathbf{z}$ *construct a basis of a coordinate system.*

The vectors $\mathbf{x}$ and $\mathbf{z}$ are determined by the axes of two cones $\mathbf{v}$ and $\mathbf{v}'$, and by the vector $\mathbf{y}$ where two great circles meet as shown in Fig 2. The vector $\mathbf{y}$ is derived as follows:

$$\mathbf{y} = \frac{\sin(\epsilon)\mathbf{v}' + \sin(\epsilon')\mathbf{v}}{\sin(\beta)\sin(\alpha)}\,, \tag{1}$$

where β is the angle between the planes bi-tangent to two cones and the plane containing the axes of the two cones as illustrated in Fig 2. This angle β is given by

$$\sin^2\beta = \frac{\sin^2(\epsilon) + 2\sin(\epsilon)\sin(\epsilon')\cos(\alpha) + \sin^2(\epsilon')}{\sin^2(\alpha)} \tag{2}$$

where α, ϵ and ϵ' are shown in Fig 2.

The vectors $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ form a basis for a coordinate system and serve to build equations of normals for the two half-spaces. From the work of [5], given a pair of matched cones on $\mathbf{v}_i \leftrightarrow \mathbf{v}'_i$, we derive the two normals $\mathbf{n}_1$ and $\mathbf{n}_2$ of half-spaces as follows:

$$\mathbf{n}_1 = \sin(\beta)\mathbf{z} + \cos(\beta)\mathbf{x} \tag{3}$$
$$\mathbf{n}_2 = \sin(\beta)\mathbf{z} - \cos(\beta)\mathbf{x}\,. \tag{4}$$

These equations provide two normals $\mathbf{n}_1$ and $\mathbf{n}_2$ for planes from a pair of matched points $\mathbf{x} \leftrightarrow \mathbf{x}'$, and eventually will be used to get an intersection of all half-spaces from all matched pair of points. This is an intersection from only one camera, and the existence of the intersection tells us whether a problem is feasible for the optimal essential matrix in one camera. In this paper, we would like to deal with multiple cameras instead of single camera to find the optimal rotation and translation.

Multiple cameras. We represent each camera by a sphere centred at the camera centre. Therefore, we have m spheres for an m-camera system. Associated with each sphere, as in Fig 1 there is a polyhedral cone with apex positioned at the centre of each camera, formed as the intersection of wedges defined by the point correspondences for that camera. These cones represent the direction of motion of each of the cameras. A correspondence of points in the k-th camera generates a constraint of the form

$$\mathbf{n}^\top(\mathbf{c}'_k - \mathbf{c}_k) \geq 0 \tag{5}$$

where $\mathbf{c}_k$ is the centre of k-th camera and $\mathbf{c}'_k$ is the center of k-th camera after the motion. The constraints from different cameras involve different variables, however. To get a set of consistent constraints, we need to transform these cones so that they constrain the final position of a specific chosen one of the cameras, let us say the final position $\mathbf{c}'_1$ of the first camera.

This problem is the same as the triangulation problem considered in [6]. We will see how the cones given by the linear constraints are transformed by the assumed rotation of the camera. This is illustrated in Fig 3.

To express (5) in terms of $\mathbf{c}'_1$ instead of $\mathbf{c}'_k$ we use the following relationship, which may be easily read from Fig 3.

$$\mathbf{c}'_1 = \mathbf{c}_k + \hat{\mathtt{R}}(\mathbf{c}_1 - \mathbf{c}_k) + (\mathbf{c}'_k - \mathbf{c}_k)$$

By substituting for $(\mathbf{c}'_k - \mathbf{c}_k)$ in (5), we obtain the inequality for multiple camera systems as follows:

$$\begin{aligned} 0 &\leq \mathbf{n}^\top(\mathbf{c}'_k - \mathbf{c}_k) \\ &= \mathbf{n}^\top(\mathbf{c}'_1 - \mathbf{c}_k - \hat{\mathtt{R}}(\mathbf{c}_1 - \mathbf{c}_k)) \\ &= \mathbf{n}^\top\mathbf{c}'_1 - \mathbf{n}^\top(\mathbf{c}_k + \hat{\mathtt{R}}(\mathbf{c}_1 - \mathbf{c}_k)) \end{aligned}$$

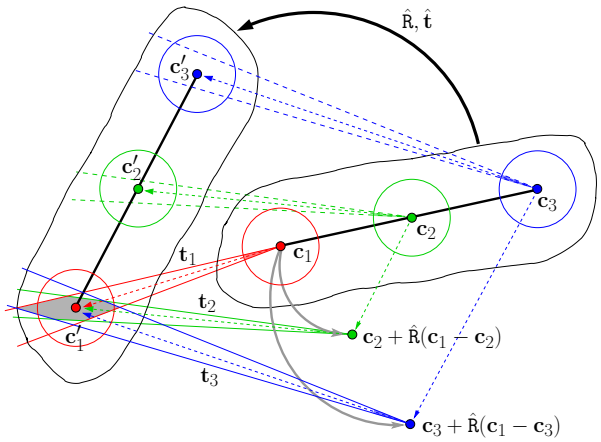


Figure 3. *The shaded region is the intersection of three polyhedra located on where each camera sees, $\mathbf{c}'_1$, the centre of the first camera after a rigid motion. The shaded region is a feasible solution of the translation of this multi-camera system.*

This is the specific inequality involving $\mathbf{c}'_1$ after the transformation. Finding a solution satisfying all these inequalities is same as finding an intersection of all half-spaces.

We find the centre of the first camera after the final motion by an intersection of all wedges defined by all pairs of matched points. In other words, we find a solution to a set of linear constraints by Linear Programming. More precisely, this feasibility problem is described as follows:

$$\begin{aligned} &\text{Does there exist} && \mathbf{c}'_1 \\ &\text{Satisfying} && \mathbf{n}_{i1}^\top \mathbf{c}'_1 - \mathbf{n}_{i1}^\top (\mathbf{c}_k + \hat{\mathrm{R}}(\mathbf{c}_1 - \mathbf{c}_k)) \geq 0 \\ & && \mathbf{n}_{i2}^\top \mathbf{c}'_1 - \mathbf{n}_{i2}^\top (\mathbf{c}_k + \hat{\mathrm{R}}(\mathbf{c}_1 - \mathbf{c}_k)) \geq 0 \\ &\text{for} && i = 1 \ldots N \end{aligned}$$

where $\mathbf{n}_{i1}$ and $\mathbf{n}_{i2}$ are the two normals derived from matched point i and k is the appropriate index of the camera generating the matched point i.

The feasible region is the region of space satisfying all these inequalities. In this problem, it is not important to know the entire polygon, but only to find one particular point of interest. Solving this feasibility problem tells us the position of the centre of the first at the final motion, and finally it gives us the optimal solution of translation direction vector and its scale value in multi-camera systems.

Feasibility Test. All half-spaces from matched pairs serve as inequalities in this LP problem. Given a total of N matched points in m cameras, the number of inequalities is $2N$. Generally, for 5 cameras with 100 points, LP requires to find an intersection of 1,000 half-spaces. If we use only LP to solve this problem, it will take too much computation time.

We introduce a way to reduce the time of computation for LP in this particular problem by testing the feasibility at an earlier stage before solving a full LP problem. The feasibility for multi-camera systems depends on the feasibility of single camera. If any feasibility observed for one single camera fails, then we do not need to look at feasibilities of other cameras. This observation gives a method to reduce the computation time greatly.

Testing a feasibility for a single camera is done by reducing the number of variables for the translation direction vector to two variables as shown in [5]. This feasibility test for a single camera can be adopted for greater speed of LP in multi-camera systems.

The order of matched points also affect the speed of the feasibility test. A larger angle α between two matched points leads to a narrower wedge in which the translation direction must lie, and gives more chance to finish the feasibility test earlier. Thus, these points should be tested first. In our experiments, using a preemptive feasibility test makes the algorithm 90 times faster than an algorithm without this feasibility test.

Degeneracy. It is important to note that if the motion from one frame to the next has no rotation, then the scale of the translation can not be computed. Because of the independence of the different cameras, there is an overall scale ambiguity, despite having known distances between the cameras. If the rotation is close to zero, the translation will be less reliable.

4. Algorithm

Given m calibrated cameras with a total of N matched points in each image, we can transform the matched points into vectors on the surface of a sphere by multiplying the inverse of the calibration matrix and the inverse of the rotation matrix of each camera. An example of these vectors is illustrated in Fig 6. With these simplified image vectors, the problem becomes easier to describe. The algorithm to find the optimal solution of motion of multi-camera systems is written as follows:

Algorithm *Optimal L_∞ Motion in Multi-Camera*
Input: Given m calibrated cameras with N matched points, $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$
Output: Estimated optimal rotation and translation with scale.

1. Obtain an initial estimate for the motion by any means (a random guess if necessary) and compute an initial estimate $\delta_{\min}$ for the minimal residual. Then carry out a branch-and-bound algorithm over rotation space, with the following steps.
2. Select a rotation block and consider its centre as an initial estimate of rotation $\hat{\mathrm{R}}$ in rotation space.

3. Multiply $\hat{\mathrm{R}}$ by $\mathbf{x}$ to get axes of two cones $\mathbf{v} = \hat{\mathrm{R}}\mathbf{x}$ and $\mathbf{v}' = \mathbf{x}'$.
4. Let $\epsilon = \delta_{\min} + r$, where r is the radius of the rotation block. Next determine whether there is a solution with rotation $\hat{\mathrm{R}}$ and residual less than ϵ by the following steps.
5. From the two cones about $\mathbf{v}$ and $\mathbf{v}'$ with half vertex-angle errors ϵ, compute two normals $\mathbf{n}_1$ and $\mathbf{n}_2$ from (3) and (4). Do this for all correspondences $\mathbf{v} \leftrightarrow \mathbf{v}'$.
6. Transform the two half-spaces to obtain inequality equations $\mathbf{n}_i^\top \mathbf{c}_1' - \mathbf{n}_i^\top (\mathbf{c}_k + \hat{\mathrm{R}}(\mathbf{c}_1 - \mathbf{c}_k)) \geq 0$
7. Solve Linear Programming with the constraints
8. If it is a feasible problem, then divide the selected rotation block into subblocks, and queue for further processing; otherwise discard the rotation block.
9. Repeat until we meet a desired error, then return the estimated rotation and translation

5. Experiments

Two experiments are conducted on synthetic and real data to show robustness and applications. A comparison with other method is presented to show improved accuracy of our proposed method.

5.1. Synthetic Data Experiments

A synthetic data set has four cameras with 50 image points randomly located in space. A total of 200 points are projected onto four image planes, and the system of four cameras is moved by a rigid motion of rotation and translation. The 200 points are also projected onto another four image planes of cameras at the final motion. When we process this synthetic data to estimate the motion by using our method, the CPU time of computation is about 3.5 seconds in a standard Intel Core 2 CPU PC based on 32-bit instructions and a single process. The implementation is written in C++ with GLPK (GNU Linear Programming Kit) [3]. As shown in Fig 4, several experiments are conducted 10 times on the same synthetic data by increasing noise parameters in pixels, and the distance error of centres is compared with the ground truth and its mean values are shown.

We have examined the performance comparison with another method [6], which we call "E+SOCP" in this paper, which uses a single essential matrix and SOCP to estimate the motion of multi-camera systems. As seen in Fig 5, our proposed method gives a better estimation for rotation and translation than E+SOCP.

5.2. Real Data Experiments

As a real example of multi-camera systems, we have used an omnidirectional camera, Point Grey's Ladybug™ [10], which consists of 6 cameras on the base unit. Its calibration information is provided by Point Grey. Six images are captured at each camera, and feature points on the images are extracted and tracked by the KLT tracker [7] through image sequences. Outliers in the tracked features are removed using RANSAC [1]. We transform these tracked features to image vectors on a sphere by multiplying the inverse calibration matrix and the inverse rotation matrix in each camera. The image vectors are shown in Fig 6. They are used in our algorithm to obtain the optimal solution of the rotation and translation in the 6-camera system of LadyBug™. Please note that we are not dealing with omnidirectional cameras but a multi-camera system.

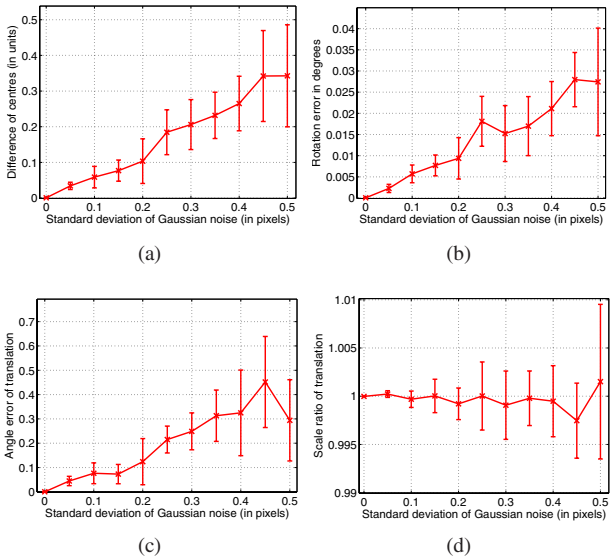


Figure 4. *Result of the synthetic data experiments. Normally distributed noise with standard deviation parameter σ is added to image coordinates in pixel units. (a) The angle difference between the estimated rotation and the true rotation of cameras, (b) The angle difference between the estimated translation direction and the true translation direction of cameras, (c) The distance between the estimated centres and the true centre of cameras and (d) the scale ratio between the estimated translation and the true translation are compared by varying noise parameters σ from 0 to 0.5 which means about 99.7% of the image points have errors from 0 to ± 1.5 pixels because of 3σ.*

5.2.1 First Real Data Set

The 6-camera system is moved on a piece of paper and the position is marked on the piece of paper. The motion of the 6-camera system, LadyBug, is a circular-like motion for 95 frames. We have selected key-frames every 5 frames from the image sequences. The estimated motion of the 6-camera system using our proposed method is shown in Fig 7 and Fig 8. The purpose of this experiment is to see how estimated motion is similar to the circular-like motion because the camera is moved randomly and the ground truth

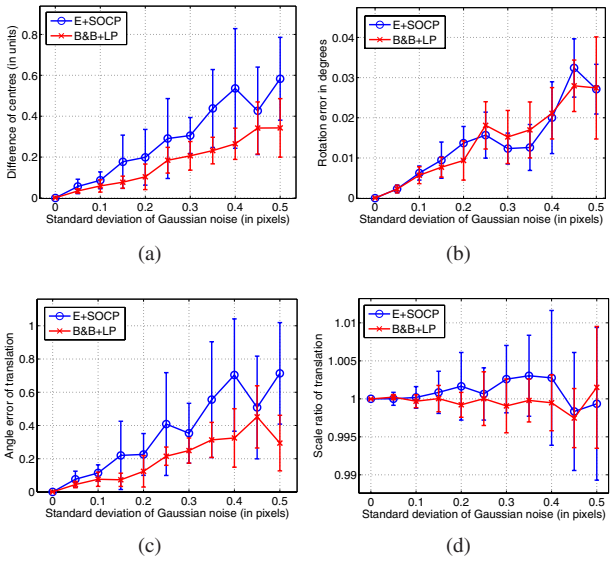


Figure 5. *Comparison of two methods which are the SOCP based on the single essential matrix method by [6]. (indicated as blue lines, "E+SOCP") and our proposed method based on Branch and Bound algorithm with LP (indicated as red lines, "B&B+LP"). (a) The difference between the true position of camera and the estimated position of the camera at the final motion. (b) Angle error of estimated rotation. (c) Angle error of estimated translation direction. (d) Scale error of estimated translation. The "B&B+LP" method gives more accurate position of camera though it has under-estimation of rotation and translation direction compared with the "E+SOCP" method. The difference of the errors is less than 1 degrees, so it is minimal. The less scale error of translation in the "B&B+LP" method shows why it estimates better position of cameras at the final position.*

for this motion is not measured. In the next experiment, we will look at how the motion is accurately estimated by locating the cameras at the pre-determined path.

5.2.2 Second Real Data Set

We have placed a piece of A2 size paper on top of a table, and the paper has 1mm grid. A trajectory of cameras is predetermined by drawing figures and marks on the paper. The LadyBug camera is moved manually to be aligned with the marked positions on the paper at every frame. The trajectory is shown in Fig 9. The images are captured for each camera at the mark positions. A total of 108 frames of image sequence are captured and matched point correspondences are found. The configuration of the camera setup is shown in Fig 10, and the images taken by the six cameras are shown in Fig 11.

Analysis of accuracy. Before we proceed with this particular "∞-shape" like motion of cameras, first, we would like to analyze how much pixel errors in images affect the accuracy of estimation for rotations and translations. For a better analysis and simulation of the experimental environment, we used the same data set which has all the measured trajectories of the Ladybug camera with rotations and translations, and we also used the camera calibration information of the Ladybug camera. With this measured ground truth and the Ladybug camera calibration information from the real experimental setup, the estimation of translations and rotations is simulated. The computed motion of cameras is

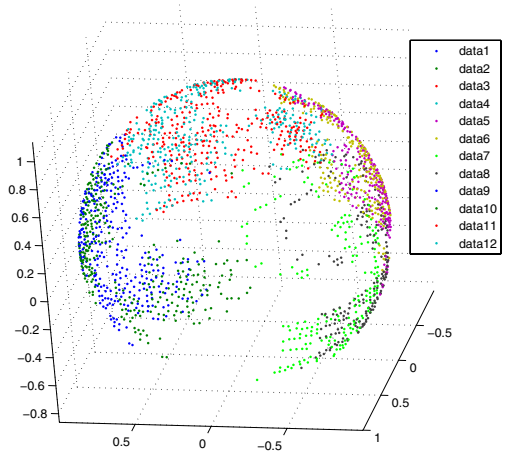


Figure 6. *Image vectors on a sphere from LadyBug^{TM} camera. These image vectors represent matched points which are transformed by the inverse of calibration matrix and the inverse of rotation matrix for our simplified model. Data 1 and 2 are from the first camera, data 3 and 4 are from the second camera, data 5 and 6 are from the third camera, and so on.*

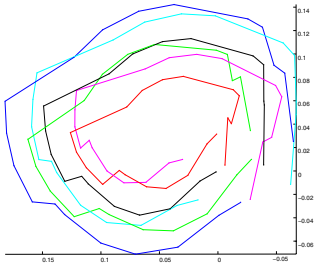


Figure 7. *Path of cameras from a top view. Each point in coloured lines represents the centre of six cameras in the system.*

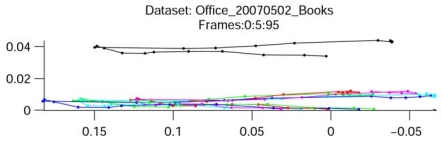


Figure 8. *Path of cameras from a side view. Each point in coloured lines represents the centre of six cameras in the system. Please note that a black coloured line is a camera on top of the base unit.*

Figure 11. *Six images captured at each camera of LadyBug. Five cameras (camera id 0 to 4) are placed to look horizontally view, and the last one (camera id 5) is located for a top view (From left to right order). There are only small overlapped fields of view across cameras.*

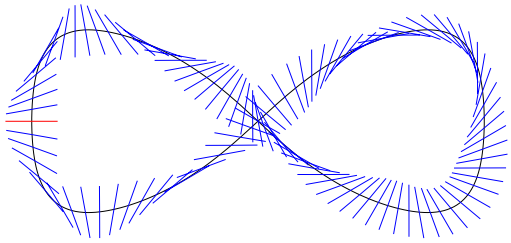


Figure 9. *The trajectory of marked positions for cameras is drawn on a piece of paper. The Ladybug camera is aligned with the blue line segments of the marked positions. A red line segment is a starting position of the camera.*

Figure 10. *Experiment setup. A LadyBug camera is placed on a piece of paper which has 1mm grids and it is surrounded by books. A trajectory of cameras is marked on the paper. Total 108 positions of ground truth are measured from the marked positions.*

shown in Fig 12.

Results. For 108 images, the motion of the 6-camera system is estimated and the results are shown and compared with the results of the "E+SOCP" method in Fig 15. The graph in Fig 15 shows that the estimated rotation and translation by our proposed method are more accurate than the estimated motion by the method uses SOCP with essential matrix from a single camera. The estimated trajectories of cameras are superimposed the ground truth of the measured trajectories of the cameras in Fig 13. Histograms of translation and rotation errors of the simulated motion are shown in Fig 14. These analysis shows that the translation direction is sensitive to noise on image coordinates. The estimated trajectories of the Ladybug camera and its consisting 6 cameras with the marker are shown in Fig 16. It shows the "∞-shape" path from the positions of the marker.

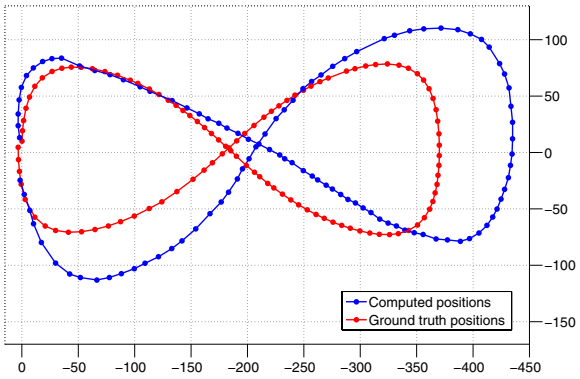


Figure 12. *Computed motion of cameras from synthetic data with the Ladybug camera calibration information and the ground truth positions. The computed motion is indicated as blue lines and the ground truth positions of cameras are drawn with red lines. The computed motion is generated with 0.1 standard deviation of the normal distribution for noises in image coordinates by pixel units. The overall scale of the computed motion is expanded compared with ground truth, perhaps largely due to the scale ambiguity caused by small rotation between frames. Nevertheless, note that the computed path is almost closes accurately. This suggests a systematic bias towards overestimating translation characteristic of Maximum Likelihood estimation.*

6. Conclusion

An optimal solution of motion for multi-camera systems under L_∞ norm is presented, and a feasibility test of Linear Programming for the multi-camera systems reduced the computation time of the problem significantly. The algorithm is optimal under L_∞ through all steps of the algorithm. Analysis of simulated motion showed that this algorithm is robust to estimate rotation angles and translation scale values (at least when the rotation is not too small) when there is noise in the image coordinates. However, we found that the estimate of the direction of translation is sensitive to the noise in the images.

Acknowledgement. We wish to thank the two anonymous reviewers for invaluable suggestions, and thank Eun

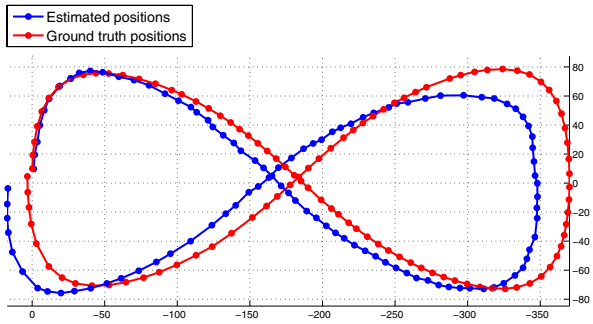
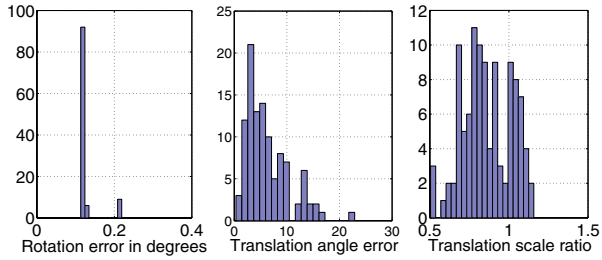


Figure 13. *Top view of the estimated trajectories of cameras and the ground truth of the cameras from frame 0 to 108. The estimated trajectories are indicated as red lines with dots on their positions of the cameras. The ground truth is illustrated as blue lines with its positions of the cameras. The starting position of the cameras is the left middle point which is (0, 0, 0) in the coordinates. There is a jittering or drift movement in the estimated motion because of accumulated errors over frames.*



(a) (b) (c)

Figure 14. *Histograms of rotation and translation errors on the simulated motion. The simulated motion is generated with 0.1 standard deviation of normal distribution as noises on the image coordinates. (a) Histogram of rotation errors. (b) Histogram of translation direction errors. (c) Histogram of translation scale errors. These shows the translation direction errors are sensitive to the noises.*

Young Kim for helping us to capture image sequences.

References

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

[2] J.-M. Frahm, K. Köser, and R. Koch. Pose estimation for Multi-Camera Systems. In *DAGM*, 2004.

[3] GNU Project. GNU Linear Programming Kit version 4.9. **http://www.gnu.org/software/glpk/**.

[4] M. D. Grossberg and S. K. Nayar. A general imaging model and a method for finding its parameters. In *iccv*, pages 108–115, 2001.

[5] R. Hartley and F. Kahl. Global optimization through searching rotation space and optimal estimation of the essential matrix. In *Proc. International Conference on Computer Vision*, Oct 2007.

[6] J.-H. Kim, R. Hartley, J.-M. Frahm, and M. Pollefeys. Visual odometry for non-overlapping views using second-order cone programming. In *8th Asian Conference on Computer Vision*, pages 353–362, 2007.

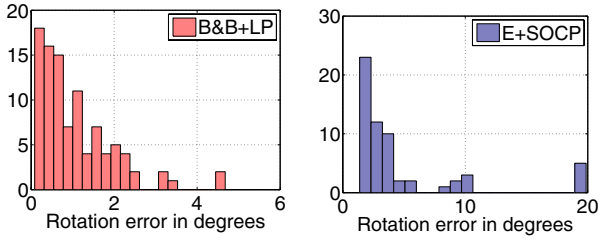
[7] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI81*, pages 674–679, 1981.

[8] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Generic and real time structure from motion. In *bmvc*, 2007.

[9] R. Pless. Using many cameras as one. In *CVPR03*, pages II: 587–593, 2003.

[10] Point Grey Research Incorporated. Ladybug™2 camera. **http://www.ptgrey.com**, 2006.

[11] H. Stewénius, D. Nistér, M. Oskarsson, and K. Åström. Solutions to minimal generalized relative pose problems. In *Workshop on Omnidirectional Vision*, Beijing China, Oct. 2005.



(a) (b)

Figure 15. *(a)Histogram of rotation error by our proposed method "B&B+LP" method. It shows 1.08 degrees of the mean and 0.83 degrees of the variance. (b)Histogram of rotation error by the "E+SOCP" method which is based on the essential matrix from single camera and SOCP by [6]. It shows 4.73 degrees of the mean and 25.61 degrees of the variance. The proposed "B&B+LP" method estimates the rotation better than the "E+SOCP" method in real data experiments.*

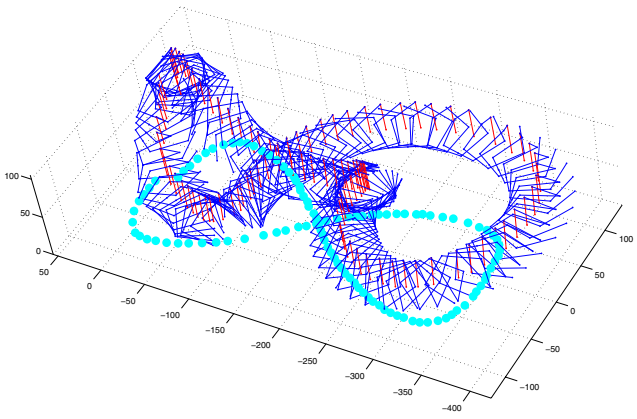


Figure 16. *The top-side view of the path of the 6 cameras (blue and red lines) and marker (cyan dots).*