

GLOBAL OPTIMIZATION FOR ONE-DIMENSIONAL STRUCTURE AND MOTION PROBLEMS

OLOF ENQVIST, FREDRIK KAHL, CARL OLSSON AND KALLE ÅSTRÖM*

Abstract. We study geometric reconstruction problems in one-dimensional retina vision. In such problems, the scene is modeled as a 2D plane, and the camera sensor produces 1D images of the scene. Our main contribution is an efficient method for computing the global optimum of the structure and motion problem with respect to the L_∞ norm of the reprojection errors.

One-dimensional cameras have proven useful in several applications, most prominently for autonomous vehicles where they are used to provide inexpensive and reliable navigational systems. Previous results on one-dimensional vision are limited to the classification and solving of minimal cases, bundle adjustment for finding local optima and linear algorithms for algebraic cost functions. In contrast, we present an approach for finding globally optimal solutions with respect to the L_∞ norm of the angular reprojection errors. We show how to solve intersection and resection problems as well as the problem of simultaneous localization and mapping (SLAM). The algorithm is robust to use when there are missing data which means that all points are not necessarily seen in all images. Our approach has been tested on a variety of different scenarios, both real and synthetic. The algorithm shows good performance for intersection, resection and for SLAM with up to five views. For more views the high dimension of the search space tends to give long running times. The experimental section also gives interesting examples showing that for one-dimensional cameras with limited field of view, the SLAM problem is often inherently ill-conditioned.

Key words. structure and motion, one-dimensional vision, SLAM, geometry

1. Introduction. One-dimensional cameras are used to provide inexpensive and reliable navigational systems for autonomous vehicles, see Figure 1.1. Strips of reflector tape are put on walls and objects along the route of the vehicle cf. [9]. A laser scanner then measures the direction from the vehicle to the different strips of tape, but not the distance. This is the one-dimensional camera. These angles can then be used to determine the position of the vehicle.

The use of one-dimensional cameras in navigation is the main motivation for our work. However, an understanding of the one-dimensional case can be of use with ordinary cameras as well. In the case of an ordinary camera moving and rotating in a plane, one-dimensional vision can be used as an efficient and accurate approximation (cf. [3]). The disadvantage of approximating is balanced against the possibility of obtaining globally optimal estimates (using the ideas we present in this article). In [1], a similar approximation is used for auto-calibration.

Another interesting application was given in [15], where it is shown that structure and motion problems using line features and an affine camera model can be reduced to the structure and motion problem in one-dimensional vision.

Apart from all this we are interested in the theoretical insights to gain from the study of one-dimensional vision and would claim that it has already yielded several ideas on how to handle similar problems in higher dimensions.

One of the key problems in one-dimensional as well as in ordinary vision is that of simultaneous localization and mapping (SLAM), also called structure and motion [8]. This is the problem of simultaneously estimating the positions of object points and cameras from a set of images. In the case of autonomous vehicles, this is normally done when the system is installed to create a map which can then be used for localization.

*Centre for Mathematical Sciences, Lund University, Box 118, 22100 Lund, Sweden. E-mail: olofe@maths.lth.se.



FIG. 1.1. An autonomous guided vehicle that uses one-dimensional vision to navigate.

High accuracy is needed, since the precision of the navigational system can never be higher than that of the map.

In this article we present a method for finding the globally optimal solution to this problem in the case of multiple views. The approach can also handle underdetermined problems, minimal cases and missing data. Using the same framework, we also show how to find optimal solutions to the intersection and resection problems. Similar ideas have been applied to compute optimal solutions for ordinary (2D) cameras in multiple view geometry, most notably in [11, 10, 14] and [6]. See also the survey paper [5]. A preliminary version of this work appeared in the conference paper [2].

The paper is organized as follows. In Section 2 we review a method for L_∞ optimization. Section 3 gives a brief introduction to the geometry of the problem and Section 4 discusses the problems of resection and intersection showing that they can be solved efficiently. An optimization method for the structure and motion problem is presented in Section 5 along with the required theoretic results. Finally, Section 6 presents some experiments illustrating the performance of the optimization method.

2. L_∞ Optimization . Many geometric problems in computer vision can be formulated as optimization problems. For example, consider the 3 spherical images of 7 object points in Figure 2.1. The objective is to estimate the structure (i.e., 2D object points) and motion (i.e., camera orientations and positions) given corresponding 1D image points.

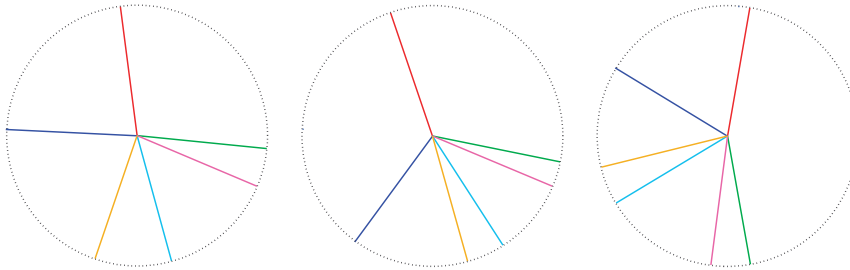


FIG. 2.1. An example of 3 spherical 1D images of 7 object points in the 2D plane.

Provided that there are no errors in the imaging process there would have been an exact solution to this problem (up to an arbitrary choice of coordinate system). In practice though the image coordinates of a given point are never exact. The deviation between measured coordinates and the *exact* coordinates - the measurement error - is

often assumed to be Gaussian noise. If so, the statistically optimal solution is given by minimizing the L_2 norm of the reprojection errors. Unfortunately it is very hard to find this solution. Existing techniques rely on a good initialization followed by local optimization and cannot guarantee global optimality.

To handle this problem L_∞ optimization was introduced in [7]. If we use the L_∞ norm of the reprojection errors instead of the L_2 norm, then it is much easier to find and verify the global optimum. The optimal structure and motion solution to the example images given in Figure 2.1 is illustrated in Figure 2.2, obtained with the algorithms developed in this paper. Further details of the computations are given in Section 6.1.

Before we continue, we will try to motivate the use of the L_∞ norm instead of the L_2 norm.

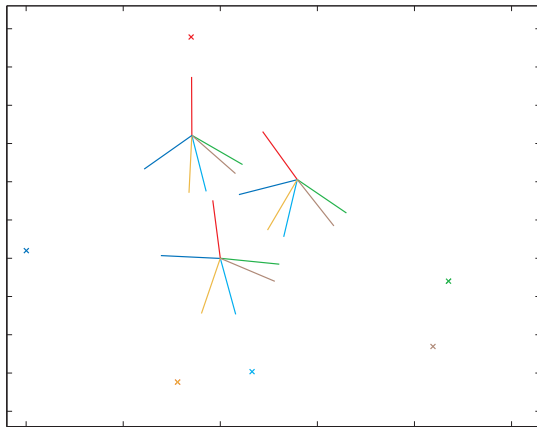


FIG. 2.2. The globally optimal solution in the L_∞ sense to the structure and motion problem for the images in Figure 2.1.

2.1. Motivation. We commented earlier that if errors in the imaging process are Gaussian, then the optimal solution with respect to the L_2 norm is also *statistically* optimal. What we mean is that it is the maximum likelihood solution. The strength of this norm is that a single point with a slightly larger error is not allowed to influence the solution that much. Thus, it has been said to be more robust to outliers than the L_∞ norm since the L_∞ norm solutions only depend on the measurements with largest errors. Note however that we are looking at the squared errors in the L_2 case, so the solution is still quite sensitive to outliers. We would argue that with or without outliers, a global optimization method with the L_∞ norm is better than a local one with the L_2 norm.

In the case of outliers both methods will yield rather poor solutions. In the L_2 case we have no way of determining whether the poor solution is due to outliers or this is simply because we have not yet found the global optimum. Using a global L_∞ approach we know that this is not the case. Thus we can try to find the outliers and remove them from our problem. Methods for removing outliers in L_∞ optimization already exist for a number of problems ([13, 12, 16]), and it is an active field of research.

In the case without outliers, the L_∞ norm has been shown to produce good solutions ([10, 11]) for ordinary (2D) cameras and we have the advantage of knowing

that we get the globally optimal solution.

2.2. Quasiconvexity. In [10, 11] it was shown that the optimization problems obtained for a number of multiview geometry problems using the L_∞ norm are examples of quasiconvex problems.

A function f is called quasiconvex if its sublevel sets $S_\Delta(f) = \{x; f(x) \leq \Delta\}$ are convex for all Δ . The reason for using the L_∞ norm when dealing with quasiconvex functions is that quasiconvexity is preserved under the max operation. That is, if $\epsilon_i(x)$, $i = 1, \dots, m$ are quasiconvex functions then $f(x) = \max_i \epsilon_i(x)$ is also a quasiconvex function. It was shown in [10, 11] that for a number of multiview geometry problems the (squared) reprojection errors are quasiconvex, and therefore the problem of minimizing the maximal reprojection error is a quasiconvex problem.

A useful property of quasiconvex functions is that checking whether there is an x such that $f(x) \leq \Delta$ is a convex feasibility problem and can usually be solved efficiently. This gives a natural algorithm for minimizing a quasiconvex function. Suppose we have bounds Δ_h and Δ_l such that $\Delta_l \leq \min_x f(x) \leq \Delta_h$ then the following bisection algorithm will solve $\min_x f(x)$, up to a predetermined precision threshold δ .

1. $\Delta = (\Delta_h + \Delta_l)/2$.
2. If there exists x fulfilling $f(x) \leq \Delta$ then $\Delta_h = \Delta$.
Otherwise $\Delta_l = \Delta$.
3. If $\Delta_h - \Delta_l > \delta$ return to 1. Otherwise quit.

3. One-Dimensional Retina Vision . We will now give a brief introduction to one-dimensional vision as used in autonomous vehicle navigation. A laser navigated vehicle is shown in Figure 1.1. Mounted on top of the vehicle is a laser scanner. A vertical laser beam generated in the scanner is deflected by a rotating mirror at the top of the scanner. When the laser beam hits a strip of retroreflective tape - a beacon - a large part of the light is reflected back to the scanner. The reflected light is processed to find sharp intensity changes. When this happens the bearing of the laser beam relative to a fixed direction in the scanner is stored. Note that only the bearing and not the distance to the beacon is measured.

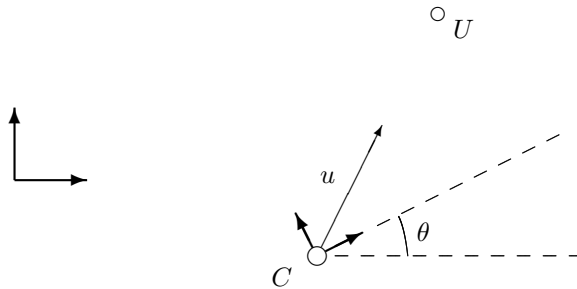


FIG. 3.1. The figure illustrates the measured bearing u (in a camera frame) as a function of scanner position C , scanner orientation θ and object position U .

Figure 3.1 shows the setup with one camera and one reflector. We introduce a world coordinate frame which will be held fixed with respect to the scene. The position of the camera in the world coordinate system is given by $C \in \mathbb{R}^2$ and the position of an object point (a beacon) in the same coordinate frame is given by U . We choose to represent the measured bearing of a beacon with a unit vector u . Note that this is given in a local camera frame. The relationship between this local frame

and the world coordinate frame is given by a rotation matrix R parameterized by a single angle θ .

In some cases it is convenient to identify the pair (R, C) with a camera matrix

$$P = \begin{pmatrix} a & -b & c \\ b & a & d \end{pmatrix} = k(R| -RC), \quad k > 0. \quad (3.1)$$

Note that every camera matrix of this form where $a \neq 0$ or $b \neq 0$ is associated with exactly one pair (R, C) . If we also allow solutions with $a = b = 0$, which corresponds to a camera position at infinity and will not yield any good solutions in practice, we can identify the set of all camera matrices with $\mathbb{R}^4 \setminus \{\mathbf{0}\}$. This will be useful when discussing quasiconvexity. Note that we do not allow the degenerate case when all elements of P are zero. When working with camera matrices we will use homogenous coordinates for the object (or scene) points, denoted by \bar{U} . Image points are represented by unit vectors u corresponding to the measured bearing. Note that the image vectors should not be regarded as a homogeneous quantity.

If there had been no errors whatsoever we would have

$$\lambda u = R(U - C) = P\bar{U}, \quad (3.2)$$

for some positive depth λ . Since we are mostly interested in overdetermined problems, (3.2) cannot be satisfied exactly. Instead we are forced to solve an optimization problem. Motivated by the previous section we choose to minimize the L_∞ norm of the reprojection errors. The reprojection error ϵ is the angle between the measured bearing and the modeled bearing. Let $\text{ang}(u, v)$ denote the angle between two vectors u and v , that is, $\arccos(u \cdot v)$. Then the reprojection error is given by

$$\epsilon = \text{ang}(u, R(U - C)) = \text{ang}(u, P\bar{U}). \quad (3.3)$$

4. Intersection and Resection. Before moving on to the general structure and motion problem, which is the main subject of this paper, we consider the simpler problems of intersection and resection.

Consider a number of cameras seeing the same object. If the positions and orientations of the cameras are known, the goal is to determine the position of the object. This is called the intersection problem.

PROBLEM 1. *Given bearings u_1, \dots, u_m of a single object from m different cameras P_1, \dots, P_m , the L_∞ intersection problem is to reconstruct the point U while minimizing*

$$f_{int}(U) = \max_i \text{ang}(u_i, P_i \bar{U}). \quad (4.1)$$

If instead the positions of a number of objects are known, then the goal is to determine the position and orientation of the camera seeing these objects. This is the resection problem.

PROBLEM 2. *Given n bearings u_1, \dots, u_n and the corresponding object points U_1, \dots, U_n the L_∞ resection problem is to find the camera matrix P such that*

$$f_{res}(P) = \max_j \text{ang}(u_j, P\bar{U}_j) \quad (4.2)$$

is minimized. These two problems are in a sense easy to solve. We shall see that both of them can be formulated as quasiconvex problems.

LEMMA 4.1. *For any $\Delta < \pi/2$ the sets*

$$\{U \mid f_{int}(U) < \Delta\} \subset \mathbb{R}^2 \quad \text{and} \quad \{P \mid f_{res}(P) < \Delta\} \subset \mathbb{R}^4$$

are convex.

Proof. First note that if $u \cdot P\bar{U} \leq 0$ then the angle between u and $P\bar{U}$ is at least $\pi/2$ or $P = \mathbf{0}$. Thus we can assume $u \cdot P\bar{U} > 0$.

For a given U , P and corresponding u we have

$$\left| \frac{u \times P\bar{U}}{u \cdot P\bar{U}} \right| = \tan \text{ang}(u, P\bar{U}), \quad (4.3)$$

where we define the cross-product for 2D vectors to be the scalar $u \times v = u_x v_y - v_x u_y$. Since $u \cdot P\bar{U} > 0$, checking whether $\text{ang}(u, P\bar{U}) \leq \Delta$ is equivalent to

$$|u \times P\bar{U}| \leq (u \cdot P\bar{U}) \tan \Delta. \quad (4.4)$$

In the intersection case, u and P are known and in the resection case, u and U are known. Hence these equations are linear in the unknowns and thus the constraints correspond to the intersection of half-planes and hence convex. Note that in the resection case, these sets do not include the degenerate $P = \mathbf{0}$. \square Note that if we use the bisection algorithm, this result also tells us that the feasibility problems can be put as linear programs.

5. Structure and Motion. In the next problem we will assume that neither the positions of the objects or the positions and orientations of the cameras are known. We will assume that the correspondence problem is solved, i.e., that it is known which measured bearings correspond to the same object. If the problem is deduced from ordinary vision this correspondence can be decided using features in the two-dimensional image. In case of one-dimensional cameras the correspondence can be estimated with a RANSAC-type algorithm [4].

To simplify notations we introduce bold-face letters, for example, \mathbf{R} to denote a set of rotation matrices $\mathbf{R} = (R_1, \dots, R_m)$.

PROBLEM 3. *Consider n different points visible in m cameras. Given the bearings u_{ij} of point j in camera i , the L_∞ structure and motion problem is to find the cameras (\mathbf{R}, \mathbf{C}) and the object positions \mathbf{U} that minimizes the maximal reprojection error,*

$$f(\mathbf{R}, \mathbf{C}, \mathbf{U}) = \max_{i,j} \text{ang}(u_{ij}, R_i(U_j - C_i)). \quad (5.1)$$

Unfortunately this problem does not have the nice properties of the intersection and resection problems in the previous section. The reason is that when both P and U are unknown (4.4) is normally not a convex condition. Nonetheless, quasiconvexity will play an important role for this problem as well.

The basic idea of our optimization scheme is to first consider optimization with fixed camera orientations, and then use branch and bound over the space of possible orientations. A problem here is that, especially with many cameras, the manifold of possible orientations is large. A method to reduce this manifold using linear conditions on the orientations is presented in Section 5.3.

5.1. Optimization with Fixed Orientations. In this section we prove that if we fix orientations in the structure and motion problem, we get a quasiconvex goal function.

DEFINITION 5.1. *We define the function*

$$d(\boldsymbol{\theta}) = \min_{\mathbf{C}, \mathbf{U}} f(\mathbf{R}(\boldsymbol{\theta}), \mathbf{C}, \mathbf{U}) \quad (5.2)$$

where $\mathbf{R}(\boldsymbol{\theta})$ are the rotation matrices corresponding to $\boldsymbol{\theta}$.

LEMMA 5.2. *For any $\Delta < \pi/2$ and a given $\boldsymbol{\theta}$, the problem of determining whether*

$$d(\boldsymbol{\theta}) \leq \Delta \quad (5.3)$$

can be cast as a linear programming feasibility problem.

Proof. Since the orientations are fixed we can without loss of generality assume that orientations have been corrected for and simply assume that $R_i = I$ for all $R_i \in \mathbf{R}$. Then, for $\epsilon = \text{ang}(u, P\bar{U})$,

$$\frac{u \times (U - C)}{u \cdot (U - C)} = \frac{|u||U - C| \sin \epsilon}{|u||U - C| \cos \epsilon} = \tan \epsilon.$$

The constraint that

$$\epsilon \leq \Delta$$

is equivalent to

$$|u \times (U - C)| \leq \tan \Delta (u \cdot (U - C)),$$

which constitutes two linear inequality constraints in the unknowns U and C . \square

This means that we can use linear programming to determine if the minimal L_∞ norm is less than some certain bound Δ . Moreover, using bisection we can get a good estimate of the minimal L_∞ norm.

To get better convergence we modify the normal bisection algorithm slightly. The idea is to seek a solution (C^*, U^*) to the problem in Lemma 5.2 that lies in the interior of the feasible space. For such a solution the L_∞ norm of the reprojection errors might be smaller than the current Δ , say Δ^* . Then one knows that the minimal L_∞ norm, $d(\boldsymbol{\theta})$ must be smaller than this Δ^* . To find such an interior solution we introduce a new variable k and try to maximize k under the constraints

$$|u \times (U - C)| + k \leq (u \cdot (U - C)) \tan \Delta.$$

We can now present an algorithm for finding the minimal L_∞ norm for fixed orientations $\boldsymbol{\theta}$.

1. Check if there is a feasible solution with all reprojected errors less than $\pi/2$. This corresponds to $\tan \Delta = \infty$ in the equations above. This can be solved by a simpler linear programming feasibility test. Use only $(u \cdot (U - C)) > 0$. If this is feasible then continue, otherwise return $d_{min} > \pi/2$.
2. Let $\Delta_l = 0$ and $\Delta_h = \pi/2$ be lower and upper bounds on the minimal L_∞ error norm.
3. Set $\Delta = (\Delta_h + \Delta_l)/2$. Examine whether $d(\boldsymbol{\theta}) \leq \Delta$. If this is the case calculate $\Delta^* = f(\mathbf{R}(\boldsymbol{\theta}), \mathbf{C}^*, \mathbf{U}^*)$ for the feasible solution and set $\Delta_h = \Delta^*$. Otherwise set $\Delta_l = \Delta$.
4. Iterate step 3 until $\Delta_h - \Delta_l$ is below a predefined threshold.

An example on how $d(\boldsymbol{\theta})$ might look is shown in Figure 6.2.

5.2. Lipschitz Continuity. To get further, we need an idea of how $d(\boldsymbol{\theta})$ depends on the camera orientations in $\boldsymbol{\theta}$. This is given by the following lemma.

LEMMA 5.3. *The function d satisfies*

$$d(\boldsymbol{\phi}) - d(\boldsymbol{\theta}) \leq \max_j |\phi_j - \theta_j| \quad (5.4)$$

which implies that it is Lipschitz continuous with Lipschitz constant 1.

Proof. The value of d is calculated as the minimum of $f(\mathbf{R}(\boldsymbol{\theta}), \mathbf{C}, \mathbf{U})$ over all \mathbf{C} and \mathbf{U} . Let $(\mathbf{C}^*, \mathbf{U}^*)$ be the minimizing camera positions and object coordinates for the orientations $\boldsymbol{\theta}$. Since (5.4) is equivalent to

$$\min_{\mathbf{C}, \mathbf{U}} f(\mathbf{R}(\boldsymbol{\phi}), \mathbf{C}, \mathbf{U}) \leq \min_{\mathbf{C}, \mathbf{U}} f(\mathbf{R}(\boldsymbol{\theta}), \mathbf{C}, \mathbf{U}) + \max_j |\phi_j - \theta_j|, \quad (5.5)$$

we see that, to prove that the (5.4) it is sufficient to find one pair (\mathbf{C}, \mathbf{U}) such that

$$f(\mathbf{R}(\boldsymbol{\phi}), \mathbf{C}, \mathbf{U}) \leq \min_{\mathbf{C}, \mathbf{U}} f(\mathbf{R}(\boldsymbol{\theta}), \mathbf{C}, \mathbf{U}) + \max_j |\phi_j - \theta_j|,$$

since the minimum $\min_{\mathbf{C}, \mathbf{U}} f(\mathbf{R}(\boldsymbol{\phi}), \mathbf{C}, \mathbf{U})$ is obviously smaller than the left hand side. We will now argue that $(\mathbf{C}^*, \mathbf{U}^*)$ satisfies this condition.

Considering an arbitrary reprojection error for $(\mathbf{C}^*, \mathbf{U}^*)$ but with orientations $\boldsymbol{\phi}$ instead of $\boldsymbol{\theta}$ it is easy to see that

$$\begin{aligned} & \text{ang}(u_{ij}, R(\phi_i)(U_j^* - C_i^*)) \\ &= \text{ang}(u_{ij}, (R(\theta_i) - R(\theta_i) + R(\phi_i))(U_j^* - C_i^*)) \\ &\leq \text{ang}(u_{ij}, R(\theta_i)(U_j^* - C_i^*)) + \text{ang}(R(\phi_i)(U_j^* - C_i^*), R(\theta_i)(U_j^* - C_i^*)) \\ &\leq \text{ang}(u_{ij}, R(\theta_i)(U_j^* - C_i^*)) + |\phi_i - \theta_i|. \end{aligned} \quad (5.6)$$

Thus the maximal reprojection error is bounded by

$$f(\mathbf{R}(\boldsymbol{\phi}), \mathbf{C}^*, \mathbf{U}^*) \leq d(\boldsymbol{\theta}) + \max_i |\phi_i - \theta_i|,$$

which proves inequality (5.4). \square

Using the fact that the function $d(\boldsymbol{\theta})$ can be evaluated and that it is Lipschitz continuous according to the above lemma, we will show how to solve globally for structure and motion. The basic idea is to perform branch and bound over the space of rotations. The Lipschitz property can be used to bound the function in the neighbourhood of a considered point. Before giving the details of the branch and bound algorithm, we will discuss how to limit the initial search space.

5.3. Initial Constraints on Orientations. When working with multiple views, the high dimension of the space of rotations can pose a problem in the branch and bound setting. This section shows how to derive simple constraints on the orientations, which can be used to limit the space we have to examine using branch and bound.

Consider two cameras, one with orientation $\theta = 0$ and position C and the other with orientation $\theta' = \phi$ and position C' . Let u and u' be the measured bearings of an object point in the two cameras. Then $C' - C$ must lie in the cone shown in Figure 5.1. This is stated in the following lemma.

LEMMA 5.4. *There exists $a \geq 0$ and $b \geq 0$ such that*

$$C' - C = a R(0)^T u - b R(\phi)^T u'. \quad (5.7)$$

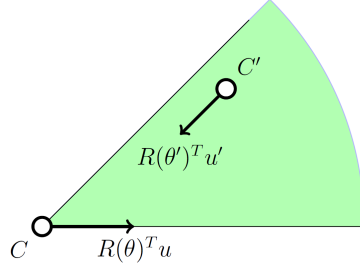


FIG. 5.1. For the beams to intersect the right camera has to lie in the green area.

Proof. Let U be the coordinates of the object point. Then, from the projection equation (3.2), we get

$$C + a R(0)^T u = U = C' + b R(\phi)^T u' \quad a, b \geq 0$$

and the result follows. \square

It is more efficient to work with angles rather than vectors. Hence, we define angles α , α' and c , such that

$$u = \begin{pmatrix} \cos \alpha \\ \sin \alpha \end{pmatrix} \quad \text{and} \quad \frac{C' - C}{|C' - C|} = \begin{pmatrix} \cos c \\ \sin c \end{pmatrix}. \quad (5.8)$$

Inserting in (5.7) and using the addition formulas for the sine and cosine we get

$$\begin{pmatrix} \cos c \\ \sin c \end{pmatrix} = a \begin{pmatrix} \cos \alpha \\ \sin \alpha \end{pmatrix} - b \begin{pmatrix} \cos(\alpha' + \phi) \\ \sin(\alpha' + \phi) \end{pmatrix}. \quad (5.9)$$

Let us first assume that $\phi = 0$ and $\alpha = 0$. Then we get the simplified relation

$$\begin{pmatrix} \cos c \\ \sin c \end{pmatrix} = a \begin{pmatrix} 1 \\ 0 \end{pmatrix} - b \begin{pmatrix} \cos \alpha' \\ \sin \alpha' \end{pmatrix} = a \begin{pmatrix} 1 \\ 0 \end{pmatrix} + b \begin{pmatrix} \cos(\alpha' - \pi) \\ \sin(\alpha' - \pi) \end{pmatrix}. \quad (5.10)$$

We get two cases. If α' can be chosen in $[\pi, 2\pi]$ we get the case on the left in Figure 5.2. The constraint that a and b be positive implies that

$$c \in [0, \alpha' - \pi]. \quad (5.11)$$

If instead α' can be chosen in $[0, \pi]$ we get the case on the right Figure 5.2 and

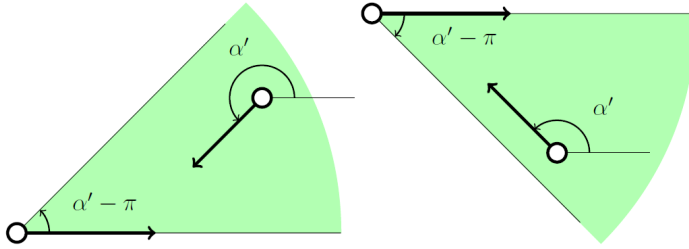
$$c \in [\alpha' - \pi, 0]. \quad (5.12)$$

In the general case, when α and ϕ are not zero, the same argument leads to

$$c \in [\alpha, \alpha' - \pi + \phi] \quad \text{or} \quad (5.13)$$

$$c \in [\alpha' - \pi + \phi, \alpha]. \quad (5.14)$$

Note that angle representations must be chosen such that interval lengths are positive but less than π . Also an angle β lies in an interval if for some $n \in \mathbb{Z}$, $\beta + n \cdot 2\pi$ does.

FIG. 5.2. *The two possible cases. Details are given in the text.*

For each point which is visible in two cameras, we get a constraint either of type (5.13) or (5.14). We introduce an index so that α_k represents the bearing of point k in the first camera and α'_k the bearing of the same point in the second camera. For a given ϕ to be feasible there must exist a c that satisfies (5.13) or (5.14) for every α_k . The following discussion will show how this can be used to produce bounds on ϕ .

First, note that (5.13) and (5.14) are linear in ϕ . Problems occur only when switching between the two types of intervals, (5.13) and (5.14). This happens when

$$\phi = \alpha_k - \alpha'_k \quad \text{or} \quad \phi = \alpha_k - \alpha'_k - \pi. \quad (5.15)$$

For each index k we get two such angles. Let

$$\{\phi^{(m)}\}, \quad m = 1 \dots M \quad (5.16)$$

be a sorted list of these angles. This divides the unit circle into M intervals. We will consider these intervals one by one. If

$$\phi \in [\phi^{(m)}, \phi^{(m+1)}], \quad (5.17)$$

then for each point we know if (5.13) or (5.14) is the relevant form of the constraint. Let K_1 be the set of the indices that generate constraints of type (5.13) and K_2 contain the indices that generate constraints of type (5.14). To determine if there is an angle c that satisfies all these constraints we need to check if the intersection

$$\left(\bigcap_{k \in K_1} [\alpha_k, \alpha'_k - \pi + \phi] \right) \cap \left(\bigcap_{k \in K_2} [\alpha'_k - \pi + \phi, \alpha_k] \right), \quad (5.18)$$

is non-empty. To make the problem cleaner we introduce variables ℓ_k for the lower interval limits and h_k for the upper limits. We get

$$\left(\bigcap_{k \in K_1} [\ell_k, h_k + \phi] \right) \cap \left(\bigcap_{k \in K_2} [\ell_k + \phi, h_k] \right). \quad (5.19)$$

Now, Algorithm 1 can be used to check for which ϕ this intersection is non-empty. Naturally, only those ϕ 's that also satisfy (5.17) are relevant. We repeat the algorithm for each of the M intervals defined by (5.16). Each interval yields different index sets K_1 and K_2 and thus different input to the algorithm.

ALGORITHM 1.

1. Choose one interval $[\ell_0 + \phi, h_0]$.
2. Adjust all angle representations such that

$$h_0 - 2\pi \leq \ell_k < h_0, \ell_k < h_k < \ell_k + 2\pi.$$
3. Let $L^a = \max_{k \in K_2} l_k, L^b = \max_{k \in K_1} l_k$
 $H^a = \min_{k \in K_2} h_k, H^b = \min_{k \in K_1} h_k.$

We have reduced the problem to

$$[L^a, H^a + \phi] \cap [L^b + \phi, H^b].$$

If $L^a > H^b$ or $L^b > H^a$ the problem is infeasible,

otherwise it is feasible for $L^a - H^a < \phi < H^b - L^b$.

Modifying this algorithm to handle an error tolerance is simple. In step 3 each interval is simply widened with the tolerance on each side. Note that when we have multiple views, we can use this method for any pair of cameras.

5.4. Branch and Bound Algorithm. In this section we present a practical algorithm to solve for structure and motion with one-dimensional cameras. The input to our algorithm is a set of m one-dimensional images. Each image is a set of bearings of object points. We assume that correspondences between the different images are known, but not that all object points are visible in all cameras.

As a consequence of Lemma 5.2, structure and motion estimation for fixed orientations is a quasiconvex problem and can be solved efficiently. Thus we propose a branch and bound algorithm over the space of orientations. With m cameras the space of orientations can be identified with $[0, 2\pi]^{m-1}$ since the first camera can be set to $R_1 = I$ by fixing the orientation of the world coordinate system. We represent the space of orientation with regions covering the whole set of feasible orientations.

To reduce this covering we first use the pairwise constraints of Section 5.3. Bounds $\theta_i - \theta_1 \in I_i$, where I_i is some interval, can be used to reduce the search space from $[0, 2\pi]^{m-1}$ to $\Pi_i I_i$. With omnidirectional data, which is the normal case in one-dimensional vision, this is often a considerable reduction.

After this initialization step we are left with a set of rectangular regions in the space of orientations. To discard such regions we use the Lipschitz continuity of the goal function (cf. Lemma 5.3) as follows. Assume that the best solution that we have found has L_∞ norm error d_{best} and that the largest side of the region we are considering to be w . Set the orientations to the centre of the region, denoted θ_c . Now assume that the goal function fulfills

$$d(\theta_c) > d_{best} + \frac{w}{2}. \quad (5.20)$$

Then according to the lemma, no orientations θ within the region can have $d(\theta) \leq d_{best}$ and thus we can discard this region. According to Lemma 5.2, checking (5.20) is a linear feasibility problem so it can be solved efficiently.

Finally we need a method to find better and better solutions and thus updating d_{best} . One method is to seek among the centres of the feasible regions. That is, when a region has passed (5.20) we also check whether $d(\theta_c) < d_{best}$. If this is true then we have found a better solution. The overall procedure is summarized in Algorithm 2.

ALGORITHM 2.

Starting with an upper bound on the optimal solution d_{best} .

Iterate until desired precision is reached:

1. *Pick a region from the queue.*
2. *Check feasibility using (5.20).*
3. *If the region cannot be discarded:*
 - *Divide the region and update the queue.*
 - *Try to update the lower bound on the optimum.*
6. *Remove the region from the queue.*

In the initialization we only used a subset of the constraints discussed in Section 5.3, namely those concerning $\theta_i - \theta_1 = \theta_i$. To further increase speed it is straightforward to use general constraints on $\theta_i - \theta_j$. In step 2 of Algorithm 2, we simply check feasibility using these bounds before using (5.20).

6. Experiments. In this section the developed theory is experimentally validated and the practical aspects of Algorithms 1 and 2 are investigated on both synthetic and real data. Further, we give examples of multiple local minima under the L_2 norm error function.

Our implementation is done in MATLAB using `linprog` for linear programming (LP) feasibility problems. For all the below experimental setups, one LP problem took around (or less) 0.05s to execute on a standard Pentium 2.8GHz processor.

6.1. Illustration of Typical Three View Problems: Synthetic Data . In a first example, study the problem of 3 views of 7 points with measured angles (in radians)

$$\alpha = \begin{pmatrix} 3.1 & -1.9 & -0.1 & -1.9 & -1.3 & 1.7 & -0.4 \\ -2.2 & -1.3 & -0.2 & -1.3 & -1 & 1.9 & -0.4 \\ 2.6 & -2.9 & -1.4 & -2.9 & -2.6 & 1.4 & -1.7 \end{pmatrix}, \quad (6.1)$$

where rows represent different views and columns represent different points, see also Figure 2.1 where the 3 images are plotted. The optimization problem consists of finding the camera orientations and positions as well as the 7 object points. In the branch-and-bound scheme, given by Algorithm 2, we can assume that the first camera orientation θ_1 is set to zero since we are free to choose the coordinate system.

Suppose that there exists a solution with $d_{best} \leq 0.05$ radians, that is, a solution with angular reprojection error no worse than 0.05 radians. We initialize the queue with a square region with a centre point at $(\theta_2, \theta_3) = (\pi, \pi)$ and width 2π , hence covering all possible orientations. In the first iteration of Algorithm 2, this region (naturally) passes the feasibility test for the bound $0.05 + \pi$ and consequently it is divided into smaller ones. This splitting leads to a quadtree structure. In the next two iterations of the algorithm there are 4 and 16 regions, respectively. None of these can be outruled. At the next level, 60 out of 64 regions of width $\pi/4$ can be outruled.

In Table 6.1, we summarize the first 10 iterations of the algorithm by describing (i) the number n_{sq} of feasible regions there are left at each level and (ii) how much area A out of the total area $A_{tot} = (2\pi)^2$ do these regions represent. After 10 iterations of the algorithm the optimal solution is bounded by $5.94 \leq \theta_2 \leq 5.98$ and $0.74 \leq \theta_3 \leq 0.86$.

Figure 6.1 illustrates the progress of the algorithm. Note that the the method quickly focuses in on the optimal solution of the problem and orientations far away from the optimum are discarded early in the iterative process. A plot of the goal function $d(\theta)$ is given in Figure 6.2. The optimal solution is plotted in Figure 2.2.

iteration	1	2	3	4	5	6	7	8	9	10
n_{sq}	4	16	4	8	20	28	40	68	104	92
$\log(A/A_{tot})$	0.0	0.0	-1.2	-1.5	-1.7	-2.2	-2.6	-3.0	-3.4	-4.1

TABLE 6.1

Progress of the branch and bound algorithm for the synthetic example in (6.1). See text for details.

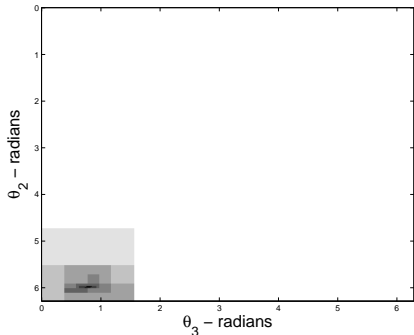


FIG. 6.1. The quadtree map of the goal function. White regions are discarded early in the branch and bound algorithm and darker areas are kept longer.

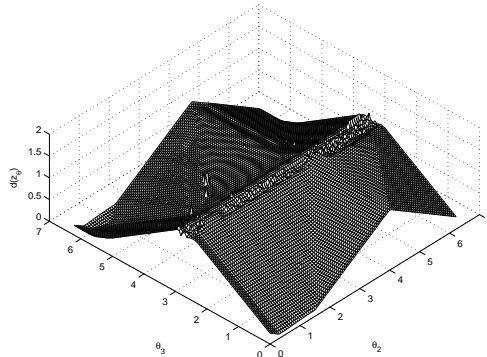


FIG. 6.2. The figure shows the goal function $d(\theta)$ as a function of θ_2 and θ_3 for the example with 3 views of 7 points in Section 6.1. Notice that the function is periodic.

See Figure 6.3 for illustrations of other typical random, synthetic three-view examples with varying number of points. In certain cases there may be several local optima and even in underconstrained cases (meaning less equations than unknowns) one can often locate the global optimum to a small region of parameter space.

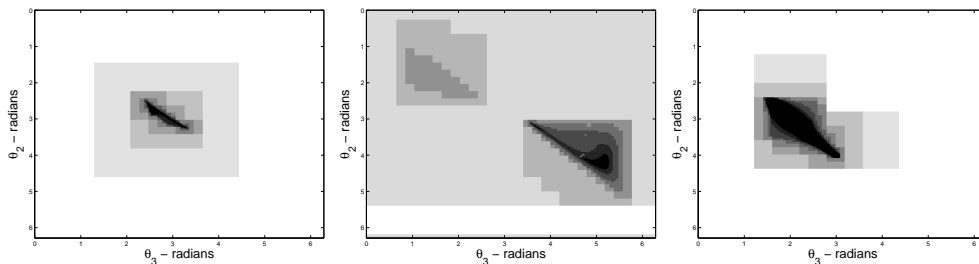


FIG. 6.3. Evolution of the quadtree map. See Figure 6.1 for explanation. Left: 4 points, 3 cameras (underconstrained). Middle: 5 points, 3 cameras with two local optima. Right: 6 points, 3 cameras (overconstrained). Note that even though the solution is underconstrained in the left example, one can locate the global optimum to a small region of parameters space.

6.2. Omnidirectional Cameras: Real Data. In the normal setup for one-dimensional vision, a rotating laser (as depicted in Figure 1.1) measures the angles between strips of reflective tape. This is an omnidirectional camera as measurements (or bearings) can be detected from a 360 degree field of view. All the experiments in this section are performed with such a sensor.

Our data were collected in a series of four experiments with varying number of object points and camera positions. The first three experiments were performed in a

single room with 5 reflective tapes on the walls of the room, with bearing measurements at 7, 8 and 21 different camera positions, respectively. In the fourth experiment, 14 reflective tapes were placed inside an ice hockey rink and the camera captured bearings to these points from 70 positions. The resolution of the angular meter is roughly 0.8 mrad.

There is no ground truth available for any of these experiments. However, as the same set of 5 reflective tapes are measured in the first three independent runs of the laser truck, the reconstruction of the scene geometry should be the same for all three experiments. This will be used to validate that the reconstructions are plausible.

To measure the uncertainty of the solution space, we use $(\frac{V}{V_0})^{1/n}$, where V is the remaining volume of rotation space, $V_0 = (2\pi)^n$ is the total volume and n the dimension. Note that the measure is normalized with respect to dimension and it can be regarded as the (normalized) geometric mean of the width of each dimension. In order to examine the running times, random subsets of 3, 4 and 5 camera positions of the ice hockey rink data were tested by measuring uncertainty as a function of the number of LP problems solved. The result is graphed in Figure 6.4. Note that already after a few hundred programs, a large portion of the rotation space can be ruled out. Also note that the execution times increase as the number of dimensions go up, as can be expected from a branch and bound scheme. The overhead for setting up the LP problems is negligible.

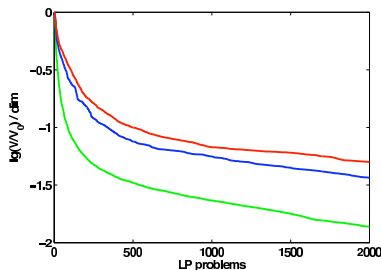


FIG. 6.4. The plot shows the uncertainty as a function of the number of linear programs for 3 views (green), 4 views (blue) and 5 views (red). The remaining volume is normalized with respect to the dimension of the space of rotations, i.e., $V^{1/n}$ is used as a measure of this uncertainty. The data are averages over respectively 25, 15 and 9 experiments.

In the first (second) experiment of the single room setup, there are $m = 7$ ($m = 8$) camera positions involved which means that one needs to perform branch and bound in $n = 6$ ($n = 7$) dimensions, respectively. The initial rotation constraints of Algorithm 1 reduce the potential rotation volume considerably to $(\frac{V}{V_0})^{1/6} = 0.17$ and $(\frac{V}{V_0})^{1/7} = 0.13$, respectively. Still, the rotational uncertainty is as much as 0.7 (0.7) radians for the most uncertain dimension.

After 5274 (6738) feasibility programs, there remain 2626 (4072) potential regions in the branch and bound queue, with relative volume $(\frac{V}{V_0})^{1/6} = 0.11$ and $(\frac{V}{V_0})^{1/6} = 0.10$, respectively, and a maximum width of 0.13 (0.13) radians for the most uncertain dimension of all rotation regions. The best solution found has an L_∞ error of 1.1 (1.3) mrad. Hence, even though solutions with low errors are obtained, it takes a long time to reduce the rotational uncertainty for such high dimensional problems. This is about the limit what is practically possible for branch and bound. The two computed solutions are plotted in the left and middle of Figure 6.5.

For the remaining two experiments with 21 and 70 camera positions we will have

to modify our strategy to a more practical approach. By combining optimal structure and motion with alternating resection and intersection, it is possible to solve for many cameras and object points in an efficient manner. The guaranteed global optimality is of course lost, but our experiments indicate that this strategy still gives very precise results.

For the third experiment with 21 camera positions viewing 5 object points we use the following two-step scheme:

1. We select the first 5 camera positions, and use Algorithm 1 for initialization (which gives rotational uncertainty $(\frac{V}{V_0})^{1/4} = 0.18$) followed by Algorithm 2 for computing a global solution (which results in rotational uncertainty $(\frac{V}{V_0})^{1/4} = 0.05$ after 2620 LP programs). The L_∞ error is 1.5 mrad for the partial structure and motion solution.
2. Then, since all 5 object points are reconstructed, we can perform optimal resection on the remaining 16 camera positions. The computed solution can be iteratively improved by alternating optimal intersection and optimal resection. This results in a solution with L_∞ error 2.6 mrad. The iterative alternation scheme (typically) converges in just a few iterations.

The three structure and motion solutions of the single room experiment are plotted in Figure 6.5. Note that the 2D object points have been identically reconstructed (modulo a choice of coordinate system) in all three experiments.

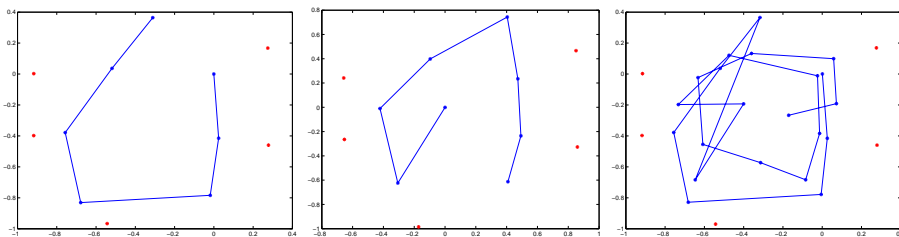


FIG. 6.5. Structure and motion solutions obtained from measurements of 5 points in 7 cameras (left), 8 cameras (middle) and 21 cameras (right), respectively.

As a final omnidirectional experiment, with measurements from an ice hockey rink of 70 images of 14 points, we solve the structure and motion problem using the same two-step procedure as above. The result is shown in Figure 6.6 and the solution has a L_∞ error of 5.2 mrad.

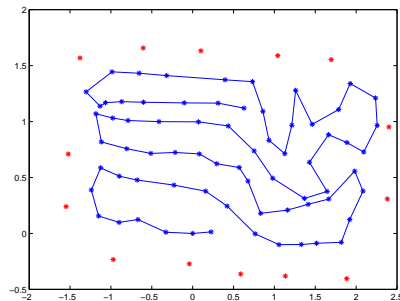


FIG. 6.6. Structure and motion for the ice hockey experiment calculated by solving a series of resection and intersection problems.

6.3. Cameras with Limited Field of View: Real Data. Pinhole cameras can only view objects in front of the camera and thus have a limited field of view. For one-dimensional cameras having a limited field of view, finding the optimal solution to structure and motion turns out to be computationally much more demanding than for omnidirectional cameras. The results in this section are based on real data measurements, but we have also validated that similar conclusions can be drawn based on (random) synthetic data with limited field of view.

The setup for the experiment is as follows. First we placed 10 black-and-white markers in the scene. Then a camera was mounted on a trolley and moved around to create a planar motion. Figure 6.7 shows five of the captured images. The blue stars show the detected markers in each image. Since the camera y -axis is not perfectly aligned with the upward direction in the images we had to reestimate the coordinate system from objects in the image that can be assumed to be aligned with the upward direction, such as the door frame and other lines parallel to it. The green lines show the estimated coordinate system and the red stars show the projections of the computed bearings. The field of view is less than 45 degrees (that is, the angle difference between maximum and minimum measurements).

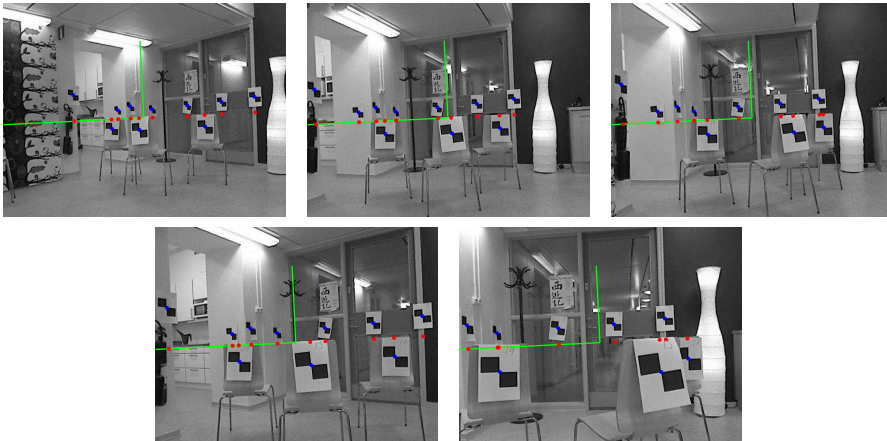


FIG. 6.7. Setup for the experiment with planar moving cameras. The blue stars show the location of the detected markers, the green lines show the estimates of the x and y axes and the red stars show the calculated bearings of the markers. Note that not all markers are visible in all views.

In order to examine the execution times, similarly to what was done in the omnidirectional case, random subsets of 3 cameras were selected and the rotation uncertainty was studied as a function of LP problems solved. Figure 6.8 shows how Algorithm 2 *typically* works for one such example compared to a typical omnidirectional example. In all examples we have encountered, the convergence of the branch and bound algorithm is slow. Not surprisingly, Algorithm 1 fails to substantially reduce the initial rotational uncertainty. For example, applying Algorithm 1 to the 5 images of 10 points described above yields $(\frac{V}{V_0})^{1/4} = 0.70$. After 3050 LP programs, the uncertainty has decreased to $(\frac{V}{V_0})^{1/4} = 0.57$ for the same instance. The best solution found at this point has L_∞ error 309 mrad. Apparently the optimization takes a lot longer time. Alternating optimal resection and optimal intersection improves the solution and a local optimal solution is obtained. The L_∞ error is 0.54 mrad.

To see if this difficulty is somehow inherent in the problem, we examined the slope

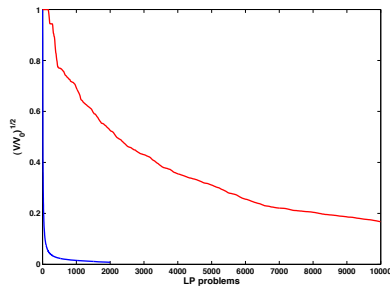


FIG. 6.8. Remaining uncertainty as a function of the number of linear programming problems solved. The plot shows the average uncertainty per dimension as a fraction of the original uncertainty. The blue curve relates to a typical omnidirectional example, whereas the red curve comes from a camera having a limited field of view.

of the goal function. As the branch and bound algorithm fails to efficiently discard regions, the goal function is flat around the optimal solution. Hence, there is a large neighbourhood of solutions that give almost the same reprojection error. Figure 6.9 shows two such solutions from one problem instance with 3 cameras viewing the 10 markers. Though the two solutions are very different the L_∞ errors are less than 2 mrad for both. In fact there is a whole continuum of low error solutions between these two. This means that the problem is very unstable. The behaviour is common in our data. Note especially that this example is not close to the minimal problem of five points in three views. Adding more views will reduce the ambiguity in determining the solution, but it is still present for smaller number of views (less than 5). This observation explains the poor performance of using narrow field of view cameras (cf. Figure 6.8).

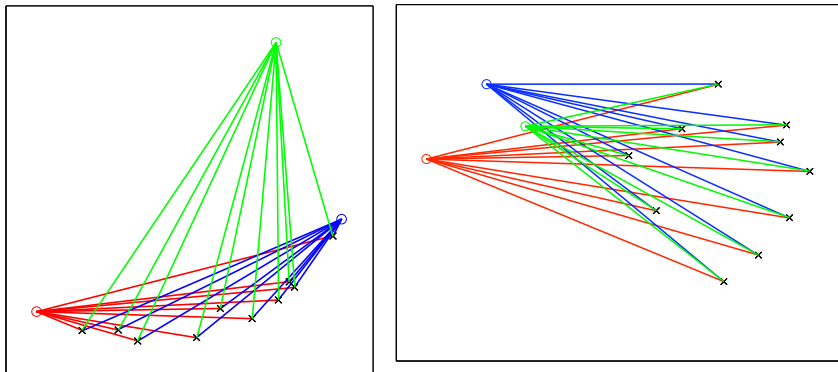


FIG. 6.9. The plots show two solutions for the same three view problem. Both solutions have a L_∞ error of less than 2 mrad.

Do local minima occur? We conclude with an experiment that shows that local minima may occur, and in some cases even quite frequently.

Using MATLAB's built in function `fmincon` we tried to minimize the L_2 norm error of the angles for the 5 images of 10 markers. We found 16 local minima in total from random initializations. However many of them can easily be discarded since one or two markers are far away (towards infinity) which is unreasonable in this setting. Figure 6.10 shows five of the detected minima, and Table 6.2 shows the corresponding

values of the goal function. The first minimum is the one with lowest value out of the 16 cases. The solution computed above with the L_∞ approach yielded an error of 0.54 mrad, and the reconstructed structure and motion is almost identical (modulo coordinate system) to the best L_2 solution, plotted in the top left of Figure 6.10.

local minima:	1	2	3	4	5
L_2 -error:	0.0013	0.0832	0.2270	0.1907	0.2145

TABLE 6.2

The L_2 angular error (in radians) for the minima shown in Figure 6.10.

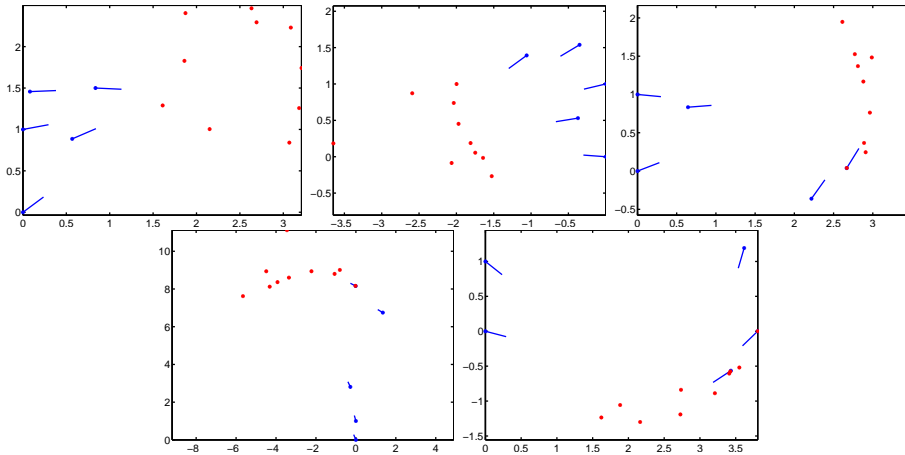


FIG. 6.10. Five detected local minima. Note that some of the reconstructions appear to have negative depths. This is because not all of the markers are visible in all views.

7. Conclusions. In this paper we have studied the problem of finding global minima to the structure and motion problem (SLAM, surveying) for one-dimensional retina cameras using the L_∞ norm on reprojected angular errors. We have shown how the problem of known camera orientations can be reduced to a series of linear programming feasibility tests. We have also shown that the L_∞ goal function as a function of orientation variables has slope less than one. This Lipschitz property gives a way to efficiently search the orientation space for the optimal solution using branch and bound, resulting in a globally optimal algorithm with good empirical performance.

Apart from algorithmic developments for the one-dimensional structure and motion problem, we have also experimentally showed that the goal function is very flat for small field of view cameras. In particular, this means that there may exist many solutions with similar reprojection errors. Hence, such problems are ill-posed and computationally hard to solve.

Acknowledgments. This work has been funded by the Swedish Research Council (grant no. 2007-6476), by the Swedish Foundation for Strategic Research (SSF) through the programme Future Research Leaders, and by the European Research Council (GlobalVision grant no. 209480).

REFERENCES

- [1] M. ARMSTRONG, A. ZISSERMAN, AND R. HARTLEY, *Self-calibration from image triplets*, in European Conf. Computer Vision, Cambridge, UK, 1996, pp. 3–16.
- [2] K. ÅSTRÖM, O. ENQVIST, C. OLSSON, F. KAHL, AND R. HARTLEY, *An L_∞ approach to structure and motion problems in 1d-vision*, in Int. Conf. Computer Vision, Rio de Janeiro, Brazil, 2007.
- [3] O. D. FAUGERAS, L. QUAN, AND P. STURM, *Self-calibration of a 1d projective camera and its application to the self-calibration of a 2d projective camera*, in European Conf. Computer Vision, vol. I, Freiburg, Germany, 1998, pp. 36–52.
- [4] M. A. FISCHLER AND R. C. BOLLES, *Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography*, Commun. Assoc. Comp. Mach., 24 (1981), pp. 381–395.
- [5] R. HARTLEY AND F. KAHL, *Optimal algorithms in multiview geometry*, in Asian Conf. Computer Vision, Tokyo, Japan, 2007.
- [6] R. HARTLEY AND F. KAHL, *Global optimization through rotation space search*, Int. Journal Computer Vision, 82 (2009), pp. 64–79.
- [7] R. HARTLEY AND F. SCHAFFALITZKY, *L_∞ minimization in geometric reconstruction problems*, in Conf. Computer Vision and Pattern Recognition, vol. I, Washington DC, USA, 2004, pp. 504–509.
- [8] R. I. HARTLEY AND A. ZISSERMAN, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2004. Second Edition.
- [9] K. HYYPPÄ, *Optical navigation system using passive identical beacons*, in Proceedings intelligent autonomous systems, Amsterdam, 1987.
- [10] F. KAHL AND R. HARTLEY, *Multiple view geometry under the L_∞ -norm*, IEEE Trans. Pattern Analysis and Machine Intelligence, 30 (2008), pp. 1603–1617.
- [11] Q. KE AND T. KANADE, *Quasiconvex optimization for robust geometric reconstruction*, IEEE Trans. Pattern Analysis and Machine Intelligence, 29 (2007), pp. 1834–1847.
- [12] H. LI, *A practical algorithm for L_∞ triangulation with outliers*, in Conf. Computer Vision and Pattern Recognition, Minneapolis, USA, 2007.
- [13] C. OLSSON, O. ENQVIST, AND F. KAHL, *A polynomial-time bound for matching and registration with outliers*, in Conf. Computer Vision and Pattern Recognition, Anchorage, USA, 2008.
- [14] C. OLSSON AND F. KAHL, *Generalized convexity in multiple view geometry*, in Journal of Mathematical Imaging and Vision, 2010.
- [15] L. QUAN AND T. KANADE, *Affine structure from line correspondences with uncalibrated affine cameras*, IEEE Trans. Pattern Analysis and Machine Intelligence, 19 (1997), pp. 834–845.
- [16] K. SIM AND R. HARTLEY, *Removing outliers using the L_∞ -norm*, in Conf. Computer Vision and Pattern Recognition, New York City, USA, 2006, pp. 485–492.