Camera Resectioning from Image Edges with the L_∞-Norm Using Linear Programming

Li Yang
yangli@cis.pku.edu.cn
Xianghua Ying*
http://www.cis.pku.edu.cn/vision/Visual&Robot/people/ying/
Lulu Hou
huyanlulu@163.com
Jing Kong
superkj@pku.edu.cn
Hongbin Zha
http://www.cis.pku.edu.cn/vision/Visual&Robot/people/zha/

Key Lab of Machine Perception, MOE
School of EECS, Peking University
Beijing, 100871 P.R. China

Abstract

This paper discusses how to resection camera using image edges extracted from projected polyhedron, with noise caused by shadows and occlusions etc. First, we propose a definition of the distance between two line segments in a 2D plane, which is an essential concept in computer vision. Based on it, the problem can be recast as a quasi-convex optimization problem in a famous and successful framewrok. Different from [8], our solution uses linear programming (LP) with the residual error defined by Euclidean distance, instead of using Second Order Cone Programming (SOCP) often utilized by most previous problems. Most related work makes use of the point to point correspondences. Intuitively, people may consider line segment correspondences should be more complex to handle than those of points. In fact, this paper shows that the problem can be simplified on the contrary.

1 Introduction

A large number of classic structure and motion problems, like the triangulation and camera resectioning have been discussed widely and deeply in recent years. Their goals are to infer the scene structure and/or the camera motion, given image data. Specially, camera resectioning is known as the computation of the camera projection matrix from corresponding 3D space and image entities. The simplest such correspondence is that between a 3D point and its image under the unknown camera mapping. The problem using this kind of correspondence has been most deeply discussed in the past.

Using line segment correspondences in addition to point ones is important in practice because low-level vision routines are relatively good at finding the middle parts of lines but

*Corresponding Author; present address: No.2 Science Building, Department of EECS, Peking University, Beijing, China.

BMVC 2011 http://dx.doi.org/10.5244/C.25.71

are much less certain about exactly where the lines terminate. Line terminations may also be widely displaced due to occlusions, shadows, or various sources of failure in the low-level edge detection algorithms. Therefore, discussing camera resectioning using line segment correspondences have such a significant meaning. This paper will discuss the problem in this way.

The problem we wish to solve is the following: given a set of known correspondences between three-dimensional line segments on a model and those in a 2D image, the values of the unknown projection will be resulted from the constraints of the known correspondences.

The procedure for reconstructing the unknowns of this inverse problem is to find the solution that reproduces the images as closely as possible. Thus, it is needed to define a distance with "geometric significance" between two different line segments. Commonly, the perpendicular distance of selected points on one line from the other line in the same plane is measured as the distance of one from another [7]. In this paper, we discover a new definition of this kind of distance by means of area. This definition is highly related to the distance mentioned above. Differently, we consider the length of line segments in the image as the weight of the perpendicular distance.

In recent years, L_∞-norm framework [3] are highly recommended to solve a variety of structure and motion problems, since the framework allows for the computation of global estimates [4, 5, 6]. All these problems in this framework can be efficiently solved using SOCP by minimizing the L_∞-norm of the vector of all residual errors. The use of SOCP is due to geometric distance, *i.e.* residual error, is defined by L_2-norm in the 2D image plane in this framework.

We can know from [8] that if the residual error is also defined by L_∞-norm, these problems could be solved by LP, but the kind of definition of residual error are not rotation-invariant, while L_2-norm is. Thus, even though those problems can be solved by LP method, the result is probably less reliable. A very important work of [2] should be mentioned here is that they discovered that camera orientation estimation problems can be solved using LP. As is mentioned, the computationally more expensive SOCP is not required.

2 Notations and Basic Equations

2.1 Camera Representation

A 2D point and a 3D point are denoted by $\mathbf{x} = (x, y, 1)^T$ and $\mathbf{X} = (X, Y, Z, 1)^T$ separately as their homogeneous coordinates. A camera can be represented as a 3x4 projection matrix denoted by $\mathbf{P}$

$$\mathbf{P} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{pmatrix} \tag{1}$$

The relationship between a 3D point $\mathbf{X}$ and its image projection $\mathbf{x}$ is given by

$$s\mathbf{x} = \mathbf{PX}, \tag{2}$$

where s is an arbitrary scale factor.

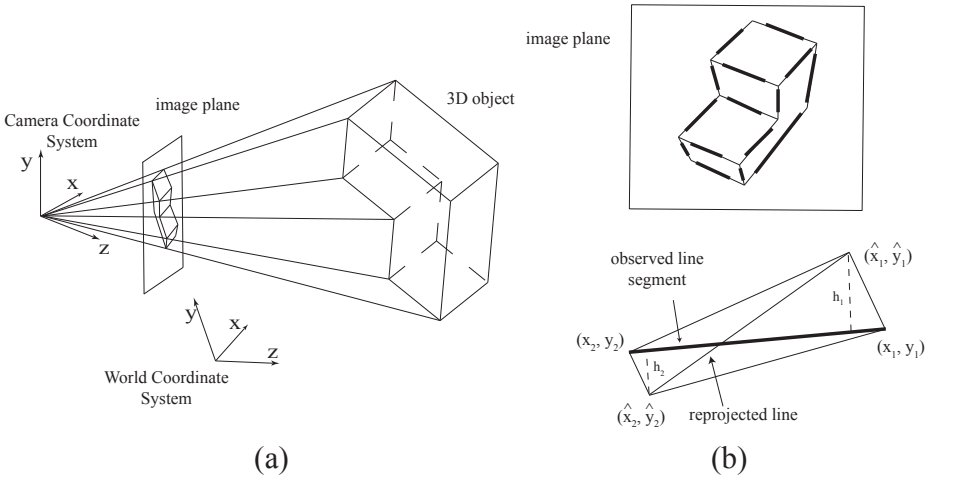

Figure 1: (**a**) Projection of an object onto a camera's image plane (**b**) In the upper figure, correspondences between reprojected 3-D line segments and observed 2-D image line segments are shown; in the lower figure, Maximum area of the two different triangles represents residual error between predicted line segment and observed line segment

2.2 Distance between Two Line Segments

Given known 3D line segment $\mathbf{X}_1\mathbf{X}_2$ and its corresponding image line segment $\mathbf{x}_1\mathbf{x}_2$, the projection matrix of camera $\mathbf{P}$ should satisfy the constraint that if we project the line segment $\mathbf{X}_1\mathbf{X}_2$ using it, the line segment $\hat{\mathbf{x}}_1\hat{\mathbf{x}}_2$ we get, where $s_1\hat{\mathbf{x}}_1 = \mathbf{P}\mathbf{X}_1, s_2\hat{\mathbf{x}}_2 = \mathbf{P}\mathbf{X}_2$, should "equal" the given image line segment $\mathbf{x}_1\mathbf{x}_2$. When noise is present in practice due to occlusion, shadows and so on, the $\hat{\mathbf{x}}_1\hat{\mathbf{x}}_2$ should be "nearly equal" to $\mathbf{x}_1\mathbf{x}_2$.

The words "equal" and "nearly equal" are always related with the concept of "distance". Therefore, the definition of the distance from one line segment to another in 2D plane should be investigated first. We define the error associated with a single image measurement as

$$d(\mathbf{x}_1\mathbf{x}_2, \hat{\mathbf{x}}_1\hat{\mathbf{x}}_2) = \max\{S_{\Delta\mathbf{x}_1\mathbf{x}_2\hat{\mathbf{x}}_1}(\mathbf{P}), S_{\Delta\mathbf{x}_1\mathbf{x}_2\hat{\mathbf{x}}_2}(\mathbf{P})\}, \tag{3}$$

where $S_{\Delta\mathbf{uvw}}$ represents the area of the triangle $\Delta\mathbf{uvw}$ in the image plane. Figure 1 illustrates the measurement of these area errors between the matching model line segment and the image one.

We would show that the individual functions $S_{\Delta\mathbf{x}_1\mathbf{x}_2\hat{\mathbf{x}}_k}$ for $k = 1, 2$ can be rewritten as

$$S_{\Delta\mathbf{x}_1\mathbf{x}_2\hat{\mathbf{x}}_k}(\mathbf{P}) = \frac{\left|\left(d_1\mathbf{X}_k^T, d_2\mathbf{X}_k^T, d_3\mathbf{X}_k^T\right)\alpha\right|}{\left(\mathbf{0}_{1\times4}, \mathbf{0}_{1\times4}, \mathbf{X}_k^T\right)\alpha}, \tag{4}$$

where

$$\alpha = (a_{11}, a_{12}, a_{13}, a_{14}, a_{21}, a_{22}, a_{23}, a_{24}, a_{31}, a_{32}, a_{33}, a_{34})^T,$$

$$d_1 = y_1 - y_2,$$

$$d_2 = x_2 - x_1,$$

$$d_3 = x_1 y_2 - x_2 y_1.$$

proof:
We know

$$\begin{aligned} S_{\Delta \mathbf{x}_1 \mathbf{x}_2 \hat{\mathbf{x}}_k} &= \left| \begin{vmatrix} \hat{x}_k & \hat{y}_k & 1 \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{vmatrix} \right| \\ &= |d_1 \hat{x}_k + d_2 \hat{y}_k + d_3|. \end{aligned} \tag{5}$$

From(2), we have

$$\begin{aligned} \hat{x}_k &= \frac{a_{11}X_k + a_{12}Y_k + a_{13}Z_k + a_{14}}{a_{31}X_k + a_{32}Y_k + a_{33}Z_k + a_{34}} = \frac{\left(\mathbf{X}_k^T, \mathbf{0}_{1\times 4}, \mathbf{0}_{1\times 4}\right)\alpha}{\left(\mathbf{0}_{1\times 4}, \mathbf{0}_{1\times 4}, \mathbf{X}_k^T\right)\alpha}, \\ \hat{y}_k &= \frac{a_{21}X_k + a_{22}Y_k + a_{23}Z_k + a_{24}}{a_{31}X_k + a_{32}Y_k + a_{33}Z_k + a_{34}} = \frac{\left(\mathbf{0}_{1\times 4}, \mathbf{X}_k^T, \mathbf{0}_{1\times 4}\right)\alpha}{\left(\mathbf{0}_{1\times 4}, \mathbf{0}_{1\times 4}, \mathbf{X}_k^T\right)\alpha}, \end{aligned} \tag{6}$$

Relating (5) (6), we could obtain that

$$\left| \frac{\left(d_1\mathbf{X}_k^T, d_2\mathbf{X}_k^T, d_3\mathbf{X}_k^T\right)\alpha}{\left(\mathbf{0}_{1\times 4}, \mathbf{0}_{1\times 4}, \mathbf{X}_k^T\right)\alpha} \right|. \tag{7}$$

Because the points $\mathbf{X}_k$ are visible in the image means they must lie in front of the camera, $\left(\mathbf{0}_{1\times 4}, \mathbf{0}_{1\times 4}, \mathbf{X}_k^T\right)\alpha$ should be greater than zero. This property is known as *Cheirality*, by which we mean the consideration of which points are in front of the camera. If we denote

$$\mathbf{c} = \left(d_1\mathbf{X}_k^T, d_2\mathbf{X}_k^T, d_3\mathbf{X}_k^T\right)^T \text{ and } \mathbf{f} = \left(\mathbf{0}_{1\times 4}, \mathbf{0}_{1\times 4}, \mathbf{X}_k^T\right)^T,$$

the form (4) could be expressed as following:

$$\frac{|\mathbf{c}^T\alpha|}{\mathbf{f}^T\alpha} \text{ and } \mathbf{f}^T\alpha \geq 0 \tag{8}$$

Geometry Interpretation. The relationship between the measurement of perpendicular errors and that of area errors is shown in Figure 1. We could note that the area equals the perpendicular distance multiply the length of the segment. Thus, in our problem, we could see the length of image edge as the weight of the perpendicular h. The Euclid distance from point to line and the distance we define both have the rotation invariance in 2D image plane. By this meaning, cost function is based on geometric distance.

2.3 Camera Resectioning Problem

In this part, we would start with the description of our problem, followed by introducing the traditional solutions for the problem.

Let $\mathbf{X}_1^i\mathbf{X}_2^i$ denote a set of 3D line segments and let $\mathbf{x}_1^i\mathbf{x}_2^i$ be the corresponding image line segments for $i = 1\cdots m$. Similarly, the unknown projection matrix of camera is denoted by $\mathbf{P}$ and $s_{ik}\hat{\mathbf{x}}_k^i = \mathbf{P}\mathbf{X}_k^i$ represents the mapping from model points $\mathbf{X}_k^i$ to the image plane via the camera matrix $\mathbf{P}$. With all those correspondences exerting constraints to the projection,

obtaining the best estimate of the $\mathbf{P}$ means that the mapped line segments $\hat{\mathbf{x}}_1^i\hat{\mathbf{x}}_2^i$ are most closely to the image line segments $\mathbf{x}_1^i\mathbf{x}_2^i$.

The words "most closely" are usually interpreted in a least squares sense. Formally, the cost function is expressed as a sum of squared residues:

$$\sum_{i=1}^{m} d^2(\mathbf{x}_1^i\mathbf{x}_2^i, \hat{\mathbf{x}}_1^i\hat{\mathbf{x}}_2^i). \tag{9}$$

Algebraic Method. There is a simple algebraic method for solving this problem. Relating (4), Each equation $S_{\Delta\mathbf{x}_1^i\mathbf{x}_2^i\hat{\mathbf{x}}_k^i} = 0$ holds only up to an unknown scale factor and can be written precisely as,

$$\mathbf{c}_j^T\alpha = 0 \tag{10}$$

in which we denote $j = \langle i,k\rangle$, $i = 1,\cdots,m$, $k = 1,2$. The complete set of equations is linear in the unknown quantities α, which can be solved involving the Singular Value Decomposition. This method may seem attractive, but is not reliable, because the cost function that it is minimizing has no particular meaning.

Nonlinear estimation. As is mentioned above, L_2-norm is sometimes considered as the gold standard. Due to the nonlinear nature of the equation (9), the estimation of the parameters involves applying an iterative algorithm, such as Levenberg-Marquardt optimization method etc. However, without proper initial parameter values the optimization may stick in a local minimum and thereby cause the calibration to fail. Usually, we could use the parameters from algebraic method as the initial values for the optimization. Nevertheless, it does not always work well.

3 Linearity of the Problem Based on the L_∞-norm

In this section, some concepts about L_∞-norm are introduced with discussing our problem. For more details, the reader is referred to the classic papers [3, 4].

3.1 L_∞-Norm Minimization

In this context, the words "most closely" are interpreted in finding the matrix $\mathbf{P}$ that minimizes the following cost function:

$$\max_i\{d(\mathbf{x}_1^i\mathbf{x}_2^i, \hat{\mathbf{x}}_1^i\hat{\mathbf{x}}_2^i)\} \tag{11}$$

Namely, the cost function measures the maximum of a set of model-fitting errors, rather than the sum of squares (9), or L_2 cost function that is commonly used. If we consider the individual errors $d(\cdot,\cdot)$ as the components of a vector, then (11) may be thought of the L_∞-norm of this vector. Obviously, our problem could also be considered as minimizing the L_∞-norm of the following vector$\langle S_{\Delta\mathbf{x}_1^i\mathbf{x}_2^i\hat{\mathbf{x}}_k^i}\rangle_{i=1,k=1}^{m,2}$. Therefore, the problem is called the minimax (L_∞) optimization problem:

$$\min_{\mathbf{P}}\max_{i,k}\left\{S_{\Delta\mathbf{x}_1^i\mathbf{x}_2^i\hat{\mathbf{x}}_k^i}\right\} \tag{12}$$

.

From (8), our problem is in the following form:

$$\min_\alpha\max_j \frac{|\mathbf{c}_j^T\alpha|}{\mathbf{f}_j^T\alpha} \text{ and } \mathbf{f}_j^T\alpha \geq 0, \quad j = 1,\cdots,2m \tag{13}$$

3.2 Polyhedral Sublevel Sets of Quasi-Convex Functions

To solve the optimization problem (13), we need work with the quasi-convex optimization [1]. Before we can introduce this concept, let us recall the definition of convex sets and quasi-convex functions, followed by the explanation of what is the quasi-convex optimization problem.

Convex Set. A subset S of $\mathbb{R}^n$ is *convex* if the line segment between any points in S lies in S, *i.e.*, if for any $\mathbf{x}_1, \mathbf{x}_2 \in S$ and any θ with $0 \leq \theta \leq 1$, we have $\theta\mathbf{x}_1 + (1-\theta)\mathbf{x}_2 \in S$.

Quasi-Convex Function. A function $f : \mathbb{R}^n \to \mathbb{R}$ is called *quasi-convex* if its domain and all its sublevel sets

$$S_\gamma = \{\mathbf{x} \in \mathbf{dom} f \,|\, f(\mathbf{x}) \leq \gamma\}$$

for $\gamma \in \mathbb{R}$, are convex.

Let us see our problem. We could prove that the functions $\frac{|\mathbf{c}_j^T \alpha|}{\mathbf{f}_j^T \alpha} \leq \gamma$ in (13) are quasi-functions. We could see it from following proof:

$$\begin{aligned} S_\gamma &= \left\{ \alpha \Big| \frac{|\mathbf{c}_j^T \alpha|}{\mathbf{f}_j^T \alpha} \leq \gamma, \mathbf{f}_j^T \alpha \geq 0 \right\} \\ &= \left\{ \alpha | -\gamma \mathbf{f}_j^T \alpha \leq \mathbf{c}_j^T \alpha \leq \gamma \mathbf{f}_j^T \alpha, \mathbf{f}_j^T \alpha \geq 0 \right\} \\ &= \left\{ \alpha | (-\gamma \mathbf{f}_j^T - \mathbf{c}_j^T)\alpha \leq 0, (-\gamma \mathbf{f}_j^T + \mathbf{c}_j^T)\alpha \leq 0, -\mathbf{f}_j^T \alpha \leq 0 \right\} \end{aligned} \tag{14}$$

Our proof shows that all the sublevel sets of these functions are *polyhedron* instead of second order cone, which means that in solving our optimization problem we could use linear programming, even though they could be seen as a special form of the second order function mentioned in [4], which means they do be quasi-convex functions without proving.

The Maximum of Quasi-Convex Functions. Now relating that $\frac{|\mathbf{c}_j^T \alpha|}{\mathbf{f}_j^T \alpha} \leq \gamma$ are quasi-convex, we could notice that our objective function (13) is a maximum of quasi-convex functions. It has also been proved that the pointwise maximum of a set of quasi-convex functions is quasi-convex (lemma3.1 [4]), therefore our problem is the following one:

Quasi-Convex Optimization Problem. This kind of optimization problem is a problem of the form

$$\text{minimize} \quad f(\mathbf{x}) \tag{15}$$

where $f(\mathbf{x})$ is quasi-convex on its domain D that is convex. Specifically, if $f(\mathbf{x}) = max_i f_i(\mathbf{x})$, in which $f_i(\mathbf{x})$ are quasi-convex functions, then the form is one kind of minimax optimization problem:

$$\min_{\mathbf{x}} \max_i f_i(\mathbf{x}), \tag{16}$$

where each $f_i(\mathbf{x})$ is a quasi-convex function on a convex domain $D = \cap_i \mathbf{dom} f_i$. This is what our problem exactly is.

4 Solving Optimization with Linear Programming

This section provides the details how to solve our problem using quasi-convex optimization.

Solving such a optimization problem (16) should commonly use a technology called convex optimization. For example, Second Order Cone Programming (SOCP) is used in the Triangulation Problem [3, 4]. Here, we use Linear Programming rather than SOCP.

In this section, we first introduce some knowledge about Linear Programming and feasibility problems, followed by the solving process of the optimization problem.

4.1 Linear Programming

Linear Programming (LP) is one of the simplest kinds of convex optimization [1]. It is also an important class of optimization problem. Its objective and all constraint functions are linear:

$$\begin{aligned} \min \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & \mathbf{a}_i^T\mathbf{x} \le b_i, \quad i = 1, \cdots, m. \end{aligned} \tag{17}$$

Here the vectors $\mathbf{c}, \mathbf{a}_1, \cdots, \mathbf{a}_m \in \mathbb{R}^n$ and scalars $b_1, \cdots, b_m \in \mathbb{R}$ are problem parameters that specify the objective and constraint functions.

LP v.s. SOCP. There are a variety of very effective methods for solving LPs and these algorithms are quite reliable. It can be said that solving linear programming is a mature technology. Linear programming solvers are embedded in many tools and applications, such as, Matlab. While, the SOCP problem includes LP as a special case. It is a kind of quadratic programming so that the time complexity for SOCP is not as good as for LP algorithms.

LP Feasibility Problem. If we omit the objective function $\mathbf{c}^T\mathbf{x}$ and ask to find any $\mathbf{x}$ satisfying the constraint, we have what is known as a *LP feasibility problem*. Formally, The LP feasibility problem is

$$\begin{aligned} \text{find} \quad & \mathbf{x} \\ \text{subject to} \quad & \mathbf{a}_i^T\mathbf{x} \le b_i, \quad i = 1, \cdots, m. \end{aligned} \tag{18}$$

The solution of such a problem could be obtained via LP with variables $t, \mathbf{x}$:

$$\begin{aligned} \min \quad & t \\ \text{subject to} \quad & \mathbf{a}_i^T\mathbf{x} \le b_i + t, \quad i = 1, \cdots, m. \end{aligned} \tag{19}$$

If minimizer $\mathbf{x}^\star, t^\star$ satisfies $t^\star \le 0$, $\mathbf{x}^\star$ is a feasible solution or there is no such $\mathbf{x}$ satisfying the constraint.

4.2 The Feasibility Problem

The feasibility problem in our case is

$$\begin{aligned} \text{find} \quad & \alpha \\ \text{subject to} \quad & \frac{|\mathbf{c}_j^T\alpha|}{\mathbf{f}_j^T\alpha} \le \gamma \quad j = 1, \cdots, 2m \\ & \mathbf{f}_j^T\alpha \ge 0 \quad j = 1, \cdots, 2m \end{aligned} \tag{20}$$

From (14), the above problem could be transformed into:

$$\begin{array}{cc} \text{find} & \alpha \\ \text{subject to} & \begin{bmatrix} -\mathbf{c}_j^T - \gamma \mathbf{f}_j^T \\ \mathbf{c}_j^T - \gamma \mathbf{f}_j^T \\ -\mathbf{f}_j^T \end{bmatrix} \alpha \leq \mathbf{0} \qquad j = 1, \cdots, 2m \end{array} \tag{21}$$

Obviously,this is a LP feasibility programming, which is in the form (18).

4.3 Solving Procedure

From (13), it is easily seen that this problem may be transformed, by introducing an additional variable γ, into an equivalent problem of the form

$$\min_{\gamma,\alpha} \gamma \text{ subject to } \frac{|\mathbf{c}_j^T \alpha|}{\mathbf{f}_j^T \alpha} \leq \gamma \text{ and } \mathbf{f}_j^T \alpha \geq 0 \tag{22}$$

However, considered as a function of both α and γ, the constraint is not a convex constraint. On the other hand, for any fixed value of γ, relating to(14), the constraint is an linear constraint.

Let $\gamma^\star$ denote the optimal value of the quasi-convex optimization problem (13). If the feasibility problem (20) is feasible, then we have $\gamma^\star \leq \gamma$. Conversely, if the problem is infeasible, then we can conclude $\gamma^\star \geq \gamma$.

This observation allows us to find the minimum value of $\gamma^\star$ via a binary searching over values of γ, which is a simple algorithm called *bisection*. We start with an interval $[\gamma_l, \gamma_u]$ known to contain the optimal value $\gamma^\star$. We then solve the linear feasibility problem (21) at its midpoint $\gamma = (\gamma_l + \gamma_u)/2$, to determine whether the optimal value is in the lower or upper half of the interval, and update the interval accordingly. This produces a new interval, which also contains the optimal value, but has half width of the initial interval. This is repeated until the width of the interval is small enough:

Algorithm Bisection
given: An interval$[\gamma_l, \gamma_u]$ known to containing the optimal value $\gamma^\star$ and tolerance $\varepsilon > 0$
repeat
$\gamma = (\gamma_l + \gamma_u)/2$.
Solve the linear feasibility problem (21)
if (21) is feasible **then**
$\gamma_u := \gamma$
else
$\gamma_l := \gamma$
end if
until $\gamma_u - \gamma_l \leq \varepsilon$.

5 Experimental Results

In order to test our method, we made use of both generated data and real data. In the simulated experiments, we randomly generate some 3-D line segments and corresponding image line segments to compare the robustness of these algorithms. For the experiment based on

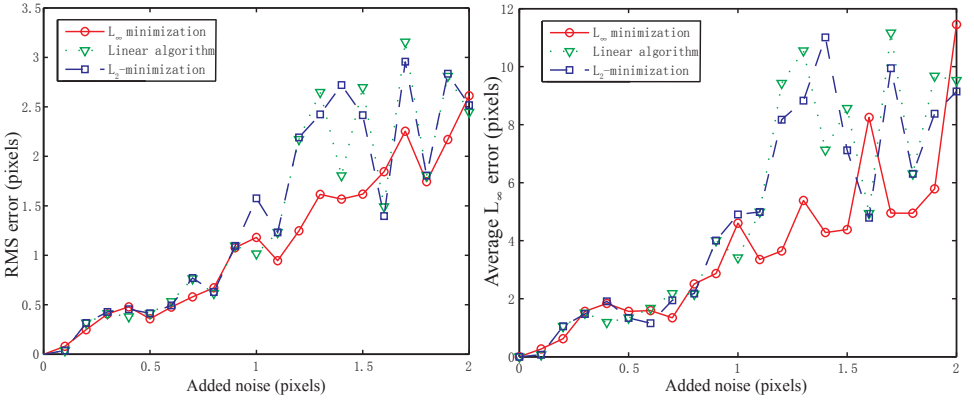

Figure 2: Camera resection results with varying added noise levels. on the left, RMS error is graphed; on the right, L_∞-norm error is graphed.

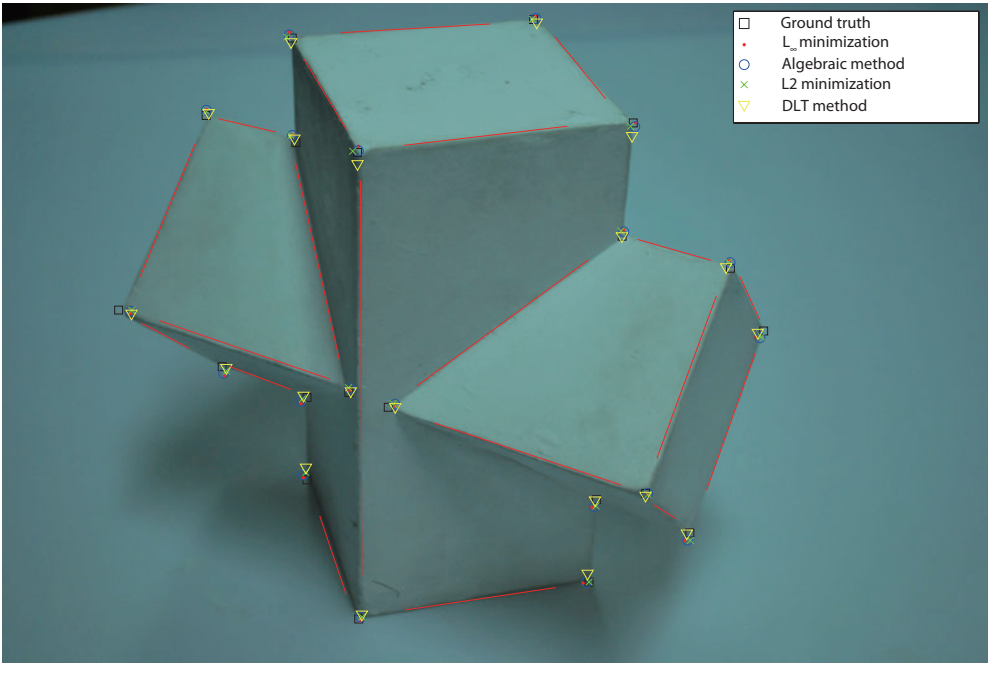

Figure 3: Comparison of reprojected points by different camera resectioning algorithms. The red line segments are extracted image edges.

real data, a polyhedron is used to resection the camera. We compare the reprojected points generated by different algorithms with the ground truth.

In the simulated experiments, besides our proposed method, we applied the algebraic method mentioned before and the nonlinear optimization using LM algorithm based on L_2-norm to the same data. For comparison, we show the performances of the algorithms with respect to the L_∞-norm. Since the L_2-norm is a common standard, the root-mean-square(RMS) error of the reprojected points, results for this measure is also given. It can be shown that the three methods all perform quite well when the added noise level is low, but our method still keeps performing well while the other two degenerate fast when the added noise level gets higher.

In the experiment based on real data, the algorithm based on point to point correspondences, *i.e.* DLT, was also applied. It shows that all the other three perform better than DLT method. DLT performs perfectly only at the points, by which DLT algorithm are computed, but could not promise good reprojections at the other points.

6 Conclusion

In this paper, we deal with camera resectioning using line segment correspondences. In this way, we discover a new definition of the distance between line segments then discuss the traditional methods to solve the problem and give a solution using linear programming based on L_∞-Norm. Camera resectioning often uses point correspondences rather than line segments and its solution under the L_∞ framework needs SCOP for problem solving. This paper may probably provide a special view to considering this problem.

7 Acknowledgements

This work was supported in part by NKBPRC 973 Grant No. 2011CB302202, NHTRDP 863 Grant No. 2009AA01Z329, and the NNSFC Grant No. 61075034.

References

[1] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521833787.

[2] R. Hartley and F. Kahl. Global optimization through rotation space search. *IJCV*, 82(1): 64 – 79, 2009.

[3] R. I. Hartley and F. Schaffalitzky. l_∞ minimization in geometric reconstruction problems. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[4] F. Kahl and R. Hartley. Multiple view geometry under the L_∞-norm. *PAMI*, 30(9):1603 – 1617, September 2008.

[5] Fredrik Kahl. Multiple view geometry and the L_∞-norm. In *International Conference on Computer Vision*, 2005.

[6] Qifa Ke and Takeo Kanade. Quasiconvex optimization for robust geometric reconstruction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29:1834–1847, October 2007.

[7] David G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artif. Intell.*, 31(3):355–395, 1987.

[8] Yongduek Seo and Richard Hartley. A fast method to minimize l error norm for geometric vision problems. In *International Conference on Computer Vision*, pages 1–8, 2007.