A practical algorithm for L_∞ triangulation with outliers

Hongdong Li
RSISE, Australian National University
VISTA, National ICT Australia

Abstract

This paper addresses the problem of robust optimal multi-view triangulation. We propose an abstract framework, as well as a practical algorithm, which finds the best 3D reconstruction with guaranteed global optimality even in the presence of outliers. Our algorithm is founded on the theory of LP-type *problem. We have recognized that the L_∞ triangulation is a concrete example of the LP-type problems. We propose a set of non-trivial* basis operation *subroutines that actually implement the idea. Experiments have validated the effectiveness and efficiency of the proposed algorithm.*

1. Backgrounds

The triangulation problem. Triangulation is the process of computing 3D structure from known camera matrices. Formally, let $\{\mathsf{P}_i; i = 1,...,n\}$ be a sequence of n known cameras, and $\mathbf{x}_i$ be the image point (at view i) of an unknown 3D point $\mathbf{X}$ in 3-space. Thus we have $\mathbf{x}_i = \mathsf{P}_i\mathbf{X}$ (up to scale). Then the problem of reconstructing the 3D point $\mathbf{X}$, given the camera matrices P_i and image points $\mathbf{x}_i$, is known as *triangulation*.

In the absence of noise or outliers, the triangulation problem is trivial, involving finding the intersection point of rays in space. However, noise and outliers are often unavoidable. In such circumstance to find a unique and optimal solution is much desirable but difficult [7][3][4].

The $\mathbf{L}_\infty$ optimization. A recent paper [3] has been sparking the interest of using L_∞-norm for various geometric vision problems. The interest is growing, as evidenced by the increasing numbers of publications devoted to the topic, e.g., [8][6][14][13].

One of the chief advantages of the L_∞ scheme is that: problems formulated by the L_∞-norm often possess a single, hence global, optimum. By contrast, hardly has any conventional L_2-based methods ensured the global optimality. Another benefit of L_∞ is that, compared with L_2, it often leads to a simpler formulation for the same problem.

However, it is well known that the L_∞-optimization is extremely vulnerable to outliers. This is because: the L_∞-norm based method is aimed at minimizing the point-wise maximal residual, whereas outliers often give rise to the maximal residual. As a result, a single outlier may destroy the whole estimation. This is the Achilles's Heel of the L_∞-based method. To salvage the method some *data cleaning* procedures (i.e., outlier-removal) must be applied first.

2. Existing approaches

The problem of outlier-removal has been extensively researched in the areas of *statistics* and *computational geometry* (c.f. [11] [2] and references therein). Statistical methods such as the robust M-estimator often assume a large-size data set (in order to be *statistically significant*). Whereas, in the triangulation context the available data set size is often small. For this reason we will not discuss the M-estimator methods here.

2.1. A remark on RANSAC

RANSAC is a powerful technique for outlier-removal, which allows the user quickly removing a large portion of outliers from the data. Particularly, for triangulation the optimal L_2 solver proposed by [4] is very suitable to the RANSAC framework, because it can be used as an efficient minimal-solver (for the RANSAC) requiring only two points from two views.

However, this is not good enough in the L_∞ setting. Being a randomized algorithm, the RANSAC may easily miss a few outliers. In other words, it is not very reliable in thoroughly removing all outliers. Whereas, even a single remaining outlier may destroy the overall L_∞ estimation.

Nevertheless, we do not oppose the use of RANSAC for solving L_∞ problems. On the contrary, we highly recommend it for quickly removing the most egregious outliers in the first stage. Our algorithm can be employed after the RANSAC, in order to further remove any remaining outliers—-if removing them does provide *further* and *substantial* decrease in the residual error.

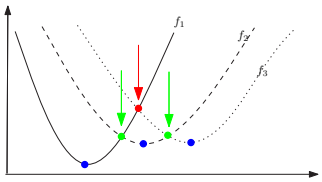

Figure 1. The existence of multiple local minima of the k-th median. This figure shows three cost functions. While there is only a single minimum for the L_∞ solution (i.e, minimax, in red dot), there are two level-1 minima (in green dots) and three level-2 minima (in blue dots).

2.2. K-th median optimization

So far, there have been only two papers that devoted specifically to the outlier problem in the L_∞ triangulation context: Ke&Kanade's [8] and Sim&Hartley's [14].

Ke and Kanade relax the objective of the L_∞ of minimizing the point-wise *maxima*. Instead, the k-th level maxima (i.e., k-th median) is to be minimized. The idea is similar to the LMeds (least median) in robust regression where the median is exactly the $\frac{n}{2}$-th level maximum.

Finding the k-th level residual is not an easy task. The authors thus suggest a convex programming algorithm, based on solving a *convex feasibility problem*. This algorithm works well, in terms of obtaining an *approximate* k-th order median. Moreover, if an *exact solution* is required, the authors further suggest an integer programming approach.

While the algorithm is novel and useful, there is however a problem that seemingly has been overlooked. That is, even if the convex programming (or integer programming) does converge to *a* k-th median solution, there is however, *no* guarantee that this solution is the *unique* global optimum at level k. In fact, k-th medians are generally not unique, but can have many local minima. In this sense, the output of the algorithm is unpredictable. It is not clear which local minimum the algorithm has eventually found. As a result, the k-th median algorithm has deviated from the original promise of the L_∞ idea which was meant to find a *single*, *unique* and *global* solution. Fig-1 illustrates the existence of multiple k-th medians.

Moreover, since at level k the α-sublevel set is not necessarily a *convex set* (otherwise there would not be multiple local minima), so the *convex feasibility problem* is no longer easy to solve. Nonetheless, since a bi-section search is adopted, their algorithm frequently ends up with a good approximation, which probably explains its success in experiments.

2.3. Throw away bad points

Sim and Hartley conduct a rigorous investigation of a simple idea for outlier-removal: *throwing away bad points with the maximal residuals*. Given that certain conditions are satisfied, the paper has proven theoretically that the above simple idea effectively reduces outliers in a predictable manner.

Their algorithm proceeds in an iterative fashion. In each iteration, at least one outlier will be removed from the measurement. Experiment for optimal L_∞ triangulation with outliers has obtained encouraging success. While effective, the algorithm is however, not efficient. Usually, in the process of throwing away a "support-set" (cf. [14]) not only outliers but also inliers are discarded. This is very 'wasteful' especially when the number of matching points is limited. For instance, in triangulation one often does not have a long track for each feature point. The authors do suggest a remedy of *reinstatement*, based on residual analysis. However, this remedy is generally not very reliable for discriminating true inliers from outliers.

In this paper, we provide an *abstract* framework, as well as a *practical* algorithm, that removes outliers in a *guaranteed* way. We present both theoretical foundation, and algorithmic implementation for L_∞ triangulation with outlier-removal. Experiments have obtained satisfactory results.

3. L_∞ triangulation, quasiconvex and SOCP

This section gives a brief review of the L_∞ triangulation. For more details the reader is referred to [3][6].

The L_∞ triangulation problem is formally cast as a min-max problem that minimizes the maximum of the reprojection errors:

$$\min_{\mathbf{X}} \max_i \frac{\|(\hat{\mathbf{x}}_i \mathsf{P}_{i;3} - \mathsf{P}_{i;1,2})\mathbf{X}\|^2}{(\mathsf{P}_{i;3}\mathbf{X})^2} \text{ subject to } \mathsf{P}_{i;3}\mathbf{X} \geq 0, \tag{1}$$

where P_i is the 3×4 camera matrix at frame i, $\mathsf{P}_{i;m}$ is the m-th row vector from P, and $\mathsf{P}_{i;m,n}$ is a sub-matrix from P consisting its m-th and n-th rows. $\hat{\mathbf{x}}$ is the *inhomogeneous* image point (i.e., a 2-vector). To ease notations we use $\|\mathsf{A}_i\mathbf{X}\|^2$ to represent the numerate part of the cost function, where A_i is an coefficient matrix depending on $\hat{\mathbf{x}}_i$.

It has been proven that the minimax cost function in (1) is a *quasiconvex* function, and the problem a *quasiconvex programming* [6] [8].

Definition 3.1. *A function $f : \mathbb{R}^n \to \mathbb{R}$ is called quasiconvex if its domain and all its α-sublevel sets $\{X \in \text{dom} f | f(X) \leq \alpha\}$ are convex.*

Every quasiconvex programming has only one local minimum, and thus it is also the global optimum. For finding this unique global optimum, we use the *bi-section* search algorithm:

Algorithm 3.2. ***(Bisection)***
***Input**: initial bounds $[\alpha_l, \alpha_u]$, two working variables $u = \alpha_u, l =$*

α_l *and a tolerance* ϵ.
Output*: optimizer* $\mathbf{X}^*$*, and optimal value of* α^*.

---

1. ***While*** $(u-l) > \epsilon$, ***do***
2. $\alpha = (u+l)/2$
3. *Solve the* SOCP feasibility problem *below.*
4. ***if*** *feasible,* ***then*** $u = \alpha$,
5. ***else*** $l = \alpha$, ***end if***.
6. ***end do.***

The SOCP feasibility problem in step-3 is:

Find $\mathbf{X}$, such that: $\forall i, \|\mathsf{A}_i\mathbf{X}\| \leq \alpha \mathsf{P}_{i;3}\mathbf{X}$, and $\mathsf{P}_{i;3}\mathbf{X} \geq 0$.
If such $\mathbf{X}$ does not exist, then reports 'infeasible'.

The first inequality constraint is easily recognized as a *second order cone*, thus the problem can be efficiently solved by a SOCP (second-order-cone-program, [1]). The second inequality is linear, known as the *chirality* (c.f. [5]).

4. Outlier removal: complexity analysis

Although there is a lack of rigorous mathematical definition of outliers, in this paper we however take a practical view: we consider outliers as a small portion of data points (say, at most k) containing in the n input data, which contributes to the greatest residual error in the estimation.

Under this view, our strategy for outlier-removal is thus to identify a subset of at least $n-k$ data points that produces the least residual error. Here the k is simply an *upper bound* estimate of the number of outliers (e.g., a percentile), and is not hard to find. In practice, k is often very small, especially after using the RANSAC for pre-filtering.

Now the outlier-removal problem has a more formal description of *subset-selection*. That is, given a set of n elements, the task is to identify a small subset containing at most $n-k$ elements that gives the minimal error of certain cost function.

To *exactly* solve the above subset-selection problem is very challenging in general. To see this, we examine the computational complexity of the problem below.

4.1. Exhaustive search

A trivial method for solving the subset-selection problem is by a *brute-force* search (over all possible combinations). This results in a typical combinatorial problem whose complexity is $\mathbf{O}(\sum_{i=0}^{k} \mathrm{C}_n^i)$ (C_n^i is the number of combinations). As the problem size n increases, this complexity grows *exponentially*. Clearly, due to such an exponential explosion the trivial search method soon becomes intractable when n or k is moderately large.

For example, consider a triangulation problem from 100 views. If one wishes to remove *at most* $k = 4$ outliers by a brute-force search, then in total he has to conduct $\mathrm{C}_{100}^0 + \mathrm{C}_{100}^1 + \mathrm{C}_{100}^2 + \mathrm{C}_{100}^3 + \mathrm{C}_{100}^4 \approx 4$ million tests of all combinations. In L_∞ triangulation context, each of these tests involves several loops of bisection iterations, and more loops of SOCP feasibility tests.

4.2. Our new method

In this paper we give a new method for exactly removing up to k outliers from n measurements. In spirit, our method belongs to exhaustive search. However, it works at a speed far more efficient than the brute force search. To demonstrate this, we will give an example below.

Consider a 100-view triangulation problem. Suppose after RANSACing there is still a small number of at most k outliers remaining. In other words, k is an upper bound of the number of outliers. Now the task is to thoroughly (exactly) eliminate all these outliers and get an accurate triangulation. By using our new method we are able to enumerate all local minima up to a desired level k. Choosing the minimum corresponding to the least residual error as the best solution, we then get a guaranteed optimal triangulation.

Empirical, we have obtained the following results shown in Table-1. It displays the numbers of tests needed (i.e. the number of local minima) in order to obtain a globally optimal solution by two different methods at different level of k. Clearly, our method is much more sensible, while the trivial exhaustive search is computationally prohibitive when k is large.

5. The geometric intuition

As mentioned above, our mew method is in principle an exhaustive search method, but, it works in a way far more efficient than brute-force search.

Before proceeding to introduce the main theory of the paper, which looks obscure at first glance, we will first present the underlying geometric intuition. Specifically, it is the following two key observations that motivate our new method.

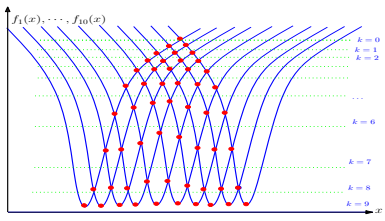

Figure 2. This figure depicts 10 quasi-convex cost functions $f_1(x), \cdots, f_{10}(x)$. The red dots indicate all the local minima. A tree structure rooted from $k=0$ connecting all local minima is formed.

Num of local minima (mthd/level k)	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$
Exhaustive search	1	101	5051	166751	4087976	79375496	1.2714×10^9
Our new method	1	4	12	35	104	302	879

Table 1. A comparison of the number of tests required (i.e. number of local minima) for obtaining a guaranteed global optimum. The data is taken from a 100-view triangulation experiment, allowing up to k outliers. Our method is much more efficient.

Consider a simple example of finding all the k-th level local minima of a set of ten one-dimensional quasi-convex cost functions $f_1(x), \cdots, f_{10}(x)$, as shown in fig-2. In the figure, all local minima are indicated by red dots.

Note that at level k there are exactly $k+1$ local minima (i.e. the number of red dots at level k). This number is much less than what is predicted by the complexity analysis conducted in the previous section, i.e. exponential.

Also note that for any given local minimum at level-$k, (k \geq 1)$, there is always a *direct path* that leads to a $(k-1)$-th level local minimum. In other words, for any node (i.e. a red dot) at level k there is always a preceding node (red dot) at level $(k-1)$. This gives rise to a *tree* data structure that links all the tree nodes (i.e. red dots) together. To recap, the two key observations are:

1. The number of local minima at level k is bounded from above, and the bound is much lower than exponential.
2. There is always a direct path from a local minimum of level k to a local minimum of level $k-1, \forall k \geq 1$.

As a result, one can perform an exhaustive search over all local minima (i.e. tree nodes) by a simple tree search.

We thus wonder: *whether these two observations (,which were made upon a* specific *one-dimensional minimization problem,) are* generic *enough so that it* applies *to the L_∞ triangulation problem as well*? If the answer is affirmative, then this will suggest an effective search method. Fortunately, for the L_∞ triangulation problem the answer is indeed "yes". To prove this, in the next section we will resort to an abstract, but elegant and powerful, optimization theory—the LP-type problems.

6. LP-type problems

6.1. Some terminologies

The theory of LP-type problems is mainly due to Sharir and Welzl [12], and Matousek with Sharir and Welzl [10], who actually coined the concept and originated the research.

An **abstract optimization** problem is specified by a pair (H, w), where H is a finite set, and $w : 2^{\mathsf{H}} \to \mathbf{W}$ is a *function* with *values* in a linearly ordered set $(\mathbf{W}, \leq)$. The elements of H are called constraints. One can consider the H as the universe of constraints for a given optimization problem (H, w). The domain space 2^{H} contains all combinations of feasibility-test applied to the constraint set H. For a subset $\mathsf{G} \subseteq \mathsf{H}$, $w(\mathsf{G})$ is called the *function value* of G. Intuitively, the value $w(\mathsf{G})$ stands for the minimal value attainable for certain objective function while satisfying all the constraints of G.

Solving an abstract optimization problem is defined as: find a minimal-size subset B of H such that their values are identical, i.e., $w(\mathsf{H}) = w(\mathsf{B}) > -\infty$. The function value $-\infty$ (standing for 'undefined') precedes all values in set $\mathbf{W}$.

Definition 6.1. ***(LP-type problem)*** *An abstract optimization problem is called an* ***LP-type problem*** *if the following* ***two axioms*** *are satisfied:*

1. ***(Monotonicity)*** *For any constraint sets F, G with $\mathsf{F} \subseteq \mathsf{G} \subseteq \mathsf{H}$, we have $w(\mathsf{F}) \leq w(\mathsf{G})$.*
2. ***(Locality)*** *For any F, G with $\mathsf{F} \subseteq \mathsf{G} \subseteq \mathsf{H}$ with $w(\mathsf{F}) = w(\mathsf{G}) > -\infty$ and any $h \in \mathsf{H}$, $w(\mathsf{G}) < w(\mathsf{G} \cup h)$ implies that also $w(\mathsf{F}) < w(\mathsf{F} \cup h)$.*

Remark. The term 'LP-type' stands for 'linear programming type'. However, an LP-type problem is not necessarily a linear programming problem. Moreover, it is not even necessarily a linear problem. In fact, many nonlinear problems (including the L_∞ triangulation) are examples of LP-type problem.

Definition 6.2. ***(Basis.)*** *A finite set of constraints B is called a* ***basis****, if any proper subset of this set $\mathsf{B}' \subset \mathsf{B}$ has strictly smaller function value, i.e., $w(\mathsf{B}') < w(\mathsf{B})$. For a subset G of H, we say that $\mathsf{B} \subset \mathsf{G}$ is a* basis *of G if $w(\mathsf{B}) = w(\mathsf{G})$.*

Definition 6.3. *We say that a constraint $h \in \mathsf{H}$* ***violates*** *a subset $\mathsf{G} \subseteq \mathsf{H}$ if we have $w(\mathsf{G} \cup h) > w(\mathsf{G})$. The violation set $\mathsf{V}(\mathsf{G})$ is all the constraints that violate G. The cardinality k of the violation set $\mathsf{V}(\mathsf{G})$ of G is called the* ***level*** *of the set G, $k = |\mathsf{V}(\mathsf{G})|$.*

Example: The violation set of the universe set $\mathsf{V}(\mathsf{H}) = \emptyset$, and the level of H is zero.

Definition 6.4. ***(Combinatorial dimension).*** *The maximal cardinality of any basis of (H) is defined as the* combinatorial dimension *of the LP-type problem, denoted by* $\dim(\mathsf{H}, w)$.

Note that for a specific LP-type d-dimension optimization problem, e.g., $\min_{\mathbf{x}} f(\mathbf{x}), \mathbf{x} \in \mathbb{R}^d$, the *domain dimension* d is not necessarily equal to its combinatorial dimension. In fact they are different in general.

6.2. Two main theorems

It turns out that the above abstract LP-type framework provides an effective way of solving a wide class of optimization problems. In particular, in a few moment we will show that under this framework the outlier-removal problem is reduced to: *finding a basis which produces the smallest function value and satisfies all but at most k constraints*.

Such a basis-finding problem, as an instance of *subset-selection*, of course could be solved exactly by a brute-force search. However, as explained before, the complexity for brute-force search is exponential.

Remarkably, an LP-type problem can be solved much more efficiently (than brute-force search), thanks to the following two main theorems ([12] [10]).

Theorem 6.5. *(upper bound of cardinality.) For a non-degenerate LP-type problem (H, w) of combinatorial dimension d with $w(\mathsf{G}) > -\infty$ for any $\mathsf{G} \subset \mathsf{H}$, the number of bases of level at most k is bounded from above by $|\mathsf{B}_{\leq k}| = O((k+1)^d)$.*

Theorem 6.6. *(basis reachability.) Every basis of level k can be reached from the basis of level $k-1$ through a direct path. Consequently, all bases are connected through a tree structure.*

These two theorems have justified our two observations in sec-5. To prove these theorems requires some knowledge of probabilistic method [2], and is beyond the scope of the paper. Interested reader is referred to [10][9] for details.

We say an LP-type problem is *non-degenerate* if for any two *distinct* bases B and B' in H we have $w(\mathsf{B}) \neq w(\mathsf{B}')$. In practice, the non-degeneracy condition may be enforced by applying infinitesimal perturbations to the function values.

6.3. Solving LP-type problems

Solving an LP-type problem is referred to as: finding all the bases at a given level k. To solve an LP-type problem, we will need the following primitive operations:

- **Violation test**: Given a set G of constraints, and a single constraint h, decide whether or not $h \in \mathsf{V}(\mathsf{G})$.
- **Basis finding**: Given a set G of constraints, find a basis B of the set, i.e., $\mathsf{B} = \mathsf{B}(\mathsf{G})$, with $w(\mathsf{B}) > -\infty$.
- **Basis change**: Given a set G and one of its basis B, for a single constraint h find some new bases of the set-union $(\mathsf{B} \cup h)$.

Follow from the two main theorems, a deterministic LP-type algorithm can be derived ([2]):

Algorithm 6.7. ***A deterministic algorithm for LP-type problems***

Input: *an LP-type problem $(\mathsf{H}; w)$, a given maximal level K.*
Output: *all the bases B_k at each level $0 \leq k \leq K$.*

1. *(Initial basis finding) find the root basis set for the universe set, i.e., $\mathsf{B}_0 = \mathsf{B}(\mathsf{H})$. Let $k = 0$;*
2. *(Basis change) generate all bases set at level $k+1$ by performing a series of basis-change operations. Specifically, for every $b \in \mathsf{B}_k$, do the following: generate a basis at level $k+1$ by $\mathsf{B}_{k+1} = \mathsf{B}(\mathsf{H} \backslash \mathsf{V}(\mathsf{B}_k) \backslash b)$, where $\mathsf{V}(\mathsf{B}_k)$ is the violation set of B_k. The symbol $\backslash$ means 'exclude' or 'deprive of';*
3. *if $k = K$ go to step 4, otherwise $k = k+1$, go back to 2.*
4. *Output all the bases, i.e., $\mathsf{B}_0, \mathsf{B}_{1,0}, \mathsf{B}_{1,1}, \cdots, \mathsf{B}_{K,1}, \cdots$.*

Intuitively, a basis B_k at level-k represents a *minimal-set* of constraints that *support* the function value at the current level—removing any of the member constraints $b \in \mathsf{B}_k$ will further lower the function value. As such, solving the LP-type problem offers a means to identify which constraints are the most active ones—in the context of robust-estimation they often relate to the outliers.

7. The L_∞ triangulation is an LP-type problem

So far, we have introduced the LP-type framework, and some of its main results.

However, in order to apply the framework to the triangulation problem, a critical question must be answered: *is the L_∞ triangulation problem an LP-type problem* ?

We now proceed to give an affirmative answer and provide a proof for it. For convenience, repeat the L_∞ triangulation formulation here. This time the objective function (i.e.,the sub-level α) is explicitly expressed.

$$\min_{\mathbf{x}} \alpha, \text{ subject to,} \tag{2}$$

$$\forall i, \|\mathsf{A}_i \mathbf{X}\| \leq \alpha \mathsf{P}_{i;3} \mathbf{X}, \text{and } \mathsf{P}_{i;3} \mathbf{X} \geq 0. \tag{3}$$

Result 7.1. ***(Main Result)***

The L_∞ multiview triangulation is an LP-type problem; Moreover, the L_∞ multiview triangulation with at most k outliers is an LP-type problem.

Proof. The proof is done by specifically constructing an LP-type problem from Eq.(2). Define the universe set H as the set of all constraints, including both the second-order-cone conditions and the chirality conditions. Define a function w whose value is the minimal value of the sub-level α, while satisfying given constraints. In other words, w is the minimal objective function value at the optimizer point $\mathbf{X}^*$ which itself is located in the intersecting region of the constraints. Clearly, all such ws are linearly ordered. So far we have constructed a pair of (H, w).

Next, we need to verify that the two axioms in the definition of the LP-type are satisfied. Axiom-1 (monotonicity) is quite obvious: adding more constraints to a set can only further constrain the set, so its function value can never decrease. Thus the monotonicity holds. It now remains to show that axiom-2 (locality) is also satisfied. Since the L_∞ triangulation problem can be solved through the SOCP level-set bisection method. Each iteration of the bisection amounts to solving a convex feasibility problem. Let us consider a non-degenerate feasible case, i.e., when α is sufficiently large so that all constraints are satisfied (i.e., their intersections are non-empty) and a single (non-degenerate) optimum exists. In this case, decreasing α will *never* increase the areas of the intersecting region. In other words, the feasible regions of different levels are *nested*. Suppose that at two different levels we have two feasible regions with the same function value w^*. Then it is easy to check that: if a constraint $h \in \mathsf{H}$ violates the smaller feasible region it must also violate the larger one. (The converse is also true: if h violates the larger region then it also violates the smaller one, due to the non-degeneracy.) This says that, the locality axiom holds. For the outlier case, the proof is similar $\square$

To analyze the complexity of the obtained LP-type problem, we have another result which has useful practical implication.

Result 7.2. *The combinatorial dimension of the L_∞-triangulation-induced LP-type problem is finite, and is bounded from above by 4, i.e., $d(\mathsf{H}_\infty, \alpha) \leq 4$.*

One way to prove this is through the use of Helly's celebrated theorem ([2]): given $\mathcal{A}$ a finite family of convex sets in $\mathbb{R}^{\mathrm{d}}$. If every $\mathrm{d}+1$ members of $\mathcal{A}$ have a point in common, then there is a point common to all members of $\mathcal{A}$. Example: if *every* 3 out of n disks in the 2D plane intersect at one point, then all these n disks must intersect at one point.

8. The proposed triangulation algorithm

The main algorithm

The top-level structure of our algorithm is given below:

Algorithm 8.1. ***LP-type for L_∞ triangulation with outliers***

Input*: a triangulation problem, and a level K.*
Output*: all bases and the global optimum up to level K.*

1. *Set level k =0. Call SOCP bisection algorithm (i.e., alg-3.2) to find an L_∞ triangulation (i.e. the solution at level $k = 0$).*
2. *Call algorithm-(6.7), which consists of the following three steps:*
   (a) *use the* basis-finding subroutine *(see below) to find the initial root basis B_0;*
   (b) ***while*** *$k < K$,* ***do*** *generate new bases by calling the* basis-change subroutine *(see below); $k = k+1$;* ***end-do****;*
   (c) *Remove any redundancies in the obtained k-th level bases, and output the bases.*
3. *For each of the obtained bases, compute their function values, find the smallest value and the corresponding global minimizer, output them as the final solution;* ***End.***

To actually apply the above algorithm some non-trivial details must be filled in. Specifically, we need to provide subroutines for violation-test, basis-finding and basis-change.

Violation-test subroutine

Violation-test is easy to implement, involving only a single step SOCP. Given a constraint set G which is initially feasible. Test a single constraint h: if adding h to G makes $\mathsf{G} \cup h$ infeasible, then report 'h violates G'. The violation-test subroutine is frequently invoked by other subroutines.

Basis-finding subroutine

The basis-finding subroutine accepts as input a set of *feasible* constraints G, and output a basis B, which is a subset of G. We propose the following algorithm which is guaranteed to produce a basis.

Algorithm 8.2. ***Basis-finding subroutine***

Input: a set of constraint G, and its optimal function value α^*_{G}.
Output: one basis $\mathsf{B}(\mathsf{G})$.

1. *let $\alpha = \alpha^*_{\mathsf{G}} - \epsilon$ to make G infeasible, where ϵ is a infinitesimal positive number;*
2. *for each member g of G, drop it temporarily from G and do an SOCP-feasibility-test; if $(\mathsf{G} \setminus g)$ is still infeasible, then drop it permanently; otherwise, return it to G;*
3. *If all members have been tested, $output\mathsf{B}(\mathsf{G}) = \mathsf{G}$.*

Based on the definition of basis it is easy to see why this subroutine works.

Basis-change subroutine

The basis-change task is: given a basis B, and a constraint h not in B_k, find a basis for $(\mathsf{B} \cup h)$. This task can be fulfilled by applying the basis-finding subroutine to the set $(\mathsf{H} \backslash \mathsf{V}(\mathsf{B}) \backslash b)$ (c.f. algorithm-(6.7)). This procedure eventually invokes an SOCP bisection procedure. Since most of the bisections are performed locally (i.e., only bisecting a small range interval), the computational overhead is low.

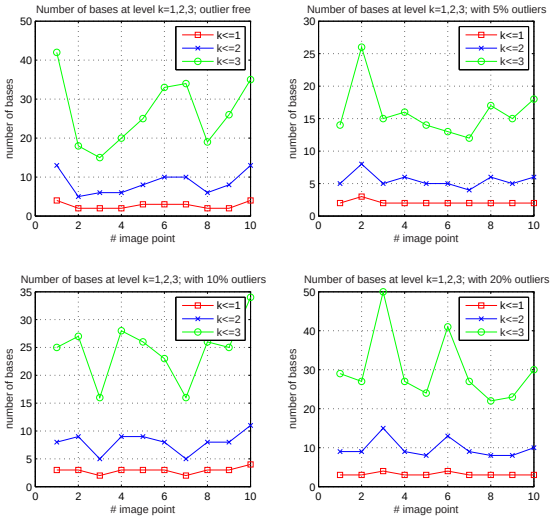

Figure 3. Y-axis: number of bases at level k; X-axis: image point.

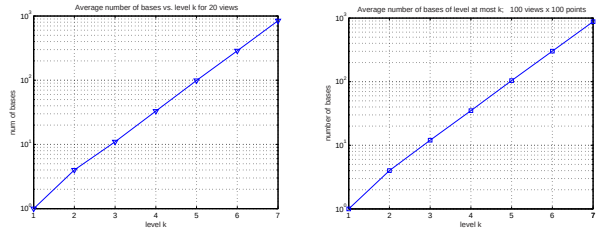

Figure 4. Number of bases vs. levels; (in semi-log-axis); left: 20-views case; right: 100-views case; two curves are nearly identical which implies that the number-of-bases is independent of the number of views.

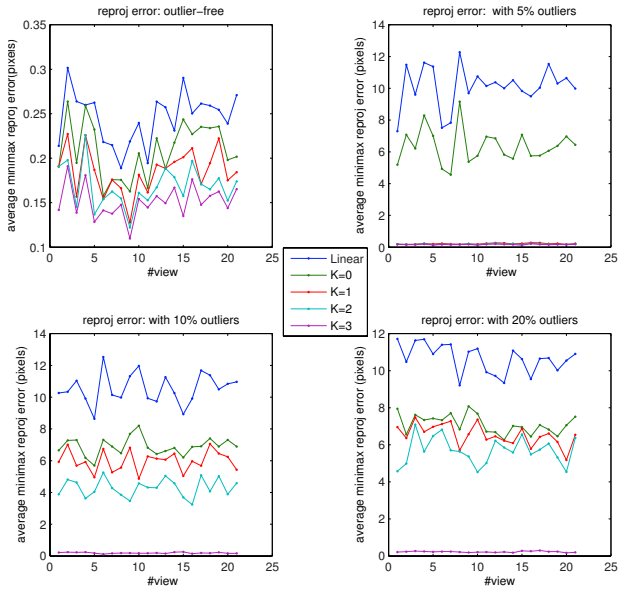

Figure 5. Min-max residual (at level k) vs. View; The percentiles of outliers are 0%, 5%; 10%,20% in sub-figure 1–4, respectively. The min-max errors obtained by a linear method are also shown for comparison. This figure is better viewed in color.

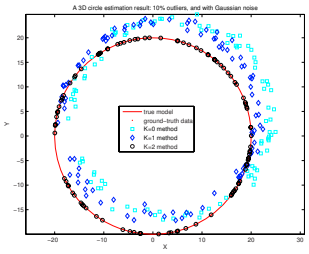

Figure 6. 3D circle experiment: Reconstructed circle at different level k. When $k = 0$ the obtained result is almost identical to the ground-truth.

9. Experimental validation

The above LP-type framework looks rather abstract. However, to actually implement the algorithm is simple. A central algorithmic component is the SOCP feasibility test, which will be frequently called by other higher-level routines. We implement the whole program in less than 200 lines of Matlab code. The adopted SOCP solver is the SeDuMi [15].

We have conducted three experiments to validate our theory and algorithm. All experiments have obtained convincing successful results. Some of which even provide new theoretical insight.

In the **first experiment**, we generate a synthetic 3D scene containing 100 points. We take 21 images from 21 positions. This amounts to a 21-view 100-point triangulation problem. We add Gaussian noise, as well as different proportions of outliers to the $21,000$ image points. We ensure that each 3D point has at most 3 corresponding image points corrupted by a 10-pixel uniform noise (as outlier), and each view has at least one such corrupted image point. This amounts to about 20% outliers. Then use our new LP-type algorithm to do the triangulation.

The purpose of the first experiment is to answer the following question: *up to a given level k, how many bases are there* ? This is actually equivalent to asking: *how many tests are needed in order to ensure a guaranteed k-level global optimality*?

Answer to this question is given in fig-3. We applied our algorithm to the 100 points. Each point is processed independently. As an example, we show the results for 10 points. It reveals that up to level $k \leq 3$ the number of bases is limited to the range of 20-50. This result is very encouraging. Otherwise this number would be 1562 if a brute-force search were used for solving the 21-view triangulation problem.

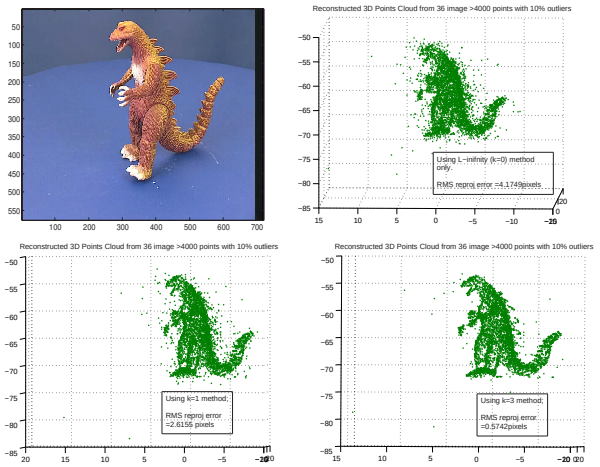

Figure 7. Experiment results on the dinosaur sequence. Top left: one of the input images; Others: the rms residual errors at k=0,1,3 are 4.17,2.61,0.57 pixels, resp.

In terms of computation time, one SOCP-test for the triangulation problem costs only about 0.03 seconds on a modest P4 machine.

Fig-5 gives the min-max residual errors over 21 views after applying our algorithm to level-k. At $k = 3$ (i.e. about $10\% - 15\%$ outliers) the min-max residual error is reduced to about 0.2 pixels, clearly indicating that most of the outliers have been removed successfully. In conclusion, this figure illustrates that our method does effectively remove all outliers.

Next, we want to examine how the number of bases increases as the level k grows. So we test for large k. Fig-4 gives the results for 20 views and 100 views cases. We plot them in $\log_{10}$-axis, and surprisingly find that it fits well with a straight line. Whilst for the moment we do not have a theoretic explanation to this, it does provide a practical guideline in predicting the number of bases.

In the **second experiment**, we synthesize 100 points on a 3D circle, and take 20 views of it. About 10% outliers are added into the image points. Our algorithm produces the results shown in fig-6. The right sub-figure gives the 3D reconstruction errors at different levels. Again, at level-3 the reconstruction error is sufficiently low, indicating that all outliers have been removed successfully.

In the **third experiment**, we perform experiments on real data set (e.g. the dinosaur sequence). About 10% image points are perturbed by 5-pixel uniform noise as an simulation of the outliers. The 3D reconstruction results obtained at k=0,1,3 are shown in fig-7. The reconstruction quality improves as k increases. The average RMS re-projection error is reduced to about 0.6 pixels at level-3, which is already close to the ideal outlier-free situation.

10. Conclusion

We have shown that the L_∞ triangulation is actually an LP-type problem. We provide a practical algorithm for implementing this idea for robust L_∞ triangulation. Our algorithm handles outliers in an exact and predictable manner. It always finds the unique global optimum at relatively low cost. Quantitative comparisons among different outlier-removal methods are planned as our future work.

Since our method has received convincing success for the triangulation problem, we expect more general applications to other problems. There have been shown that many geometric vision problems accommodate a simpler L_∞ solution, see [6] [3] for example. So long as these problems *are* LP-type problems, our method could be applied to find a guaranteed global optimum, even in the presence of outliers.

Currently, the computational burden time is very heavy. In fact, if we strictly follow the LP-type theory framework, much faster algorithms (e.g., in linear expectation time, or sub-exponential time) are clearly possible.

Potentially, the central idea of our outlier-removal method can also be applied to other problems where an exact robustness is desired.

Acknowledgements

The author wishes to thank R.Hartley for generous research support, many encouragements and inspiring discussions. Thank all three anonymous reviewers for their invaluable comments. Thank Dr C Shen for discussion and proofreading. The proof of the Main-Result is adapted from Carl Olsson's comment: personal communication with C. Olsson and F Kahl.

References

[1] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004. 3

[2] J. Goodman and J. O'Rourke. *Handbook of dis. and comp. geometry, 2nd ed*. CRC Press, 2004. 1, 5, 6

[3] R. Hartley and F. Schaffalitzky. L-infty minimization in geometric reconstruction problems. *CVPR*, 1:504–509, 2004. 1, 2, 8

[4] R. Hartley and P. Sturm. Triangulation. *CVIU*, 62:146–157, 1997. 1

[5] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, UK, 2004. 3

[6] F. Kahl. Multiple view geometry and the l_∞-norm. *In Proc. ICCV*, 2005. 1, 2, 8

[7] F. Kahl and D. Henrion. Globally optimal estimates for geometric reconstruction problems. *In Proc. ICCV*, 2005. 1

[8] Q. Ke and T. Kanade. Quasiconvex optimization for robust geometric reconstruction. *In Proc.ICCV*, pages 986–993, 2005. 1, 2

[9] J. Matousek. On geometric optimization with few violated constraints. *Symposium on Computational Geometry*, pages 312–321, 1994. 5

[10] J. Matousek, M. Sharir, and E.Welzl. A subexponential bound for linear programming. *Algorithmica*, 16:498, 1996. 4, 5

[11] P. J. Rousseeuw and A. M. Leroy. *Robust regression and outlier detection*. John Wiley & Sons, Inc., New York, NY, USA, 1987. 1

[12] M. Sharir and E. Welzl. A combinatorial bound for linear programming and related problems. *STACS '92: Proceedings of the 9th Annual Symposium on Theoretical Aspects of Computer Science*, pages 569–579, 1992. 4, 5

[13] K. Sim and R. Hartley. Recovering camera motion using linfty minimization. *in Proc. CVPR*, 2006. 1

[14] K. Sim and R. Hartley. Removing outliers using the l-infty norm. *in Proc. CVPR*, 2006. 1, 2

[15] J. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(12):625–653, 1999. 7