An Adversarial Optimization Approach to Efficient Outlier Removal

Jin Yu Anders Eriksson Tat-Jun Chin David Suter
The Australian Centre for Visual Technologies
The University of Adelaide, South Australia
{jin.yu, anders.eriksson, tjchin, dsuter}@cs.adelaide.edu.au

Abstract

This paper proposes a novel adversarial optimization approach to efficient outlier removal in computer vision. We characterize the outlier removal problem as a game that involves two players of conflicting interests, namely, optimizer and outlier. Such an adversarial view not only brings new insights into various existing methods, but also gives rise to a general optimization framework that provably unifies them. Under the proposed framework, we develop a new outlier removal approach that is able to offer a much needed control over the trade-off between reliability and speed, which is otherwise not available in previous methods. The proposed approach is driven by a mixed-integer minmax (convex-concave) optimization process. Although a minmax problem is generally not amenable to efficient optimization, we show that for some commonly used vision objective functions, an equivalent Linear Program reformulation exists. We demonstrate our method on two representative multiview geometry problems. Experiments on real image data illustrate superior practical performance of our method over recent techniques.

1. Introduction

Model fitting as a fundamental problem in computer vision underlies numerous applications. It is typically posed as minimization of an objective function on some input data. For instance, a particularly successful line of research in multiview geometry [5, 7] casts various multiview reconstruction problems as minimizing the maximal reprojection error (*i.e*. the L_∞ norm of the error vector) across all measurements. Such methods have demonstrated excellent performance on a variety of applications. However, they are also known [19] to be extremely vulnerable to outliers, which unfortunately are often unavoidable, due to imperfections in data acquisition and preprocessing. This paper aims to provide a principled and unified framework for outlier removal by viewing it as a zero-sum two-player game. This not only brings new insights into various existing outlier removal schemes, but also gives rise to a general problem formulation that unifies them.

RANSAC [1] is probably the most commonly used method to cope with outliers. The hope is that a "clean" data sample would be drawn such that a model that is representative of the genuine structure in the data can be estimated purely from the noise-free sample. RANSAC has met with great success in computer vision, though its efficiency largely relies on fast computation of candidate models. Where it is not possible, RANSAC has to resort to "cheap" but less accurate alternatives. For instance, when applied to multiview geometry problems, RANSAC (or its accelerated variants, *e.g*. [13]) can only afford to compute candidate models from 2 or 3 views, instead of the full track of images, thus resulting in some outliers being undetected in long tracks. Even though the undetected outliers often only constitute a small portion of the data ($< 15\%$), they can still cause disastrous results, especially when L_∞-norm-based methods are used. Such a limitation of RANSAC motivates the development of various alternative outlier removal techniques [8, 9, 14, 18, 19].

Among them, the work of [8, 9] focuses on the optimization of robust objective functions tailored for multiview geometry problems. Similar to the Least-Median-of-Squares method [16], the method proposed in [8] approximately minimizes the m^{th} smallest reprojection error, m being less than the overall number of the data. Unfortunately, since the resulting optimization problem has multiple local minima, the obtained solution is most likely to be sub-optimal. Li [9] later proposed a search-based method that seeks to identify a pre-specified number of data that produce the smallest cost, but it is done at considerable computational expense.

Another branch of recent work casts outlier removal as identifying a subset I of the input data such that a model $\mathbf{w}$ (*e.g*. a homography matrix) exists with

$$f_i(\mathbf{w}) \leq \epsilon, \quad \forall i \in I \subseteq \{1, \cdots, N\}, \tag{1}$$

where the cost function $f_i(\mathbf{w})$ evaluates the discrepancy between datum i and $\mathbf{w}$, and $\epsilon \geq 0$ is a given error tolerance. Ideally, one would aim for the largest I, and there exists

2011 IEEE International Conference on Computer Vision
978-1-4577-1102-2/11/$26.00

work [3, 10] that targets this goal. However, these methods are either computationally intractable for high dimensional $\mathbf{w}$ (with a worst-case exponential complexity), or only tailored for a very specific class of applications. Sim and Hartley [19] relaxed the goal to finding an I of sufficient size that is representative of the underlying structure in the data. Exploiting the special properties of strictly quasi-convex objective functions, as commonly used in multiview geometry, they proposed to iteratively exclude data that generate the largest residual. This intuitive approach guarantees that at least one outlier is among the removed data. Although quite effective, it is computationally expensive, due to the need of solving a quasi-convex problem at each iteration.

More recent research [14, 18] introduces slack variables into the objective function, and devises outlier removal schemes by analysing slack values. Take the following *1-slack* objective function as an example:

$$\min_{s,\mathbf{w}} \; s \quad \text{s.t.} \quad f_i(\mathbf{w}) \leq \epsilon + s, \;\; \forall i, \tag{2}$$

where $s \in \mathbb{R}$ is a slack variable. Under such a problem formulation, data that generate the largest positive residual are guaranteed to contain at least one outlier. Based on this, an iterative outlier removal scheme was developed and shown to be equivalent to the strategy of [19], yet significantly faster. However, since each time the 1-slack method can only remove a very limited number of outliers, it often requires many iterations to entirely clean the data, thus is still too expensive to be widely applicable for practical use. In an attempt to accelerate the process, Olsson et al. [14] examined a variant of (2) that uses N slack variables:

$$\min_{s_i,\mathbf{w}} \; \sum_i s_i \quad \text{s.t.} \quad f_i(\mathbf{w}) \leq \epsilon + s_i, \;\; s_i \geq 0, \;\; \forall i \tag{3}$$

Essentially, (3) computes the L_1 norm of the vector of the "slacks" (or sum of infeasibilities). Using this problem formulation, data associated with positive s_i can be removed as outliers *in one shot*. The L_1 method is therefore fast and shown to work well in practice. Its limitation, however, is that the removed data may not include genuine outliers. This can lead to arbitrarily bad models (see *e.g.* Fig.1 in [14]). Also observed in some of our experiments is that as it removes outliers, the L_1 method tends to remove many inliers as well (Fig. 1, right), which renders it unreliable.

Noting that the 1-slack and L_1 methods represent two ends of the spectrum of reliability versus speed, we aim for a general problem formulation that is able to strike a balance between these two performance factors. Our method is inspired by the classical minimax formulation of a zero-sum two-player game [17] in which one player makes moves to maximize his payoff, while the other player tries to minimize the payoff of his opponent. In the context of outlier removal, the optimizer can be seen as a player who tries to minimize the objective function, whereas the interest of outliers is to make the minimal objective as worst as possible. Such an adversarial view can be translated to a convex-concave optimization problem. Solving it allows us to effectively identify a set of "offending" data that are responsible for a "bad" objective value. To identify all outliers, one can remove this set of data, and repeat the process until the remaining data are clean. Fig. 1 depicts one run of the three outlier removal schemes on sample line-fitting data with 35% of outliers. It shows that our method (center) identifies noticeably more outliers than the 1-slack approach (left), yet achieving so in a less "aggressive" manner than the L_1 approach (right). Later, we prove that the two existing methods are special cases of our approach.

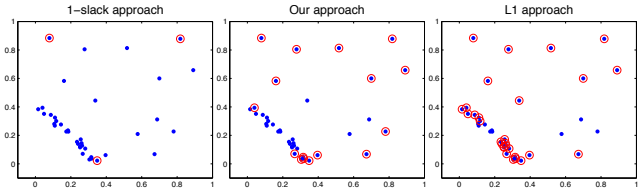

Figure 1. Behaviour of three outlier removal schemes on sample 2D line-fitting data (dots). Left: the 1-slack method (2) removes only 3 data points (circles), hence many runs are needed to clean the data, whereas the overly "aggressive" L_1 method (3) (right) mistakenly removes many inliers. The proposed approach (center) offers a more balanced result: it is clearly more productive than the 1-slack method, yet not sacrificing too many inliers.

The paper proceeds as follows: We first provide a general description of our problem formulation. Sec. 3 shows how to optimally solve the resulting optimization problem. In Sec. 4 we discuss related work. Experiments on real multiview geometry data are reported in Sec. 5. Sec. 6 concludes the paper with an outlook and discussion.

2. Problem Formulation

We introduce an adversarial problem formulation for outlier removal, and relate our method to recent techniques.

2.1. Outlier Removal as a Minmax Problem

We view the outlier removal problem as a zero-sum game that involves two competing players: optimizer and outlier. The optimizer aims to find a model that achieves the minimal objective value, whereas outliers' strategy is to make the minimal objective value as worse as possible. To mathematically formulate such a game, we first introduce a binary variable $\boldsymbol{\pi} \in \{0,1\}^N$ (All vectors are by default column vectors.); each of its entries, denoted by π_i, corresponds to a datum; $\pi_i = 1$ indicates that datum i is an outlier; otherwise $\pi_i = 0$. Taking into account the fact that outliers only constitute part of the data, we enforce $\boldsymbol{\pi}^\top \mathbf{1} = K$, where $\mathbf{1}$ is a vector of all ones and $K \in [1, N]$. We use the sum

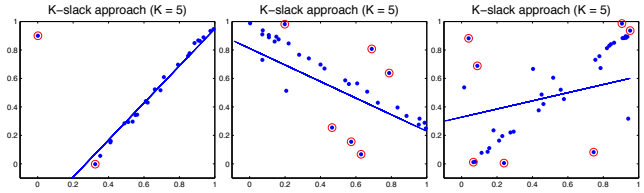

Figure 2. The model (line) and the potential outliers (circles) found by solving (5) on various 2D line-fitting data (dots). The potential outliers produce larger residuals (w.r.t. the found model) than other data, and clearly contain genuine outliers.

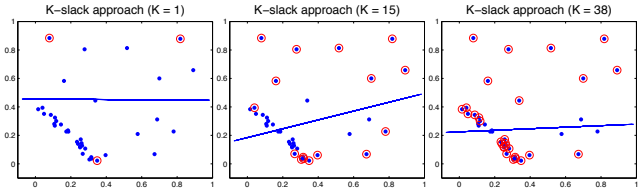

Figure 3. The optimal model (line) and the potential outliers (circles) found by solving (5) with various K values on the same data as in Fig. 1. Our approach reduces to the 1-slack (Fig. 1, left) *resp.* L_1 (Fig. 1, right) methods with $K = 1$ *resp.* N (38 here).

of non-negative slacks to measure model fitting error. The resulting adversarial problem takes the following form:

$$\min_{\mathbf{s},\mathbf{w}} \max_{\boldsymbol{\pi}} \ \boldsymbol{\pi}^\top \mathbf{s} \tag{4a}$$

$$\text{s.t.} \quad f_i(\mathbf{w}) \leq \epsilon + s_i, \ \ s_i \geq 0, \ \ \forall i \tag{4b}$$

$$\boldsymbol{\pi} \in \{0,1\}^N, \ \ \boldsymbol{\pi}^\top \mathbf{1} = K, \tag{4c}$$

where $\mathbf{s} \in \mathbb{R}_+^N$ collects N slacks s_i. The outer $\min_{\mathbf{s},\mathbf{w}}$ and inner $\max_{\boldsymbol{\pi}}$ operations characterize the strategies of the optimizer and outliers, respectively. Denote the objective of (4) by $J(\mathbf{s},\mathbf{w},\boldsymbol{\pi})$. Then, for a joint action $(\mathbf{s},\mathbf{w},\boldsymbol{\pi})$, the payoffs of the optimizer and outliers are $-J(\mathbf{s},\mathbf{w},\boldsymbol{\pi})$ and respectively, $J(\boldsymbol{\pi},\mathbf{s},\mathbf{w})$. In the game equilibrium, the optimizer minimizes the sum of the K-largest slacks, which can be identified by the K positive entries of $\boldsymbol{\pi}$.

2.2. A Relaxed Minmax Formulation

The problem (4) seems difficult to optimize due to the presence of the discrete constraint on $\boldsymbol{\pi}$. Fortunately, the linearity of (4) in $\boldsymbol{\pi}$ allows us to relax the constraint $\boldsymbol{\pi} \in \{0,1\}^N$ to $\boldsymbol{\pi} \in [0,1]^N$ without changing the optimal objective value. To see this, for the moment, let us relax the discrete constraint to obtain the following relaxation:

$$\min_{\mathbf{s},\mathbf{w}} \max_{\boldsymbol{\pi}} \ \boldsymbol{\pi}^\top \mathbf{s} \tag{5a}$$

$$\text{s.t.} \quad f_i(\mathbf{w}) \leq \epsilon + s_i, \ \ s_i \geq 0, \ \ \forall i \tag{5b}$$

$$\boldsymbol{\pi} \in [0,1]^N, \ \ \boldsymbol{\pi}^\top \mathbf{1} = K. \tag{5c}$$

The inner maximization problem of (5) is simply a Linear Program (LP). This ensures that the optimal $\boldsymbol{\pi}$ can be attained at vertices of the linear constraints on $\boldsymbol{\pi}$ (5c). Since the vertices of these constraints are integral, the optimal $\boldsymbol{\pi}$ is integral. The relaxation (5) is therefore an *equivalent reformulation* of its mixed-integer counterpart (4).

We can see that for a given $\mathbf{s}$, the optimal solution to the inner $\max_{\boldsymbol{\pi}}$ problem can be achieved by choosing the K-largest entries of $\mathbf{s}$ as outliers. Denote the optimal solution to (5) by $(\mathbf{s}^*,\mathbf{w}^*,\boldsymbol{\pi}^*)$. Then, for the purpose of outlier removal, one can simply remove the data in the following *potential outlier set*:

$$\mathcal{O} := \{i \in \{1,\cdots,N\} \mid s_i^* \geq s^k > 0\}, \tag{6}$$

where s^K is the K^{th} largest positive entry of $\mathbf{s}^*$. Note that $\mathcal{O}$ can contain more than K data (indices) if several entries of $\mathbf{s}^*$ share the value of s^K; in this case $\boldsymbol{\pi}^*$ is not unique, and can be fractional as it can spread the mass of 1 over the data with the same slack value of s^K. Conversely, if $\mathbf{s}^*$ has fewer than K positive entries, then $|\,\mathcal{O}\,| < K$, where $|\cdot|$ computes the cardinality of a set.

Before discussing the optimization strategy for solving (5), in Fig. 2 we first provide a pictorial illustration of the solution to (5) on several sample 2D line-fitting data (dots). Given a datum $(\mathbf{x}_i, y_i)$, we define the cost function as:

$$f_i(\mathbf{w}) := |\,\mathbf{x}_i^\top \mathbf{w} - y_i|, \tag{7}$$

where $\mathbf{x}_i$ consists of the x coordinate of datum i and 1, and y_i is the corresponding y coordinate. We used $\epsilon = 0.05$ and $K = 5$ throughout. Fig. 2 shows that the data in $\mathcal{O}$ (circles) produce larger residuals with respect to the optimal model $\mathbf{w}^*$ (line) than other data. Clearly, among the data in $\mathcal{O}$ exist outliers. In Th. A.1 (Appendix A), we prove that if $|\,\mathcal{O}\,| \geq K$ (Fig. 2, center and right), then $\mathcal{O}$ contains at least one outlier. Note that in practical sense this is a very pessimistic lower bound on the number of identified outliers. In the case of $|\,\mathcal{O}\,| < K$ (Fig. 2, left), we immediately know that $f_i(\mathbf{w}^*) \leq \epsilon, \forall i \notin \mathcal{O}$, because $s_i^* = 0, \forall i \notin \mathcal{O}$.

2.3. Connections with Existing Methods

This section connects our method with the 1-slack and L_1 approaches through its parameter K. It is clear that if $K = N$, the minmax problem (4) (and hence (5)) reduces to the L_1 problem (3), since in this case all π_i are forced to take on the value of 1. At the other extreme of $K = 1$, (4) is equivalent to the 1-slack problem (2). To see this, first note that solving (2) is equivalent to minimizing the maximal infeasibility measured by the single slack s. This can be formulated as the following N-slack minmax problem:

$$\min_{s_i,\mathbf{w}} \max_{i} \ s_i \quad \text{s.t.} \quad f_i(\mathbf{w}) \leq \epsilon + s_i, \ \ \forall i. \tag{8}$$

If there exist outliers in the data, the maximal s_i for (8) must be positive, hence we can add the constraints: $s_i \geq 0, \forall i$ to (8) without changing its optimal objective value. It is then easy to recognize that (8) is simply (4) with $K = 1$.

Fig. 3 illustrates the influence of K on the solution to (5). The data in Fig. 1 were again used here. By comparing to Fig. 1 (left and right), we can see that our method reduces to the 1-slack and L_1 methods with $K = 1$ and respectively, N. As can be seen, with different K values, models (lines) returned by solving (5) are different. The 1-slack model (left) is determined by the worst residual (*i.e.* the largest slack), while the L_1 model (right) is influenced by the residuals of all data. These two settings overlook the fact that the size of outlier population is almost always *between* 1 and N and it is the joint effort of this subpopulation of the data that degrades the model quality. To capture this phenomenon, the introduction of the K parameter as in our problem formulation is therefore necessary. Fig. 3 (center) also shows that with an intermediate K, more "offending" data are identified than setting $K = 1$, while unlike the $K = N$ case, the majority of them are indeed outliers.

Note that the three methods share one limitation, that is, they can mistakenly remove inliers. Therefore, they are not recommended for use on severely contaminated data, that may require many runs to clean, hence increasing the chance of removing many inliers. For this reason, all three methods are best used at a refinement stage after a crude RANSAC-like outlier filtering. Nevertheless, our experiments suggest that 1-slack and our method with an intermediate K can still handle 15%-35% of outliers, while the performance of the L_1 method is unstable and data-dependent.

3. An Equivalent LP Reformulation

One potential technical difficulty with the minmax formulation (5) is that it is generally not amenable to efficient convex optimization. However, if the constraints in (5b) are linear in $\mathbf{w}$, we can derive an equivalent LP reformulation. This significantly simplifies the optimization.

Since the objective function (5a) is bilinear in $\mathbf{s}$ and $\boldsymbol{\pi}$, our first step is to decouple these two terms so as to avoid introducing non-convexity into the optimization. We do so by analysing the dual of the inner $\max_{\boldsymbol{\pi}}$ problem. Introducing Lagrange multipliers $\alpha \in \mathbb{R}$ and $\boldsymbol{\beta} \in \mathbb{R}^N_+$, we can write the dual of the inner $\max_{\boldsymbol{\pi}}$ problem as:

$$\min_{\alpha,\boldsymbol{\beta}} \ \alpha K + \boldsymbol{\beta}^\top \mathbf{1} \quad \text{s.t.} \ \ \alpha \mathbf{1} + \boldsymbol{\beta} \geq \mathbf{s}, \ \ \boldsymbol{\beta} \geq \mathbf{0}, \tag{9}$$

where $\mathbf{0}$ denotes a column vector of all zeros. Replacing the inner $\max_{\boldsymbol{\pi}}$ problem in (5) with (9) gives the following

Algorithm 1 Outlier Removal by Minmax Optimization

input data and the choice of K
output a subset I of the data such that
$\exists\, \mathbf{w} \in \mathbb{R}^d : f_i(\mathbf{w}) \leq \epsilon, \ \forall i \in I$
repeat
  solve (10) on the current data to obtain the optimal solution $(\alpha^*, \boldsymbol{\beta}^*, \mathbf{w}^*, \mathbf{s}^*)$
  if $\alpha^* K + \boldsymbol{\beta}^{*\top} \mathbf{1} > 0$ **then**
    construct the potential outlier set $\mathcal{O}$ via (6)
    remove the data collected in $\mathcal{O}$
  end if
until $\alpha^* K + \boldsymbol{\beta}^{*\top} \mathbf{1} = 0$
optional: restore inliers from the removed data

equivalent minimization problem:

$$\min_{\alpha,\boldsymbol{\beta},\mathbf{s},\mathbf{w}} \ \ \alpha K + \boldsymbol{\beta}^\top \mathbf{1} \tag{10a}$$

$$\text{s.t.} \ \ \ f_i(\mathbf{w}) \leq \epsilon + s_i, \ \ \forall i \tag{10b}$$

$$\alpha \mathbf{1} + \boldsymbol{\beta} \geq \mathbf{s}, \ \ \boldsymbol{\beta} \geq \mathbf{0}, \ \ \ \mathbf{s} \geq \mathbf{0}. \tag{10c}$$

It is a convex problem if $f_i(\mathbf{w})$ is convex; all cost functions considered in this paper are convex. Furthermore, if $f_i(\mathbf{w})$ is linear, then (10) is simply an LP. In this case we can represent (10b) in compact matrix form as

$$\mathbf{A}_j \, \mathbf{w} \leq \mathbf{b}_j + \mathbf{s}, \ \ \forall j, \tag{11}$$

where the matrix $\mathbf{A}_j \in \mathbb{R}^{N \times d}$ and the vector $\mathbf{b}_j \in \mathbb{R}^N$ contain the coefficients of linear constraints enforced by (10b). The values of the coefficients and the range of j are determined by the exact form of the cost function $f_i(\mathbf{w})$. Take (7) for instance, it can be realized by two linear constraints:

$$\mathbf{x}_i^\top \, \mathbf{w} \leq y_i + \epsilon \quad \text{and} \quad -\mathbf{x}_i^\top \, \mathbf{w} \leq -y_i + \epsilon.$$

Expressing these constraints for all i in the matrix form of (11), we obtain two coefficient matrices: $\mathbf{A}_1 := [\mathbf{x}_1^\top; \cdots; \mathbf{x}_N^\top]$ and $\mathbf{A}_2 := -\mathbf{A}_1$, with their corresponding coefficient vectors given by $\mathbf{b}_1 := [y_1; \cdots; y_N] + \epsilon \mathbf{1}$ and $\mathbf{b}_2 := -[y_1; \cdots; y_N] + \epsilon \mathbf{1}$, respectively. Having solved the LP, one can simply sort the entries of the optimal $\mathbf{s}$ in non-ascending order to form the potential outlier set $\mathcal{O}$ (6), remove all the data in $\mathcal{O}$, and repeat the process on the remaining data until the optimal objective value of (10) reduces to 0. Alg. 1 details our approach (named K-slack).

4. Related Work

Perhaps the closest in spirit to our paper is the work of Nguyen and Welsch [12], who tailored a maxmin formulation for outlier removal in least squares regression. Their

objective function takes the following form:

$$\max_{\boldsymbol{\pi}} \quad \min_{\mathbf{w}\in\mathbb{R}^d} \quad \sum_i \pi_i \, f_i(\mathbf{w})^2 \tag{12a}$$

$$\text{s.t.} \quad \boldsymbol{\pi} \in [0,1]^N, \quad \boldsymbol{\pi}^\top \mathbf{1} = K, \tag{12b}$$

where $f_i(\mathbf{w})$ is defined as (7). To optimize (12), they first substitute the inner $\min_{\mathbf{w}}$ problem with its closed-form solution so as to convert (12) to a maximization problem in $\boldsymbol{\pi}$ only, which is then further reformulated as a semi-definite program (SDP) for convex optimization. The resulting SDP has $(N+1)$ variables and $(N+1)$ corresponding real symmetric matrices of size $(d+1)\times(d+1)$ in the linear matrix inequality constraints of the SDP.

Compared to our approach, this SDP-based method is clearly less general since it restricts the cost function to be in the least-squares form, which may not be desirable for some computer vision problems [4]. Moreover, the computational complexity of solving the SDP reformulation of (12) via an SDP solver such as SeDuMi [20] is in the order of $(N+1)^2(d+1)^{2.5}$. This can be prohibitively expensive for high dimensional problems such as the Structure from Motion problems considered in Sec. 5.2, where we need to estimate 3D coordinates of all the observed 2D image points as well as translation parameters of a set of cameras.

One possible remedy for the computational issue of the SDP-based method is to use our minmax formulation (5) to rewrite (12) as an *equivalent* but less expensive Second-Order Cone Program (SOCP). The equivalence of (5) and (12) is derived from the fact that the $\min_{\mathbf{s},\mathbf{w}}$ and $\max_{\boldsymbol{\pi}}$ operations in (5) are interchangeable if $f_i(\mathbf{w})$ is continuous and convex. With such an $f_i(\mathbf{w})$, the feasible region of $\mathbf{w}$, as specified by (5b), is convex and closed. It is also clear that the feasible region of $\mathbf{s}$ is convex and closed; and that of $\boldsymbol{\pi}$ is convex and bounded. Invoking the minimax theorem [15, Corollary 37.3.2] allows us to swap $\min_{\mathbf{s},\mathbf{w}}$ and $\max_{\boldsymbol{\pi}}$ in (5) without changing the optimal object value. Setting $\epsilon = 0$ and replacing $f_i(\mathbf{w})$ in (5b) with the convex and continuous squared cost as used in (12), we obtain an equivalent reformulation of (12). Squaring $f_i(\mathbf{w})$ in (5b) produces a set of quadratic constraints, which can be easily converted to Second Order Cone constraints. This leads to a computationally cheaper SOCP than the SDP. If further speedup is preferred, then a linear cost function is probably more appropriate, as in that case one only need to solve an LP (10).

5. Experiments

We tested our method (K-slack, Alg. 1) on two multiview geometry problems: Homography estimation and Structure from Motion (SfM) estimation with known camera rotation. We implemented our method in MATLAB with MOSEK LP solver.[1] All experiments were run on a machine with 2.67GHz Intel quad core processors and 4 GB of RAM.

[1] Available from http://www.mosek.com.

Figure 4. SIFT keypoint matches on two image pairs: Keble (left) and Graf (right), for Homography estimation in Sec. 5.1.

5.1. Homography Estimation

Our first set of experiments were conducted on keypoint matches for homography estimation, where a pair of keypoints: $\mathbf{u} := (x_i, y_i, 1)$ and $\mathbf{u}' := (x_i', y_i', 1)$ (in homogeneous plane coordinates) are related by $\mathbf{u} \simeq \mathbf{H}\,\mathbf{u}'$, where $\mathbf{H}$ is a 3×3 homography matrix. We fixed the right bottom entry of $\mathbf{H}$ to 1. (See *e.g.* [7] for more details on such a parameterization.) The number of unknowns is 8. We used the same reprojection error function as considered in [14]:

$$\begin{aligned} V_i(\mathbf{H}) &:= \max(|\,\mathbf{h}_1^\top\,\mathbf{u}_i' \,/\, \mathbf{h}_3^\top\,\mathbf{u}_i' - x_i|, |\,\mathbf{h}_2^\top\,\mathbf{u}_i' \,/\, \mathbf{h}_3^\top\,\mathbf{u}_i' - y_i|) \\ &= \max(|\frac{(\mathbf{h}_1^\top - x_i\,\mathbf{h}_3^\top)\,\mathbf{u}_i'}{\mathbf{h}_3^\top\,\mathbf{u}_i'}|, |\frac{(\mathbf{h}_2^\top - y_i\,\mathbf{h}_3^\top)\,\mathbf{u}_i'}{\mathbf{h}_3^\top\,\mathbf{u}_i'}|), \end{aligned} \tag{13}$$

where $\mathbf{h}_j^\top$ denotes the j^{th} row of $\mathbf{H}$. The *cheirality* condition: $\mathbf{h}_3^\top\,\mathbf{u}_i' > 0$ is enforced for all data. Given an $\mathbf{H}$, a datum i is regarded as an inlier if $V_i(\mathbf{H}) \le \epsilon$, this requires

$$|(\mathbf{h}_1^\top - x_i\,\mathbf{h}_3^\top)\,\mathbf{u}_i'| \le \epsilon\,\mathbf{h}_3^\top\,\mathbf{u}_i', \tag{14}$$

similarly for the error along the y coordinate. Rearranging (14) and introducing a slack give the following constraint:

$$|(\mathbf{h}_1^\top - x_i\,\mathbf{h}_3^\top)\,\mathbf{u}_i'| - \epsilon\,\mathbf{h}_3^\top\,\mathbf{u}_i' \le s_i. \tag{15}$$

The slack $s_i \ge 0$ was also used to bound the deviation from the cheirality condition via the following constraint:

$$-\,\mathbf{h}_3^\top\,\mathbf{u}_i' \le s_i. \tag{16}$$

Note that (15) and (16) are in different forms from (5b), but since they are linear, the LP reformulation (10) still applies.

Two image pairs[2] were used in our experiments. Keypoint matches (Fig. 4) were established by SIFT matching [11]. The Keble data contain 167 matches and the Graf data 437. They were directly fed to K-slack without outlier prefiltering. The error tolerance ϵ was set to 2 pixels.

We investigated the performance of K-slack under the influence of various K values. Three performance measures were considered: overall CPU time required to clean the data, quality of the model returned by the last LP, and the overall number of removed matches. We measured model quality by Root Mean Square (RMS) reprojection error on

[2] Both obtained from http://www.robots.ox.ac.uk/∼vgg.

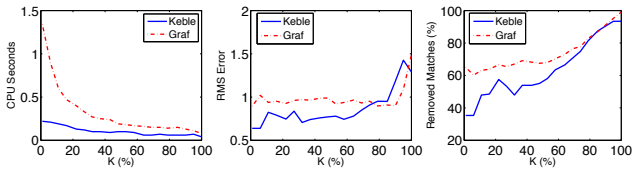

Figure 5. The influence of K on CPU time (left), RMS error (center), and the number of removed matches (in percentage of N).

Figure 6. The mosaics obtained by setting K to $\lceil 15\%N \rceil$ (top) versus the skewed results for $K = N$ (bottom).

the cleaned data. Fig. 5 (left) shows that on both datasets the required CPU time reduces with increasing values of K. This is not surprising because more data (hence potentially more outliers) are removed for a larger K, therefore fewer runs of LPs are needed. This also explains why the L_1 (*resp.* 1-slack) method is fast (*resp.* slow) in general. In terms of RMS error, Fig. 5 (center) shows that it maintains at a relatively low level for a wide range of K values from 1 (plotted as $K = \%0N$) to around $\lceil 60\%N \rceil$. A K value towards the upper bound of this range may seem a good choice for achieving good performance in both runtime and model quality. However, note from Fig. 5 (right) that there is a general increasing trend in the number of removed data as the value of K increases. As already illustrated in Fig. 3 (right), being too "aggressive" in removing data may lead to disastrous model fitting results. Therefore, to obtain a good compromise between reliability and speed, an intermediate K is preferable. Empirically, we found it sufficient to set $K = \lceil 5\%N \rceil$ to $\lceil 20\%N \rceil$. Fig. 6 (top) provides the mosaics obtained for a sample choice of K within this range. These results are clearly more satisfactory than the skewed mosaics (bottom) obtained by setting $K = N$. Meanwhile, the speedup gained from using a $K > 1$ is also apparent (Fig. 5, left), especially on the larger dataset (Graf).

5.2. SFM with Known Camera Rotation

We also evaluated our approach on the more challenging SfM estimation problem, where given 2D image points and rotation parameters of several cameras, we infer the 3D structure (3D point positions) of the scene and camera translation parameters. The calibration of each camera is given.

Let $\mathbf{P}_j = [\mathbf{R}_j \,|\, \mathbf{t}_j]$ be the 3×4 camera matrix of camera j, where the 3×3 rotation matrix $\mathbf{R}_j$ is assumed known, while the translation vector $\mathbf{t}_j$ is to be estimated. Parameterizing an unknown 3D point by $\mathbf{u}_i := (x_i, y_i, z_i, 1)$ and denoting the coordinates of its corresponding observation in image j by (x_{ij}, y_{ij}), we measured the reprojection error of point i in image j by the following linear cost function:

$$V(\mathbf{u}_i, \mathbf{t}_j) := \max(|\frac{\mathbf{r}_{j1}^\top \mathbf{u}_i + t_{j1}}{\mathbf{r}_{j3}^\top \mathbf{u}_i + t_{j3}} - x_{ij}|, |\frac{\mathbf{r}_{j2}^\top \mathbf{u}_i + t_{j2}}{\mathbf{r}_{j3}^\top \mathbf{u}_i + t_{j3}} - y_{ij}|),$$

where $\mathbf{r}_{jk}^\top$ and t_{jk} denote the k^{th} row of $\mathbf{R}_j$ and respectively, the k^{th} entry of $\mathbf{t}_j$. We required the depth of the reconstruction to be within a positive range:

$$d_1 \leq \mathbf{r}_{j3}^\top \mathbf{u}_i + t_{j3} \leq d_2. \tag{17}$$

The error along the x (similarly y) coordinate is bounded by a non-negative slack via a linear constraint:

$$|(\mathbf{r}_{j1} - x_{ij}\,\mathbf{r}_{j3})^\top \mathbf{u}_i + t_{j1} - x_{ij} t_{j3}| - \epsilon(\mathbf{r}_{j3}^\top \mathbf{u}_i + t_{j3}) \leq s_{ij}.$$

Similar to (16), the slack s_{ij} was also used to bound the deviation from the depth condition (17).

Tab. 1 lists the datasets used in our experiments. All data with camera calibration and rotation information were obtained from the web.[3] The Dino data are relatively clean, so we randomly perturbed 15% of the original data by up to 5 pixels to create a more realistic test setting. Other data were pre-cleaned by RANSAC, and only contain around 1% outliers. Although the noise level is low, it is shown [14] that directly applying a bundle adjustment method on these data results in a RMS reprojection error of around 3 pixels. To test if a lower reprojection error can be achieved by removing outliers, we set the error tolerance ϵ to 2 pixels[4] for all experiments. Furthermore, to examine the influence of outlier ratio on the performance of K-slack, we also tested it on a noisier version of the Church data (denoted Church2) with 15% of the observations perturbed by up to 5 pixels. We set the structure depth range to [0.1, 100].

The performance of K-slack with various K values is shown in Fig. 8. As expected, a larger K generally gives

Table 1. Image sequences used in the SfM experiments (Sec. 5.2), the number of cameras, 3D points (visible in at least 2 cameras), and observed 2D image points.

Data	Cameras	3D points	Observations
Dino	36	4983	16432
UWO	57	8990	27722
House	12	12475	35470
Church	17	16961	46045

[3] The Dino data are from http://www.robots.ox.ac.uk/∼vgg/data; other data as well as the camera rotation parameters were obtained from www.maths.lth.se/matematiklth/personal/calle.

[4] Note that this is a more challenging task than that considered in [14], where the error tolerance was set to 5 pixels.

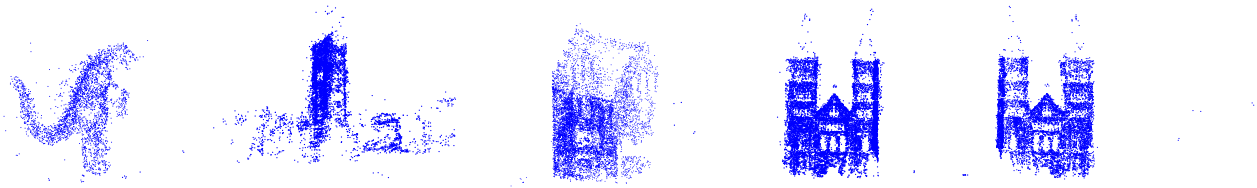

Figure 7. Left to right: 3D reconstructions produced by K-slack with $K = \lceil 10\%N \rceil$ from the SfM data listed from top to bottom in Tab. 2.

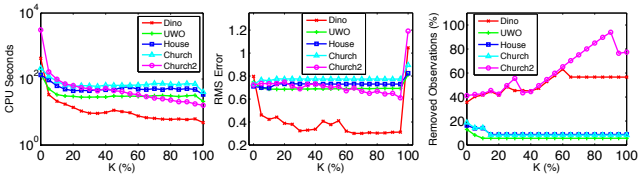

Figure 8. The influence of K on CPU time (left, on a log scale), RMS error (center), and the number of removed data (right).

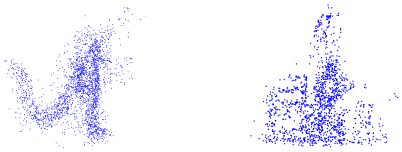

Figure 9. The L_1 method is particularly unreliable when the noise level is relatively high, as evidenced by the disastrous 3D reconstruction results on the Dino (left) and Church2 (right) data.

better runtime performance (left). The RMS error is again found to stay at a relatively low level over a wide range of K values (center), but the error increases when $K = N$ (the L_1 case). On the less noisy UWO, House, and Church data, the degradation in model quality is small as K approaches N, while on the more challenging Dino and Church2 data, the L_1 results (Fig. 9) are incomprehensible, indicating that it is particularly unstable when the outlier ratio is relatively high. Also, on these two datasets, the L_1 method removes noticeably more data than using a smaller K (Fig. 8, right); for instance, it removes 80% of the Church2 data, while the 1-slack method only removes 40%. On the three cleaner datasets, the influence of K on the number of removed data is small; on these data, the 1-slack method actually removes slightly more data than using a larger K since it executes more outlier removal steps. Overall, the SfM experiments confirm that an intermediate K provides a better compromise between various performance measures.

Tab. 2 compares the K-slack approach with K fixed to $\lceil 10\%N \rceil$ against its two extremes. Although the 1-slack and K-slack methods are slower than the L_1 method, in all cases the quality of their models is noticeably better. Especially, on the Dino and Church2 data, their advantage in this regard is evident: their RMS errors on these two datasets are at least 24% and respectively, 39% lower than those of the L_1 method. Between the 1-slack and K-slack methods, K-slack requires much fewer runs of LPs to clean the data, exhibiting substantial speedups of up to 97% (see the result on Church2). More importantly, such a superior runtime performance is achieved without compromising the quality of final 3D reconstructions (Fig. 7); and in fact, on 3 out of 5 datasets, K-slack produces better models. This is because besides removing outliers, K-slack also removes some "noisy" inliers, *i.e.* data with non-zero (but $\leq \epsilon$) residuals, hence achieving lower errors than the 1-slack method.

Table 2. Performance of various methods on the SfM data in Tab. 1 in terms of the number of LPs solved by each method in order to clean the data and the total number of removed 2D observations (with the corresponding number of removed 3D points in parentheses). We also report the required CPU seconds and the RMS reprojection error (in pixels) of the model returned by each method.

Data	Method	LP	Removed	CPU	RMS
Dino	L_1	1	9301 (2503)	4.8	1.05
	1-slack	234	5861 (1437)	424.0	0.80
	K-slack	6	6843 (1707)	21.3	0.42
UWO	L_1	1	1566 (262)	20.9	0.81
	1-slack	25	3593 (627)	145.6	0.75
	K-slack	2	1565 (262)	33.7	0.69
House	L_1	1	3119 (422)	34.0	0.82
	1-slack	21	5790 (1166)	133.50	0.71
	K-slack	3	4989 (847)	62.4	0.70
Church	L_1	1	3804 (554)	40.5	0.90
	1-slack	24	8613 (2189)	213.1	0.73
	K-slack	3	6650 (1491)	85.6	0.76
Church2	L_1	1	35651 (15440)	16.0	1.19
	1-slack	590	18955 (6380)	3078.1	0.72
	K-slack	6	19622 (6200)	98.5	0.73

Since among the removed data there often exist inliers, it may be desirable to have an inlier restoring scheme for some applications. Noting that after the outlier removal procedure, all the camera parameters are known to us (since they are part of the model), we can recover inliers by solving a triangulation problem [6] on the 2D observations of each removed 3D point, and restore observations that have a reprojection error less than a given tolerance.

6. Outlook and Discussion

We proposed a novel adversarial view of the outlier removal problem that not only gives rise to a general optimization framework that unifies various previous methods, but also brings new insights into the outlier removal problem from a game-theoretic perspective. To solve the resulting mixed-integer program, we also developed an equiva-

lent LP reformulation that significantly simplifies the process. Owing to its general formulation, our method is able to control the trade-off between reliability and speed, which is otherwise not possible using existing methods. Experiments on real image data demonstrate the superior practical performance of our method over recent approaches.

Our current optimization framework is designed for outlier removal in single-structure model fitting. For future work, we plan to extend it to deal with multiple structures.

Acknowledgements

This work is supported by the Australian Research Council grant DP0878801.

References

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

[2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[3] O. Enqvist, K. Josephson, and F. Kahl. Optimal correspondences from pairwise constraints. In *ICCV*, 2009.

[4] R. Hartley and F. Kahl. Optimal algorithms in multiview geometry. In *ACCV*, 2007.

[5] R. Hartley and F. Schaffalitzky. L_∞ minimization in geometric reconstruction problems. In *CVPR*, 2004.

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

[7] F. Kahl and R. Hartley. Multiple-view geometry under the L_∞-norm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(9):1603–1617, 2008.

[8] Q. Ke and T. Kanade. Quasiconvex optimization for robust geometric reconstruction. In *ICCV*, 2005.

[9] H. Li. A practical algorithm for L_∞ triangulation with outliers. In *CVPR*, 2007.

[10] H. Li. Consensus set maximization with guaranteed global optimality for robust geometry estimation. In *ICCV*, 2009.

[11] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[12] T. Nguyen and R. Welsch. Outlier detection and least trimmed squares approximation using semi-definite programming. *Computational Statistics and Data Analysis*, 2009.

[13] D. Nistér. Preemptive RANSAC for live structure and motion estimation. *Machine Vision and Applications*, 16(5): 321–329, 2005.

[14] C. Olsson, A. Eriksson, and R. Hartley. Outlier removal using duality. In *CVPR*, 2010.

[15] R. Rockafellar. *Convex analysis*. Princeton University Press, 1997.

[16] P. Rousseeuw. Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880, 1984.

[17] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2 edition, 2003.

[18] Y. Seo, H. Lee, and S. Lee. Outlier removal by convex optimization for L-infinity approaches. *Advances in Image and Video Technology*, pages 203–214, 2009.

[19] K. Sim and R. Hartley. Removing outliers using the L_∞ norm. In *CVPR*, 2006.

[20] J. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization methods and software*, 11(1):625–653, 1999.

A. A Theoretical Performance Bound

Th. A.1 below establishes a performance bound for our method. Its proof uses a known property of *active constraints*. In convex optimization, an inequality constraint $c(\mathbf{z}) \geq 0$ at a solution $\mathbf{z}$ is *active* if $c(\mathbf{z}) = 0$. The objective value achieved at a solution is determined only by its associated active constraints [2]. This property can be extended to problems that involve strictly quasi-convex functions [19].

Theorem A.1 *If $f_i(\mathbf{w})$ in (4) is strictly quasi-convex or convex, then the potential outlier set $\mathcal{O}$ as defined in (6) contains at least one outlier given $|\mathcal{O}| \geq K$.*

Proof First, we show that the optimal objective value of (4) (*resp.* (5)) is only influenced by the potential outliers identified in $\mathcal{O}$. To this end, we rewrite (4) as

$$\min_{\mathbf{s},\mathbf{w}} \max_{\boldsymbol{\psi}\in\Psi} \; g(\boldsymbol{\psi},\mathbf{s}) \;\text{ s.t. }\; f_i(\mathbf{w}) \leq \epsilon + s_i,\; s_i \geq 0,\; \forall i, \quad (18)$$

where $\Psi := \{\boldsymbol{\pi} \in \{0,1\}^N \mid \boldsymbol{\pi}^\top \mathbf{1} = K\}$, and $g(\boldsymbol{\psi},\mathbf{s}) := \boldsymbol{\psi}^\top \mathbf{s}$ is an auxiliary function. Further reformulation of (18) results in the following equivalent minimization problem:

$$\min_{\mathbf{s},\mathbf{w},\delta} \quad \delta \quad (19a)$$

$$\text{s.t.} \quad g(\boldsymbol{\psi},\mathbf{s}) \leq \delta, \;\; \forall \boldsymbol{\psi} \in \Psi \quad (19b)$$

$$f_i(\mathbf{w}) \leq \epsilon + s_i, \;\; s_i \geq 0, \;\; \forall i, \quad (19c)$$

where $\delta \in \mathbb{R}$. Let $(\mathbf{s}^*, \mathbf{w}^*, \delta^*)$ be the solution to (19). It is clear that δ^* must equal the sum of the K-largest entries of $\mathbf{s}^*$. By the definition of $\mathcal{O}$, $g(\boldsymbol{\psi},\mathbf{s}^*) = \delta^*$ only if $\boldsymbol{\psi} \in \Psi^* := \{\boldsymbol{\pi} \in \{0,1\}^N \mid \boldsymbol{\pi}^\top \mathbf{1} = K, \pi_i = 0, \forall i \notin \mathcal{O}\}$. Therefore, the constraints $g(\boldsymbol{\psi},\mathbf{s}) \leq \delta, \forall \boldsymbol{\psi} \notin \Psi^*$ are inactive. This means we can reduce Ψ in (19b) to Ψ^* without changing the optimal objective value. In this case $\pi_i = 0, \forall i \notin \mathcal{O}$, hence the data not in $\mathcal{O}$ are redundant to the optimization.

Suppose that a $\mathbf{w}$ exists with $f_i(\mathbf{w}) \leq \epsilon, \forall i \in \mathcal{O}$. This ensures a zero optimal objective value for (19) (hence also for (4) and (5)). This contradicts the fact that $|\mathcal{O}| \geq K$. ■