

# Consensus Set Maximization with Guaranteed Global Optimality for Robust Geometry Estimation

Hongdong Li

NICTA and Australian National University  
Canberra ACT 0200, Australia

<http://rsise.anu.edu.au/~hongdong>

## Abstract

*Finding the largest consensus set is one of the key ideas used by the original RANSAC for removing outliers in robust-estimation. However, because of its random and non-deterministic nature, RANSAC does not fulfill the goal of consensus set maximization exactly and optimally. Based on global optimization, this paper presents a new algorithm that solves the problem exactly. We reformulate the problem as a mixed integer programming (MIP), and solve it via a tailored branch-and-bound method, where the bounds are computed from the MIP's convex under-estimators. By exploiting the special structure of linear robust-estimation, the new algorithm is also made efficient from a computational point of view.*

## 1. Introduction

This paper is about *robust estimation* of multi-view geometry. We propose a new algorithm to estimate the parameters of a geometric model from a set of observed image data containing outliers, based on the idea of “consensus set maximization”. This idea is motivated from the conventional RANSAC algorithm ([5]).

However, different from RANSAC which is a randomized and non-deterministic method, our new algorithm is *exact* and *globally optimal*, in the sense that by our algorithm the obtained consensus set is guaranteed to return the largest (*i.e.* globally maximum) consensus set. To make this point clearer and to put our discussions in context, let us first briefly review the RANSAC algorithm.

RANSAC (RANDOM SAmple Consensus) is a well-known algorithm for robust estimation, and is widely used in multi-view geometry in vision. As its name suggests, there are two steps in RANSAC: (i) random sampling and (ii) consensus. These two steps are performed iteratively, in a hypothesize-and-test manner. More precisely, a subset of the data points is first randomly sampled from the input and fitted to a tentative model; this model is then evaluated

by counting how many points are inliers (whose fitting error below a prescribed tolerance). The two steps iterate many times until one can tell, with a very high probability, that a good model has been found that fits to the *largest* subset of the input [7].

One of the key wisdoms of RANSAC (as originally presented in [5]) is to find the largest (or large enough) consensus set that satisfy a prescribed tolerance. This *consensus set maximization* idea is *key* to RANSAC, and proves to be very effective in removing outliers in practice.

However, with conventional RANSAC, the goal of consensus-set-maximization is not fulfilled *exactly*. Being a non-deterministic heuristic algorithm, RANSAC provides no guarantee to the optimality of its solution (in terms of maximizing the consensus-set's cardinality). In any instance run of RANSAC, the obtained consensus set may be far from the optimal one. In addition, RANSAC's randomized nature also renders it somewhat unpredictable. For example, running the algorithm twice on the same data with same parameter settings may produce different results.

To illustrate this point, we did a simple test by running RANSAC 1000 times for camera motion estimation using the eight-point algorithm, applied to the same set of feature correspondences from Oxford's Corridor data [8]. The results are shown in Figure-1. The histogram gives the distribution of the cardinalities (size) of the consensus sets (inliers) obtained by RANSAC. In the 1000 runs there is a clear variation in the cardinalities (ranging from 170 to 226); this variation may not be thoroughly eliminated by a further nonlinear refinement, as the inlier-sets found in different runs can be very different.

Some modern variants of RANSAC have abandoned the original idea of consensus set maximization, and use instead more sophisticated measures of goodness-of-fit such as likelihood or posterior, *e.g.* in MLESAC, MAPSAC, IMPSAC, WALDSAC, R-RANSAC, Lo-RANSAC and preemptive-RANSAC, to just name a few (cf. [20, 19, 17, 4, 14] and references therein). However, since they largely follow the same heuristic of random sampling, none of them guaran-

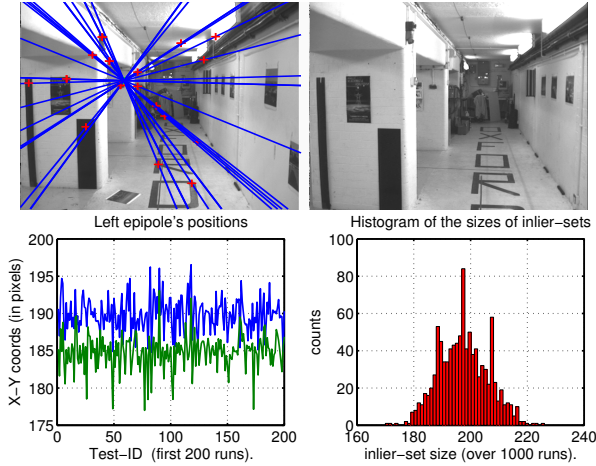


Figure 1. Run the RANSAC 1000 times on the Corridor image pair (top row). Top row: Two sample images with some epipolar lines overlaid on the left image. Bottom left: X-y coordinates of the left epipole obtained by 200 runs of RANSAC. Bottom right: histogram of the cardinalities of the inlier-sets found by 100 runs of RANSAC. The results vary from 170 to 226 inliers. While RANSAC’s results vary a lot, our new algorithm of this paper always finds the maximum consensus set (with cardinality 226) in a single run.

tees an exact solution.

In fact, throughout the large RANSAC literature, little is devoted to addressing such exactness and global optimality issues. The only exception we are aware of is [15] where an interesting post-validation idea is proposed to verify whether or not a solution is the optimal one. But, such a hit-or-miss type procedure gives no upper bound on the number of trials needed.

### 1.1. Contributions

In this paper we propose an *exact* (non-heuristic) method to solve robust geometry estimation based on consensus-set-maximization. Our method is globally optimal, in the sense that it always find the exactly maximum (hence also generally unique) consensus set. We hope such an exact and optimal algorithm will be useful for certain applications where the unpredictability of the conventional RANSAC is undesirable.

Our method is based on *mixed integer programming* (MIP). Solving a generic MIP is very expensive in general (NP-hard). However, by taking advantage of certain structures of the problem, we develop a specially-tailored global-optimization code (based on *branch-and-bound*) that solves the problem efficiently. From a computational point of view, our algorithm is also efficient.

Our primary contribution of the work lies in the theoretical aspect. It provides a principled way to solve the consensus set maximization problem in the context of ro-

bust estimation, and offers a fresh perspective on the classic RANSAC paradigm.

## 2. Mathematical Program Formulations

In this section we formulate the consensus-based robust geometry estimation as a mathematical program. In particular, we give a rigorous Mixed Integer Program (MIP) formulation of the cardinality-maximization task.

### 2.1. Notational preparations

For ease of exposition, we use notations and assumptions similar to RANSAC. We denote  $S$  the input data set, containing  $N$  data points. The unknown parameters of the geometric model to be fitted are denoted by a homogeneous vector  $\Theta$  with entries  $\theta_j, j=1..M$ .  $M$  is the degree-of-freedom of the model, which is typically small; otherwise RANSAC cannot handle it very efficiently.

Treating the parameters as a *homogeneous* vector is common practice in multi-view geometry. For simplicity’s sake, in order to fix the unknown scale of the homogenous vector we use a linear constraint  $\mathbf{c}^T \Theta = 1$ , rather than the commonly used  $\|\Theta\| = 1$ , where  $\mathbf{c}$  is a problem-dependent vector. Usually  $\mathbf{c}$  can be chosen without much difficulty and with care (*e.g.* to avoid singularity, see [8], pp-286).

We assume an initial *bounding-box* on the value of  $\Theta$ , *i.e.*  $[\underline{\Theta}, \bar{\Theta}]$ , such that for any  $\Theta$  we have  $\underline{\Theta} \leq \Theta \leq \bar{\Theta}$ . The operator  $\leq$  is element-wise. The bounding-box is introduced here to facilitate algorithm derivations. It is also problem-dependent. For a given problem, guessing a rough (broad) initial box is not very difficult. Sometimes, the context of the problem provides useful clues, *e.g.*, the radius of a 2D circle cannot be less than zero, or the angle of a rotation must within  $[0, 2\pi)$ , *etc.* In the paper, we assume without further explanation that an initial bounding-box on  $\Theta$  is always available.

### 2.2. DLT-based robust geometry fitting

We use the algebraic DLT (Direct Linear Transformation [8]) as the *solver* to estimate model parameters. Such a linear algebraic (rather than nonlinear geometric) solver is often considered adequate for the sole purpose of outlier detection by RANSAC.

DLT works by solving a homogeneous linear equation  $\mathbf{A}\Theta = \mathbf{0}$  where  $\mathbf{A}$  is known as the design matrix. Each row vector of  $\mathbf{A}$  is denoted  $\mathbf{a}_i^T$ , with elements formed from the  $i$ -th input data point. For example, in 2D line-fitting ( $ax_i + by_i + c = 0$ ) we have  $\mathbf{a}_i^T = [x_i, y_i, 1]$ . The algebraic residual  $d_i$  at point- $i$  is defined as  $d_i = |\mathbf{a}_i^T \Theta|$ . When there is no outlier in the input, it is well-known that DLT leads to a simple SVD.

When outliers are present, the DLT estimate must be purely based on inlying points, and must not take into ac-

count any outlier. To this end, let us imagine that there exists an oracle (or Maxwell's demon) that can partition the set of input data into an inlier-set  $\mathcal{S}_I \subseteq \mathcal{S}$  and an outlier-set  $\mathcal{S}_O \subseteq \mathcal{S}$  with  $\mathcal{S}_I \cap \mathcal{S}_O = \emptyset$ ,  $\mathcal{S}_I \cup \mathcal{S}_O = (\mathcal{S})$ , then the DLT estimation can be equivalently formulated as a *mathematical program*:

$$\min_{\Theta} \sum_{i \in \mathcal{S}_I} d_i^2, \quad (1)$$

$$s.t. \quad |\mathbf{a}_i^T \Theta| = d_i, \quad i = 1..N, \quad (2)$$

$$\mathbf{c}^T \Theta = 1, \quad \underline{\Theta} \leq \Theta \leq \overline{\Theta}. \quad (3)$$

The last row describes the scale and bounding-box constraints. To simplify presentation, from now on, whenever no confusion arises, we omit the last row in the rest of the paper. But the reader should bear in mind that the two constraints are always enforced.

In the original RANSAC, inliers are distinguished from outliers by solving the following consensus set maximization problem, where  $T$  is the residual threshold:

$$\max_{\Theta} \text{card}(\mathcal{S}_I), \quad (4)$$

$$s.t. \quad \forall i \in \mathcal{S}_I \subseteq \mathcal{S}, \quad |\mathbf{a}_i^T \Theta| \leq T.$$

This formulation simply says that, given a proper residual tolerance  $T$ , the method attempts to find the largest consensus set based on  $T$ .

Conventional RANSAC uses random sampling heuristic to approximately solve the above maximization problem; hence the exact optimality is not guaranteed.

### 2.3. An MIP formulation

The max-cardinality formulation in Eq.-4 is not easy to handle. We now perform a convenient transformation converting it into an equivalent Mixed Integer Program (MIP).

Introduce  $N$  auxiliary 0-1 variables  $z_i \in \{0, 1\}$ ,  $i = 1..N$ , with each  $z_i$  indicating whether the  $i$ -th point is an inlier ( $z_i = 1$ ) or an outlier ( $z_i = 0$ ). We multiply from both sides of the  $i$ -th constraint of Eq.2 by  $z_i$ . We then reach the following MIP minimization (denoted as MIP-MIN):

$$\min_{\mathbf{z}, \Theta} \sum_{i=1}^N (-z_i), \quad (5)$$

$$s.t. \quad z_i |\mathbf{a}_i^T \Theta| \leq z_i T, \quad z_i \in \{0, 1\}, \quad i = 1..N.$$

Note that if  $z_i = 0$ , denoting an outlier, then the  $i$ -th constraint is excluded (eliminated) automatically.

A different approach in formulating the cardinality problem as an MIP is via the (in)famous big-M method. However, this is not recommended in general, because the behaviour of the big-M method is infamously unpredictable, as it depends excessively on the value of  $M$  [13].

Inspecting the MIP-MIN formulation, one can get a better understanding of the *complexity* of the problem. The target here is to *classify* every data point into either inlier or outlier so that the size of inlier-set is maximized. A naive approach would be to exhaustively enumerate all possible combinations of the  $z_i$ s. This would lead to an algorithm of complexity  $O(2^N)$  which is prohibitive when  $N$  is big. Alternatively, the attempt to solve it directly as a general non-linear program proves to be extremely difficult too, since the constraint set is non-convex and the variables are mixed.

Fortunately, all hope is not lost. By examining the problem's special structure more carefully, we find that we can *relax* it to some handy forms that are easily manageable and even practically solvable. In the next two sections we will first *reformulate* the MIP-MIN to a Bilinear Programming (BLP), and further *relax* it to a linear programming (LP). The relaxed LP will be used for branch-and-bound global optimization.

## 3. Bilinear Reformulation and Linear Relaxation

### 3.1. Bilinear program reformulation

Inspecting MIP-MIN again, a significant observation that can be made is that: we may replace the integer constraints of  $z_i \in \{0, 1\}$  with  $0 \leq z_i \leq 1$  without harming any optimal solution of the MIP. This way, the original mixed-integer program problem (MIP) is reformulated to a general (continuous) Bilinear Program problem given below (denoted BLP).

$$\min_{\mathbf{z}, \Theta} \sum_{i=1}^N (-z_i), \quad (6)$$

$$s.t. \quad z_i |\mathbf{a}_i^T \Theta| \leq z_i T, \quad 0 \leq z_i \leq 1, \quad i = 1..N.$$

Why is this possible? Suppose that if any solved  $z_i$  is neither 0 nor 1, then minimizing the objective function (of BLP) will automatically force the  $z_i$  to increase to 1. In other words, fractional solution is not possible. Moreover, doing so will not affect any constraint, since each  $z_i$  appears homogeneously on both sides of the constraint.

Therefore, the two formulations of MIP-MIN and BLP are in fact equivalent. For this reason, we call the BLP a *reformulation* as opposed to a *relaxation*.

### 3.2. Linear program relaxation

The above BLP reformation has removed all the discrete (integer) constraints of MIP-MIN. However, to globally solve the BLP is still not a trivial task, because the bilinear constraints are still non-convex.

We now introduce a further simplification, intending to remove the bilinear terms too. Consider the left hand side

of a bilinear constraint,  $z_i |\mathbf{a}_i^T \Theta|$ . Since  $z_i \geq 0$ , if we substitute the bilinear term with a single vector  $\mathbf{w}_i = z_i \Theta$  whose entries are  $w_{ij} = z_i \theta_j$ , then the BLP problem becomes

$$\begin{aligned} \min_{\mathbf{z}, \Theta} \sum_{i=1}^N (-z_i), \quad (7) \\ \text{s.t. } |\mathbf{a}_i^T \mathbf{w}_i| \leq z_i T, \quad 0 \leq z_i \leq 1, \quad i = 1..N \\ w_{ij} = z_i \theta_j, \quad i = 1..N, \quad j = 1..M. \end{aligned}$$

Note that now the only non-convexity of the problem is contained entirely in the  $(N \times M)$  bilinear constraints.

Now we want to replace the bilinear constraints with some convex terms. To do so, we make use of convex approximation [13]. Recall that in the bilinear equality  $w_{ij} = z_i \theta_j$ , both  $z_i$  and  $\theta_j$  are (assumed) bounded by some bounding boxes. For any bilinear equality, say,  $\gamma = \alpha\beta$  with their bounding-boxes  $[\underline{\alpha}, \bar{\alpha}]$  and  $[\underline{\beta}, \bar{\beta}]$ , one may relax it using its convex and concave envelopes:

$$\begin{aligned} \gamma &\geq \max(\underline{\alpha}\beta + \underline{\beta}\alpha - \underline{\alpha}\bar{\beta}, \bar{\alpha}\beta + \bar{\beta}\alpha - \bar{\alpha}\bar{\beta}) \\ \gamma &\leq \min(\bar{\alpha}\beta + \bar{\beta}\alpha - \bar{\alpha}\underline{\beta}, \underline{\alpha}\beta + \underline{\beta}\alpha - \underline{\alpha}\underline{\beta}) \end{aligned}$$

Such a relaxation is based on the work of [13] and [18] in the optimization literature, which was also adopted by computer vision researchers recently ([3, 16, 9]).

To ease symbols, we collectively represent them as  $\text{conv}(\alpha\beta) \leq \gamma \leq \text{conc}(\alpha\beta)$ . We finally arrive at a *linear program* (denoted key-LP).

$$\begin{aligned} \min_{\mathbf{z}, \Theta} \sum_{i=1}^N (-z_i), \quad (8) \\ \text{s.t. } |\mathbf{a}_i^T \mathbf{w}_i| \leq z_i T, \quad i = 1..N \\ c^T \Theta = 1, \quad \underline{\Theta} \leq \Theta \leq \bar{\Theta}, \quad 0 \leq z_i \leq 1, \quad i = 1..N, \\ \text{conv}(z_i \theta_j) \leq w_{ij} \leq \text{conc}(z_i \theta_j), \quad i = 1..N, \quad j = 1..M. \end{aligned}$$

Since this LP is the *key* to the paper, hereafter we will refer to it as *key-LP*. This LP is in fact a convex underestimator to the original BLP: the minimal objective function of key-LP gives a lower bound to the BLP and the original MIP. This is to say, the *maximal*  $\sum_i (z_i^*)$  offers a *lower bound* of the (neg) cardinality of the consensus set given  $T$ .

Before proceed further, let us summarize what we have done so far:  $\text{MIP-MIN} \rightarrow \text{BLP} \rightsquigarrow \text{key-LP}$ . Note that while BLP is equivalent to MIP-MIN, key-LP is only a relaxation (*i.e.* a lower-bound approximation) of BLP. In order to design an exact algorithm (*i.e.* to reduce the relaxation gap to zero), we will resort to branch-and-bound method (in the next section).

## 4. Partial Branch and Bound

Key-LP offers a cheap way to compute a lower bound to the original MIP-MIN (Eq.-5). Having such a good lower

bounds actually invites the use of the branch-and-bound (BnB) idea to solve the non-convex BLP. Branch-and-bound is a well-known non-heuristic global-optimization method. When terminated it returns a solution with a certificate proving that the solution is globally optimal (up to a preset accuracy).

Branch-and-bound is normally very slow. In the worst case the required computation time is exponential to the size of the problem. For large-scale problems, the space (memory usage) requirement is also excessively large. This has limited the BnB method to very small problems only, *e.g.*, having merely a few, or up to a dozen variables (see [18]).

However, in our MIP-MIN problem the number of variables (including both  $z_i$ s and  $\Theta$ ) is typically in the hundreds, which is already too large to solve by a general-purpose BnB solver. Moreover, it is almost certain (unless P=NP) that no polynomial time algorithm exists for exactly solving the MIP. Therefore, to make use of the BnB idea, a special treatment is needed to exploit the problem's special structures further.

**Partial branch and bound.** We propose a *partial branch-and-bound* idea. The idea was inspired by recent work [3], which solves bilinear factorization problems (*e.g.*, Tomasi-Kanade Factorization) with global optimality.

The *partial branch-and-bound* is described as follows. Note that the only non-convexity of the BLP lies in the bilinear terms of  $z_i \Theta$ . While the  $z_i$  part has a total dimension of  $N$ , the dimension of  $\Theta$  is much smaller. Therefore, instead of branching in the entire parameter space involving both  $z_i$ s and  $\Theta$ , we branch in the  $\Theta$  space only. A rigorous mathematical exposition of this idea can be found in [18].

**Branching strategy.** We use a binary branching scheme [2]. At each iteration, we select a parameter box as a node, and split it into two children boxes. We select the box that contains the best-so-far lower bound to split, and split it along the axis that has the longest side-length.

**Bounding strategy.** The *lower bound* at each iteration is obtained by solving a key-LP within a proper box at that iteration. When the box is smaller, the lower bound is tighter.

We utilize as an *upper bound* the actual objective value of BLP (*i.e.* neg cardinality), evaluated at the current solution of the key-LP (*i.e.* the optimizer for  $\Theta$ ). This way is extremely inexpensive and turns out to work remarkably well in our experiments.

**Initialization and termination.** With valid bounds, a BnB algorithm always converges. However, a good initial upper bound may speed up the convergence, because it helps pruning those un-promising boxes more quickly. The (neg) cardinality detected by conventional RANSAC can be used as a good upper bound, because RANSAC's solution is always sub-optimal. This step is optional; in fact in all



our tests of section-6 we avoided using this, allowing a fair comparison.

The BnB algorithm terminates when the gap between current lower bound and upper bound is lower than a prescribed accuracy  $\epsilon$ .

## 5. The Branch-and-Bound Algorithm

Our branch-and-bound algorithm is listed in Algorithm-1. It is an adapted version of the one presented in [1].

The bounding boxes for  $\Theta$  are denoted as  $Q_s$ . Denote the upper bound and lower bound after  $k$  iterations as  $U_k$  and  $L_k$  respectively, and the procedures for finding them as  $\Phi_{lb}$  and  $\Phi_{ub}$ . Then,  $\Phi_{lb}$  is simply the result of key-LP, *i.e.*  $\Phi_{lb}(Q) \leftarrow \text{key-LP}(A, c, T, Q)$ . A linked-list  $\Omega$  is used to maintain the candidate boxes.

The algorithm is very simple, and easy to implement. In each iteration, the only non-trivial computation is to solve two key-LPs, one for each child box of  $Q_I$  and  $Q_{II}$ .

---

### Algorithm 1: BnB-based Consensus set maximization.

---

**Input:** A DLT problem  $(A, c)$ ; threshold  $T$ ; accuracy  $\epsilon$ ;  
initial bounding-box  $Q_0$  for  $\theta$ ;  
**Output:** An optimal box  $Q^*$ .

**begin**  
 $k = 0$ ;  $\Omega = \{Q_0\}$ ;  $L_0 = \Phi_{lb}(Q_0)$ ;  $U_0 = \Phi_{ub}(Q_0)$ ;  
 (optional) use the standard RANSAC to refine  $U_0$ ;  
**while**  $U_k - L_k > \epsilon$ , **do**  
   Pick a  $Q \in \Omega$  for which  $(\Phi_{lb}(Q) == L_k)$ ;  
   Split  $Q$  along its longest side to  $Q_I$  and  $Q_{II}$ ;  
    $\Omega = \{\Omega \setminus \{Q\}\} \cup \{Q_I, Q_{II}\}$ ;  
    $L_{k+1} = \min_{Q \in \Omega} \Phi_{lb}(Q)$ ;  
    $U_{k+1} = \min_{Q \in \Omega} \Phi_{ub}(Q)$ ;  
   Prune  $\Omega$  by discarding  $\forall Q \in \Omega$  if  $\Phi_{lb}(Q) > U_{k+1}$ ;  
    $k = k+1$ ;  
**end**  
**return**  $Q^* = Q \in \Omega$  for which  $(\Phi_{lb}(Q) == L_k)$ ;  
**end**

---

## 6. Experiments

We have implemented the BnB-based algorithm in Matlab, and have tested it on a modest machine (Intel P4 laptop 1.66G 780MB RAM). For ease of experimenting we use Matlab's native linprog as the LP solver.

We did extensive experiments on different model-fitting problems using both synthetic data with various levels of Gaussian noise and different fractions of outliers, and real test images. They are sufficient to validate the main theory and the algorithm of the paper.

The test problems include line-fitting, circle-fitting, ellipse-fitting, image-alignment, 3d registration, affine and perspective epipolar geometry, homography and the PnP problem. Despite the diversity of these problems, all their implementations are largely the same: the only thing differ-

entiating them is the design matrix  $A$  and scale constraint  $c$  used in their DLT equations.

From all these experiments we intend to check whether the BnB idea works, and how well it works (*e.g.* in terms of optimality and convergence). To illustrate the convergence of the algorithm, we use two types of figures: one is the upper- and lower-bound evolution curve; the other is the shrink of the total volume of active parameter boxes. To measure its speed, it is better to use the number of iterations, rather than absolute CPU time. This is because the computational burden at each iterations of the BnB algorithm is dominated by solving key-LPs.

On synthetic data, we have found that neither the levels of added noise nor the fractions of outliers affects the BnB algorithm's convergence in any significant sense. This is not surprising, as the BnB theory guarantees this.

Some details of our experiments are in order. In simulations we randomly generate 100 points on the model to be fitted. We add Gaussian noise to their  $x$  and  $y$  coordinates, followed by perturbing a fraction (*e.g.* 30% ~ 70%) of them by a large uniform noise to mimic outliers. So the number of outliers can be more than half of the input size.

To run our algorithm is particularly simple, and no harder than conventional RANSAC. The only required extra user-inputs are an initial bounding-box  $Q_0$  and accuracy tolerance  $\epsilon$  (*e.g.*  $\epsilon=0.1$ ). In all experiments we adopt a very conservative initial bounding-box, say, the ground-truth (or a conventional RANSAC's solution)  $\pm 100 \sim \pm 1000$ , depending on the context. We find that using an excessively-big initial box does not affect the speed much, as the space typically shrinks quickly as the iterations proceed.

We compare our BnB algorithm with RANSAC on the same input. Both use the same DLT and the same algebraic threshold. The RANSAC is repeated 1000 times and produces two types of figures: one is the estimated parameters versus iterations; the other is a histogram of the sizes of the detected inlier-sets.

Because all results look similar, here we will only present some typical examples, mainly for their demonstrative and pedagogic values.

### 6.1. Line fitting

We use the simplest linear regression technique (rather than the total least squares) to fit a 2D line. Figure-2 shows an instance of the results. On the left we see that 68 inliers have been found by RANSAC, while on the right 76 inliers have been found by the BnB.

A question naturally arises: *Is this BnB's solution of 76 inliers really the optimal one?* We repeat RANSAC 1000 times on the same input, and get the result in figure-3. After 1000 runs RANSAC does find solutions of size 76 located at the right-most histogram bin. This at least suggests that we cannot be too wrong about the optimality of the BnB's

solution compared with 1000 runs of RANSAC. Moreover, the quality of our solution is also assured, because all the inlying residuals are below  $T$ .

Figure-4 illustrates the convergence of the global lower and upper bounds. It takes only 60 iterations to find that the maximum cardinality is 76. The total volume of the parameter space drops as quickly. Roughly, on a modest PC, the time spent was around 10-15 seconds. We consider this to be very fast, compared with our previous experience with branch-and-bound [12].

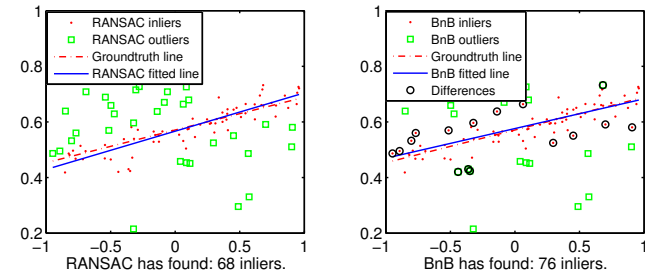


Figure 2. A line-fitting result: on the left is the result of the standard RANSAC; On the right is our BnB result. The ‘Differences’ indicates the differences between RANSAC’s result and our result.

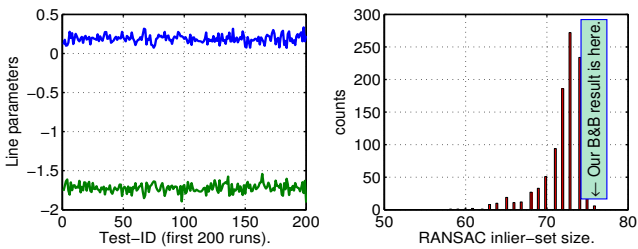


Figure 3. Test RANSAC 1000 times on the same input. Left: variation of the estimated line parameters; Right: histogram of the sizes of the inlier-sets declared by the RANSAC.

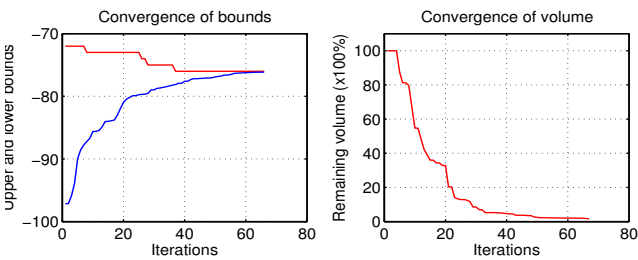


Figure 4. Line fitting: Convergence of the upper and lower bounds (left) and convergence of the volume of the parameter space (right).

## 6.2. Conic fitting

We apply our algorithm to the conic-fitting problems. We test for the cases of circle and ellipse. In the ellipse case

an ellipse is treated as a generic conic without enforcing its specific type (*e.g.* positive definiteness, see [6]). However, it is worth noting that even with such type-specification the new algorithm is still workable (at the expense of solving SDP at each iterations).

The convergence curves for a test on ellipse fitting are given in figure-5, showing that BnB terminated after 86 iterations to find a maximum cardinality of 65. We have verified that this is indeed the optimal solution against 1000 RANSAC tests.

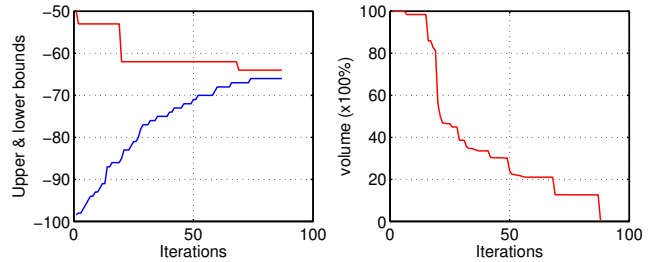


Figure 5. Ellipse fitting: Convergence of the upper and lower bounds (left) and convergence of the volume of the parameter space (right).

## 6.3. Epipolar geometry

RANSAC has played a significant role in multi-view geometry in sifting out wrong matches. We test BnB in the cases of affine and projective fundamental matrix and homography, on real images. These problems have higher degrees-of-freedom than the previous examples. They typically take about a few hundreds to thousands of iterations (in about a few minutes on a modest PC) to converge. Nevertheless we still count it efficient to find a provably optimal solution by solving merely a few hundreds or thousands of LPs, for the problem itself is NP-hard. Moreover, it is not our intention to compete with RANSAC in speed. RANSAC is much faster, but lacks exactness. In addition, the speed of our current BnB algorithm seems adequate for some off-line, non realtime applications, or merely as a mean of post validating RANSAC.

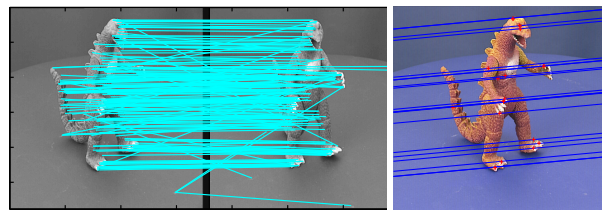


Figure 6. Affine fundamental matrix estimation result using the proposed BnB algorithm. On the left are the tentative matches (limited to 100 points); On the right are some estimated epipolar lines.

Figure-6 and -7 show the matching and convergence results for affine fundamental matrix estimation using our

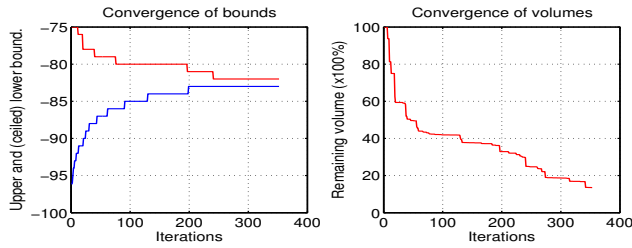


Figure 7. Affine FM fitting: Convergence of the upper and lower bounds (left) and convergence of the total volume of all the active boxes (right). It converged after 360 iterations.

method.

To give an indication of the algorithm’s memory usage during iterations, we plot the numbers of active boxes versus the iterations as shown in figure-8. The result is also satisfactory, as the peak memory usage is relatively small ( $\leq 150$  boxes).

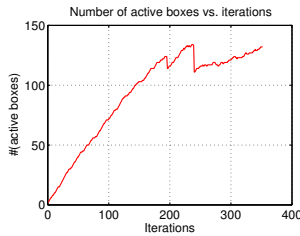


Figure 8. Memory requirement for affine FM fitting: number of active boxes versus iterations.

## 7. Extension and Conclusion

We have described a new algorithm for robust estimation, based on the idea of consensus set maximization. This algorithm offers a certificated and provably optimal solution to the problem. By utilizing special structures of the problem, we have developed tailored optimization codes to find the optimum efficiently.

Because the BnB algorithm is relatively novel in the robust estimation context, many things are left open in the paper. There is still much to be improved in the future. For example, our current algorithm is limited to the linear DLT case. In principle, this may be improved by using nonlinear geometric error (*e.g.* reprojection error) or nonlinear minimal solvers (*e.g.* 5-point [11], 6-point solver [10]); the current convex-concave bounds may be upgraded by more sophisticated RLT (reformulation linearization [18]) technique, *etc.* Doing the above extensions may result in much more complicated (or even intractable) mathematical formulations; nevertheless, the to-be-gained theoretical insights are invaluable.

We hope this work will be useful for certain applications, and hope it may inspire other researches pursuing globally-optimal solutions in robust estimation.

**Acknowledgement.** NICTA is a world-class research institute funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and ARC Centre of Excellence program. The author thanks the anonymous reviewers’ for insightful comments which greatly improve the presentation of the paper. Thanks are also given to J.Lim, R.Hartley, N.Barnes, C.Shen, P.Lieby, Y.Dai and P.Carr for helpful discussions.

## References

- [1] V. Balakrishnan, S. Boyd, and S. Balemi. Branch and bound algorithm for computing the minimum stability degree of parameter-dependent linear system. *International Journal of Robust and Non-linear Control*, 1(4):295–317, 1991.
- [2] T. M. Breuel. Implementation techniques for geometric branch-and-bound matching methods. *CVIU*, 90(3):258 – 294, 2003.
- [3] M. Chandraker and D. Kriegman. Globally optimal bilinear programming for computer vision applications. *CVPR’08*, 2008.
- [4] O. Chum and J. Matas. Optimal randomized ransac. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(8):1472–1482, 2008.
- [5] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.
- [6] A. Fitzgibbon, M. Pilu, and R. B. Fisher. Direct least square fitting of ellipses. *IEEE PAMI.*, 21(5):476–480, 1999.
- [7] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, 2002.
- [8] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision 2nd ed.* Cambridge University Press, 2004.
- [9] F. Kahl, S. Agarwal, M. K. Chandraker, D. Kriegman, and S. Bologlie. Practical global optimization for multiview geometry. *Int. J. Comput. Vision*, 79(3):271–284, 2008.
- [10] H. Li. A simple solution to the six-point two-view focal-length problem. In *9th European Conference on Computer Vision (ECCV 2006)*, pages 200–213, 2006.
- [11] H. Li and R. Hartley. Five-point motion estimation made easy. In *ICPR 2006*, pages 630–633. IEEE Computer Society.
- [12] H. Li and R. Hartley. The 3d-3d registration problem revisited. In *ICCV 2007*, pages 1–8. IEEE, 2007.
- [13] G. McCormick. Computability of global solutions to factorable non-convex programs—part i—convex underestimating problems. *Math. Prog.*, 10:147–175., 1976.
- [14] D. Nister. Preemptive ransac for live structure and motion estimation. *Proc. ICCV*, 1:199–206., 2003.
- [15] C. Olsson, O. Enqvist, and F. Kahl. A polynomial-time bound for matching and registration with outliers. *Proc. CVPR’08*, 2008.
- [16] C. Olsson, F. Kahl, and M. Oskarsson. Branch and bound methods for euclidean registration problems. *IEEE PAMI*, May 2008.
- [17] R. Raguram, J.-M. Frahm, and M. Pollefeys. A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. *Proc. ECCV’08*, pages 500–513, 2008.
- [18] H. Serali and A. Alameddine. A new reformulation-linearization technique for bilinear programming problems., *Journal of Global Optimization*, (2):379–410, 2002.
- [19] B. Tordoff and D. W. Murray. Guided sampling and consensus for motion estimation. *ECCV ’02: Proc. of the 7th European Conference on Computer Vision*, pages 82–98, 2002.
- [20] P. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *CVIU*, pages 138–156., 2000.