Signal Processing: Image Communication 84 (2020) 115834

Contents lists available at ScienceDirect

Signal Processing: Image Communication

journal homepage: www.elsevier.com/locate/image

GORFLM: Globally Optimal Robust Fitting for Linear Model

Yiru Wang[1], Yinlong Liu[1,2], Xuechen Li, Chen Wang, Manning Wang [*], Zhijian Song [*]

Digital Medical Research Center, School of Basic Medical Sciences, Fudan University, Shanghai 200032, China
Shanghai Key Laboratory of Medical Image Computing and Computer Assisted Intervention, Shanghai 200032, China

ARTICLE INFO

Keywords:
Robust linear model fitting
Globally optimization
Branch and bound
Gaussian function

ABSTRACT

Fitting a model to data contaminated by noise and outliers is a common task in computer vision, and it is often solved by maximizing inlier set. Most existing methods cannot guarantee global optimality, due to the two techniques widely utilized in inlier maximization: randomized sampling to generate candidate models and a hard threshold on residuals to classify inliers and outliers. In this paper, we propose a deterministic globally optimal linear model fitting method, in which we use the negative Gaussian function as a soft loss function over the residual and formulate model fitting as minimizing the sum of the Gaussian functions. We derive a convex quadratic function as the lower bound function of the objective function so that it can be globally minimized by a branch-and-bound algorithm. Experiments showed that the proposed method outperformed the state-of-the-art methods on several typical CV problems, especially when there are multiple models with different noise levels and large number of data points.

1. Introduction

Fitting a model to a set of data is a common task in experimental sciences, including Computer Vision. In the CV field, many algorithms rely on the ability to recover ideal mathematical models with imperfect data contaminated by both noise and outliers [1,2]. The data points that do not belong to any models are called gross-outliers. When there are multiple models in the data, the data points belonging to one model can be regarded as outliers to the other models, and they are called pseudo-outliers. Therefore, the proportion of outliers may be very high when we fit a model to the data, and this is one source of difficulty in robust model fitting. Another difficulty comes from the fact that different models may have different levels of noise, which is very common in practice but is seldom considered in existing model fitting methods.

There is a great body of literature on robust model fitting, and the most popular method is the celebrated RANSAC (RANdom Sample Consensus) [3]. It combines two basic ideas: random sampling and inlier counting, and initiates a major line of research on robust model fitting. RANSAC generates a series of hypothetical models by randomly sampling the original data iteratively and selects the model with maximum consensus (inlier maximization). In the original RANSAC algorithm, a data point is regarded as an inlier when its residual with respect to a model is less than a given inlier threshold. Many variants of RANSAC have been proposed such as Lo-RANSAC [4], PROSAC [5] and so on [6]. They concentrate on sampling [5,7] and evaluation strategies [8] to improve the robustness, efficiency and accuracy of RANSAC. However, the basic ideas of random sampling and inlier counting change little. Theoretically, a model fitting method can sample all possible combinations of the data points to obtain all possible models to evaluate, but it is computationally infeasible in practice. Therefore, the random sampling strategy has to be adopted to generate hypothetical models, which cannot guarantee that the best model can be found. In the inlier counting part, most methods classify inliers and outliers with a hard threshold on the residuals to select underlying model, which makes the result very sensitive to the choice of the threshold. Fig. 1 shows a common scenario in real applications, where different models have different levels of noise. For example, when we scan a room with Kinect, the noise levels of points at different distances from the Kinect are usually different. As shown in Fig. 1a, if the inlier threshold is smaller than the intrinsic noise level of the model, maximizing the consensus set may not find the true model. On the other hand, if the threshold is larger than the intrinsic noise level of the model, the model found may be affected by nearby outliers as in Fig. 1a. In this scenario, it is difficult to choose a proper inlier threshold to fit different models.

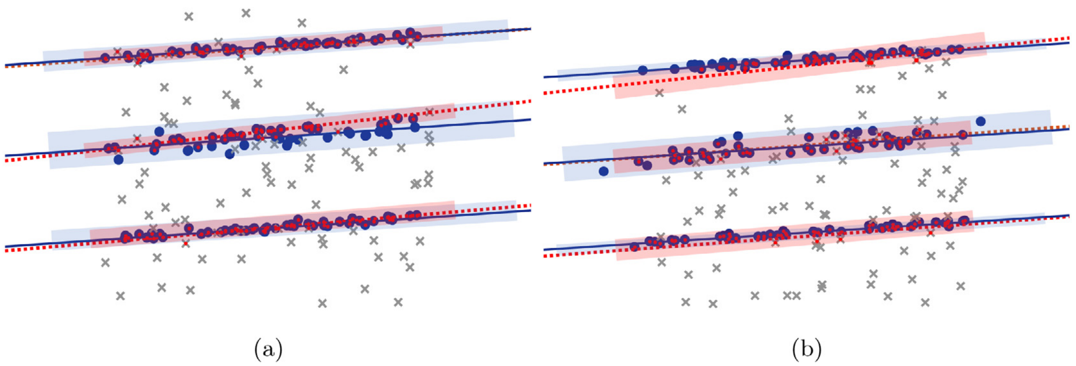
Recently, solving a robust model fitting problem within the optimization framework is a new trend. Deterministic methods [9,10] try to resolve the uncertainty by iteratively performing deterministic updates on the initial solution by relaxing the objective function of consensus maximization. Unlike them, [11] reformulates the consensus maximization into an instance of biconvex programming, which enables

[*] Corresponding authors at: Digital Medical Research Center, School of Basic Medical Sciences, Fudan University, Shanghai 200032, China.
E-mail addresses: mnwang@fudan.edu.cn (M. Wang), zjsong@fudan.edu.cn (Z. Song).

[1] The two authors contributed equally to this work.

[2] Current address: Department of Informatics, Technical University of Munich, 85748 Garching, Munich, Germany.

https://doi.org/10.1016/j.image.2020.115834
Received 12 September 2019; Received in revised form 16 December 2019; Accepted 15 March 2020
Available online 20 March 2020



(a) (b)

Fig. 1. Illustration of the scenario that an improper inlier threshold leads to wrong models for methods based on consensus maximization with a hard inlier threshold. The blue solid lines are the ground truth, and the red dashed lines are the detected ones. Some points are first generated on the ground truth lines, and then for generating noisy inliers with different levels of noise, the y coordinates on the upper and lower lines are perturbed by small Gaussian noise and the points on the middle line are perturbed by large Gaussian noise. These points are the blue dots in the figure. Finally, for generating gross outliers, some points with random coordinates are added and they are the gray crosses in the figure. A blue rectangle represents a suitable threshold for the fitted model and a red rectangle represents an inlier threshold actually used in fitting the three lines. The small red dots falling in a red rectangle are the detected inliers of the corresponding model. (a) A small inlier threshold suitable for the upper and the lower lines leads to the wrong model for the middle line, which has a big noise level. (b) A large inlier threshold suitable for the middle line makes the fitted models of the upper and the lower lines be affected by nearby outliers.

the use of bisection search without an objective function relaxation. However, these methods do not strive for the global optimality and may fail if their initial solutions are poor. Additionally, all these methods introduce binary indicator variables in the process of relaxing or reformulating the objective function of consensus maximization, which leads to an extremely long processing time in the case of fitting large number of data points.

Another line of research is the global optimization methods [12–18], in which the Branch-and-Bound (BnB) is the most used approach [12–14]. BnB is a general framework of optimizing non-convex functions by searching the parameter space intelligently, and almost every BnB-based method involves deriving new bounds of the optimized objective function. BnB has been employed in many computer vision model fitting applications: registration [19–24] and relative pose estimation [25] and so on. In [16,18], inlier set maximization is cast as a tree search problem that is globally optimized by a novel A*-search algorithm. This method only traverses a small subset of the tree compared to exhaustive methods such as [15,17] to improve time efficiency. Although these methods guarantee the global optimality of the result, the intrinsic problem of consensus maximization illustrated in Fig. 1 is still unsolved. Ask et al. [26] proposed a truncated L_2 loss function over residuals and developed a global optimization strategy that avoids the problem of defining inliers using a hard threshold. However, this method becomes infeasible to compute the optimal estimate for a large number of data points.

Our contributions in this paper can be summarized as follows:

- We propose a robust linear model fitting method with guaranteed global optimality. We use a negative Gaussian function as the loss function over the residuals and formulate the model fitting by minimizing the sum of the negative Gaussian functions. The objective function is highly non-convex, but we derive a convex quadratic function as its lower bound function and globally optimize it using a BnB algorithm. We also provide a solution for the linear model problem with unit-norm constraint, which is more difficult to optimize because of its non-convex domain.
- The loss function we use is close to the robust loss function suggested in [26,27] and it is very robust to outliers. The soft loss function instead of a hard inlier threshold makes the proposed method be able to better address the situation with different levels of noise.
- Each bound evaluation takes linear time w.r.t the input size M. If there are $\mathcal{K}$ branches that should be evaluated in BnB until convergence, the runtime of the whole procedure increases at most linearly with $\mathcal{K}M$. The performance of Section 3.3.1 indicates our advantage when the number of points is large.

The rest of this paper is organized as follows: in Section 2, we describe a convex quadratic lower bound and the BnB framework for the proposed method. In Section 3, we present the practical advantages compared with several other methods on both synthetic and real data. We draw conclusions in Section 4.

2. Method

2.1. Problem formulation and objective function with linear models

This paper addresses the problem of estimating the parameters of a prescribed linear model with the guaranteed global optimality from a set of data points contaminated by noise and outliers. Given a set of data $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{M}$ and a parameterized linear model $\boldsymbol{\theta} \in \mathbb{R}^d$ in the D-dimensional space, the algebraic residual of a data point $\mathbf{x}_i$ is:

$$r_i(\boldsymbol{\theta}) = |\mathbf{x}_i^T \boldsymbol{\theta}| \tag{1}$$

Please note that here we regard a data point $\mathbf{x}_i$ and the model $\boldsymbol{\theta}$ as column vectors. Two kinds of constraints are usually utilized to fix the unknown scale of $\boldsymbol{\theta}$. The first constraint is the unit-norm constraint: $\|\boldsymbol{\theta}\| = 1$, which results in a non-convex domain. The second constraint is setting the last element of $\boldsymbol{\theta}$ to be minus one: $\boldsymbol{\theta} = [\boldsymbol{\theta}'^T \ -1]^T$, which leads to a linear regression problem. Note that this constraint makes the dimension of the parameter space decrease one. Let $\mathbf{x}_i = [\mathbf{a}_i^T \ b_i]^T$ and the algebraic residual of linear regression problem is:

$$r_i(\boldsymbol{\theta}) = |\mathbf{a}_i^T \boldsymbol{\theta}' - b_i| \tag{2}$$

For a 3D plane fitting $\theta_1 x_{i1} + \theta_2 x_{i2} + \theta_3 \cdot 1 + \theta_4 x_{i3} = 0$, we have $\boldsymbol{\theta} = [\boldsymbol{\theta}'^T \ -1]^T = [-\frac{\theta_1}{\theta_4}, -\frac{\theta_2}{\theta_4}, -\frac{\theta_3}{\theta_4}, -1]^T$, $\mathbf{a}_i = [x_{i1}, x_{i2}, 1]^T$ and $b_i = x_{i3}$.

The objective of linear model fitting is to find the best $\boldsymbol{\theta}$ to explain the data points contaminated by noise and outliers. The traditional least square method consists of minimizing the sum of the squared residuals, which has been generally applied to the scenarios with "clean" data [28]. Since it implies the idea of maximum likelihood estimation under the assumption of independent and normally distributed measurement noise [29], the least square method lacks the robustness to the outliers. Indeed, one single outlier can have an arbitrarily high impact on the estimation [28]. To resist the impact of the outliers, we propose using the negative Gaussian function as the loss function, which is similar to the robust loss function suggested in [27]. Then, the problem becomes minimizing the sum of the negative Gaussian functions defined on $r_i(\theta)$. Specifically, the objective function is:

$$O(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M} O_i(\boldsymbol{\theta}) = \frac{1}{M}\sum_{i=1}^{M} -exp(-\frac{r_i(\boldsymbol{\theta})^2}{2\sigma^2}) \tag{3}$$

Compared to consensus maximization using a hard threshold, the proposed loss function models the errors of inliers better and this is the key for the proposed method to avoid the problems illustrated in Fig. 1.

The objective function in Eq. (3) is the sum of the negative Gaussian functions, which is highly non-convex. In this paper, we find its global minimum using the BnB optimization framework. The BnB is a powerful deterministic global optimization tool for non-convex problems. By branching the parameter space and estimating the upper and/or lower bounds in each branch, the BnB can shrink the search range, thus leading to an effective global optimization. The key of our BnB optimization is to find the lower bounds of the objective function in each branch efficiently. In Section 2.2, we will derive a tight convex quadratic lower bound function for our objective function in Eq. (3). The BnB based algorithm and its implementation details including how to handle the unit-norm constraint are described in Sections 2.3 and 2.4, respectively.

2.2. Convex quadratic lower bound function

Given a function $O(\theta)$, in a branch $\mathcal{B}$ of its parameter space, a constant $\underline{O}_*$ is a lower bound of the minimum of $O(\theta)$ if $\underline{O}_* \leq O(\theta)$ for every θ in $\mathcal{B}$. In addition, a function $\underline{O}(\theta)$ defined on domain $\mathcal{B}$ is a lower bound of $O(\theta)$ if $\underline{O}(\theta) \leq O(\theta)$ for every θ in $\mathcal{B}$. Therefore, in each branch $\mathcal{B}$, $\underline{O}_*$ can be chosen as the minimum of $\underline{O}(\theta)$, and the key is to find a $\underline{O}(\theta)$ that can be globally minimized in $\mathcal{B}$. To avoid confusion, we call $\underline{O}_*$ and $\underline{O}(\theta)$ lower bound and lower bound function, respectively. In this section, we derive a tight lower bound $\underline{O}_*$ of the sum of the negative Gaussian functions $O(\theta)$. We start from the lower bound function of a single Gaussian function in the following form:

$$G(r) = -exp(-\frac{r^2}{2\sigma^2}) \tag{4}$$

Let $u(r) = \frac{r^2}{2\sigma^2}$, and we have $G(u) = -exp(-u(r))$. Given the range of the variable r, the range of $u(r)$ can be obtained by the interval extension, and it is designated as $[\underline{u}, \overline{u}]$. As depicted in Fig. 2a, the line intercepting $-exp(-u)$ at $(\underline{u}, e^{-\underline{u}})$ and $(\overline{u}, e^{-\overline{u}})$ is the lower bound function of $-exp(-u)$ in the range of $[\underline{u}, \overline{u}]$. Therefore, the lower bound function $\underline{G}(r)$ of a single Gaussian function $G(r)$ is:

$$\underline{G}(r) = \xi \frac{r^2}{2\sigma^2} + \eta \tag{5a}$$

$$\xi = \frac{(-e^{-\overline{u}}) - (-e^{-\underline{u}})}{\overline{u} - \underline{u}}, \eta = -e^{-\overline{u}} - \xi\overline{u} \tag{5b}$$

Obviously, $\underline{G}(r)$ is a quadratic function of r, and it is convex because ξ is positive.

When we substitute (1) into (5a), we can obtain a convex quadratic lower bound function $\underline{O}_i(\theta)$ of each data point $\mathbf{x}_i$ with respect to θ. Therefore, in each branch $\mathcal{B}$, we can get a convex quadratic lower bound function $\underline{O}(\theta)$ of the objective function (3) by adding $\underline{O}_i(\theta)$ together:

$$\underline{O}(\theta) = \frac{1}{M}\sum_{i=1}^{M} \underline{O}_i(\theta) = \frac{1}{M}\sum_{i=1}^{M} (\xi_i \frac{|\mathbf{x}_i^T\theta|^2}{2\sigma^2} + \eta_i) \leq O(\theta), \theta \in \mathcal{B} \tag{6}$$

$\theta \in \mathcal{B}$ is a linear constraint and minimizing equation (6) is a convex quadratic programming in the linear regression problem formulation. How to address the unit-norm constraint problem formulation in our BnB based algorithm will be introduced in Section 2.4. Therefore, by minimizing the convex quadratic lower bound function, the lower bound $\underline{O}_*$ in each branch $\mathcal{B}$ can be calculated very quickly:

$$\underline{O}_* = \min_{\theta \in \mathcal{B}} \underline{O}(\theta) \leq \min_{\theta \in \mathcal{B}} O(\theta) \tag{7}$$

This value will be used to indicate if there is a potential better solution in this branch. A linear lower bound function of the exponential function and a quadratic lower bound function of the Gaussian function are illustrated in Figs. 2a and 2b, respectively. According to [17], the integral of the difference between the original function and its bound function is a measure of tightness of the bound, and we show the integral in the range of $r = [-1, 1]$ with respect to the number of branches used in this interval in Fig. 2c. We can see that the integral decreases rapidly as the number of branches increases, which means that the lower bound will become very tight when we branch the parameter space increasingly finer in the BnB optimization framework. A small example is given in Fig. 3 to show the splits of the parameter space and the changes of the lower bound functions.

2.3. BnB based algorithm

We first show how to solve the model fitting problem in the linear regression formulation using a BnB-based algorithm in this section, and then introduce how to incorporate the unit-norm constraint in the BnB-based algorithm in the next section. For a typical BnB-based framework to minimize a function, such as the objective function (3), an upper bound and a lower bound of the global minimum of the objective function are stored. These two bounds are updated in iteration to narrow the gap between them and the algorithm terminates when the gap falls below a given threshold. In our method, we use a queue to store all branches in which the global minimum may exist and calculate an upper bound and a lower bound for each branch in the queue. The lower bound is calculated by the method introduced in Section 2.2 and the calculation of the upper bound is straightforward because any functional value of the objective function can be an upper bound. In the linear regression problem formulation, we evaluate the functional value at the center of a branch as the upper bound. The process of evaluating the upper and lower bounds for each branch is shown in Algorithm 1, and the upper bound is denoted by $\overline{O}^*$.

Algorithm 1: Evaluate upper and lower bounds of the minimum of (3) in branch $\mathcal{B}$.

```
Input: Observed data $\mathcal{X}$, sigma $\sigma$ in Eq. (3) and branch $\mathcal{B}$.
Output: Upper bound $\overline{O}^*$ and lower bound $\underline{O}_*$ of $\mathcal{B}$.
$\theta_c$ ← center point of $\mathcal{B}$. $\overline{O}^*$ ← $O(\theta_c)$ or $O(\frac{\theta_c}{\|\theta_c\|})$
for each data $\mathbf{x}_i$ do
    $[\underline{u}_i, \overline{u}_i]$ ← the range of $u(r_i)$ w.r.t $\mathcal{B}$ by the interval extension.
    $\xi_i$ and $\eta_i$ ← Eq. (5b).
    $\underline{O}_i(\theta)$ ← substitute $\xi_i$, $\eta_i$ and Eq. (2) to (5a).
end
$\underline{O}$ ← equation (6).
$\underline{O}_*$ ← minimize $\underline{O}$ in Eq. (7) using Guribo [30] mathematical optimization solver.
```

We use a Depth-First-Search BnB algorithm to minimize the objective function (3). The method is formalized in Algorithm 2. This is a typical BnB algorithm and its goal is to guarantee the global optimality of the current optimal θ^*. The key to guaranteeing the global optimality of the BnB based algorithm is to narrow the gap between two bounds (O_L and O_U).

$$O_L \leq O_{BEST} \leq O_U \tag{8}$$

O_L and O_U are the lowest $\underline{O}_*$ and $\overline{O}^*$ among all branches, respectively. O_{BEST} is the current optimal objective function value.

The initial branch $\mathcal{B}_0$ is a $(D-1)$-dimension or D-dimension hypercube in the linear regression problem formulation and unit-norm constraint problem formulation, respectively. In the linear regression problem formulation, the initial branch needs to be large enough to contain the globally optimal solution because the range of the parameters is unlimited. In each iteration, the branch $\mathcal{B}_L$ with the lowest lower bound O_L will be subdivided into smaller branches because it is the branch that currently has the greatest potential to contain the globally optimal solution. After evaluating the upper and lower bounds of new branches,

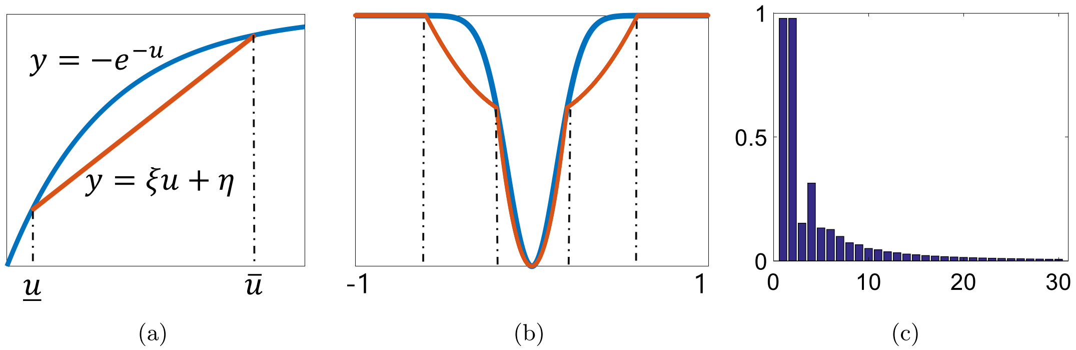


Fig. 2. (a) The linear lower bound function of the exponential function $G(u) = -exp(-u(r))$ when $u \in [0.25, 2.25]$. (b) The quadratic lower bound function of the Gaussian function $G(r) = -exp(-\frac{r^2}{2\sigma^2})$ when $2\sigma^2 = 0.04$ and the range of parameter space [−1,1] is divided into five branches. The exponential and Gaussian functions are blue, and the lower bound functions are brown. (c) Integral of the difference between the objective function in (b) and its bounds in [−1,1] with respect to the number of branches.

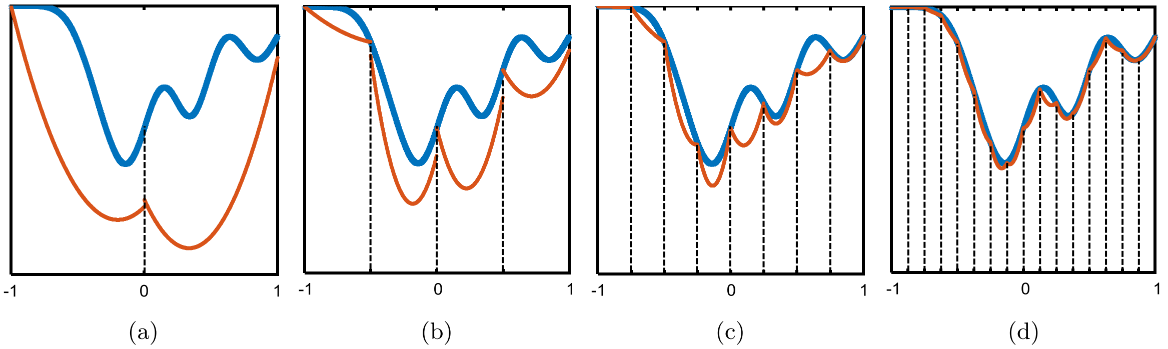


Fig. 3. A 1D example showing the changes of the lower bound function (brown) of the original function (blue) as we split the parameter space finer and finer, where the original function is a sum of three Gaussians and the lower bound functions are quadratic functions.

the lowest upper bound O_U is used to update the current optimal objective function value O_{BEST}. And then the current optimal θ^* comes from the center of the branch whose upper bound is O_{BEST}. Meanwhile, the branch $\mathcal{B}_i$ whose lower bound $\underline{O}_{*i}$ is larger than O_{BEST} can be removed from the branch queue q since the globally optimal solution cannot exist in it. The algorithm terminates when the gap between O_U and O_L is below ϵ, which is a given threshold. The algorithm achieves the ϵ-suboptimality because the difference between the global minimum value and the θ^* is guaranteed to be smaller than ϵ.

2.4. Incorporate unit-norm constraint in BnB

The unit-norm constraint influences the branching, the calculation of the upper and lower bound of each branch and the initial search branch of the BnB algorithm, and we will discuss these influences in this section.

The unit-norm constraint makes the feasible domain a hyper-hemisphere, which is a non-convex set. Actually, it is really difficult to branch the hyper-hemisphere and provide a bounding function for each branch. Therefore, we incorporate the constraint in the branching process in the following way. When we split the parameter space finer and finer, the branches that intersect with the unit hemisphere are inserted into the queue for further processing. And all the other branches are directly discarded because there is not any feasible solution in these branches.

To calculate the upper bound of a branch, in the unit-norm constraint formulation, we normalize the centric parameter of a branch to make its norm to be one and then evaluate its function value as the upper bound of the branch. This normalization makes the evaluation a feasible point of the unit-norm constraint problem. It should be noted that normalizing the centric parameter of a branch may result in a feasible parameter falling out of the branch, but evaluating the function at this point can also work for our BnB algorithm, because what we really need is a feasible solution to be used as an upper bound of the global minimum. The calculation of the low bound needs not to be changed, because the lower bound for the objective defined on a squared branch can also work as a lower bound defined on a subset of it. Regarding the initial search branch, in the unit-norm constraint, we have the nature parameters range from $\|\theta\| = 1$:

$$0 \le \theta_1 \le 1, -1 \le \theta_n \le 1, n = 2, 3, \dots, D \tag{9}$$

In this paper, all experiments were conducted with unit-norm constraint problem formulation for the proposed method because of its nature parameters range without prior.

3. Experiments

In this section, we investigate the performance of GORFLM and compare it against Lo*-Ransac [2], IBCO [11], A* search [18], and ILP-RansaCov [31] in four typical Computer Vision Applications: 2D line fitting, 3D plane fitting, the estimation of affine fundamental matrix and video motion segmentation. Lo*-Ransac, IBCO and A* search were selected for comparison because they are the state-of-the-art algorithms concentrated on random, deterministic and global robust model fitting, respectively. In particular, Lo*-Ransac and the proposed GORFLM share some common heuristics in using soft threshold so that they are more robust to different levels of noise of the models to be fitted. In order to investigate the sensitivity of the algorithms to different noise levels of multiple models in the same scene, experiments on multiple model fitting were conducted. In these experiments, GORFLM, Lo*-Ransac, IBCO, and A* search ran in a sequential way in fitting multiple models, which means that these methods are applied to find the first structure, the inliers of that structure are removed and the methods are then applied to the remaining data. A fixed inlier threshold is used in removing the inliers of each structure for all these methods.

Algorithm 2: Globally Optimal Robust Fitting for Linear Model.

```
Input: Observed data $\mathcal{X}$, sigma $\sigma$ in Eq. (3), gap $\epsilon$ between
       lowest upper bound $O_U$ and lowest lower bound $O_L$, and
       the dimension $K$ of model parameter $\theta$.($K$ is equal to
       $(D-1)$ and $D$ for linear regression problem formulation
       and unit-norm constraint problem formulation,
       respectively.)
Output: Optimal $\theta^*$ with accuracy $\epsilon$.
Initialize branch $\mathcal{B}_0$ with possible model parameters range, and
insert branch $\mathcal{B}_0$ with $\overline{O}_0^*$ and $\underline{O}_0^*$ into queue $q$. $\theta^* \leftarrow$ center of
the branch $\mathcal{B}_0$, best objective function value $O_{BEST} \leftarrow O(\theta^*)$.
while $q$ is not empty do
    Remove cube $\mathcal{B}_L$ with lowest lower bound $O_L$ from $q$.
    Subdivide $\mathcal{B}_L$ into $2^K$ cubes $\{\mathcal{B}_d\}_{d=1}^{2^K}$.
    Evaluate $\overline{O}_d^*$ and $\underline{O}_{*d}$ for all $\{\mathcal{B}_d\}_{d=1}^{2^K}$. (Algorithm 1)
    Insert $\mathcal{B}_d$ with $\overline{O}_{*d}$ and $\underline{O}_{*d}$ into $q$, and update $O_U$ and $O_L$.
    if $O_U < O_{BEST}$ then
        $O_{BEST} \leftarrow O_U$, $\theta_c$ ←center of $\mathcal{B}^*$ corresponding to $O_{BEST}$.
        $\theta^* \leftarrow \theta_c$ or $\frac{\theta_c}{\|\theta_c\|}$
    end
    for each $\mathcal{B}_i$ of branch queue $q$ do
        if $\underline{O}_{*i} > O_{BEST}$ then
            Eliminate $\mathcal{B}_i$ with $\overline{O}_i^*$ and $\underline{O}_{*i}$ from $q$.
        end
    end
    if $O_{BEST} - O_L < \epsilon$ then
        Terminate.
    end
end
```

Additionally, we compared with ILP-RansaCov, which is a state-of-the-art multiple model fitting method and all models are found in one run of this method. All the algorithms, including ILP-RansaCov, were provided the true number of models when fitting multiple models. The code of Lo*-Ransac and IBCO were obtained from [32]. The code of A* search is from its conference version [18]. A latter version A*-prune [16] improved the performance of A* search slightly, while it is still not practical when the number of data and the outlier ratio increase to a certain extent (e.g. 500 points with 20% outliers), but the code of A*-prune is not available. The code of ILP-RansaCov was obtained from [33]. GORFLM was implemented in MATLAB R2018a. The specific parameters of these algorithms were tuned as follows:

- GORFLM (**G**): There are three parameters: sigma σ in formulation (3), gap ϵ for ϵ-suboptimality guarantee and inlier threshold ε for removing inliers of each structure in the multiple model fitting experiments. Theoretically, there exists some potential connection between σ and ε, because both parameters are used to model how far an inlier can deviate from its ideal position. This property provides the basis for jointly setting these two parameters. After balancing precision and efficiency, we chose to set the sigma σ and the inlier threshold ε to be equal in all experiments for keeping consistent. Indeed, in practice, the reader can adjust the relationship between the sigma σ and the inlier threshold ε to obtain better performance.
- Lo*-Ransac (**L**): We chose the confidence $\rho = 0.99$ for the stopping criterion in all experiments. All of the other parameters except inlier threshold ε are set as described in [11].
- IBCO (**B**): It needs one parameter: inlier threshold and we executed this algorithm with a random initialization.
- A* search (**A**): It needs one parameter: inlier threshold and it usually cannot converge within an acceptable time in our experiments. We therefore set a time limit for **A**. It means that the current solution without guaranteed global optimality is returned if **A** cannot converge within the time limit.
- ILP-RansaCov (**R**): It needs one parameter: inlier threshold.

The inlier thresholds of all the methods are tuned to be the same in all line and plane fitting experiments. In the experiments of affine fundamental matrix and video motion segmentation, the inlier thresholds of all the methods are adjusted to their best performance. All the experiments were done on a computer with an Intel Core i7 2.8 GHz CPU and 16 GB RAM.

3.1. Convergence and parameter choice of GORFLM

Convergence. We first show the convergence of GORFLM in a challenging 2D line fitting experiment as shown in Fig. 4a, in which two lines consisting of different number of inliers, different levels of noise and biased gross outliers are to be fitted. We randomly generated 50 and 100 inliers along the yellow and pink lines, respectively. For each inlier point, both its x and y coordinates were perturbed by Gaussian noise with standard deviation of δ_{in}, which was set to 0.02 and 0.01 for the yellow and pink line, respectively. After that, additional 50 random points were added to generate gross outliers. These 50 points were first randomly generated on the bottom (pink) line, and then its y coordinate was added the absolute value of a Gaussian noise with $\delta_{out} = 0.2$. The sign of the Gaussian noise was forced to be positive to make the outliers fall on one side of the line just as in [9].

The evolution of the upper and the lower bound of the minimum of the objective function and the remaining volume of the parameter space with respect to iteration in fitting the pink line in Fig. 4a are illustrated in Fig. 4b. In this experiment, the sigma σ and the inlier threshold ε were set to 0.02 and the gap ϵ were set to 0.01. We can see that the gap between the upper and lower bound and the remaining volume are very small after 120 iterations. Meanwhile, Figs. 4c, 4d show the contribution of each point to the value of the objective function on the estimated models in fitting the pink and the yellow lines, which reveals the capability of GORFLM in addressing models with different noise levels and biased outliers. The dark blue points have little contribution to the objective function and the existence of them has little influence on the fitted parameters of the line.

Inlier threshold ε & **sigma** σ. Then, we study the influence of inlier threshold ε and sigma σ on its accuracy and runtime in fitting the two lines as shown in Fig. 4a. Please note that in all experiments the sigma σ and the inlier threshold ε were set to be the same. The gap $\epsilon = 0.01$ was kept unchanged. For each of the nine different choices of ε, we repeated GORFLM with 50 randomly generated data points of Fig. 4a. The average fitting errors of each line and average runtime were plotted in Figs. 5a and 5b, respectively. Since the slope k and intercept b of the parameters have a clear meaning in 2D line fitting, we have listed their average fitting error separately. The fitting errors are all acceptable with different choices of ε, which means that the performance of the proposed algorithm is not sensitive to the choice of ε. This property is very useful when we need to fit multiple models with different levels of noise. Fig. 5b shows that the runtime is fairly stable across different choices of ε ranging from 0.025 to 0.05 and it tends to increase when ε becomes very small.

Gap ϵ. The gap ϵ for ϵ-suboptimality guarantee also has an effect on the runtime and accuracy of GORFLM. For each of five different ϵ, we ran GORFLM with 50 randomly generated data points of Fig. 4a. The inlier threshold ε = sigma $\sigma = 0.02$ were constant and unchanged. The average runtime with different choice of ϵ are shown in Fig. 5c. In fitting the pink line, the differences of the objective function value in Eq. (3) between $\epsilon = 0.01$ and $\epsilon = \{0.05, 0.1, 0.15, 0.2\}$ are 3.9e−4, 4.2e−4, 4.2e−4 and 4.2e−4. The gap ϵ is an upper bound of the difference between the globally optimal functional value and estimated functional value. Since the objective function is a continuous function and the gap ϵ cannot be exactly zero, we can roughly consider the

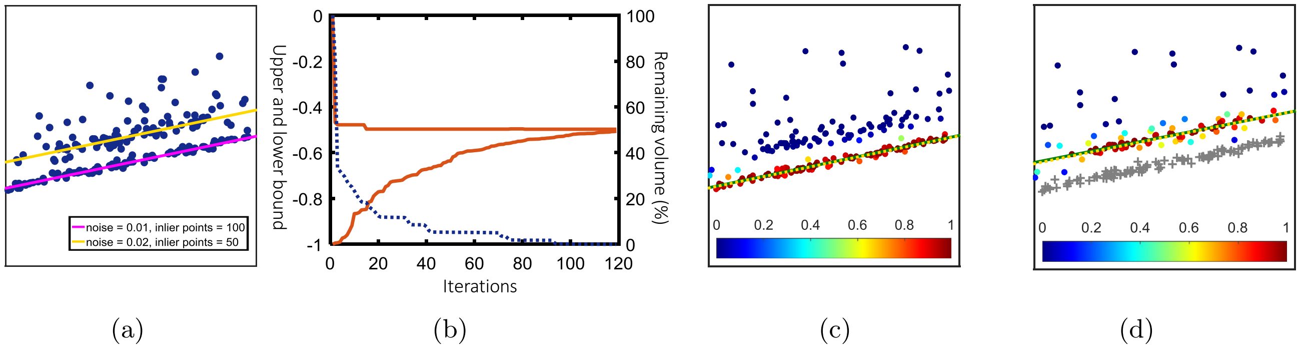


Fig. 4. Convergence of the GORFLM algorithm and the contribution exhibition of each point to the value of the objective function. (a) A challenging example of two parallel lines to be fitted. The yellow and pink lines consist of 50 and 100 inliers, respectively, and they are contaminated by 50 severely biased gross outliers. The noise of the inlier points of the yellow and the pink lines are $N(0, 0.02^2)$ and $N(0, 0.01^2)$. (b) The evolution of the upper and lower bound of the objective function (brown line) and the remaining volume of the parameter space (blue dashed line) with respect to the iterations in fitting the pink line. (c), (d) The contribution of each point to the value of the objective function in fitting the pink and the yellow line in figure (a), respectively. The contribution values are color coded according to the bar at the bottom. Yellow dashed line indicates ground truth, and green solid line indicates fitted line. "+" in (**d**) indicates removed points after fitting a structure.

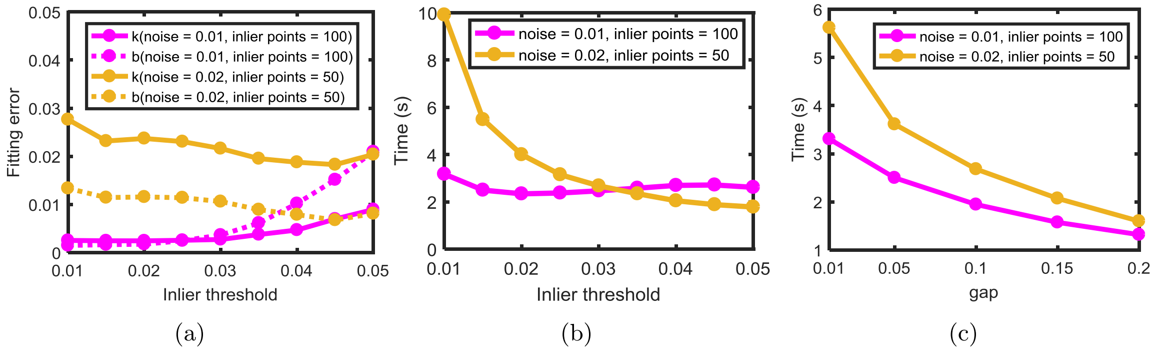


Fig. 5. The influence of GORFLM "inlier threshold ε" and "gap ϵ" parameters on its accuracy and runtime in the two lines in Fig. 4a.

solution with $\epsilon = 0.01$ to be the globally optimal solution. Then, the gap of the objective function value between the estimated result and globally optimal solution is much smaller than the gap ϵ itself. It indicates that the gap ϵ can be relaxed to obtain an increase in speed without losing too much accuracy.

3.2. Synthetic data

3.2.1. Robustness to outliers

In this section, we first demonstrate the robustness of GORFLM with increasing number of outliers under different choices of inlier threshold ε. We generated inlier points on the two lines with different levels of noise and different number of gross outliers as in Fig. 4a from 0 to 250 with an interval of 50. For each number of outliers and each of nine different choices of ε, we ran GORFLM on 50 randomly generated data. The average fitting errors of the slope k and intercept b of each line were plotted in Fig. 6. As shown in Fig. 6, ε can be selected in a large range, which means that it has a strong tolerance for different number of outliers. Even for the difficult situations, where there are more than 150 outliers, the average fitting errors are still small when ε is less than 0.035. The reason why the average fitting errors in Figs. 6c, 6d are larger than that in Figs. 6a, 6b is that the yellow was perturbed by a larger noise than the pink line in Fig. 4a. The result in Fig. 6 further confirms that the choice of ε is not sensitive to noise level, even in the case of big proportion of biased outliers.

Then, we compared GORFLM against Lo*-Ransac, IBCO, A* search and ILP-RansaCov. The same inlier threshold $\varepsilon = 0.02$ was used for all methods and the gap $\epsilon = 0.01$ was used for GORFLM. The time limit of A* search is 100 s for each model. In order to evaluate a method in detecting all of the models, we calculated a comprehensive metric "overall error" as in [34,35]. For a particular 2D line fitting, $P = \{p_1, \dots, p_N\}$ and $P = \{\hat{p}_i, \dots, \hat{p}_N\}$ are the set of ground truth and estimated lines with unknown correspondences between p_i and $\hat{p}_j$, where N is the number of estimated lines. The pairwise error between a pair of parameters is defined as the L_2 distance and the overall error between P and $\hat{P}$ is:

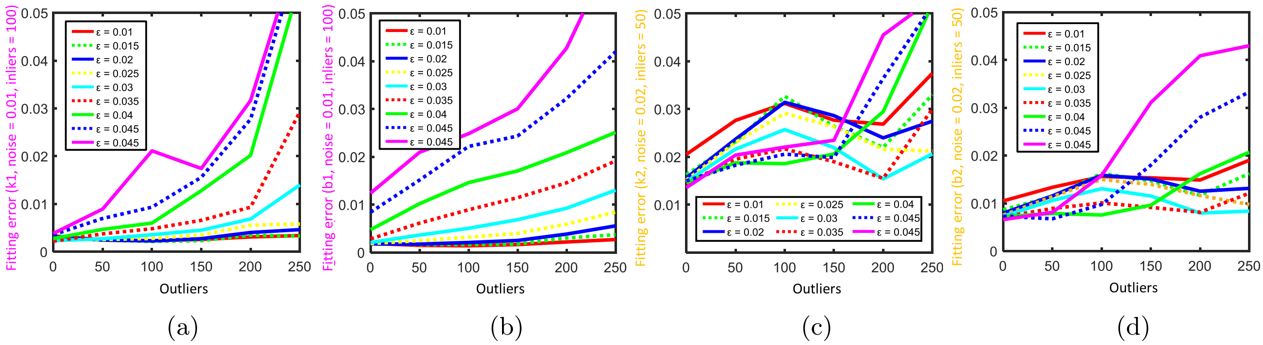
$$overall\ error = \sum_{n=1}^{N} min\ \varphi_n \tag{10}$$

where φ_n represents the set of all pairwise errors at the nth summation, and the pair with the lowest pairwise error should be removed from P and $\hat{P}$ before calculating the next summation.

The overall error of each method with respective to outlier numbers are plotted in Fig. 7. The performance of our method is the best in most cases, except a slightly inferior result of in the 100-outlier case compared with ILP-RansaCov. To further demonstrate the advantage of the proposed method, we listed the fitting error of slope k and intercept b of each line separately in Appendix A, which exposes the unstable performance of the compared algorithms in fitting two models with different levels of noise. The difficulty of this experiment is that the optimal parameters required for fitting lines with different levels of noise are different, and it is hard for the compared algorithms to balance the performance on the two lines. Fig. 7b is the average runtime in fitting the two lines. A* search, a globally optimal method, cannot converge within 100 s for each line even with 50 outliers, and this is the reason why A* search has a poor precision as shown in Fig. 7a.

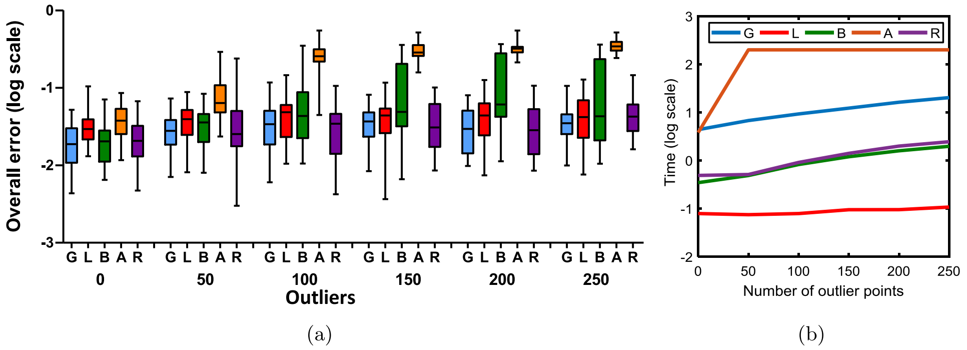
3.2.2. Time complexity

We compared the time complexity of GORFLM and the other methods with respect to the increasing number of points: 1000, 3000, 5000, 7000 and 9000 in fitting a 3D plane in 3D point sets. These points



(a) (b) (c) (d)

Fig. 6. The average fitting error of slope k and intercept b with increasing number of outliers in fitting the two lines in Fig. 4a with nine different choices of ϵ: 0.01, 0.015, 0.02, 0.025, 0.03, 0.035, 0.04, 0.045, 0.05.



(a) (b)

Fig. 7. (a) shows the box plots of the overall error (log scale) over 50 runs of GORFLM (**G**), Lo*-RANSAC (**L**), IBCO (**B**), A* search (**A**) and ILP-RansaCov (**R**) under different number of outliers. (b) Average runtime (log scale) of GORFLM (**G**), Lo*-RANSAC (**L**), IBCO (**B**), A* search (**A**) and ILP-RansaCov (**R**) with respect to the number of outliers.

Table 1
Average time of GORFLM (**Ggap0.01**, **Ggap0.05**), Lo*-Ransac (**L**), IBCO (**B**), A* search (**A**) and ILP-RansaCov (**R**) w.r.t the total point number M.

Average time (s)						
Num	**Ggap0.01**	**Ggap0.05**	**L**	**B**	**A**	**R**
1000	58.67	41.66	0.62	4.57	3600.11	27.30
3000	60.97	44.45	3.88	33.65	3600.24	263.53
5000	63.86	46.86	6.92	88.91	3600.78	787.77
7000	66.86	49.07	10.56	190.49	3600.84	2319.19
9000	70.10	50.60	20.70	283.05	3600.23	4488.67

were first created on the plane passing through the point [1, 1, 1] with a random normal, and then 50% and 50% points are perturbed by adding Gaussian noise with a standard deviation of 0.02 and 0.5 to each of its three coordinates to generate inliers and outliers, respectively. The inlier thresholds of all methods are set to 0.02 and we chose two different gap $\epsilon = 0.01$, 0.05 for GORFLM. For A* search, the time limit is 3600 s in this experiment. For each number of point set, the average runtime of GORFLM, Lo*-Ransac, IBCO, A* search and ILP-RansaCov for 10 randomly generated point sets was recorded in Table 1.

Globally optimal robust linear model fitting is an intractable problem, and it has been shown to be NP-hard [36–38]. Concretely, for a K-dimensional robust linear model fitting problem with M inputs, we cannot expect to obtain its exact solution faster than $O(M^K)$. Moreover, it is almost impossible to remove K from the exponent of the run time of a globally optimal algorithm [36]. In GORFLM, we utilized the BnB [39] framework to provide certainty of finding optimal solution. The proposed convex quadratic lower bound makes each bound evaluation take linear time w.r.t the input size M. However, solving the problems with high dimensions is still difficult for our proposed method. Therefore, GORFLM is recommended for the problems with low dimensions and large input size. From the results of Table 1, we can see that the relaxation of gap ϵ increases the speed to a certain extent and the average runtime of GORFLM is very stable with the increasing number of points. A* search cannot converge within 3600 s so that its average runtime remains constant. However, the average runtime of the other three methods increases drastically.

3.2.3. A more challenging line fitting example with different levels of noise

The estimated results of a more challenging line fitting experiment with different levels of noise and gross outliers are shown in Fig. 8. There are four line segments with 100 inliers on each of them and there are 400 uniformly distributed random outliers in the data, as shown in Fig. 8a. The intersection of lines to be fitted introduces additional difficulties for using a sequential way to solve multi-model fitting problems. The inlier points were perturbed by adding different levels of Gaussian noise ($N(0, 0.001^2)$, $N(0, 0.01^2)$, $N(0, 0.02^2)$ and $N(0, 0.03^2)$) to each of its coordinates. The inlier threshold of all these methods was 0.02 and the gap ϵ of GORFLM was 0.01. The time limit of A* search is 100 s for each line. Overall, GORFLM correctly fitted all four lines with high accuracy in this challenging scenario, and there are some errors in the line parameters or some wrongly fitted lines for the other methods.

3.3. Real data

3.3.1. Fitting planes in the point cloud of a room with large number of points

In this section, we compared GORFLM against Lo*-Ransac, IBCO, A* search and ILP-RansaCov in fitting multiple planes in the 3D point cloud of a room. The data was taken from the "livingRoom.mat" of the MATLAB example 3D Point Cloud Registration and Stitching, which consists of a series of 3D point sets obtained from continuously scanning a living room. We used the first frame of this data, and it is shown in Fig. 9a. There are two challenges in fitting planes in these data. First, the number of points in these data set is very big (M = 235,306 points). The second challenge of these data is that the two planes marked with "1" and "2" in Fig. 9a have different levels of noise. The specific parameter settings are: inlier threshold = 0.14 for all methods, three

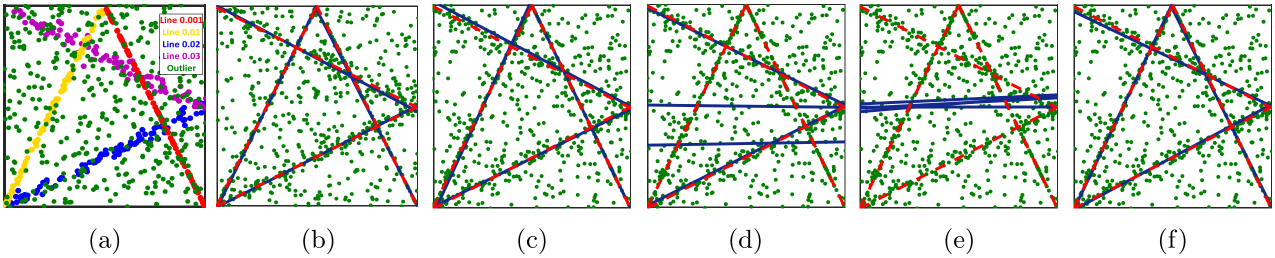


Fig. 8. (a) Four lines with different levels of noises to be fitted in this experiment. (b–f) Examples of the worst overall error results of GORFLM (**G**), Lo*-Ransac (**L**), IBCO (**B**), A* search (**A**) and ILP-RansaCov (**R**), respectively. (c)/(f): low accuracy for Line 0.03; (d): wrong fitting of Line 0.001 and Line 0.01; (e) wrong fitting of all four lines.

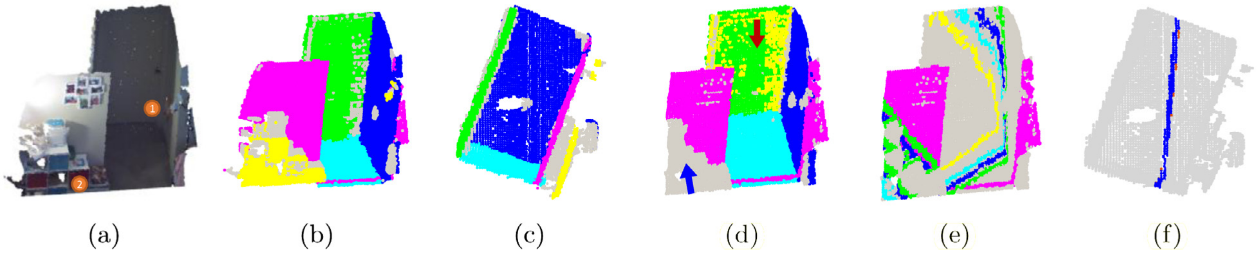


Fig. 9. Living room data set and its detection results colored in yellow, pink, green, dark blue and bright blue. The gray dots are the outliers of the detection results. (a) The original data set with textures. There are five major planes in this data set and "1" and "2" mark the two planes with different noise levels. (b,c) Detection results of GORFLM on the down-sampled data ($M = 15{,}830$ points) from two different views. The following figures are examples of errors in Lo*-Ransac (d), IBCO (e) and A* search (f). The example of error in ILP-RansaCov is similar to (d). In addition, the blue and red arrows indicate the circumstances of missing detection and redundant detection, respectively.

different gaps $\epsilon = 0.1, 0.15, 0.2$ were experimented for GORFLM, time limit = 720 s for A* search. To compare the performance of all these methods, we firstly down-sampled the data to 4,015 points and 15,830 points using a box grid filter of Matlab build-in function *pcdownsample* with the specified input *gridStep* = 0.06 and 0.03, respectively. We ran each method 50 times and recorded the number of planes detected correctly. Therefore, the highest possible score of each run is 5, which means it correctly detects all five planes and there is no missing detection or redundant detection. Table 2 lists the average runtime and scores of all methods with different number of points. And the examples of the detection results in GORFLM from two views are illustrated in Figs. 9b, 9c. Some typical mistakes that occurred in the four competing methods are illustrated in Figs. 9d, 9e, 9f.

Generally, GORFLM achieved the highest score of 5 in all experiments. The relaxation of the gap ϵ accelerated GORFLM obviously and the average runtime did not increase much w.r.t the point number M. Especially, in the case of $M = 15{,}830$, the average runtime of IBCO and ILP-RansaCov is one order of magnitude larger than the average runtime of GORFLM with the gap $\epsilon = 0.15$. Lo*-Ransac, a ransac-type method, is the most efficient, while it cannot detect all the planes correctly because of its random property. We set the time limit to 720 s for each run and then the total time limit of fitting the five planes is 3600 s, which is slightly longer than GORFLM. Overall, this experiment reveals the potential ability of the proposed method in fitting planes with different levels of noise and large number of points ($M = 15{,}830$).

3.3.2. Affine fundamental matrix

In this section, we examined the feasibility of GORFLM in estimating the affine fundamental matrix from real images, which can be formulated as a linear model fitting problem [13] and the dimension is higher than that of the line and plane fitting in previous sections. We used Wadham, Merton College I and Merton College III data from the website of [40] and estimated the affine fundamental matrix between the first two frames of each data. These three data were chosen because the fundamental matrix can be well approximated by an affine one. Feature points are provided on each image and the ground truth matches between some of these points are given. For every unmatched point in the first frame we randomly chose an unmatched point in the second frame for it to form a false-matching pair. These inliers and outliers can be perfectly separated by a fundamental matrix. For each data set, we first calculated an affine fundamental matrix by the least squares method using only ground truth matching pairs. And then the inlier/outlier label of each point set was updated by the maximum residual between the affine fundamental matrix and ground truth matching pairs. We ran GORFLM, Lo*-Ransac, IBCO, A* search and ILP-RansaCov 50 times and calculated the average misclassification error [41,42]:

$$ME(Misclassification\ error) = \frac{number\ of\ mislabeled\ data\ points}{total\ number\ of\ data\ points} \tag{11}$$

The inlier threshold of all the methods are adjusted to their best performance. Concretely, for GORFLM, Lo*-Ransac and ILP-RansaCov, the inlier thresholds of are 0.05, 0.06, and 0.04 in Merton College III, Merton College I and Wadham data, respectively. And the inlier thresholds of other methods are 0.08, 0.09 and 0.08 in Merton College III, Merton College I and Wadham data. The gap ϵ is 0.1 for GORFLM and the time limit is 1000 s for A* search. Examples of the misclassified points by all methods are shown in Fig. 10. The box plots of the results are illustrated in Fig. 11. We find that GORFLM achieved stable performance owing to its global optimization. The accuracy of GORFLM, ILP-RansaCov and Lo*-Ransac are similar in the experiment of MertonCollegeIII, which may be because there is only one model in this problem.

3.3.3. Video motion segmentation with different levels of noise

In this experiment, we evaluated the performance of GORFLM on video motion segmentation with outliers. We used the traffic2 group from the Hopkins 155 data-set [43], corrected by removing mistrackings using the method in [44]. The traffic2 group consists of 31 sequences, each containing 2 motions. The Cars2_07_g13 and the Cars5_g12 sequences were not used in this experiment, because the ground-truth labels of these two sequences do not correspond to the images well in the corrected data set [44]. We used the code provided on [45] to generate 20 outliers for these 29 sequences in the traffic2 group. Under the affine camera projection assumption, [44] has shown that the trajectory of points belonging to a single rigid moving object are contained in a subspace of rank 4 to 2 according to different level of degeneration of the motion. Following [46], we projected the data

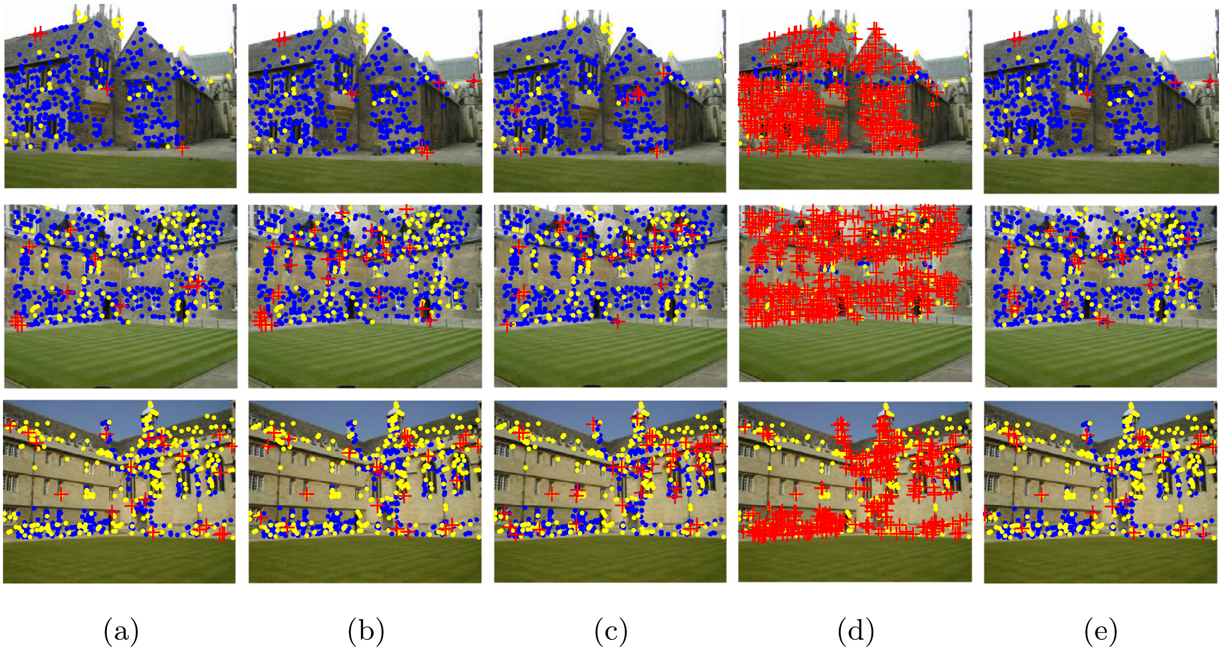


Fig. 10. Images used in estimating the affine fundamental matrix: MertonCollegeIII, MertonCollegeI and Wadham in the top-down order. Their inlier ratios are 89.86%, 78.04% and 51.60% respectively. (a–e) The worst misclassification result obtained by GORFLM, Lo*-Ransac, IBCO, A* search and ILP-RansaCov respectively. Yellow dot—outlier, blue dot—inlier, and red cross—misclassified point.

Table 2
Average runtime and scores in detecting the five planes in the Living room data set with point number of M.

Ggap0.20				**Ggap0.15**				**Ggap0.10**			
M	Gap	Time (s)	Score	M	Gap	Time (s)	Score	M	Gap	Time (s)	Score
4105	0.20	516.81	**5.00**	4105	0.15	848.72	**5.00**	4105	0.10	1470.93	**5.00**
15 830	0.20	591.54	**5.00**	15 830	0.15	997.45	**5.00**	15 830	0.10	1755.33	**5.00**

L			**B**			**A**			**R**		
M	Time (s)	Score	M	Time (s)	Score	M	Time (s)	Score	M	Time (s)	Score
4105	**10.14**	4.90	4105	589.16	1.00	4105	3602.46	0.00	4105	658.76	4.10
15 830	**87.26**	4.94	15 830	11 678.91	1.00	15 830	3601.74	0.00	15 830	14 610.09	4.00

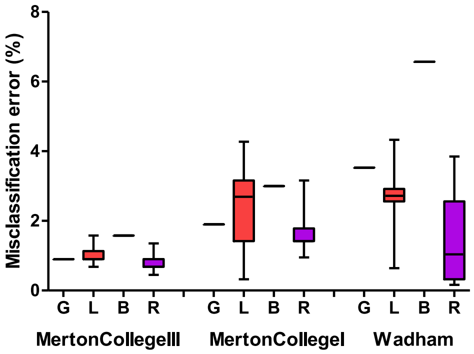


Fig. 11. The box plots of misclassification number for fitting affine fundamental matrix. The box plot of each data set in A* search is not listed because of its visually poor performance as illustrated in Fig. 10d.

onto an affine 4D space and the motion segmentation is translated into a 3D plane-fitting problem. We chose the best inlier threshold parameters for each algorithm. The gap ϵ of GORFLM is 0.15 and the time limit of A* search is 200 s in detecting one motion. We ran each of them 50 times for each sequence and the average Misclassification Error (ME) in segmenting the total 29 sequences of each algorithm is listed in Table 3. To further reflect the performance of all methods in the total 29 sequences, we listed ME of each sequence in Appendix B.

Table 3
Misclassification error (ME %) on traffic2 group with outliers.

traffic2 with outliers	G	L	B	A	R
ME	**2.15%**	2.56%	20.14%	45.49%	6.18%
Time (s)	28.88	**0.05**	0.82	226.12	2.89

GORFLM achieved the best performance among them. The ME of IBCO and ILP-RansaCov is high because it misclassified many trajectories, where the number of inliers and the noise level of the two planes in the 3D space is quite different. From these challenging sequences, we chose five examples of their segmentation results as shown in Fig. 12. Fig. 12a is the trajectory projection in 3D space, where the planes with fewer inlier points are always have larger noise level. The ME of A* is high because it cannot converge in many sequences. Consequently, the fitting result indicates that GORFLM is robust to the data with different number of inliers and different noise levels.

4. Discussion and conclusion

In this paper, we propose a globally optimal and robust algorithm, which is named GORFLM, for solving the problem of linear model fitting. This algorithm abandoned the traditional idea of maximizing the consensus set, and formulated the model fitting into an optimization problem of the minimizing the sum of negative Gaussian functions. So

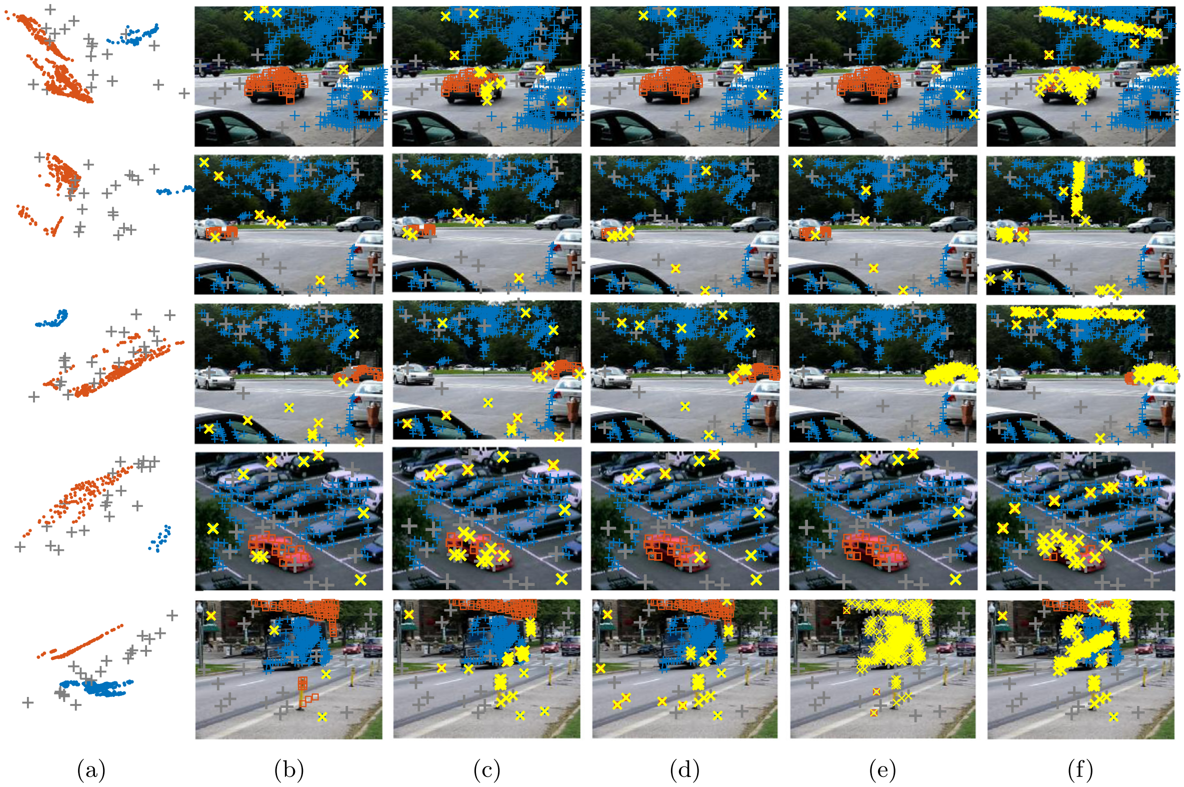


Fig. 12. Examples of Hopkins 155 notion segmentation results with quite different inlier quantities and noise levels: Cars2, Cars5_g13, Cars5_g23, Kanatani1, and Truck2 in the top-down order. (a) Trajectory projection in 3D space. (b–f) The results obtained by GORFLM, Lo*-Ransac, IBCO, A* search and ILP-RansaCov, respectively. Blue "+" and orange "□" detected inliers of the two motions, gray "+" detected outliers, and yellow "×" misclassified points.

that from the theoretical point of view, the existence of the globally optimal solution for model fitting is guaranteed.

Through applying our algorithm to solve a variety of computer vision problem, it is demonstrated that our method is superior than the state-of-the-art methods of model fitting. Especially when fitting models with different levels of noise and large proportion of outliers, our method shows better robustness and global optimality. In the experiments of low dimensions, the advantage of speed of the proposed method is revealed gradually with the increase of the number of sampling points.

In the studying of model fitting, to maximize the inliers of the models is the final goal of the optimization in traditional methods. The random and deterministic methods cannot guarantee the global optimization of the fitting results. And the methods looking for the global solution tend to catastrophically fail when the model fitting problem is increasingly hard. In addition, most traditional methods used the hard inlier threshold to determine whether a point is an inlier or an outlier. Therefore, it inevitably leads to the result that the choice of the inlier threshold has a great influence on the fitting result. Although recently many solutions have been proposed, they have not been able to get rid of the traditional idea of maximizing the consensus set, so the effect in not satisfactory. The main reason, why our method has the advantages mentioned above, is that we can use the soft loss function to construct as an object function to transform the problem of the model fitting into the problem of minimizing the sum of negative Gaussian functions. Then we derived a convex quadratic lower bound of this objective function, and the lower bound derived is very tight. This idea ensures the existence of the globally optimal solution on theory, and the optimal solution can be obtained by the BnB algorithm. This fundamentally solves the problems of local optimum and poor robustness problem brought by the two computing processes of random sampling and inlier counting. As a result, our method has the advantages of global optimization and strong robustness.

There is no doubt that there are still many room of improvements in our method. For example, although the method we propose is faster in solving the low dimensional problem, the computation cost will increase rapidly with the increase of the dimensions, which is proved in previous experiments in this paper. Therefore, in the future work, we will further explore more effective strategies of the branching and searching. In addition, using GPU to realize parallel computing is one of the methods to improve the speed for use in high dimensions.

CRediT authorship contribution statement

Yiru Wang: Methodology, Software, Investigation, Writing - original draft, Writing - review & editing. **Yinlong Liu:** Methodology, Software, Investigation, Formal analysis, Writing review & editing. **Xuechen Li:** Validation, Visualization, Formal analysis. **Chen Wang:** Validation, Formal analysis. **Manning Wang:** Conceptualization, Writing - original draft, Writing - review & editing, Funding acquisition, Project administration. **Zhijian Song:** Writing - review & editing, Funding acquisition, Supervision.

Acknowledgments

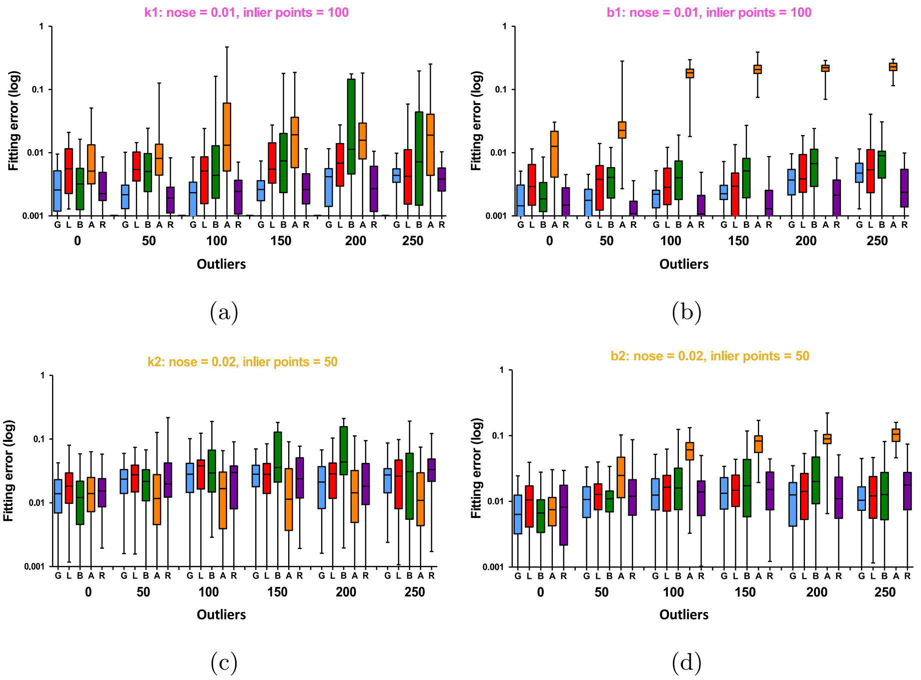
This work was supported in part by the National Natural Science Foundation of China under Grant 81701795 and Grant 81471758.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A

See Fig. A.13.



(a)

(b)

(c)

(d)

Fig. A.13. The box plots of fitting errors of the slop ($k1$, $k2$) and intercept ($b1$, $b2$) of the two lines in the experiment in Section 3.2.1.

Table B.4
ME of each of the 29 sequences in Section 3.3.3.

traffic2 with outliers	G		L		B		A		R	
	ME	Time (s)	ME	Time (s)	ME	Time (s)	ME	Time (s)	ME	Time (s)
cars1	**0.00%**	15.00	0.08%	0.05	31.48%	4.19	92.13%	400.11	0.34%	3.02
cars2	**1.00%**	7.50	1.98%	0.09	1.20%	1.20	1.20%	200.04	11.84%	7.86
cars4	**0.63%**	17.31	1.68%	0.04	7.50%	0.56	86.88%	183.48	6.08%	0.82
cars2_06_g12	3.23%	64.47	2.84%	0.04	17.74%	0.32	61.29%	1.96	**1.77%**	0.14
cars2_06_g13	**1.87%**	23.34	1.91%	0.02	12.15%	0.36	87.85%	400.05	5.03%	0.43
cars2_06_g23	**1.75%**	23.58	1.77%	0.02	10.53%	0.45	82.46%	400.05	7.14%	0.43
cars2B_g12	12.61%	51.14	19.93%	0.04	54.95%	0.46	72.07%	400.03	**6.20%**	0.39
cars2B_g13	**1.15%**	9.08	1.33%	0.18	1.92%	0.74	0.77%	27.48	11.49%	8.39
cars2B_g23	1.01%	9.01	0.88%	0.04	4.45%	0.93	**0.00%**	20.65	12.61%	7.52
cars2_07_g12	10.00%	138.70	9.00%	0.03	41.67%	0.37	61.67%	18.38	**4.10%**	0.13
cars2_07_g23	**1.42%**	16.23	2.25%	0.03	3.77%	0.28	2.36%	122.29	6.49%	1.35
cars3_g12	**0.79%**	38.86	0.95%	0.03	38.89%	0.59	73.02%	23.10	1.27%	0.49
cars3_g13	1.03%	10.42	**0.96%**	0.07	1.23%	0.73	7.60%	400.06	10.30%	7.92
cars3_g23	**0.19%**	9.39	0.30%	0.03	**0.19%**	0.89	0.57%	31.89	11.14%	8.58
cars5_g13	1.96%	8.29	**1.46%**	0.04	2.79%	0.70	1.68%	154.98	11.73%	3.86
cars5_g23	**2.40%**	8.93	2.84%	0.09	3.73%	0.58	12.80%	400.06	16.35%	4.18
cars6	1.66%	11.34	1.77%	0.14	3.52%	0.66	**1.45%**	10.67	11.52%	7.51
cars7	0.96%	10.40	0.40%	0.04	0.96%	0.84	**0.38%**	29.72	10.57%	8.14
cars8	1.52%	30.07	**1.10%**	0.03	50.25%	0.75	82.23%	400.04	2.04%	1.16
cars9_g12	2.30%	98.88	5.89%	0.04	54.02%	0.24	77.01%	108.98	**1.70%**	0.25
cars9_g13	3.48%	26.08	1.31%	0.04	27.86%	1.21	38.31%	400.07	**1.23%**	1.23
cars9_g23	**0.00%**	19.81	**0.00%**	0.03	5.13%	0.68	**0.00%**	400.03	1.94%	1.21
cars10_g12	**0.49%**	24.87	0.96%	0.03	34.80%	0.62	93.63%	400.08	0.98%	1.23
cars10_g13	**0.45%**	22.39	0.51%	0.03	42.99%	0.67	91.40%	400.05	**0.45%**	1.51
cars10_g23	1.20%	35.83	0.48%	0.03	36.14%	0.48	39.16%	400.03	**0.00%**	0.82
kanatani1	5.77%	10.81	4.82%	0.06	5.77%	2.28	**3.21%**	17.70	6.68%	0.87
kanatani2_	**0.00%**	30.74	0.88%	0.03	56.10%	0.56	75.61%	5.41	1.00%	0.22
truck1	**2.15%**	51.80	2.81%	0.04	24.73%	0.60	77.42%	400.04	3.22%	1.03
truck2	**1.22%**	13.31	3.18%	0.06	7.62%	0.93	95.12%	400.11	14.13%	3.23

Appendix B

See Table B.4.

References

[1] T.-J. Chin, D. Suter, The maximum consensus problem: Recent algorithmic advances, Synthesis Lectures on Computer Vision, Morgan and Claypool, 2017, pp. 1–194.

[2] K. Lebeda, J. Matas, O. Chum, Fixing the locally optimized ransac, in: British Machine Vision Conference, Citeseer, 2012, pp. 1–11.

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

[4] O. Chum, J. Matas, J. Kittler, Locally optimized ransac, in: Joint Pattern Recognition Symposium, Springer, 2003, pp. 236–243.

[5] O. Chum, J. Matas, Matching with prosac-progressive sample consensus, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2005, pp. 220–226.

[6] S. Choi, T. Kim, W. Yu, Performance evaluation of ransac family, in: British Machine Vision Conference, Citeseer, 2009, pp. 1–12.

[7] B.J. Tordoff, D.W. Murray, Guided-mlesac: Faster image transform estimation by using matching priors, IEEE Trans. Pattern Anal. Mach. Intell. 27 (10) (2005) 1523–1535.

[8] O. Chum, J. Matas, Optimal randomized ransac, IEEE Trans. Pattern Anal. Mach. Intell. 30 (8) (2008) 1472–1482.

[9] H. Le, T.-J. Chin, D. Suter, An exact penalty method for locally convergent maximum consensus, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2017, pp. 379–387.

[10] P. Purkait, C. Zach, A. Eriksson, Maximum consensus parameter estimation by reweighted L_1 methods, in: International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition, Springer, 2017, pp. 312–327.

[11] Z. Cai, T.-J. Chin, H. Le, D. Suter, Deterministic consensus maximization with biconvex programming, in: European Conference on Computer Vision, Springer, 2018, pp. 685–700.

[12] J.C. Bazin, Y. Seo, C. Demonceaux, P. Vasseur, K. Ikeuchi, I. Kweon, M. Pollefeys, Globally optimal line clustering and vanishing point estimation in manhattan world, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 638–645.

[13] H. Li, Consensus set maximization with guaranteed global optimality for robust geometry estimation, in: IEEE International Conference on Computer Vision, IEEE, 2009, pp. 1074–1080.

[14] Y. Zheng, S. Sugimoto, M. Okutomi, Deterministically maximizing feasible subsystem for robust model fitting with unit norm constraint, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2011, pp. 1825–1832.

[15] O. Enqvist, E. Ask, F. Kahl, Robust fitting for multiple view geometry, in: European Conference on Computer Vision, Springer, 2012, pp. 738–751.

[16] T.-J. Chin, P. Purkait, A. Eriksson, D. Suter, Efficient globally optimal consensus maximisation with tree search, IEEE Trans. Pattern Anal. Mach. Intell. 39 (4) (2017) 758–772.

[17] C. Olsson, O. Enqvist, F. Kahl, A polynomial-time bound for matching and registration with outliers, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2008, pp. 1–8.

[18] T.-J. Chin, P. Purkait, A. Eriksson, D. Suter, Efficient globally optimal consensus maximisation with tree search, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2015, pp. 2413–2421.

[19] T.M. Breuel, On the use of interval arithmetic in geometric branch and bound algorithms, Pattern Recognit. Lett. (ISSN: 0167-8655) 24 (9) (2003) 1375–1384.

[20] H. Li, R. Hartley, The 3d-3d registration problem revisited, in: IEEE International Conference on Computer Vision, IEEE, 2007, pp. 1–8.

[21] C. Olsson, F. Kahl, M. Oskarsson, Branch-and-bound methods for euclidean registration problems, IEEE Trans. Pattern Anal. Mach. Intell. 31 (5) (2009) 783–794.

[22] J. Yang, H. Li, D. Campbell, Y. Jia, Go-icp: A globally optimal solution to 3d icp point-set registration, IEEE Trans. Pattern Anal. Mach. Intell. 38 (11) (2016) 2241–2254.

[23] D. Campbell, L. Petersson, L. Kneip, H. Li, S. Gould, The alignment of the spheres: Globally-optimal spherical mixture alignment for camera pose estimation, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2019, pp. 11796–11806.

[24] D. Campbell, L. Petersson, L. Kneip, H. Li, Globally-optimal inlier set maximisation for simultaneous camera pose and feature correspondence, in: IEEE International Conference on Computer Vision, IEEE, 2017, pp. 1–10.

[25] R.I. Hartley, F. Kahl, Global optimization through rotation space search, Int. J. Comput. Vis. 82 (1) (2009) 64–79.

[26] E. Ask, O. Enqvist, F. Kahl, Optimal geometric fitting under the truncated l2-norm, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2013, pp. 1722–1729.

[27] A. Blake, A. Zisserman, Visual reconstruction, Math. Comp. 373 (2) (1987) 131–135.

[28] P.J. Rousseeuw, Least median of squares regression, J. Amer. Stat. Assoc. 79 (388) (1984) 871–880.

[29] C. Olsson, F. Kahl, Generalized convexity in multiple view geometry, J. Math. Imaging Vision 38 (1) (2010) 35–51.

[30] https://www.gurobi.com.

[31] L. Magri, A. Fusiello, Multiple models fitting as a set coverage problem, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2016, pp. 3318–3326.

[32] https://github.com/ZhipengCai/Demo---Deterministic-consensus-maximization-with-biconvex-programming.

[33] http://www.diegm.uniud.it/fusiello/demo/cov/.

[34] G. Xiao, H. Wang, T. Lai, D. Suter, Hypergraph modelling for geometric model fitting, Pattern Recognit. 60 (2016) 748–760.

[35] T.-J. Chin, H. Wang, D. Suter, Robust fitting of multiple structures: The statistical learning approach, in: IEEE International Conference on Computer Vision, IEEE, 2009, pp. 413–420.

[36] T.-J. Chin, Z. Cai, F. Neumann, Robust fitting in computer vision: easy or hard?, in: European Conference on Computer Vision, Springer, 2018, pp. 715–730.

[37] J. Erickson, S. Har-Peled, D.M. Mount, On the least median square problem, Discrete Comput. Geom. 36 (4) (2006) 593–607.

[38] D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, A.Y. Wu, On the least trimmed squares estimator, Algorithmica 69 (1) (2014) 148–183.

[39] http://cophy-wiki.informatik.uni-koeln.de/index.php/Branch-and-bound.

[40] http://www.robots.ox.ac.uk/∼vgg/data.

[41] L. Magri, A. Fusiello, T-linkage: A continuous relaxation of j-linkage for multi-model fitting, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014, pp. 3954–3961.

[42] S. Mittal, S. Anand, P. Meer, Generalized projection based m-estimator: Theory and applications, IEEE Trans. Pattern Anal. Mach. Intell. 34 (12) (2012) 2351–2364.

[43] R. Tron, R. Vidal, A benchmark for the comparison of 3-d motion segmentation algorithms, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2007, pp. 1–8.

[44] Y. Sugaya, Y. Matsushita, K. Kanatani, Removing mistracking of multibody motion video database hopkins155, in: British Machine Vision Conference, Citeseer, 2013, pp. 26.1–26.10.

[45] http://vision.jhu.edu/code/.

[46] Y. Sugaya, K. Kanatani, Geometric structure of degeneracy for multi-body motion segmentation, in: Statistical Methods in Video Processing, Springer, 2004, pp. 13–25.