Searching for Representative Modes on Hypergraphs for Robust Geometric Model Fitting

Hanzi Wang, *Senior Member, IEEE*, Guobao Xiao, Yan Yan, *Member, IEEE*, and David Suter

Abstract—In this paper, we propose a simple and effective geometric model fitting method to fit and segment multi-structure data even in the presence of severe outliers. We cast the task of geometric model fitting as a representative mode-seeking problem on hypergraphs. Specifically, a hypergraph is first constructed, where the vertices represent model hypotheses and the hyperedges denote data points. The hypergraph involves higher-order similarities (instead of pairwise similarities used on a simple graph), and it can characterize complex relationships between model hypotheses and data points. In addition, we develop a hypergraph reduction technique to remove "insignificant" vertices while retaining as many "significant" vertices as possible in the hypergraph. Based on the simplified hypergraph, we then propose a novel mode-seeking algorithm to search for representative modes within reasonable time. Finally, the proposed mode-seeking algorithm detects modes according to two key elements, i.e., the weighting scores of vertices and the similarity analysis between vertices. Overall, the proposed fitting method is able to efficiently and effectively estimate the number and the parameters of model instances in the data simultaneously. Experimental results demonstrate that the proposed method achieves significant superiority over several state-of-the-art model fitting methods on both synthetic data and real images.

Index Terms—Geometric model fitting, hypergraph construction, mode-seeking, multi-structure data

1 Introduction

Geometric model fitting is a challenging research problem for a variety of applications in computer vision, such as optical flow calculation, motion segmentation and homography/fundamental matrix estimation. Given that data usually contain outliers, the task of geometric model fitting is to robustly estimate the number and the parameters of model instances in data. A number of robust geometric model fitting methods (e.g., [1], [2], [3], [4], [5], [6], [7], [8]) have been proposed. One of the most popular robust fitting methods is RANSAC [2] due to its efficiency and simplicity. However, RANSAC is sensitive to a threshold specified by a user and it is originally designed to fit single-structure data. During the past few decades, many robust model fitting methods have been proposed to deal with multi-structure data, such as KF [1], PEARL [3], AKSWH [6], T-linkage [4], SCAMS [9] and PM [7]. However, current fitting methods are still far from being practical to deal with real-world problems, due to the limitations of speed or accuracy. In this paper, we aim to accurately detect model instances in data within reasonable time.

- *H. Wang and Y. Yan are with the Fujian Key Laboratory of Sensing and Computing for Smart City, School of Information Science and Engineering, Xiamen University, Xiamen, Fujian 361005, China. E-mail: hanzi.wang@ieee.org, yanyan@xmu.edu.cn.*
- *G. Xiao is with the School of Aerospace Engineering, Xiamen University, Xiamen, Fujian 361005, China. E-mail: guobaoxiao@xmu.edu.cn.*
- *D. Suter is with the School of Computer Science, University of Adelaide, Adelaide, SA 5005, Australia. E-mail: dsuter@cs.adelaide.edu.au.*

Manuscript received 23 Dec. 2016; revised 30 Sept. 2017; accepted 3 Feb. 2018. Date of publication 6 Feb. 2018; date of current version 13 Feb. 2019. (Corresponding author: Hanzi Wang.)
Recommended for acceptance by H. Li.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2018.2803173

Note that a hypergraph involves high-order similarities, and some works have been proposed to deal with the model fitting problem based on hypergraphs, e.g., [10], [11], [12], [13]. Although using a hypergraph is beneficial for a model fitting method in terms of the fitting accuracy, it also causes the problem of high computational complexity due to complex relationships in the hypergraph. To reduce the computational complexity, these hypergraph based fitting methods usually fix the degree of each hyperedge in the hypergraph to be a small constant value. However, such a way cannot characterize the complex relationships between model hypotheses and data points (for the model fitting problem) very well.

In this paper, we propose a simple and effective Mode-Seeking on Hypergraphs Fitting method (MSHF) to fit and segment multi-structure data. The proposed method (MSHF) starts from hypergraph construction, where vertices and hyperedges respectively correspond to model hypotheses and data points (as shown in Fig. 1). We also develop a novel hypergraph reduction technique to remove insignificant vertices, which improves the effectiveness of the constructed hypergraph. After that, we propose a novel mode-seeking algorithm to search for representative modes on the hypergraph. Finally, MSHF simultaneously estimates the number and the parameters of all model instances in data (according to the detected modes).

The proposed MSHF method has three main advantages over previous model fitting methods. First, the constructed hypergraph is able to effectively characterize the complex relationships between model hypotheses and data points, and the size of each hyperedge in the hypergraph is data-driven. Moreover, the hypergraph can be directly used for geometric model fitting. That is, MSHF avoids constructing a pairwise affinity matrix as used in [14] and [15]. Note that the projection from a hypergraph to an induced graph

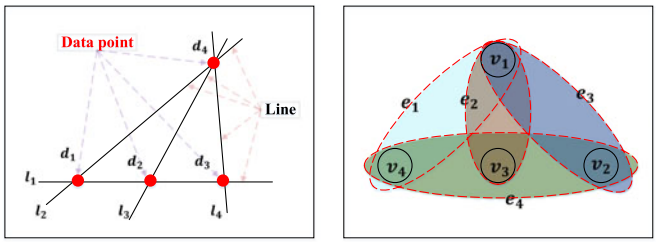


Fig. 1. An example of hypergraph construction for line fitting. (a) The input data including four data points and four model hypotheses (i.e., lines). (b) A hypergraph with four vertices $\{v_i\}_{i=1}^4$ and four hyperedges $\{e_i\}_{i=1}^4$. In the hypergraph, each vertex v_i and each hyperedge e_i denote a model hypothesis l_i and a data point d_i in (a), respectively.

usually causes information-loss (except for the methods using the affinity tensor). Second, MSHF deals with geometric model fitting in the parameter space, which can effectively handle severe unbalanced data (i.e., the numbers of inliers belonging to different model instances in data are significantly different). Third, MSHF performs mode seeking by analyzing the weighting scores and the similarity between vertices on a hypergraph, which shows great scalability to solve the model fitting problem. We demonstrate that MSHF is a highly robust method for geometric model fitting by conducting extensive experimental evaluations and comparisons in Section 5.

This paper is an extension of our previous work in [16]. We have made several significant improvements, including a novel hypergraph reduction technique to improve the performance of the original proposed method on fitting accuracy (Section 3.2), more theoretical analyses (Sections 2, 4.2 and 6) and more experimental justification (Section 5). We have also used the neighboring constraint to reduce the computational cost of MSHF (Section 3.3).

The rest of the paper is organized as follows: We provide an overview of the related work in Section 2. We describe the components of the proposed fitting method in Section 3 and summarize the complete fitting method in Section 4. We present the experimental results on both synthetic and real data in Section 5. We analyze the limitations of the proposed method in Section 6, and draw conclusions in Section 7.

2 Related Work

In this section, we briefly review the related work on robust geometric model fitting including the hypergraph-based fitting methods, the mode-seeking based fitting methods, and several other state-of-the-art fitting methods.

2.1 Hypergraph Based Fitting Methods

Recently, some hypergraph based methods, e.g., [10], [11], [12], [13], have been proposed for robust model fitting due to its effectiveness. For example, Liu and Yan [11] proposed the random consensus graph (RCG) to fit multiple structures in data. Purkait et al. [13] proposed to use large hyperedges for face clustering and motion segmentation.

Compared with the hypergraph constructed in the previous methods (e.g., [10], [11], [12], [13]), where a hyperedge is constrained to connect with a fixed number of vertices, the hyperedge of hypergraphs constructed in this paper can connect with a varying number of vertices (that is we construct non-uniform hypergraphs as those in [17]). In addition, the vertices of the hypergraph constructed in the previous methods (e.g., [10], [11], [12], [13]) represent data points, while the vertices of the hypergraph constructed in this paper denote model hypotheses. Therefore, we can directly deal with the model fitting problem in the parameter space.

2.2 Mode-Seeking Based Fitting Methods

Mode-seeking is a simple and effective data analysis technique, and it can be extended to deal with model fitting problems (e.g., [18], [19], [20], [21]). These mode-seeking based fitting methods select model instances by seeking the peaks of the underlying distributions in the parameter space. Each point in the parameter space corresponds to a model hypothesis, and the detected modes represent the estimated model instances. For example, Mean Shift [18] and its variant [19] attempt to find peaks in the parameter space to estimate the model instances in the data. Hough [20] proposed a robust fitting method, called the Hough Transform (HT), which discretizes the parameter space into bins and then votes for these bins according to the information derived from a set of sampled data points. The bins with higher votes correspond to the estimated model instances in the data. Xu et al. [21] proposed an extended version of HT, i.e., Randomized Hough Transform (RHT). RHT uses model hypotheses to vote for the bins in the parameter space to reduce the computational cost of HT.

The above-mentioned mode-seeking based fitting methods can estimate the number of model instances in data, but their performance largely depends on the proportion of good model hypotheses in the generated model hypotheses derived from a set of sampled data points. As a result, these fitting methods often wrongly estimate the number of model instances when the proportion of good model hypotheses is low. In contrast, the proposed mode-seeking based fitting method alleviates this drawback. Specifically, the proposed method can effectively seek modes by analyzing both the weighting scores of vertices, and the similarity between the vertices of hypergraphs. The vertices corresponding to good model hypotheses usually show unique characteristics even when the proportion of good model hypotheses is low. Thus, we can select these vertices as modes for model fitting.

In this paper, we integrate hypergraph construction with mode seeking for solving the model fitting problems. The constructed hypergraph can effectively capture the correlation information of model hypotheses and data points, and the proposed mode-seeking method can efficiently search for representative modes, which correspond to model instances in data, on the hypergraph. The proposed method tightly couples both hypergraph construction and mode-seeking, by which it yields better performance for model fitting.

2.3 Other Related Fitting Methods

In addition to the above-mentioned robust fitting methods, there are several other related fitting methods, such as KF [1], J-linkage [5], T-linkage [4], SCAMS [9], PM [7], PEARL [3], AKSWH [6], HS [22], RELRT [23] and GMD [24]. KF, J-linkage, T-linkage and SCAMS directly deal with data points for model fitting but they are sensitive to unbalanced data distributions that are quite common in practical applications. In addition, these methods have difficulties in dealing with the

data points near the intersection of model instances. The computational costs of J-linkage and T-linkage are high due to the use of the agglomerative clustering procedure. The other robust fitting methods also have some problems. For example, PM requires the input of the number of model instances in data; PEARL is sensitive to the initial generated hypotheses; AKSWH may remove some good model hypotheses corresponding to the correct model instances involving a small number of data points, during the procedure of selecting significant hypotheses; HS encounters the computational complexity problem due to the expansion and dropping strategy used; both RELRT and GMD only work for single-structure data.

The proposed method in this paper is based on the mode-seeking technique, which is related to the clustering technique. However, the proposed method directly searches for the cluster centers in the parameter space, which can avoid dealing with the data points near the intersection of model instances. In addition, the proposed method can achieve more accurate fitting results for multiple-structure data within reasonable time.

3 The Methodology

In this paper, the geometric model fitting problem is formulated as a mode-seeking problem on a hypergraph. We describe the details of the proposed MSHF method in this section. Specifically, we first construct hypergraphs for model fitting in Section 3.1. Then, we develop a novel hypergraph reduction technique to remove the "insignificant" vertices in the hypergraph in Section 3.2. After that, we propose a novel mode-seeking algorithm to search for representative modes on the hypergraph in Section 3.3.

3.1 Hypergraph Construction

A hypergraph $G = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ consists of vertices $\mathcal{V}$, hyperedges $\mathcal{E}$, and weights $\mathcal{W}$. Each vertex v is weighed by a weighting score $w(v)$. When $v \in e$, a hyperedge e is incident with a vertex v. Then an incident matrix $\mathbf{H}$, whose entry at (v, e) satisfies $h(v, e) = 1$ if $v \in e$ and 0 otherwise, is used to represent the relationships between vertices and hyperedges in the hypergraph G. For a vertex $v \in \mathcal{V}$, its degree is defined by $\delta(v) = \sum_{e \in \mathcal{E}} h(v, e)$.

In our case, a vertex in a hypergraph represents a model hypothesis and a hyperedge denotes a data point. The detailed procedure of hypergraph construction is described as follows: Given a set of data points $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$, we first sample a set of minimal subsets from $\mathbf{X}$. A minimal subset contains the minimum number of data points, which is necessary to estimate a model hypothesis (e.g., 2 for line fitting and 4 for homography fitting). Then we generate a set of model hypotheses using the minimal subsets and estimate their inlier noise scales. In this paper, we use IKOSE [6] as the inlier noise scale estimator due to its efficiency. After that, we connect each vertex (i.e., a model hypothesis) to the corresponding hyperedges (i.e., the inliers of the model hypothesis). We can see that, the constructed hypergraph effectively characterizes the relationship between model hypotheses and data points. In this manner, we can directly perform mode-seeking on the hypergraph for model fitting.

The constructed hypergraph usually includes a large number of vertices, and we assign a weighting score $w(v)$ to each vertex v to measure its quality. Inspired by [6], we employ the density estimate technique through the following equation to compute $w(v)$ (see Section 3.2 in [6])

$$w(v) = \frac{1}{n} \sum_{e \in \mathcal{E}} \frac{\Psi(r_e(v)/b(v))}{\hat{s}(v) b(v)}, \tag{1}$$

where $\Psi(\cdot)$ is a kernel function (such as the Epanechnikov kernel); $r_e(v)$ is a residual measured with the Sampson distance [25] from the model hypothesis to a data point; n and $\hat{s}(v)$ are the number of hyperedges and the inlier noise scale of the model hypothesis, respectively, and $b(v)$ is the window radius (bandwidth), which is estimated using [26]

$$b(v) = \left[\frac{243 \int_{-1}^{1} \Psi(\lambda)^2 d\lambda}{35n \int_{-1}^{1} \lambda^2 \Psi(\lambda) d\lambda} \right]^{0.2} \hat{s}(v). \tag{2}$$

Since the good model hypotheses corresponding to the model instances in the data have significantly more inliers with smaller absolute residuals than the bad model hypotheses, the weighting scores of the vertices corresponding to the good model hypotheses should be higher than those of the vertices corresponding to the bad model hypotheses. However, weighing a vertex based on residuals may not be robust to outliers, especially for extreme outliers. To weaken the influence of outliers, we only consider the residuals of the corresponding inliers belonging to the model hypotheses (note that [6] considers the residuals of all data points, which is less effective and robust). Thus, based on a hypergraph G, Eq. (1) can be reformulated as

$$w(v) = \frac{1}{\delta(v)} \sum_{e \in \mathcal{E}} \frac{h(v, e) \Psi(r_e(v)/b(v))}{\hat{s}(v) b(v)}, \tag{3}$$

where $\delta(v)$ is the degree of a vertex v and $h(v, e)$ is an entry of the incident matrix $\mathbf{H}$ corresponding to the hypergraph G. Recall that $h(v, e)$ will be assigned 0 if the corresponding data point is not an inlier belonging to the corresponding model hypothesis. Thus, compared with Eq. (1), the weighting score computed by Eq. (3) is not greatly influenced by outliers.

3.2 Hypergraph Reduction

In [16], we originally use a weight-aware sampling technique (WAS) to sample vertices according to their weighting scores to avoid ineffective mode-seeking results. The main task of WAS is to remove a few insignificant vertices corresponding to bad model hypotheses with low weighting scores. However, although WAS can effectively remove some bad model hypotheses, some good model hypotheses may be also discarded in some cases, which will affect the accuracy of the mode-seeking algorithm (see Section 3.3). Furthermore, WAS has a low probability to sample some bad model hypotheses. This will make the whole algorithm unstable. Therefore, we propose a new hypergraph reduction technique in this paper, which is inspired by the information theoretic approach proposed in [27], to remove insignificant vertices corresponding to bad model hypotheses with low weighting scores while preserving most of the significant vertices corresponding to good model hypotheses.

Given a hypergraph G with vertices $\mathcal{V} = \{v_1, v_2, \ldots, v_m\}$ and the associated weighting scores $\mathcal{W} = \{w_1, w_2, \ldots, w_m\}$, where m is the number of vertices, i.e., the number of the generated model hypotheses, let $q_i = mean(\mathcal{W}) - w_i$ denote the gap between the average weighting score of vertices and the weight of v_i. Thus, as in [27], we can compute the prior probability p_i of the vertex v_i by normalizing q_i as follows:

$$p_i = \begin{cases} \frac{q_i}{\sum_{j=1}^{m} q_j}, & if\ q_i > 0, \\ \xi \quad , & otherwise, \end{cases} \tag{4}$$

where ξ denotes a small positive value to avoid zero division. Then we can obtain the entropy of the prior probability, which is used as an adaptive threshold to select significant vertices

$$E = -\sum_{i=1}^{m} p_i \log\, p_i. \tag{5}$$

Finally, we retain the vertices with higher quantities of information than the entropy E, and the retained vertices $\mathcal{V}$ are defined as

$$\mathcal{V} = \{v_i | -\log p_i > E\}. \tag{6}$$

Note that [6] also applies the information theoretic approach to select significant hypotheses. However, in [6], the gap between the weight of a model hypothesis and the maximum weight is used to select significant hypotheses. However, such a strategy may remove not only many bad model hypotheses but also some good model hypotheses having less number of inliers. In contrast, the proposed method removes less significant model hypotheses than [6] and retains more good ones. Thus, it is more effective than the method used in [6] for removing insignificant model hypotheses while preserving significant ones.

3.3 The Mode-Seeking Algorithm

Given a hypergraph G^*, we aim to seek modes by searching for the authority peaks that correspond to model instances in data. The "authority peaks" in a hypergraph can be defined as follows.

Definition 1. *Authority peaks are the vertices that have the local maximum values of weighting scores in the hypergraph.*

The vertices that have the local maximum values of weighting scores correspond to the modes in a hypergraph. Here, "local" refers to the neighbors of a vertex in a hypergraph. This definition is consistent with the conventional concept of modes, which are defined as the significant peaks of the density distribution in the parameter space [18], [21], [28].

Inspired by [29], where each cluster center is characterized by two attributes (i.e., a higher local density than its neighbors and a relatively large distance from any point that has higher densities to the cluster center itself), we search for the authority peaks, which are the vertices that are surrounded by their neighbors with the lower local weighting scores, and are significantly dissimilar to any other vertices that have higher local weighting scores.

To describe the relationships between two vertices in a hypergraph, we propose an effective similarity measure based on the Tanimoto distance [30] (referred to as T-distance), which is able to effectively measure the degree of overlap between two hyperedge sets connected by the two vertices. Given two vertices v_p and v_q, their T-distance is computed as

$$\mathcal{T}(\mathcal{C}_{v_p}, \mathcal{C}_{v_q}) = 1 - \frac{\langle \mathcal{C}_{v_p}, \mathcal{C}_{v_q} \rangle}{\|\mathcal{C}_{v_p}\|^2 + \|\mathcal{C}_{v_q}\|^2 - \langle \mathcal{C}_{v_p}, \mathcal{C}_{v_q} \rangle}, \tag{7}$$

where $\langle \cdot, \cdot \rangle$ and $\| \cdot \|$ indicate the standard inner product and the corresponding induced norm, respectively. $\mathcal{C}_{v_p}$ and $\mathcal{C}_{v_q}$ denote the preference function of v_p and v_q to hyperedges $\mathcal{E}$, respectively.

We define the preference function (see Section 2.1 in [4]) of a vertex v_p to a hyperedge $e \in \mathcal{E}$ as

$$\mathcal{C}^e_{v_p} = \begin{cases} \exp\{-\frac{r_e(v_p)}{\hat{s}(v_p)}\}, & if\ r_e(v_p) \leq E\hat{s}(v_p), \\ 0 \quad , & otherwise, \end{cases} \tag{8}$$

where E is a threshold (the value of E is usually set to 2.5 to include 98 percent inliers of a Gaussian distribution). The preference function is used to compute a rank of a vertex that indicates the degree of preference of the vertex to a hyperedge, where the most preferred hyperedge is ranked in the top (a high value), and the least preferred hyperedge is ranked in the last (a low value). Note that the preference function of each vertex can be effectively expressed by Eq. (8), which takes advantages of the information of residuals of data points with regard to model hypotheses. That is, a vertex prefers to hyperedges that correspond to a data point with a small absolute residual.

Considering a hypergraph, we rewrite Eq. (8) for the preference function of each vertex v_p to hyperedges $\mathcal{E}$ as

$$\mathcal{C}_{v_p} = h(v_p, e) \exp\left\{-\frac{r_e(v_p)}{\hat{s}(v_p)}\right\}, \forall e \in \mathcal{E}. \tag{9}$$

Although [4] also employs the T-distance as a similarity measure, the T-distance defined in this paper has significant differences: 1) We define the preference function of a vertex (i.e., a model hypothesis) towards a hyperedge set (i.e., the inliers), while the authors in [4] define the preference function of a data point towards model hypotheses. Thus, the similarity between model hypotheses can be more effectively discovered by the corresponding preference functions in our case. 2) The T-distance used in the proposed method is calculated without using iterative processes. In contrast, the T-distance in [4] is iteratively calculated until an agglomerative clustering algorithm segments all data points. Therefore, the T-distance used in this paper is much more efficient than that in [4].

Based on the similarity measure and weighting scores, we then compute the Minimum T-Distance (MTD) $\eta^{v_i}_{min}$ of a vertex v_i in G^* as follows:

$$\eta^{v_i}_{min} = \min_{v_j \in \Omega(v_i)} \{\mathcal{T}(\mathcal{C}_{v_i}, \mathcal{C}_{v_j})\}, \tag{10}$$

where

$$\Omega(v_i) = \{v_j | w(v_j) > w(v_i), v_j \in \mathcal{N}(v_i)\}, \tag{11}$$

$$\mathcal{N}(v_i) = \{v_j | \frac{\sum_{e \in \mathcal{E}} h(v_i, e) h(v_j, e)}{\sum_{e \in \mathcal{E}} (h(v_i, e) + h(v_j, e))} > \epsilon, v_j \in \mathcal{V}\}. \tag{12}$$

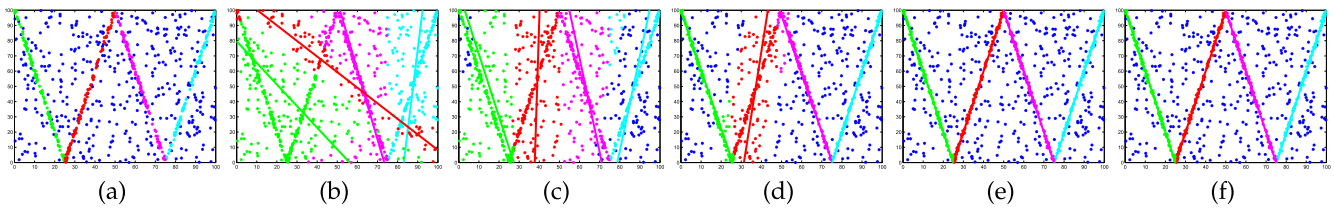


Fig. 2. Some results obtained by NCut based on different hypergraphs for line fitting. (a) The input data. The data points with blue color are outliers, and the other data points with the same color belong to the inliers of the same model instance. (b) to (e) The results obtained by NCut based on the uniform hypergraphs with three, six, nine and twelve degrees, respectively. (f) The results obtained by NCut based on the proposed non-uniform hypergraph.

That is, $\Omega(v_i)$ contains all the vertices that have higher weighting scores than $w(v_i)$ in the neighbors of v_i. $\mathcal{N}(v_i)$ contains the neighbors of v_i in G^*. $\frac{\sum_{e\in\mathcal{E}} h(v_i,e)h(v_j,e)}{\sum_{e\in\mathcal{E}}(h(v_i,e)+h(v_j,e))}$ denotes the ratio of the common hyperedges connected by two vertices (v_i and v_j). We fix $\epsilon = 0.8$, which means that two vertices share at least 80 percent of common hyperedges in a hypergraph. For the vertex v_{max} with the highest weighting score, we set $\eta_{min}^{v_{max}} = \max\{\mathcal{T}(\mathcal{C}_{v_{max}}, \mathcal{C}_{v_i})\}_{v_i\in\mathcal{N}(v_{max})}$.

Note that a vertex with the local maximum value of weighting score, usually has a larger MTD value than the other vertices in G^*. Therefore, we propose to seek modes by searching for the authority peaks, i.e., the vertices with significantly large MTD values.

4 The Complete Method and Analysis

Based on the components described in the previous section, we present the complete fitting method in Section 4.1. We also analyze how MSHF is able to perform well for the model fitting problem in Section 4.2.

4.1 The Complete Method

We summarize the proposed Mode-Seeking on Hypergraphs Fitting method in Algorithm 1. The proposed MSHF seeks modes by directly searching for authority peaks (i.e., representative modes) without requiring iterative processes. As a result, the number and the parameters of model instances can be simultaneously derived from the detected modes.

The computational complexity of MSHF is mainly governed by Step 3 of the algorithm for computing the T-distance between pairs of vertices (here, we do not consider the time for generating model hypotheses since we focus on model selection for model fitting). The other steps in MSHF take much less time than Step 3. For Step 3, the complexity of computing the neighbors of each vertex and the T-distance between all pairs of vertices are $O(M \log M)$ and $O(MM')$, respectively. Here, M is the number of vertices in G^* (M is empirically about $15\% \sim 30\%$ of the total number of vertices in G), and M' ($>> \log M$) is the average number of the neighbors of vertices in G. Therefore, the total complexity of MSHF approximately amounts to $O(MM')$.

4.2 How Does the Proposed Method Find Representative Modes on Hypergraphs

MSHF includes three main parts: hypergraph construction, hypergraph reduction and mode-seeking. MSHF tightly combines these three parts, by which it can effectively search for representative modes on hypergraphs for model fitting.

Algorithm 1. The Mode-Seeking on Hypergraphs Fitting (MSHF) Method for Geometric Model Fitting

Input: Data points X, the K value for IKOSE

1: Construct a hypergraph G and compute the weighting score for each vertex (described in Section 3.1).
2: Use the information theoretic approach for hypergraph reduction and generate a new hypergraph G^* (described in Section 3.2).
3: Compute the minimum T-distance η_{min}^v for each vertex v of G^* by Eq. (10).
4: Sort the vertices in G^* according to their MTD values satisfying $\eta_{min}^{v_1} \geq \eta_{min}^{v_2} \geq \cdots$.
5: Find the vertex v_i whose MTD value ($\eta_{min}^{v_i}$) has the largest drop from $\eta_{min}^{v_i}$ to $\eta_{min}^{v_i+1}$ and reject the vertices whose values of η_{min}^v are smaller than $\eta_{min}^{v_i}$.
6: Derive the inliers/outliers dichotomy from the hypergraph G^* and the remaining vertices (modes).

Output: The modes (model instances) and the hyperedges (inliers) connected by the modes.

For the hypergraph construction, we construct a non-uniform hypergraph to represent the relationships between model hypotheses and data points. As mentioned in [13], we argue that using larger hyperedges is more effective for model fitting. In Fig. 2, we show some results obtained by NCut [31] based on several uniform hypergraphs with different degrees and one non-uniform hypergraph constructed by the proposed MSHF method for line fitting. From the results, we can see that NCut achieves better accuracy based on the hypergraphs with larger degrees than that based on the hypergraphs with smaller degrees. However, how large to set the size of the hyperedges is still an unsolved problem and unaddressed in those most works. Recall that the proposed hypergraph construction can adaptively estimate the degree of each hyperedge. It is worth pointing out that, NCut tends to find a balanced cut, and it cannot effectively deal with unbalanced data.

MSHF searches for representative modes on hypergraphs which shares the similar idea of detecting cluster centers in [29]. Specifically, [29] computes the density of each data point and the minimum T-distance between the data point and any other data point with higher density, to detect cluster centers. Similarly, MSHF computes the weighting score $w(v)$ of each vertex in a hypergraph and the minimum T-distance η_{min}^v between the vertex and its neighbors with higher weighting scores. We show an example of mode-seeking on hypergraphs for line fitting on the "star5" data in Fig. 3. We show the plot of η_{min}^v with respect to the weighting scores of vertices in non-decreasing order in Fig. 3b, and this representation is called the decision graph.

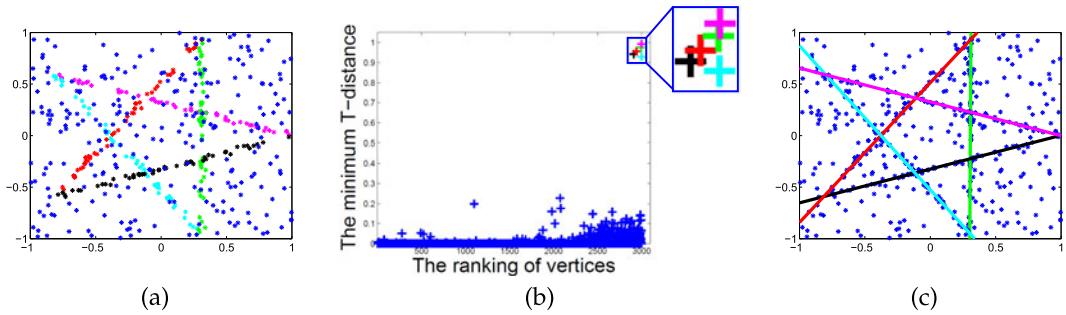


Fig. 3. An example shows that MSHF fits the five lines on the "star5" data. (a) The input data. The data points with blue color are outliers, and the other data points with the same given color belong to the inliers of the same model instance. (b) The obtained decision graph. The vertices are ranked according to their weighting scores in non-decreasing order. The five vertices with the first five highest values of the minimum T-distance (shown in different colors in the zoomed portion of the figure except for blue) are the sought modes. (c) The five lines corresponding to the five sought modes.

MSHF can find the representative modes according to the decision graph, and then estimate the model instances in the data (as shown in Fig. 3c).

In [29], cluster centers can be intuitively determined by the corresponding decision graph. However, it is nontrivial to detect cluster centers in our case, since some isolated data points also show large values of the minimum distance. For the model fitting problem, the proposed mode-seeking algorithm works well for line fitting. This is because the distribution of model hypotheses generated for line fitting is dense in the parameter space, and there do not exist any isolated vertices (corresponding to bad model hypotheses with low weighting scores) showing large MTD values. However, the distribution of model hypotheses generated for higher-order model fitting applications, such as homography based segmentation or two-view based motion segmentation, is often sparse, where a few isolated vertices corresponding to bad model hypotheses may also have anomalously large MTD values as good model hypotheses (with high weighting scores). This problem will cause the proposed mode-seeking algorithm to work ineffectively.

To solve the above problem, we propose to remove some vertices corresponding to bad model hypotheses in Step 2 of Algorithm 1. Therefore, hypergraph reduction (described in Section 3.2) is a critical step to improve the effectiveness of the proposed MSHF algorithm. To show the importance of hypergraph reduction on the performance of the proposed mode-seeking algorithm, we evaluate the algorithm for fitting multiple homographies based on the two constructed hypergraphs, i.e., G (the hypergraph without hypergraph reduction) and G^* (the hypergraph with hypergraph reduction), as shown in Fig. 4. We show the obtained decision graphs in Figs. 4a and 4b, which respectively correspond to G and G^*. We can see that the proposed mode-seeking algorithm based on G has difficulty in distinguishing the three significant model hypotheses according to the MTD values. This is because a vertex corresponding to a bad model hypothesis with a low weighting score also has a large MTD value (as pointed by the arrow in Fig. 4a). Therefore, for the feature points of the middle wall (shown in cyan in Fig. 4c), there are two model instances estimated by the proposed MSHF method based on G. In contrast, the vertex corresponding to a bad model hypothesis is successfully removed by the step of hypergraph reduction, and the proposed mode-seeking algorithm based on G^* can correctly find all the three significant model hypotheses by seeking the largest drop in the MTD values. As shown in Figs. 4c and 4d, the segmentation results further show the importance of hypergraph reduction for the proposed MSHF method—leading to more accurate results.

5 Experiments

In this section, we compare the proposed MSHF with several state-of-the-art model fitting methods, including KF [1], RCG [11], AKSWH [6], and T-linkage [4], on both synthetic data and real images. We choose these representative methods because KF is a data clustering based method, RCG is a hypergraph based method, and AKSWH is a parameter space based method. These fitting methods are closely related to the proposed method (recall that MSHF seeks modes on hypergraphs and it fits multi-structure data in the parameter space). We also choose T-linkage as a competing method due to its good performance. Moreover, we compare with our original method (MSH) in [16] to show the improvements of the proposed MSHF. For MSHF, we test two versions: MSHF1, which does not use the neighboring constraint in Eq. (10) and MSHF2, which uses the neighboring constraint in Eq. (10).

To be fair, we first generate a set of model hypotheses by using the proximity sampling [5], [33] for all the competing

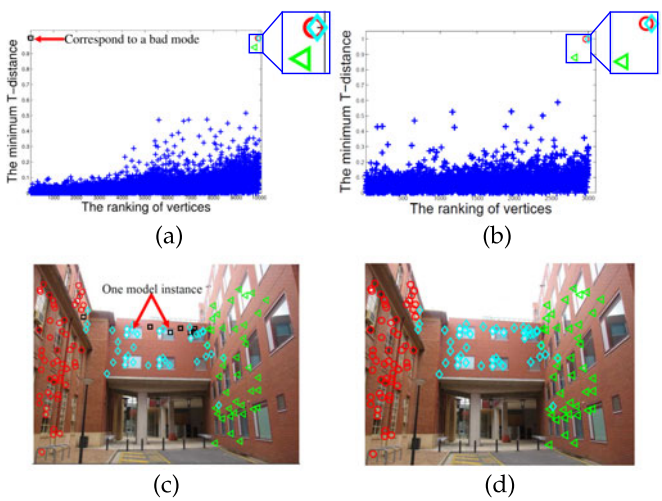
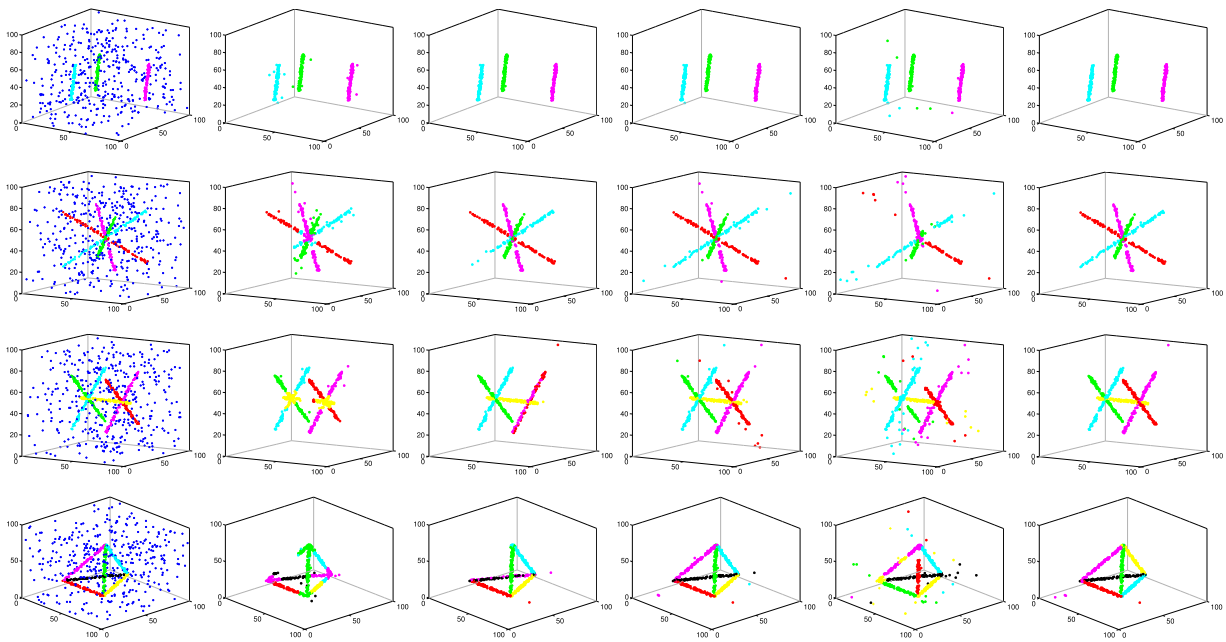


Fig. 4. Homography based segmentation on "Neem" [32]. (a) and (b) The decision graphs obtained by the proposed mode-seeking algorithm based on G and G^*, respectively. (c) and (d) The segmentation results obtained by the proposed MSHF method based on G and G^*, respectively.



(a) Data (b) KF (c) RCG (d) AKSWH (e) T-linkage (f) MSHF2

Fig. 5. Examples for line fitting in the 3D space. 1st to 4th rows respectively fit three, four, five and six lines. The corresponding outlier percentages are respectively 86, 88, 89 and 90 percent. The inlier noise scale is set to 1.0 and each line includes 100 inliers. Each data includes 400 outliers. We do not show the results of MSH/MSHF1, which are similar to those of MSHF2, due to the space limit.

algorithms in each experiment. Then all the competing methods perform model fitting based on the same set of model hypotheses. We generate the same number of model hypotheses as [6], i.e., there are 5,000 model hypotheses generated for line fitting (Sections 5.1.1 and 5.2.1) and circle fitting (Sections 5.1.2 and 5.2.2), 10,000 model hypotheses generated for homography based segmentation (Section 5.2.3), and 20,000 model hypotheses generated for two-view based motion segmentation (Section 5.2.4).

We have optimized the parameters of all the competing fitting methods[1] on each dataset for the best performance. For our methods (i.e., MSH and MSHF1/MSHF2), we only slightly adjust the value of K for IKOSE. In most cases we fix $K = 10\% * n$, where n is the number of input data points. In some challenging cases (i.e., where the data do not include at least 10 percent inliers), we adjust the value of K to obtain good inlier noise scales. All experiments are run on MS Windows 7 with Intel Core i7-3630 CPU 2.4 GHz and 16 GB RAM. The fitting error is computed as follows [4], [34]:

$$error = \frac{\#\ mislabeled\ data\ points}{\#\ data\ points} \times 100\%. \qquad (13)$$

5.1 Synthetic Data

5.1.1 Line Fitting

We evaluate the performance of the seven fitting methods on line fitting using four challenging synthetic data in the 3D space (as shown in Fig. 5). We repeat the experiment 50 times and report the standard variances, the average and the best results of the fitting errors (in percentage) and the average CPU time (in seconds) obtained by the competing methods in Table 1 (we exclude the time used for sampling and generating potential hypotheses for all the fitting methods). We also show the corresponding fitting results obtained by all the competing methods from Figs. 5b, 5c, 5d, 5e, and 5f.

From Fig. 5 and Table 1, we can see that: (1) For the "three lines" data, the three lines are completely separable in the 3D space, and the seven fitting methods succeed in fitting all the three lines. However, MSH and MSHF1/MSHF2 achieve the best performance on the fitting accuracy among

TABLE 1
Quantitative Comparison Results of Line Fitting on Four Synthetic Data

Data		M1	M2	M3	M4	M5	M6	M7
3 lines	Std.	0.01	0.01	0.01	0.01	0.01	0.01	0.01
	Avg.	1.76	0.33	0.34	1.87	0.16	**0.14**	**0.14**
	Min.	1.71	0.29	0.29	1.71	**0.14**	**0.14**	**0.14**
	Time	13.74	**0.41**	1.17	155.92	0.99	1.77	1.04
4 lines	Std.	3.16	2.11	1.02	4.73	0.59	**0.26**	**0.26**
	Avg.	18.25	4.13	3.00	31.40	1.29	**1.23**	**1.23**
	Min.	13.25	1.63	2.88	23.75	0.88	**0.75**	**0.75**
	Time	17.09	**0.53**	1.18	210.39	1.68	2.92	2.57
5 lines	Std.	3.07	7.42	5.34	4.53	0.21	**0.17**	**0.17**
	Avg.	15.27	18.00	3.78	17.29	1.76	**1.72**	**1.72**
	Min.	11.42	2.44	2.67	11.89	1.44	**1.22**	**1.22**
	Time	20.36	**0.68**	1.23	274.08	1.88	2.81	2.61
6 lines	Std.	3.32	5.63	2.87	3.15	**0.47**	**0.47**	**0.47**
	Avg.	33.71	15.69	4.57	16.26	3.34	**3.32**	**3.32**
	Min.	27.10	5.00	2.70	11.70	**2.30**	**2.30**	**2.30**
	Time	25.53	**0.69**	1.37	326.48	2.07	3.06	2.77

(M1-KF; M2-RCG; M3-AKSWH; M4-T-linkage; M5-MSH; M6-MSHF1; M7-MSHF2. M1-M7 in the following tables denote the same meaning.) The best results are boldfaced.

1. For KF and T-linkage, we use the code published on the web: http://cs.adelaide.edu.au/~tjchin/doku.php and http://www.diegm.uniud.it/fusiello/demo/jlk/, respectively. For RCG and AKSWH, we use the code provided by the authors.

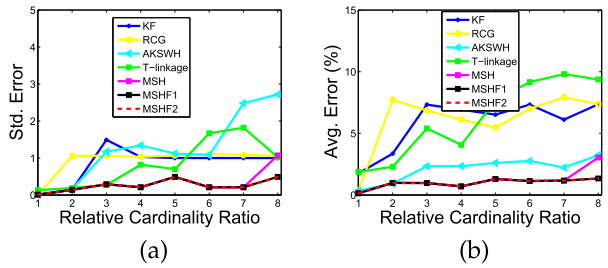


Fig. 6. The fitting errors obtained by the seven competing methods for data with different cardinality ratios of inliers: (a) and (b) show the performance comparison of the standard variances and the average fitting errors for data with different inlier cardinality ratios, respectively.

the seven fitting methods due to their high robustness to outliers. (2) For the "four lines" data, the four lines intersect at one point. The seven fitting methods succeed in estimating the number of the lines in the data, but the data clustering based methods (i.e., KF and T-linkage) cannot effectively segment the data points near the intersection. In contrast, RCG, AKSWH, MSH and MSHF1/MSHF2 correctly fit all the four lines with lower fitting errors, while MSHF1/MSHF2 achieve the lowest fitting error. (3) For the "five lines" data, there exist two intersections. The data points near the intersections are not correctly segmented by both KF and T-linkage, which causes these two methods to obtain high fitting errors. RCG correctly fits four lines but it wrongly fits one. This is because the dense subgraph representing a potential structure in the data is not effectively detected by RCG. In contrast, the parameter space based methods (i.e., AKSWH, MSH and MSHF1/MSHF2) do not directly deal with data points. AKSWH, MSH and MSHF1/MSHF2 correctly fit all the five lines with low fitting errors. (4) For the "six lines" data, RCG only correctly fits five of the six lines. T-linkage wrongly estimates the number of lines in the data. KF achieves the worst performance among the five fitting methods. In contrast, AKSWH, MSH and MSHF1/MSHF2 correctly fit the six lines. These results on the challenging dataset further shows the superiority of the parameter space based methods over the other competing fitting methods.

For the performance of computational time, RCG achieves the fastest speed among the seven fitting methods, but it cannot achieve good fitting accuracy. AKSWH, MSH and MSHF1/MSHF2 achieve similar computational speed. Here, the speed of these four fitting methods depends on the number of the selected significant model hypotheses/vertices. MSHF1/MSHF2 retain the maximum number of significant vertices to avoid missing model instances with less number of inliers. Thus, MSHF1/MSHF2 are slower than AKSWH and MSH. MSHF2 is faster than MSHF1 due to the use of the neighboring constraint. MSHF1 is faster than KF and T-linkage for all four data (about $5.85 \sim 8.34$ times faster than KF and about $72.05 \sim 106.69$ times faster than T-linkage).

We also evaluate the performance obtained by the seven fitting methods for the data with different cardinality ratios between the inliers of each line, to show the ability of the model fitting methods to deal with unbalanced data. We use the "three lines" data from Fig. 5 for evaluation since all competing methods can successfully estimate the three lines when the cardinality ratio is low. We set the inlier numbers of the three lines to be the same at the beginning, and we gradually increase the inlier numbers of two lines while reducing the inlier number of the third line to make the cardinality ratios between the inliers of lines increase from 1.0 to 8.0. We repeat each experiment 20 times and show the standard variances and the average fitting errors in Fig. 6.

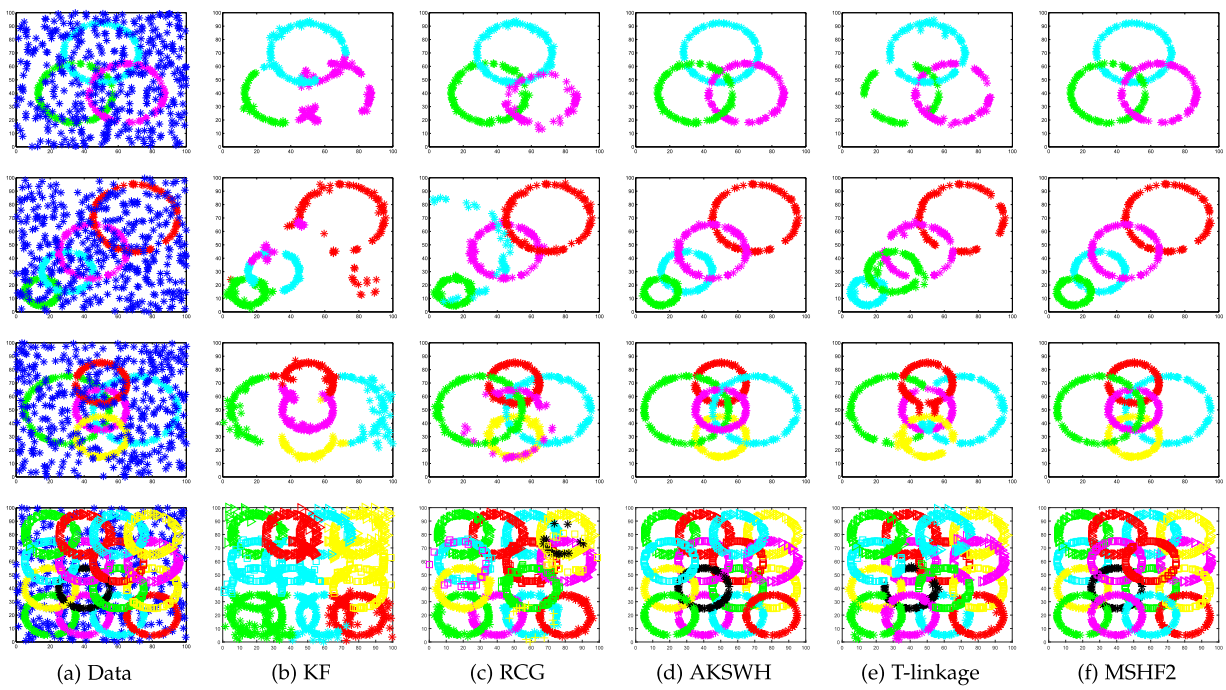


Fig. 7. Examples for circle fitting in the 2D space. 1st to 4th rows respectively fit three, four, five and sixteen circles. The inlier noise scale is set to 0.5 and each circle has 100 inliers. Each data includes 400 outliers. We do not show the results of MSH/MSHF1, which are similar to those of MSHF2, due to the space limit.

TABLE 2
Quantitative Comparison Results of Circle Fitting on Four Synthetic Data

Data		M1	M2	M3	M4	M5	M6	M7
3 circles	Std.	3.79	3.93	2.78	2.04	6.33	**0.63**	**0.63**
	Avg.	27.85	24.01	4.63	15.50	3.07	**1.64**	**1.64**
	Min.	22.00	17.28	2.00	13.00	**0.57**	**0.57**	**0.57**
	Time	51.66	**1.50**	2.91	164.27	2.49	3.70	1.76
4 circles	Std.	4.38	5.65	5.62	4.53	0.68	**0.60**	**0.60**
	Avg.	32.79	23.98	9.12	16.62	2.46	**2.14**	**2.14**
	Min.	26.25	18.12	3.75	10.37	1.50	**1.37**	**1.37**
	Time	64.95	2.21	2.00	235.34	2.91	3.03	**1.48**
5 circles	Std.	3.74	4.87	1.87	5.54	6.34	**1.14**	**1.14**
	Avg.	36.17	24.78	6.50	18.86	4.72	**2.95**	**2.95**
	Min.	31.11	18.22	3.33	13.88	1.55	**1.44**	**1.44**
	Time	70.94	2.80	3.40	287.94	4.54	4.72	**1.83**
16 circles	Std.	3.00	6.30	7.58	1.62	3.83	**1.56**	**1.56**
	Avg.	57.92	32.72	14.46	24.75	4.46	**3.59**	**3.59**
	Min.	54.35	23.70	**1.55**	23.15	2.25	1.90	1.90
	Time	254.43	7.38	3.97	1589.13	7.75	9.46	**1.47**

From Fig. 6, we can see that MSHF1/MSHF2 achieve the same results and both achieve low standard variances and average fitting errors for the "three lines" data with different inlier cardinality ratios. In contrast, KF, RCG and T-linkage achieve low average fitting errors when the cardinality ratio of inliers is smaller than 2.0, but they begin to break down when the inlier cardinality ratios are larger than 2.0, 3.0 and 3.0, respectively. AKSWH and MSH obtain large fitting errors when the inlier cardinality ratios are larger than 2.0 and 7.0, respectively. As a result, the parameter space based methods (i.e., AKSWH, MSH and MSHF1/MSHF2) show better performance than the other competing fitting methods for the unbalanced data.

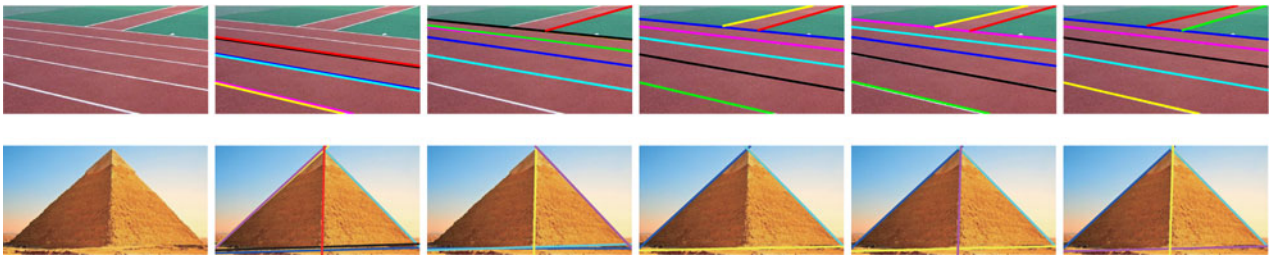
TABLE 3
The CPU Time Used by the Seven Fitting Methods (in Seconds)

Data	M1	M2	M3	M4	M5	M6	M7
Tracks	133.71	9.76	8.81	23,256.00	7.21	8.26	**7.12**
Pyramid	79.28	8.12	**7.23**	13,600.00	8.05	8.65	7.62
Coins	65.02	6.29	5.84	8,746.50	5.02	5.16	**4.48**
Bowls	9.01	4.33	3.68	862.34	3.81	3.98	**3.64**

5.1.2 Circle Fitting

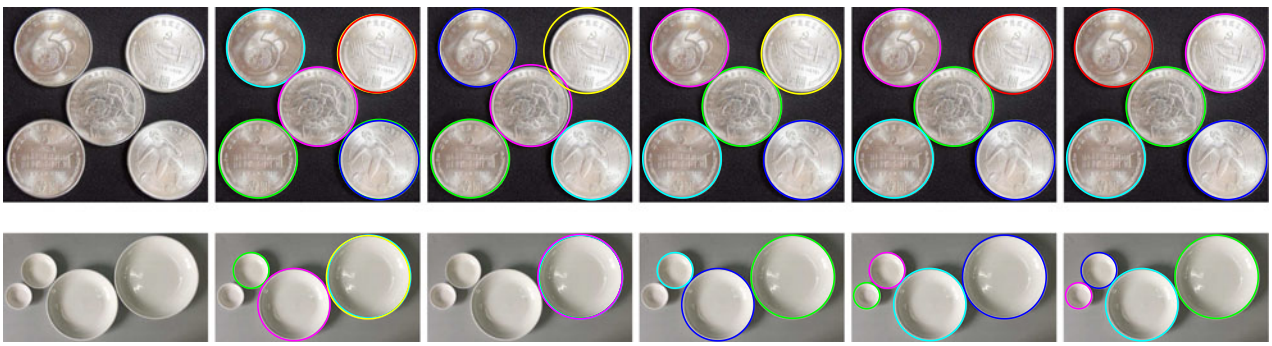
We further evaluate the performance of the seven fitting methods on circle fitting using four challenging synthetic data in the 2D space (see Fig. 7). We repeat the experiment 50 times and report the standard variances, the average and the best results of the fitting errors (in percentage) and the average CPU time (in seconds) obtained by the seven competing methods, in Table 2 (we exclude the time used for sampling and generating potential hypotheses for all the fitting methods). We also show the corresponding fitting results obtained by all the competing methods from Figs. 7b, 7c, 7d, 7e, and 7f.

From Fig. 7 and Table 2, we can see that: (1) For the "three circles" data, the three circles with the same diameter intersect each other. All of the seven competing methods can correctly estimate the number of circles in the data. However, MSH and MSHF1/MSHF2 achieve the top-three lowest average and minimal fitting errors among all competing methods. And MSHF1/MSHF2 are the most stable methods (achieving the lowest standard deviation of fitting errors). This is because that MSHF1/



(a) Data (b) KF (c) RCG (d) AKSWH (e) T-linkage (f) MSHF2

Fig. 8. Examples for line fitting. First ("Tracks") and second ("Pyramid") rows respectively fit seven and four lines. We do not show the results of MSH/MSHF1, which are similar to those of MSHF2, due to the space limit.



(a) Data (b) KF (c) RCG (d) AKSWH (e) T-linkage (f) MSHF2

Fig. 9. Examples for circle fitting. First ("Coins") and second ("Bowls") rows respectively fit five and four circles. We do not show the results of MSH/MSHF1, which are similar to those of MSHF2, due to the space limit.

TABLE 4
Quantitative Comparison Results of Homography Based Segmentation on 19 Image Pairs

Data (#)		M1	M2	M3	M4	M5	M6	M7
	Std.	5.29	5.56	0.71	16.54	**0.01**	**0.01**	**0.01**
Bonython (1)	Avg.	31.86	5.66	4.54	28.28	**0.00**	**0.00**	**0.00**
	Time	1.01	**0.49**	1.44	11.65	3.40	3.63	0.96
	Std.	0.29	**0.01**	1.91	14.87	**0.01**	**0.01**	**0.01**
Physics (1)	Avg.	10.47	**0.00**	22.54	39.43	**0.00**	**0.00**	**0.00**
	Time	3.02	**0.24**	3.45	12.68	1.43	2.33	1.75
	Std.	2.78	**0.01**	1.64	26.66	**0.01**	**0.01**	**0.01**
Unionhouse (1)	Avg.	27.16	**0.30**	2.74	24.81	**0.30**	**0.30**	**0.30**
	Time	1.42	1.08	1.63	38.05	2.44	2.11	**1.02**
	Std.	0.72	0.37	**0.15**	0.45	**0.15**	**0.15**	**0.15**
Elderhalla (2)	Avg.	12.15	10.37	0.98	1.17	**0.93**	**0.93**	**0.93**
	Time	3.34	**1.66**	2.79	15.28	3.38	4.57	2.16
	Std.	**0.00**	2.42	1.37	0.58	0.96	1.10	1.10
Elderhallb (3)	Avg.	34.51	10.12	13.06	12.63	3.37	**2.94**	**2.94**
	Time	3.29	**1.14**	2.34	30.47	2.63	2.87	2.18
	Std.	7.07	2.91	0.43	0.32	2.98	**0.31**	**0.31**
Hartley (2)	Avg.	15.31	4.88	4.06	2.50	2.81	**1.90**	**1.90**
	Time	2.92	**1.21**	2.12	62.16	2.01	2.14	1.63
	Std.	6.24	**0.01**	5.11	3.38	5.89	0.82	0.82
Library (2)	Avg.	13.19	9.77	5.79	4.65	2.79	**2.37**	**2.37**
	Time	3.34	**1.71**	2.13	16.04	2.31	3.65	1.80
	Std.	6.12	5.44	5.78	0.37	**0.13**	**0.13**	**0.13**
Sene (2)	Avg.	12.08	10.00	2.00	0.44	**0.24**	**0.24**	**0.24**
	Time	5.24	**0.91**	2.73	22.78	2.59	2.11	1.80
	Std.	9.57	**0.01**	0.55	0.51	0.27	0.27	0.27
Nese (2)	Avg.	28.03	36.61	3.54	1.88	**0.20**	**0.20**	**0.20**
	Time	5.40	**0.67**	3.13	24.15	1.91	2.61	2.32
	Std.	2.67	**0.01**	3.53	2.58	0.81	0.86	0.86
Ladysymon (2)	Avg.	16.46	22.36	5.74	5.06	2.87	**2.62**	**2.62**
	Time	3.06	**0.83**	2.87	20.86	2.76	3.44	2.39
	Std.	**0.01**	8.34	0.14	0.25	0.41	0.33	0.33
Oldclassicswing	Avg.	18.73	10.34	1.29	1.27	1.13	**1.08**	**1.08**
(2)	Time	2.80	**1.66**	2.25	74.89	2.44	3.56	1.81
	Std.	10.75	5.77	7.04	4.96	5.40	**0.48**	**0.48**
Neem (3)	Avg.	10.25	11.17	5.56	3.82	2.90	**1.78**	**1.78**
	Time	6.32	**0.83**	2.49	21.40	2.81	2.78	2.13
	Std.	3.18	1.84	12.45	4.73	1.77	**1.43**	**1.43**
Johnsona (4)	Avg.	25.74	23.06	8.55	4.03	3.73	**3.02**	**3.02**
	Time	16.53	**1.36**	2.93	57.11	3.63	2.96	2.24
	Std.	4.85	**1.81**	6.45	10.51	5.99	4.96	4.96
Johnsonb (7)	Avg.	48.32	41.45	26.49	18.39	16.75	**16.61**	**16.61**
	Time	14.52	**4.18**	4.73	261.62	5.67	6.48	5.00
	Std.	**2.49**	5.22	4.26	4.54	16.53	3.26	3.26
Napiera (2)	Avg.	28.24	30.96	30.86	**23.37**	32.51	27.78	27.78
	Time	3.14	**0.89**	1.94	29.88	2.76	3.44	2.18
	Std.	5.52	**0.01**	0.38	5.14	4.19	4.12	4.12
Napierb (3)	Avg.	30.42	33.59	36.33	19.92	14.21	**13.12**	**13.12**
	Time	2.45	**0.65**	3.33	21.93	2.18	3.31	1.87
	Std.	4.09	3.74	4.53	6.65	15.84	9.50	9.50
Barrsmith (2)	Avg.	22.28	54.64	**20.08**	29.33	37.80	24.48	24.48
	Time	6.06	**0.62**	2.20	18.91	1.92	2.51	1.42
	Std.	8.54	8.60	**0.14**	4.98	5.39	0.38	0.38
Unihouse (5)	Avg.	38.32	41.70	14.91	14.04	10.99	**9.29**	**9.29**
	Time	31.27	9.40	8.67	2908.61	**6.25**	13.50	10.50
	Std.	4.85	**1.81**	5.10	0.13	8.16	8.69	8.69
Bonhall (6)	Avg.	48.32	41.45	38.77	**29.06**	31.89	31.65	31.65
	Time	14.52	4.18	7.58	835.38	**4.15**	9.22	7.87
	Mean	24.83	20.97	13.04	13.89	8.70	**7.38**	**7.38**
Total	Std.	11.94	16.59	12.43	12.32	12.30	**10.30**	**10.30**
	Median	25.74	11.17	5.79	12.63	2.87	**2.37**	**2.37**

'#' denotes the actual number of model instances in data.

MSHF2 can keep the representative modes corresponding to model instances in most cases, while MSH may remove some representative modes in some cases, which will increase the average fitting errors. AKSWH can also achieve a low average fitting error with the third lowest standard deviation. In contrast, KF, RCG and T-linkage obtain high average fitting errors. KF removes some inliers during the procedure of outlier removal. RCG cannot effectively find the representative sub-graphs corresponding to the model instances. T-linkage cannot segment the data points of intersection with high accuracy. (2) For the "four circles" data, the four circles with different diameters intersect each other. MSH and MSHF1/MSHF2 achieve the lowest average and minimal fitting errors again. Among the other four competing methods, AKSWH and T-linkage successfully estimate all four circles, while KF and RCG miss one of the circles. (3) For the "five circles" data, the five circles with different diameters intersect together. All of AKSWH, T-linkage, MSH and MSHF1/MSHF2 successfully estimate the five circles, but MSHF1/MSHF2 achieve the best results of the standard deviation, the average and minimal fitting errors. In contrast, KF and RCG obtain high fitting errors. (4) For the "sixteen circles" data, the sixteen circles with the same diameter intersect together. This data include a large number of model instances. MSHF1/MSHF2 can effectively estimate the number of model instances. MSH wrongly estimates the number during the repeating experiments in 1 out of 50 times. However, MSH still achieves a low average fitting error. In contrast, KF, RCG, AKSWH and T-linkage cannot achieve low average fitting errors, especially for KF, which fails to estimate the number of model instances in most cases.

For the performance of computational time, MSHF2 achieves the fastest speed among the seven fitting methods for three data (i.e., the "four circles", "five circles" and "sixteen circles" data). AKSWH and MSH/MSHF1 achieve the similar computational speed for the "three circles", "four circles" and "five circles" data. KF and T-linkage are relatively slow. Especially for T-linkage which takes 1,589.13 seconds for the "sixteen circles" data, it is more than 1,000 times slower than the proposed MSHF2. Among MSH and MSHF1/MSHF2, MSHF2 is significantly faster than MSH and MSHF1 (MSHF2 is about $1.41 \sim 5.27$ and $2.04 \sim 6.44$ times faster than MSH and MSHF1, respectively). This is because MSHF2 uses the neighboring constraint to reduce the computational cost. RCG achieves fast speed for the "three circles", "four circles" and "five circles" data, but obtains slow speed for the "sixteen circles" data due to the large number of data points.

5.2 Real Images

5.2.1 Line Fitting

We evaluate the performance of all the competing fitting methods using real images for line fitting (see Fig. 8). For the "Tracks" image, which includes seven lines, there are 6,704 edge points detected by the Canny operator [35]. As shown in Fig. 8 and Table 3, AKSWH, T-linkage, MSH and MSHF1/MSHF2 correctly fit all the seven lines. However, MSH, MSHF1/2 are faster than AKSWH and T-linkage. T-

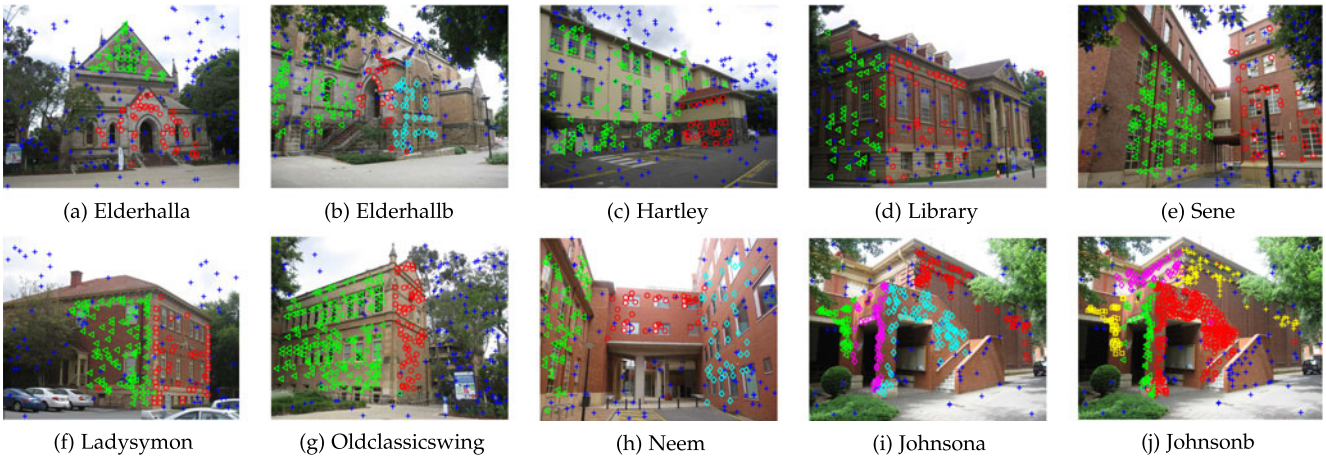


Fig. 10. Some fitting results obtained by MSHF2 for homography based segmentation on the AdelaideRMF dataset.

linkage is very slow due to the large number of input data points. RCG correctly estimates the number of lines, but some estimated lines are overlapped and two lines are missed. This is because the potential structures in the data are not correctly estimated by RCG during the detection of dense subgraphs. KF only correctly fits three out of the seven lines because many inliers belonging to the other four lines are wrongly removed.

For the "Pyramid" image shown in Fig. 8, which includes four lines with a large number of outliers, there are 5,576 edge points detected by the Canny operator. Only T-linkage, MSH and MSHF1/MSHF2 succeed in fitting all the four lines. In contrast, although KF also fits the four lines, it wrongly estimates the number of lines (which is five lines instead of four lines). Both RCG and AKSWH only correctly fit three out of the four lines, although RCG successfully estimates the number of lines in the data. AKSWH can detect four lines after the clustering step, but two lines are wrongly fused during the fusion step. For the CPU time (see Table 3), RCG, AKSWH, MSH and MSHF1/MSHF2 achieve similar time, but KF and T-linkage are much slower than the other five methods.

5.2.2 Circle Fitting

Next we evaluate the performance of the seven fitting methods using real images for circle fitting (see Fig. 9). For the "Coins" image, which includes five circles with similar numbers of inliers, there are totally 4,595 edge points detected by the Canny operator. As shown in Fig. 9 and Table 3, AKSWH, T-linkage, MSH and MSHF1/MSHF2 correctly fit all the five circles, and MSH and MSHF1/MSHF2 are the top-three fastest among all the seven fitting methods. In contrast, two model hypotheses estimated by KF correspond to one circle, and RCG correctly fits only four out of the five circles.

For the "Bowls" image, which includes four circles with significantly unbalanced numbers of inliers, 1,565 edge points are detected by the Canny operator. We can see that two circles estimated by both KF and RCG overlap in the image. AKSWH correctly fits three circles but it misses one circle, because most of the model hypotheses generated for the circle with a small number of inliers are removed during the process that AKSWH selects significant model hypotheses. In contrast, T-linkage, MSH and MSHF1/MSHF2 succeed in fitting all the four circles in this challenging case. However, MSH and MSHF1/MSHF2 are much faster than T-linkage (see Table 3).

5.2.3 Homography Based Segmentation

We also evaluate the performance of the seven fitting methods using the 19 real image pairs from the AdelaideRMF dataset [32][2] (the dataset contains 19 image pairs designed for homography fitting—which we use here—and 19 image pairs for motion segmentation—which we use in Section 5.2.4 devoted to that topic) for homography based segmentation. We repeat each experiment 50 times, and show the standard variances, the average fitting errors (in percentage) and the average CPU time (in seconds) in Table 4 (we exclude the time used for sampling and generating potential hypotheses, which is the same for all the fitting methods). Some fitting results obtained by MSHF2 are also shown in Fig. 10.

From Fig. 10 and Table 4, we can see that MSHF1/MSHF2 obtain good results, achieving the lowest average fitting errors in 16 out of 19 image pairs. Although MSHF1 is slightly slower than MSH, it significantly improves the fitting accuracy over MSH in 12 out of 19 image pairs. The reason behind this is that MSHF1 removes less vertices corresponding to model hypotheses than MSH, and thus MSHF1 takes more time to seek modes in a hypergraph. However, MSHF1 retains more good vertices corresponding to significant model hypotheses, which improves its fitting accuracy. MSHF2 achieves the same fitting errors as MSHF1, but it is faster than MSHF1 in all the 19 image pairs. In contrast, AKSWH only succeeds in fitting 10 out of 19 image pairs with low fitting errors. Although T-linkage can also achieve low fitting errors in most of image pairs, it is much slower than the other six competing methods. Both KF and RCG achieve bad results in most cases. We note that KF clusters many outliers together with inliers, and RCG is very sensitive to its parameters when there exist many bad model hypotheses in the generated model hypotheses. For the overall fitting errors, MSH and MSHF1/MSHF2 achieve the top-three best performance on the mean and median

2. http://cs.adelaide.edu.au/~hwong/doku.php?id=data

TABLE 5
Quantitative Comparison Results of Two-View Based Motion Segmentation on 19 Image Pairs

Data (#)		M1	M2	M3	M4	M5	M6	M7
	Std.	4.30	**0.35**	0.42	23.06	0.55	0.55	0.55
Biscuit (1)	Avg.	**0.61**	14.39	1.41	25.84	1.30	1.30	1.30
	Time	6.08	**2.13**	6.19	79.34	5.09	6.01	5.27
	Std.	0.55	0.73	1.24	18.58	**0.42**	**0.42**	**0.42**
Book (1)	Avg.	5.88	7.54	3.47	24.54	**0.64**	**0.64**	**0.64**
	Time	6.02	**0.75**	5.49	24.97	4.08	6.56	4.81
	Std.	**0.22**	0.88	0.86	22.34	0.66	0.66	0.66
Cube (1)	Avg.	8.70	22.48	2.21	23.07	**2.08**	**2.08**	**2.08**
	Time	7.02	**1.80**	5.93	79.48	6.94	6.01	5.11
	Std.	**0.11**	2.11	0.97	32.65	1.07	0.74	0.74
Game (1)	Avg.	18.81	19.31	2.61	38.15	**2.44**	**2.44**	**2.44**
	Time	7.11	**1.18**	5.87	42.70	6.99	5.50	4.95
	Std.	4.30	1.04	3.10	**0.85**	1.26	0.98	0.98
Cubechips (2)	Avg.	8.42	13.43	4.72	5.63	3.80	**3.55**	**3.55**
	Time	7.94	**1.69**	5.10	64.87	6.45	6.71	5.18
	Std.	10.80	1.38	3.78	0.80	1.27	**0.79**	**0.79**
Cubetoy (2)	Avg.	12.53	13.35	7.23	5.62	3.21	**2.16**	**2.16**
	Time	6.08	**1.34**	4.97	51.65	5.74	6.30	4.89
	Std.	3.92	3.27	6.06	1.32	0.95	**0.78**	**0.78**
Breadcube (2)	Avg.	14.83	12.60	5.45	4.96	2.69	**2.31**	**2.31**
	Time	7.07	**1.53**	6.10	46.17	6.01	6.05	4.82
	Std.	**0.19**	9.53	10.74	1.85	2.71	0.74	0.74
Gamebiscuit (2)	Avg.	13.78	9.94	7.01	7.32	3.72	**1.95**	**1.95**
	Time	7.66	**2.36**	6.44	91.49	6.93	7.55	5.81
	Std.	3.41	5.81	6.72	**1.50**	8.07	7.76	7.76
Breadtoy (2)	Avg.	8.36	20.48	15.03	7.33	5.90	**4.86**	**4.86**
	Time	22.51	**2.12**	15.18	68.62	9.48	13.02	5.87
	Std.	8.22	8.15	8.58	**1.43**	1.96	1.96	1.96
Breadtoycar (3)	Avg.	16.87	26.51	9.04	**4.42**	6.63	5.42	5.42
	Time	5.70	**0.98**	4.56	24.15	5.48	6.18	5.06
	Std.	10.71	5.31	1.40	**1.16**	1.82	1.82	1.82
Biscuitbook (2)	Avg.	12.90	3.82	2.89	2.55	2.60	**2.40**	**2.40**
	Time	7.84	**2.03**	6.59	129.47	8.52	9.60	6.57
	Std.	4.00	1.98	3.17	1.60	0.92	**0.90**	**0.90**
Biscuitbookbox (3)	Avg.	16.06	16.87	8.54	1.93	**1.54**	**1.54**	**1.54**
	Time	8.50	**1.71**	5.11	53.44	6.11	6.35	5.44
	Std.	7.26	6.64	3.41	7.03	4.39	**1.75**	**1.75**
Breadcubechips (3)	Avg.	33.43	26.39	7.39	**1.06**	1.74	1.74	1.74
	Time	16.53	**1.36**	2.93	57.11	8.35	13.28	4.35
	Std.	4.99	12.18	0.95	7.72	7.18	6.62	6.62
Cubebreadtoychips	Avg.	31.07	37.95	14.95	**3.11**	4.28	4.25	4.25
(4)	Time	25.68	**1.83**	5.99	91.05	9.16	13.09	4.70
	Std.	7.90	**1.29**	5.26	9.45	8.96	6.14	6.14
Breadcartoychips	Avg.	26.96	49.36	42.86	**16.96**	33.92	25.06	25.06
(4)	Time	6.91	**1.96**	4.92	40.76	6.23	5.14	4.02
	Std.	9.09	1.59	1.67	**0.44**	5.10	7.56	7.56
Carchipscube (3)	Avg.	**10.96**	38.96	51.75	17.51	20.72	25.51	25.51
	Time	6.52	**1.58**	4.60	18.52	4.84	4.32	3.81
	Std.	10.23	**0.48**	6.06	1.20	9.74	6.57	6.57
Toycubecar (3)	Avg.	27.05	38.75	34.55	16.20	20.35	**14.00**	**14.00**
	Time	7.77	**0.68**	5.72	25.35	3.94	6.06	4.73
	Std.	9.90	**1.50**	5.96	1.91	4.13	7.77	7.77
Boardgame (3)	Avg.	30.21	45.16	48.13	28.60	21.68	**21.57**	**21.57**
	Time	2.36	**1.39**	7.57	58.07	4.34	5.11	4.63
	Std.	3.41	**0.92**	10.74	1.85	5.32	2.08	2.08
Dinobooks (3)	Avg.	30.86	54.27	24.72	19.52	**16.08**	18.05	18.05
	Time	12.93	**2.33**	6.66	118.96	5.54	5.61	4.89
	Mean	17.27	24.81	15.47	13.38	8.17	**7.41**	**7.41**
Total	Std.	9.81	15.00	16.55	10.91	9.48	**8.63**	**8.63**
	Median	14.83	20.48	7.39	7.33	3.72	**2.44**	**2.44**

'#' denotes the actual number of model instances in data.

fitting errors among all the seven competing fitting methods. MSHF1/MSHF2 also achieve the lowest standard variances of fitting errors. For the performance of computational time, RCG achieves the lowest values in 17 out of 19 image pairs, but it cannot obtain low fitting errors. In short, MSH and MSHF1/MSHF2 can achieve low fitting errors within reasonable time for most image pairs.

5.2.4 Two-View Based Motion Segmentation

For the two-view based motion segmentation problem, we use the other 19 image pairs of the AdelaideRMF dataset to evaluate the performance of the competing fitting methods. The results are shown in Table 5 and Fig. 11.

From Table 5 and Fig. 11, we can see that both KF and RCG achieve high fitting errors and both fail in most cases. This is because when a large number of model hypotheses are generated for two-view based motion segmentation to cover all the model instances in the data, a large proportion of bad model hypotheses may lead to inaccurate similarity measure between data points, which results in the wrong estimation of the number of model instances by KF and RCG. AKSWH achieves better results than both KF and RCG on the average fitting errors. T-linkage, MSH and MSHF1/MSHF2 achieve low fitting errors, while MSHF1/MSHF2 obtain relatively better results than T-linkage and MSH (as shown in Table 5). MSHF1/MSHF2 achieve the lowest average fitting errors in 12 out of 19 image pairs and the second lowest ones in 6 out of the remaining 7 image pairs. MSHF1/MSHF2 also achieve the best performance on the standard variances, the mean and median fitting errors for the overall results. It is worth pointing out that MSHF1/MSHF2 achieve lower average fitting errors than MSH in 11 out of 19 image pairs (and they achieve the same average fitting errors in 6 out of the remaining 8 image pairs). This benefits from the step of hypergraph reduction.

For the computational time, MSH and MSHF1/MSHF2 do not achieve better results than RCG. However, MSH and MSHF1/MSHF2 are significantly faster than KF in most data. Compared with T-linkage, which achieves good performance on fitting accuracy, MSH and MSHF1/MSHF2 show significant superiority regarding to the computational time for all the 19 image pairs. Note that MSHF2 is faster than MSHF1 for all the 19 image pairs due to the use of neighboring constraint, which can effectively reduce the computational cost.

We also evaluate the performance of the five competing methods (i.e., AKSWH, T-linkage, MSH and MSHF1/MSHF2) on the estimated number of model instances and the average fitting errors with different numbers of generated model hypotheses for homography based segmentation and two-view based motion segmentation, as shown in Fig. 12 (we do not show the results of KF and RCG since they cannot achieve good results). We can see that, the number of generated model hypotheses has significant influence on the results of all the five fitting methods. When the number of model hypotheses is large, all the five fitting methods achieve relatively lower fitting errors. For the estimated number of model instances, T-linkage achieves the best results for both homography based segmentation and two-view based motion segmentation.

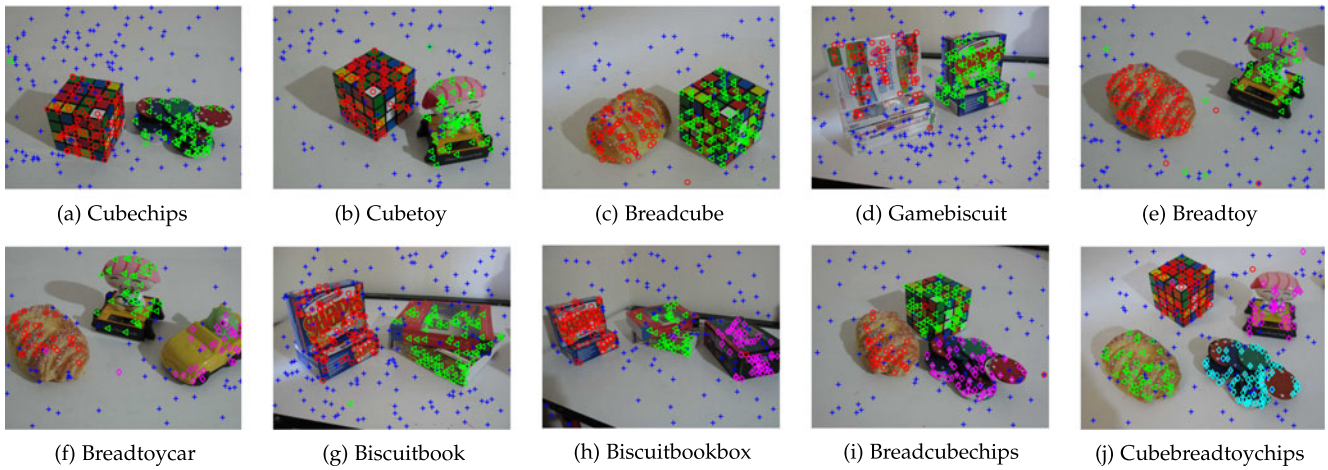


Fig. 11. Some fitting results obtained by MSHF2 for two-view based motion segmentation on the AdelaideRMF dataset.

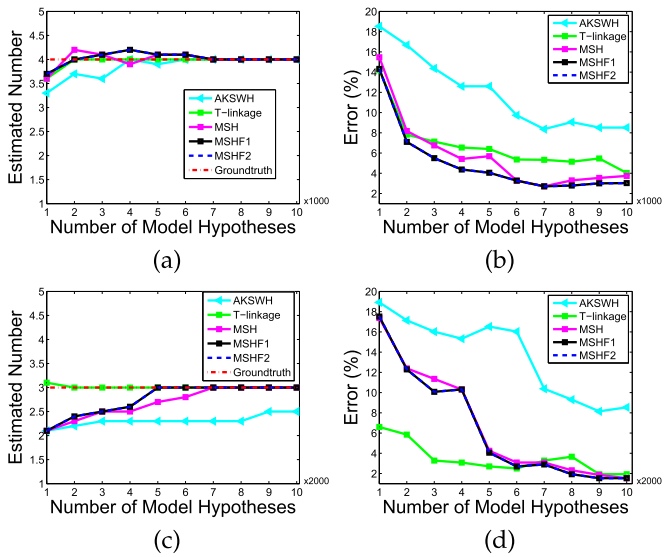


Fig. 12. Quantitative comparison for the estimated number of model instances and the average fitting errors with different numbers of generated model hypotheses: (a) and (b) respectively show the performance comparison for homography based segmentation on the "Johnsona" image pair; (c) and (d) respectively show the performance comparison for two-view based motion segmentation on the "Biscuitbookbox" image pair.

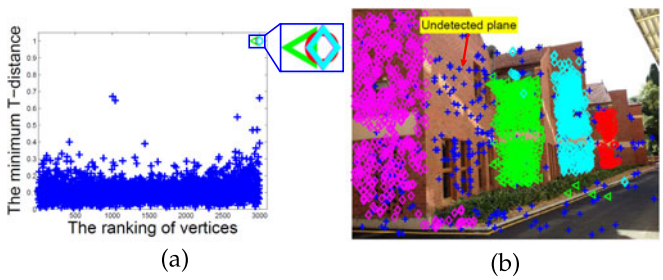


Fig. 13. An example shows that the proposed method fails to estimate the number of model instances in homography based segmentation on the "Unionhouse" data (only one of the two views is shown for each case). (a) The decision graph obtained by MSHF2. (b) The results obtained by MSHF2.

However, MSH and MSHF1/MSHF2 also correctly estimate the number of model instances when sufficient model hypotheses are generated. For the fitting errors, MSH and MSHF1/MSHF2 achieve the top-three best results for homography based segmentation, and they also achieve the top-three best results for two-view based motion segmentation when the number of model hypotheses is larger than 12,000.

6 Limitations

The proposed method (MSHF) is a parameter space based fitting method, which can effectively segment data points near the intersection of model instances. However, MSHF cannot effectively estimate the model instance if there are no (or there are very low proportions of) model hypothesis candidates corresponding to the model instance in the initial generated model hypotheses.

For example, as shown in Fig. 13, the model instance corresponding to the "undetected plane" (in Fig. 13b) includes only 4.17 percent inliers, and there are only about 0.06 percent of generated model hypothesis candidates corresponding to the model instance in the initial 10,000 generated model hypotheses. MSHF cannot find the representative modes by using the decision graph (in Fig. 13a). Note that this limitation also affects the other parameter space based fitting methods, e.g., AKSWH.

7 Conclusion

This paper formulates geometric model fitting as a mode-seeking problem on a hypergraph, in which each vertex represents a model hypothesis and each hyperedge denotes a data point. Based on the hypergraph, we propose a novel mode-seeking algorithm, which searches for representative modes by analyzing the weighting score of vertices and the similarity between the vertices and their neighbors. Hypergraph construction, hypergraph reduction and mode-seeking are effectively combined by the proposed fitting method (MSHF) to simultaneously estimate the number and the parameters of model instances in the parameter space. MSHF can also effectively alleviate sensitivity to unbalanced data. Experimental results on both synthetic data and real images have demonstrated that the proposed method significantly outperforms several other start-of-the-art fitting methods for geometric model fitting (i.e., line fitting, circle fitting,

homography-based segmentation and motion segmentation) when the data involve a large percentage of outliers.

Acknowledgments

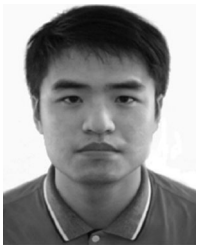
This work was supported by the National Natural Science Foundation of China under Grants U1605252, 61472334, 61571379 and 61702431, by the Natural Science Foundation of Fujian Province of China under Grant 2017J01127, and by the China Postdoctoral Science Foundation 2017M620272. David Suter acknowledged funding under ARC DPDP130102524. Hanzi Wang and Guobao Xiao contributed equally to this work.

References

[1] T.-J. Chin, H. Wang, and D. Suter, "Robust fitting of multiple structures: The statistical learning approach," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 413–420.
[2] 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*, vol. 24, no. 6, pp. 381–395, 1981.
[3] H. Isack and Y. Boykov, "Energy-based geometric multi-model fitting," *Int. J. Comput. Vis.*, vol. 97, no. 2, pp. 123–147, 2012.
[4] L. Magri and A. Fusiello, "T-linkage: A continuous relaxation of J-linkage for multi-model fitting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 3954–3961.
[5] R. Toldo and A. Fusiello, "Robust multiple structures estimation with J-linkage," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 537–547.
[6] H. Wang, T.-J. Chin, and D. Suter, "Simultaneously fitting and segmenting multiple-structure data with outliers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 6, pp. 1177–1192, Jun. 2012.
[7] R. Tennakoon, A. Bab-Hadiashar, Z. Cao, R. Hoseinnezhad, and D. Suter, "Robust model fitting using higher than minimal subset sampling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 350–362, Feb. 2016.
[8] J. Fredriksson, V. Larsson, and C. Olsson, "Practical robust two-view translation estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 2684–2690.
[9] Z. Li, L.-F. Cheong, and S. Z. Zhou, "SCAMS: Simultaneous clustering and model selection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 264–271.
[10] S. Jain and V. M. Govindu, "Efficient higher-order clustering on the Grassmann manifold," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 3511–3518.
[11] H. Liu and S. Yan, "Efficient structure detection via random consensus graph," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 574–581.
[12] P. Ochs and T. Brox, "Higher order motion models and spectral clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 614–621.
[13] P. Purkait, T.-J. Chin, H. Ackermann, and D. Suter, "Clustering with hypergraphs: The case for large hyperedges," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 672–687.
[14] L. Pu and B. Faltings, "Hypergraph learning with hyperedge expansion," in *Proc. Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2012, pp. 410–425.
[15] S. R. Bulo and M. Pelillo, "A game-theoretic approach to hypergraph clustering," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2009, pp. 1571–1579.
[16] H. Wang, G. Xiao, Y. Yan, and D. Suter, "Mode-seeking on hypergraphs for robust geometric model fitting," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2902–2910.
[17] D. Ghoshdastidar and A. Dukkipati, "Consistency of spectral hypergraph partitioning under planted partition model," *Ann. Statist.*, vol. 45, no. 1, pp. 289–315, 2017.
[18] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.
[19] R. Subbarao and P. Meer, "Nonlinear mean shift over Riemannian manifolds," *Int. J. Comput. Vis.*, vol. 84, no. 1, pp. 1–20, 2009.
[20] P. V. Hough, "Method and means for recognizing complex patterns," U.S. Patent 3 069 654, Dec. 18, 1962.
[21] L. Xu, E. Oja, and P. Kultanen, "A new curve detection method: Randomized hough transform (RHT)," *Pattern Recognit. Lett.*, vol. 11, no. 5, pp. 331–338, 1990.
[22] Y. Wang, X. Lin, Q. Zhang, and L. Wu, "Shifting hypergraphs by probabilistic voting," in *Proc. Pacific-Asia Conf. Advances Knowl. Discovery Data Mining*, 2014, pp. 234–246.
[23] A. Cohen and C. Zach, "The likelihood-ratio test and efficient robust estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2282–2290.
[24] R. Litman, S. Korman, A. Bronstein, and S. Avidan, "Inverting RANSAC: Global model detection via inlier rate estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5243–5251.
[25] P. H. Torr and D. W. Murray, "The development and comparison of robust methods for estimating the fundamental matrix," *Int. J. Comput. Vis.*, vol. 24, no. 3, pp. 271–300, 1997.
[26] M. Wand and M. Jones, *Kernel Smoothing*. London, U.K.: Chapman and Hall, 1994.
[27] L. Ferraz, R. Felip, B. Martínez, and X. Binefa, "A density-based data reduction algorithm for robust estimators," in *Proc. 3rd Iberian Conf. Pattern Recognit. Image Anal.*, 2007, pp. 355–362.
[28] M. Cho and K. M. Lee, "Mode-seeking on graphs via random walks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 606–613.
[29] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Sci.*, vol. 344, no. 6191, pp. 1492–1496, 2014.
[30] T. Tanimoto, "An elementary mathematical theory of classification and prediction," *IBM Internal Report.*, 1958.
[31] D. Zhou, J. Huang, and B. Schölkopf, "Learning with hypergraphs: Clustering, classification, and embedding," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 1601–1608.
[32] H. S. Wong, T.-J. Chin, J. Yu, and D. Suter, "Dynamic and hierarchical multi-structure geometric model fitting," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 1044–1051.
[33] Y. Kanazawa and H. Kawakami, "Detection of planar regions with uncalibrated stereo using distributions of feature points," in *Proc. Brit. Mach. Vis. Conf.*, 2004, pp. 1–10.
[34] S. Mittal, S. Anand, and P. Meer, "Generalized projection-based M-estimator," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 12, pp. 2351–2364, Dec. 2012.
[35] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

Hanzi Wang received the PhD degree in computer vision from Monash University. He is currently a distinguished professor of "Minjiang Scholars" in Fujian province and a founding director of the Center for Pattern Analysis and Machine Intelligence (CPAMI), Xiamen University in China. His research interests concentrate on computer vision and pattern recognition including visual tracking, robust statistics, object detection, video segmentation, model fitting, optical flow calculation, 3D structure from motion, image segmentation, and related fields. He was an associate editor of the *IEEE Transactions on Circuits and Systems for Video Technology* (T-CSVT) from 2010 to 2015. He was the general chair for ICIMCS2014, program chair for CVRS2012, area chair for ICPR2018, ACCV2016, DICTA2010, tutorial chair for VALSE2017, publicity chair for ICIG2015 and IEEE NAS2012. He also served on the program committee (PC) of ICCV, ECCV, CVPR, ACCV, PAKDD, ICIG, etc, and he serves on the reviewer panel for more than 40 journals and conferences. He is a senior member of the IEEE.

Guobao Xiao received the PhD degree in computer science and technology from Xiamen University, China, in 2016. He is currently a research assistant in the School of Aerospace Engineering, Xiamen University, China. He has published more than 10 papers in the international journals and conferences including *Pattern Recognition*, *Pattern Recognition Letters*, *Computer Vision and Image Understanding*, ICCV, ECCV, ACCV, etc. His research interests include computer vision and pattern recognition.



Yan Yan received the PhD degree in information and communication engineering from Tsinghua University, China, in 2009. He is currently an associate professor in the School of Information Science and Engineering, Xiamen University, China. He worked at Nokia Japan R&D center as a research engineer (2009-2010) and Panasonic Singapore Lab as a project leader, in 2011. He has published around 60 papers in the international journals and conferences including the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, the *IEEE Transactions on Image Processing*, the *IEEE Transactions on Multimedia*, the *IEEE Transactions on Cybernetics*, the *IEEE Transactions on Intelligent Transportation Systems*, the *Pattern Recognition*, the *Knowledge-Based Systems*, ICCV, ECCV, ACM MM, ICPR, ICIP, etc. His research interests include computer vision and pattern recognition. He is a member of the IEEE.

David Suter received the BSc degree in applied mathematics and physics from the Flinders University of South Australia, in 1977, the Graduate diploma in computing from the Royal Melbourne Institute of Technology, in 1984, and the PhD degree in computer science from La Trobe University, in 1991. He was a lecturer with La Trobe from 1988 to 1991; and a senior lecturer in 1992, associate professor in 2001, and professor from 2006 to 2008 with Monash University, Melbourne, Australia. Since 2008, he has been a professor in the School of Computer Science, University of Adelaide. He served on the ARC College of Experts from 2008 to 2010. He is currently a member of the editorial board of the journal *Pattern Recognition*. He has previously served on the editorial boards of the *Journal of Machine Vision and Applications*, the *International Journal of Computer Vision* and the *Journal of Mathematical Imaging and Vision*. He was a general co-chair of ACCV 2002 and ICIP 2013.