Efficient Structure Detection via Random Consensus Graph

Hairong Liu
National University of Singapore, Singapore
lhrbss@gmail.com

Shuicheng Yan
National University of Singapore, Singapore
eleyans@nus.edu.sg

Abstract

In this paper, we propose an efficient method to detect the underlying structures in data. The same as RANSAC, we randomly sample MSSs (minimal size samples) and generate hypotheses. Instead of analyzing each hypothesis separately, the consensus information in all hypotheses is naturally fused into a hypergraph, called random consensus graph, with real structures corresponding to its dense subgraphs. The sampling process is essentially a progressive refinement procedure of the random consensus graph. Due to the huge number of hyperedges, it is generally inefficient to detect dense subgraphs on random consensus graphs. To overcome this issue, we construct a pairwise graph which approximately retains the dense subgraphs of the random consensus graph. The underlying structures are then revealed by detecting the dense subgraphs of the pairwise graph. Since our method fuses information from all hypotheses, it can robustly detect structures even under a small number of MSSs. The graph framework enables our method to simultaneously discover multiple structures. Besides, our method is very efficient, and scales well for large scale problems. Extensive experiments illustrate the superiority of our proposed method over previous approaches, achieving several orders of magnitude speedup along with satisfactory accuracy and robustness.

1. Introduction

Structure detection is the task of fitting models to data. For example, fitting lines in a point set. Structure detection has been widely used in many vision problems, such as motion segmentation and multi-view geometry estimation [4]. In practice, structure detection is a non-trivial task, since real-world data may contain single or multiple structures, and may also be contaminated by severe noises and large amount of outliers.

1.1. Structure Detection Methods

RANSAC [12] is the most popular structure detection technique [1]. The "hypothesize-and-verify" framework of RANSAC is very robust to outliers, which represents the

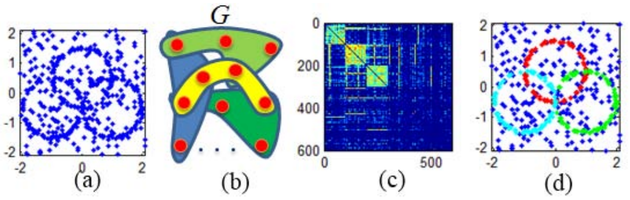

Figure 1. An exemplar illustration of our proposed method. (a) Input data of 600 points, with 100 inliers per circle (three circles) and 300 gross outliers. The inliers are perturbed by Gaussian noise with $\sigma = 0.03$. (b) The random consensus graph G. Each hyperedge contains four vertices, representing whether these four vertices lying on a circle or not. (c) Adjacency matrix of the pairwise graph constructed from the random consensus graph. Each dense subgraph corresponds to a circle. (d) Three detected circles. For better viewing, please see original color pdf file.

main challenge in many structure detection problems. However, to obtain a good hypothesis, RANSAC usually needs a large number of samples [8], especially when the portion of outliers is large or the noise is severe. When the portion of outliers is large, the probability of obtaining an all-inlier sample is very small; while when the noise is severe, the hypotheses estimated from many all-inlier samples are also bad, which decrease the probability of obtaining a good hypothesis. At the same time, RANSAC is usually not a good choice to detect multiple structures and its threshold parameter is hard to set [19].

Numerous methods have been derived based on RANSAC [20, 8, 7, 17, 14, 16], and for a recent survey please refer to [6]. Generally speaking, these methods try to improve RANSAC from three perspectives: accuracy, speed and robustness. Accuracy is improved by choosing better loss function [21, 22] or adopting a local optimization step [9]; speed is improved by reducing the number of samples [20, 7] or applying partial evaluation [16]; and robustness is improved by adaptively adjusting the two parameters of RANSAC, the threshold parameter [22] and the number of iterations [21]. These methods may partially overcome the drawbacks of RANSAC, but the two key issues still remain: how to detect multiple structures in data and how to robustly fit structures under severe noises.

Another category of methods directly detects clusters in the parameter space of the models, and each cluster in the parameter space is the indicator of a structure in data [10, 18, 23]. Such methods can detect multiple structures; however, they suffer from non-trivial parameter space digitization and poor computational efficiency.

Recently, by analyzing the residual distribution of random hypotheses to a point, a new category of structure detection methods has been proposed [19, 3, 4, 24]. Zhang and Kosecka [24] revealed that multiple structures appear as multiple modes in the residual distribution, and proposed to detect them via nonparametric mode seeking. Toldo and Fusiello [19] proposed a "conceptual representation", essentially a reduction of the parameter space to a one-dimensional discrete space of hypothesis index. Robust fitting then proceeds by agglomerative clustering of the conceptual representations of the data points. Chin [3, 5, 4] proposed to sort the residuals and construct an affinity measure based on the sorted residuals. Such affinity measure can then guide hypothesis generation [5] or be used as kernel to analyze the clusters [4]. Such methods essentially rely on the assumption that there are multiple hypotheses close to real structures. To reliably detect structures, a large number of samples are required.

From the perspective of hypergraph clustering, Bulo and Pelillo [2] proposed to detect structures by mining dense subgraphs. Liu et al. [15] further improved this method and proposed a strategy to detect all dense subgraphs in a hypergraph. The hypergraph is an elegant tool to fuse various useful information, and detecting structures by mining dense subgraphs is inherently robust to both noises and outliers. However, these methods suffer from two issues: 1) high computational complexity to construct hypergraphs, and 2) high memory requirement to store hypergraphs.

1.2. Our Contributions

Our proposed method adopts the hypergraph clustering framework, but utilizes the "hypothesize-and-verify" strategy to approximately construct hypergraphs. Since the constructed hypergraph expresses the consensus information in all random samples, we call it *random consensus graph*. In the random consensus graph, each vertex represents a data point and the weight of each hyperedge represents the degree of consensus of its associated vertices. When an ***MSS*** is sampled, a hypothesis is generated and the weights of its related hyperedges are updated. Thus, the sampling process is essentially a progressive refinement procedure of the random consensus graph. As the number of samples increases, the weights of hyperedges become more accurate and the dense subgraphs become more ***significant***. Due to the significance of the dense subgraphs, usually a small number of samples is sufficient to correctly reveal them. To efficiently detect the underlying structures, we construct a pairwise graph which approximately retains the dense subgraphs of the random consensus graph, and then detect the dense subgraphs on the pairwise graph. Figure 1 illustrates the overall framework for our proposed method.

2. Hypergraph-based Structure Detection

Suppose the input data contains n points, $\mathcal{P} = \{p_1, \dots, p_n\}$. The model to fit is M_θ, with k parameters $\theta = \{\theta_1, \dots, \theta_k\}$, e.g., $k = 2$ for 2d lines, $k = 3$ for 2d circles. Each instance of the model is called a structure, which corresponds to a θ. When saying $\mathcal{P}$ contains a structure with parameter θ, it means that the model M_θ fits many points in $\mathcal{P}$. Obviously, the core to detect a structure is to estimate the corresponding parameter θ. Since there are k components in θ, to obtain a hypothesis θ, generally at least k points are required. A sample of k points is called a minimal size sample (MSS). Note that some MSSs may be degenerated, from which we cannot obtain a deterministic hypothesis.

To capture the relations in $\mathcal{P}$, we can construct a hypergraph $G = \{V, E, W\}$. $V = \{v_1, \dots, v_n\}$ is the vertex set, with each v_i representing a point $x_i \in \mathcal{P}$. E is the edge set, with each $e \in E$ being a hyperedge composed by $k+1$ vertices, $e = \{v_{e_1}, \dots, v_{e_{k+1}}\}$. $W : E \to R$ is the weight function over hyperedges in E, with each hyperedge $e \in E$ being attached a weight $w_e \in W$.

Each hyperedge $e \in E$ represents the relation among $k+1$ points $\{p_{e_1}, \dots, p_{e_{k+1}}\}$. For an instance of the model M_θ with parameter set θ, the deviation of the point p_{e_i} to this model is $d_{e_i}(\theta) \geq 0$, then the deviations of all these $k+1$ points form a set $\{d_{e_1}(\theta), \dots, d_{e_{k+1}}(\theta)\}$. The deviation of the hyperedge e to the model M_θ is then defined as:

$$d_e(\theta) = \max_{i=\{1,\dots,k+1\}} d_{e_i}(\theta), \tag{1}$$

where $d_e(\theta)$ is the maximal deviation of all $k+1$ points associated with hyperedge e to the model M_θ. The weight w_e can then be defined as the deviation of e to the best model,

$$w_e = \min_\theta d_e(\theta), \tag{2}$$

where w_e measures the potential of all $k+1$ points associated with e belonging to the same instance of the model. The smaller the w_e is, the more probable that they belong to the same instance of the model.

For a structure in data, any combination of $k+1$ points belonging to the structure should form a hyperedge e with small w_e, thus, all points in this structure form a dense subgraph of G. In [15], Liu et al. utilize this phenomenon to detect structures in data and achieve the state-of-the-art performance. However, precisely constructing G is computationally expensive, especially when n is large, since there are $\binom{n}{k+1}$ hyperedges in E.

3. Random Consensus Graph

Inspired by RANSAC, we adopt the "hypothesize-and-verify" framework to approximately construct the hypergraph G. We randomly select a minimal size sample (MSS) and estimate a hypothesis θ, and then the deviations of all points in $\mathcal{P}$ to the model M_θ are computed, that is, we obtain a deviation vector $d(\theta) = \{d_1(\theta), \ldots, d_n(\theta)\}$ for all points in $\mathcal{P}$. For each edge $e \in E$, we can obtain $d_e(\theta) = \max_{i=1}^{k+1} d_{e_i}(\theta)$. If $d_e(\theta)$ is smaller than the current weight of e, then we set $w_e = d_e(\theta)$. The obtained hypergraph G' is called *random consensus graph*, since it is constructed by random sampling and retains the consensus information in all hypotheses. Suppose there are T hypotheses in total, with parameters being $\theta_1, \ldots, \theta_T$, respectively. Then as the number of hypotheses increases from 1 to T, we obtain a series of successively better approximation of the hypergraph G, that is, $\{G'(1), \ldots, G'(T)\}$ with $G' = G'(T)$.

In $G'(i)$, for any hyperedge e,

$$w'_e(i) = \min_{t=1}^{i} d_e(\theta_t). \tag{3}$$

Obviously, $w'_e(1) \geq \ldots \geq w'_e(T) \geq w_e$, and as the number of hypotheses T increases, $w'_e(T)$ gradually approaches w_e along a non-increasing trajectory.

Generally speaking, the sampling process can be considered to be a progressive refinement procedure of the random consensus graph G'. All the off-the-shelf sampling strategies [20, 7] for improving RANSAC can be utilized to improve the approximation of G'; however, usually a coarse approximation is sufficient to reveal all structures in data. Since G' is an approximation of G, we use them interchangeably hereafter.

Since the number of hyperedges is huge, for the consideration of efficiency, we neither really compute the weight w_e for each hyperedge e, nor store each hyperedge. Instead, we store all T deviation vectors $\{d(\theta_1), \ldots, d(\theta_T)\}$, which contain only nT elements but retain all the information of the random consensus graph G.

4. From Hypergraph to Pairwise Graph

As stated before, it is inefficient to detect dense subgraphs of the random consensus graph G', due to the huge number of hyperedges. Since our aim is to detect structures in data, which correspond to dense subgraphs of G', we propose to overcome this issue by constructing a pairwise graph $G^* = \{V^*, E^*, W^*\}$ which approximately retains the dense subgraphs of G.

If a point p_i is an inlier of a structure with m inliers, then any k inliers in this structure, together with p_i, form a hyperedge with small weight. That is, p_i is a vertex of at least $\binom{m-1}{k}$ hyperedges with small weights. On the contrary, if p_i is not an inlier of any structure, then the number of hyperedges containing p_i should be small. This phenomenon inspires us to compute the weights of G^* by counting the number of common hyperedges with small weights.

Algorithm 1 Computing W^* based on T deviation vectors

```
1: **Input:** $T$ deviation vectors and $\delta$.
2: Set all entries in $W^*$ to zeros.
3: **for** Each deviation vector $d(\theta)$ **do**
4:     Compute $I(\theta) = \{i | d_i(\theta) \leq \delta\}$ and $\nu = \binom{|I(\theta)|-2}{k-1}$.
5:     **for** each pair $\{i, j\}, i < j$ in $I(\theta)$ **do**
6:         $w^*(i,j) = w^*(i,j) + \nu$ and set $w^*(j,i) = w^*(j,i) + \nu$;
7:     **end for**
8: **end for**
9: **Output:** $W^*$.
```

Suppose $E(\delta) = \{e | e \in E, w_e \leq \delta\}$ and $E_{p_i}(\delta) = \{e | p_i \in e, e \in E, w_e \leq \delta\}$. That is, $E(\delta)$ contains all hyperedges in G with weights not larger than δ, and $E_{p_i}(\delta)$ is a subset of $E(\delta)$ with all hyperedges containing p_i. The weight of the pairwise graph G^* is then defined as follows:

$$w^*_{ij} = |E_{p_i}(\delta) \cap E_{p_j}(\delta)|. \tag{4}$$

That is, the weight $w^*_{ij}, i \neq j$, is the number of hyperedges in $E(\delta)$ containing both p_i and p_j. Since $w^*(i,i)$ contains no consensus information, we simply set $w^*(i,i) = 0$ for all i, that is, the pairwise graph G^* has no self-edges.

The adjacency matrix W^* of the graph G^* captures the intuition that, if two points are in the same structure, then they appear in many common hyperedges with small weights, otherwise not. Thus, real structures form dense subgraphs of G^*. To automatically determine the number of structures and obtain them, we only need to robustly locate the dense subgraphs of G^*.

Since we store all T deviation vectors, we need to efficiently compute W^* from these T deviation vectors. For a deviation vector $d(\theta) = \{d_1(\theta), \ldots, d_n(\theta)\}$, suppose $I(\theta) = \{i | d_i(\theta) \leq \delta\}$. Since any $k+1$ points in $I(\theta)$ forms a hyperedge in $E(\delta)$, for any two points in $I(\theta)$, they share $\binom{|I(\theta)|-2}{k-1}$ hyperedges whose vertices are in $I(\theta)$. Obviously, for any pair of points, simply adding the number of their shared hyperedges in all deviation vectors will result in duplicate counting of some hyperedges; however, since our aim is to detect dense subgraphs of G^* and duplicate counting usually enhances these dense subgraphs (note that a hypothesis θ close to real structures tends to have larger $I(\theta)$), we allow the duplicate computation and compute W^* as in Algorithm 1.

In Algorithm 1, for each deviation vector $d(\theta)$, we only compute a value ν and then add them to all edges whose both vertices are in $I(\theta)$. Thus, it is very efficient, as verified by latter experimental results.

5. Dense Subgraph Detection over Pairwise Graph G^*

To detect dense subgraphs of G^*, we utilize the method proposed in [15], which has been proven to be very robust to noise and outliers, compared with spectral method. However, we make some modifications to further improve the algorithmic robustness in our problem setting.

5.1. Dense Subgraph Detection

We now briefly introduce the dense subgraph detection algorithm in [15], which is suitable for both graphs and hypergraphs. In a pairwise graph G^*, if all the vertices in a subset $U \subseteq V^*$ form a dense subgraph, then the weights of all pairwise edges in its edge set, denoted as $E^*_U \subseteq E^*$, are relatively larger, compared with other subgraphs with $|U|$ vertices. In [15], Liu et al. proposed to measure such ensemble phenomenon by the sum of weights of all edges in E^*_U, that is:

$$S(U) = \sum_{e \in E^*_U} w^*_e. \tag{5}$$

For a subset $U \subseteq V^*$, suppose y is an $n \times 1$ indicator vector of the subset U, such that $y_{v_i} = 1$ if $v_i \in U$ and zero otherwise, then $S(U)$ can be expressed as:

$$S(U) = S(y) = \frac{1}{2} \sum_{i,j} w^*(i,j) y_i y_j. \tag{6}$$

The aim is to optimize the average weights, which can be expressed as:

$$\begin{aligned} S_{av}(U) &= \frac{1}{m^2} S(y) \\ &= \frac{1}{2} \sum_{i,j} w^*(i,j) \frac{y_i}{m} \frac{y_j}{m} \\ &= \frac{1}{2} x^T W^* x, \end{aligned} \tag{7}$$

where $x = y/m$. As $\sum_i y_i = m$, $\sum_i x_i = 1$ is a natural constraint over x.

To approximately maximize $S_{av}(U)$, Liu et al. [15] proposed to optimize the following problem:

$$\begin{cases} \max f(x) = \frac{1}{2} x^T W^* x, \\ \text{subject to } x \in \Delta^n \text{ and } x_i \in [0, \varepsilon], \end{cases} \tag{8}$$

where $\Delta^n = \{x \in \mathrm{R}^n : x \geq 0 \text{ and } \sum_i x_i = 1\}$ is the standard simplex in R^n and $\varepsilon \leq 1$ is a constant.

As pointed out in [15], in the formulation (8), the only parameter ε offers us a tool to control the least number of vertices in a dense subgraph. Since each component of x does not exceed ε, the number of selected points is at least $\lceil \frac{1}{\varepsilon} \rceil$, where $\lceil \frac{1}{\varepsilon} \rceil$ represents the smallest integer larger than or equal to $\frac{1}{\varepsilon}$. In our settings, we can utilize ε to control the minimal number of inliers for a structure, which is useful and usually easy to set.

The formulation (8) is not convex, which is in fact desirable in our settings. The data may contain multiple structures, with each structure corresponding to a significant cluster in G^*, and thus a significant maximum of (8).

From any $x(0)$, [15] optimizes it in the following pairwise way:

$$x'_l = \begin{cases} x_l, & l \neq i, l \neq j; \\ x_l + \alpha, & l = i; \\ x_l - \alpha, & l = j. \end{cases} \tag{9}$$

Then

$$f(x') - f(x) = -w^*(i,j)\alpha^2 + ((W^*x)_i - (W^*x)_j)\alpha \tag{10}$$

Obviously, when $(W^*x)_i > (W^*x)_j$, we can select a proper $\alpha > 0$ to increase $f(x)$. By iteratively optimize (8) according to (9), we can obtain a solution of (8). For the algorithmic details, please refer to [15].

5.2. Our Modifications

According to our experiments, in our settings, the product of weights of all edges in E^*_U is a better measure than $S(U)$, which is defined as:

$$P(U) = \prod_{e \in E^*_U} w^*_e. \tag{11}$$

To utilize the optimization framework proposed in [15], we transform $P(U)$ in the following way:

$$\log(P(U)) = \sum_{e \in E^*_U} \log(w^*_e). \tag{12}$$

That is, we simply replace the weight of each edge e of graph G^* by $\log(w^*_e)$.

Why is the product of weights better than the sum of weights in our settings? This is because in the dense subgraphs of G^*, some weights may be extremely large, compared with other weights. The sum of weights may be dominated by such extreme large weights; after replacing w^*_e by $\log(w^*_e)$, large weights are suppressed and then the sum better reflects the ensemble effect.

Although the pairwise graph G^* has $\frac{n(n-1)}{2}$ edges, most of them have relatively small weights. Since edges with small weights have little influence on the computation of dense subgraphs, we discard them and only store the sparse version of G^* for efficiency in dense subgraph detection.

To obtain multiple clusters, we need to construct multiple initializations for (8). Suppose the minimal number of points in a cluster is h, then we set $\varepsilon = \frac{1}{h}$. We propose to construct N initializations in the following way: for each of the T hypotheses obtained in the sampling process, we rank it according to the h smallest deviations in the deviation vector $d(\theta)$, that is, we define a measure $q(i)$

Algorithm 2 Construct an initialization $x(0)$ from a deviation vector $d(\theta_i)$

```
1: Input: The deviation vector $d(\theta_i)$ and $h$;
2: Sort all deviations in ascending order;
3: Select the $h$ points with the $h$ smallest deviations to form a set $L$.
4: For each point $p_j \in L$, set the corresponding component $x_j(0) = \frac{1}{h}$;
5: Output: an initialization $x(0)$.
```

for the i-th hypothesis, which is the sum of the h smallest values in $d(\theta_i) = \{d_1(\theta_i), \ldots, d_n(\theta_i)\}$. Then we select N hypotheses corresponding to the N smallest $q(i) \in \{q(1), \ldots, q(T)\}$ to construct initializations. The method of constructing an initialization from a deviation vector is described in Algorithm 2.

If there are hypotheses close to real structures in data, our method will choose them to construct initializations and the optimization procedure is to utilize information from other hypotheses to improve them. If there is no hypothesis close to a real structure in data, our method will try to deduce a correct one according to the information from all T hypothesis. The number of initializations, N, is usually much smaller than T. In fact, the initializations corresponding to true clusters are usually constructed from high-ranked hypotheses, thus the result is not sensitive to N.

6. Structure Recovery and Overall Algorithm

From the N initializations, we can obtain N dense subgraphs of G^*, with each dense subgraph representing a potential structure in data. First, each dense subgraph contains at least $h = [\frac{1}{\varepsilon}]$ points, which may be only part of the corresponding structure. From these points, we can fit the structure M_θ precisely and then obtain all inliers. Second, there are many dense subgraphs corresponding to the same underlying structure. We can eliminate such duplications by comparing either the points in cluster or the estimated structure parameter θ. Third, after eliminating duplications, the remaining dense subgraphs are usually real structures in data. In some cases, there may be some detected subgraphs not corresponding to any real structures in data. Such subgraphs correspond to small local maxima of (8) and are in fact not dense, thus can be easily classified and eliminated.

The overall algorithm is summarized in Algorithm 3. Although there are four parameters, δ, h, T and N, our proposed method is in fact not sensitive to them, unless they are set unreasonably. T is the number of samples and can be roughly estimated as in RANSAC. N is the number of initializations, usually set to a fixed number, such as 100. δ determines the hyperedges in consideration when computing W^* and is usually set to a small number depending on the level of noises. h is the minimal cluster size, which is usually easy to determine. Since subgraphs corresponding to real structures are very dense, adjusting these parameters in a wide range usually has small impact on the results, as demonstrated in latter experiments.

Algorithm 3 Structure Detection Algorithm

```
1: Input: The dataset $\mathcal{P}$, the deviation threshold $\delta$, the minimal cluster size $h$, the number of samples $T$, the number of initializations $N$.
2: Sample $T$ MSSs and generate $T$ hypotheses, obtain $T$ deviation vectors.
3: Compute the adjacency matrix $W^*$ by Algorithm 1.
4: Set $\varepsilon = \frac{1}{h}$, rank all $T$ hypotheses and generate $N$ initializations by Algorithm 2.
5: From $N$ initializations, detect $N$ dense subgraphs of $G^*$.
6: Fit structures to dense subgraphs and eliminate duplicate structures.
7: Output: Detected structures.
```

7. Complexity Analysis

The time complexity of constructing random consensus graph is $O(Tn)$, where T is the number of hypotheses and n is the number of points in data. The time complexity of constructing N initializations is $O(Nn\log(n))$, since we need to sort n deviations when constructing each initialization. The time complexity of optimizing (8) from N initializations is $O(Ntu)$, where t is the maximal number of iterations for the pairwise optimization and u is the maximal number of edges involved in optimizing (8). Note that graph G^* is very sparse and only a small number of edges around the local cluster are involved in optimizing (8). Thus, u is usually small. The overall time complexity is then $O(Tn + Nn\log(n) + Ntu)$. The space complexity of our method is also very low. The T deviation vectors need $O(Tn)$ space, and the graph G^* needs $O(v)$ space, where v is the number of edges in sparse graph G^* and $v \ll n^2$. Thus, our proposed method is very efficient, and can deal with large scale problems.

8. Experiments

We evaluate our proposed algorithm on three tasks: k-line fitting, plane fitting, and homography estimation, and compare our proposed method with three state-of-the-art structure detection methods, namely, J-Linkage [19], KF method [4] and affinity clustering [15]. The time complexity of J-Linkage is $O(n^2)$, the time complexity of KF method is about $O(n^3)$, due to the SVD decomposition, and the time complexity of affinity clustering is about $O(n^{k+1})$, due to the time-consuming hypergraph construction procedure. For the J-Linkage and KF method, we use the codes published on the web, and for the affinity clustering method, we use the code provided by the author.

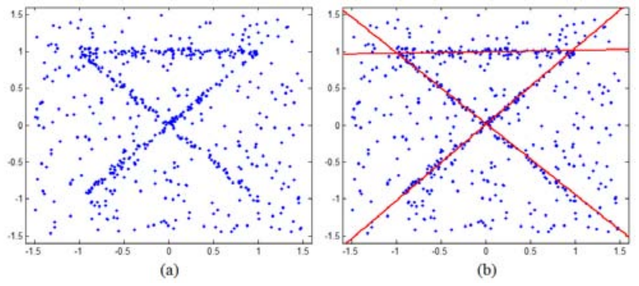

Figure 2. (a) A point set with 3 lines. (b) The detected 3 lines by our proposed method.

8.1. k-Line Fitting

In this experiment, we detect lines in randomly generated point sets. The point sets are generated as follows: first generate 3 lines, with each line containing n_i points, then add Gaussian noise $N(0, \sigma)$ on these points. Finally we add n_o uniformly distributed outliers to the point set. Figure 2 illustrates one such point set and the detected 3 lines by our proposed method. In this experiment, we uniformly sample MSSs with replacement.

We score the performances of all methods by two measures. From the viewpoint of clustering, we compute the *clustering precision*, that is, for each structure, in the n_i best fitted points, how many points agree with the ground truth. From the view point of fitting, we compute the *fitting error*, that is, for a detected structure, the total deviation of ground truth to it. All algorithms ran on the same data sets over 30 trials for each value of the varying parameter and the mean performance curves are plotted. To demonstrate the variances, we plot the curves of one standard deviation below (above) the mean for clustering precision (fitting error).

In Figure 3, we fix $n_i = 30$, $n_o = 90$ and vary the noise σ from 0.01 to 0.08. The parameters of our method are $T = 1000$, $N = 100$, $\delta = 0.01$ and $h = 30$. Clearly, the affinity clustering method performs best, since it precisely computes the whole hypergraph and thus utilizes all ensemble information. The KF method is not stable, sometimes, it can only detect one or two lines in the point set, which is reflected by the large variance of its performance curves; while our proposed method can robustly detect all clusters. This may be due to two reasons: 1) the ordered residual kernel utilized in KF method is not as good as the adjacency matrix W^* in our method, and 2) KF method detects clusters by spectral method, which is easily distributed by noises and outliers, while our proposed method detects clusters by the method proposed in [15], which is very robust to outliers. An adjacency matrix W^* and Ordered Residual Kernel in the same experiment are shown in Figure 3(c) and Figure 3(d), respectively. Obviously, the adjacency matrix W^* exploits better cluster structure than the ordered residual kernel, which is in accordance with our intuition. The J-Linkage method produces too many clusters, for each line in data, we select the cluster that best fits it as the final result.

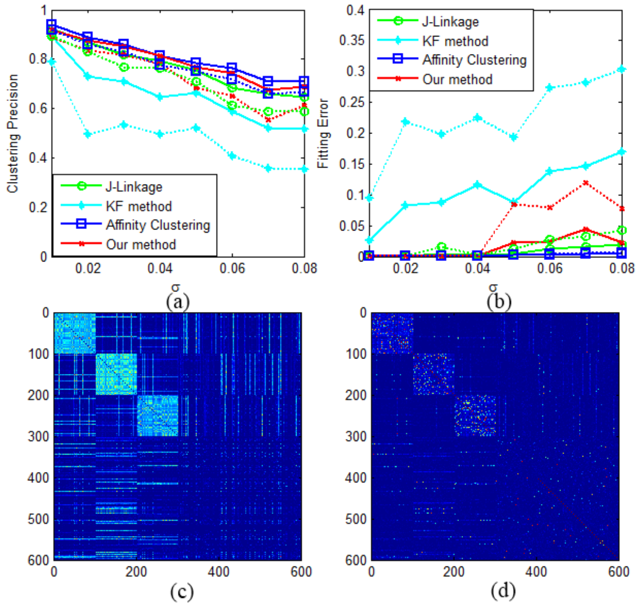

Figure 3. (a) Clustering precisions of all four methods as noises increases. (b) Fitting errors of all four methods as noise amount increases. (c) The adjacency matrix W^* from our method. (d) The Ordered Residual Kernel in KF method.

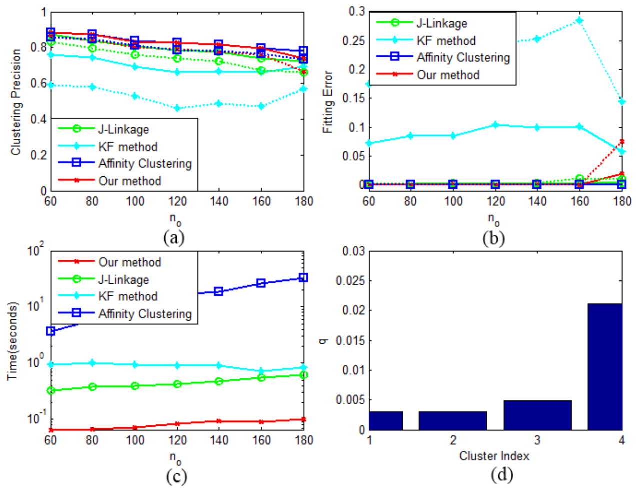

Figure 4. (a) Clustering precisions of all four methods as the number of outliers increases. (b) Fitting errors of all four methods as the number of outliers increases. (c) Average time of all four methods as the number of outliers increases. (d) The total deviation of h best fitted points for detected clusters.

In Figure 4, we fix $n_i = 30$, $\sigma = 0.03$ and increase the number of outliers from 60 to 180. The parameters of our method are the same as in Figure 3. The result reveals the same fact as in Figure 3: the affinity clustering works best, the KF method is not stable, while J-Linkage and our method work well. Figure 4(c) demonstrates the average

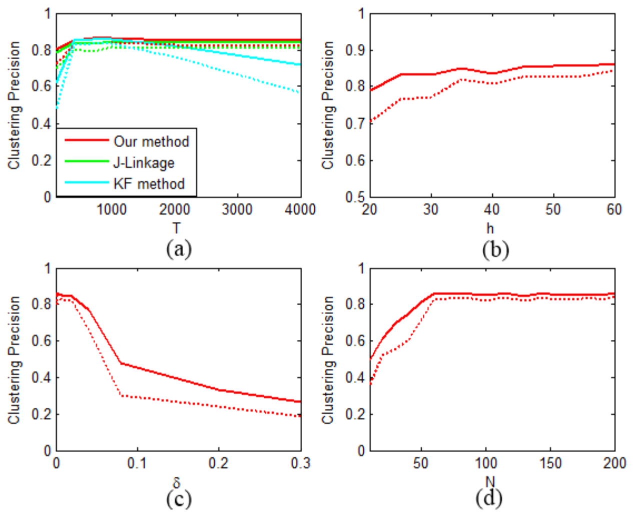
Figure 5. Performance curves under different parameters. (a) The number of samples T. (b) The minimal cluster size h. (c) The deviation threshold δ. (d) The number of initializations N.

time of all methods. As expected, the affinity clustering is slow, since it is computationally very expensive to precisely construct the hypergraph and obtain its dense subgraphs. Our method is much more efficient than J-Linkage and KF methods. For our method, in some cases, it may detect more clusters than expected; however, the measure q (the total deviation of h best fitted points) of the fake cluster is usually significantly larger than those of the real clusters, as demonstrated in Figure 4(d). In Figure 4(d), there are only three lines, thus the measure q of the fourth cluster is significantly larger than the three real clusters.

The performance curves of our method under different parameters are plotted in Figure 5. In this experiment, we fix $n_i = 50$, $n_o = 150$ and $\sigma = 0.03$. For the parameter T, we also plot the performance curves of J-Linkage and KF method. Obviously, our method is not sensitive to these parameters under wide ranges. As expected, the performance decreases in the following situations: 1) the number of samples (T) is too small, 2) h is too small, which weakens the ensemble effect, 3) the deviation threshold (δ) is too large, which leads to unreliable weights for the pairwise graph, and 4) the number of initializations (N) is too small, which may miss some real clusters. Note that in Figure 5(a), when T is very small, the performance of J-Linkage and KF method drops faster than our method.

Finally, we test the performance of J-Linkage, KF method and our method in large point sets. In this experiment, we fix $n_i = 1000$, $\sigma = 0.03$ and change n_i from 1000 to 16000. We set $T = 5000$, $N = 100$, $h = 1000$ and $\delta = 0.001$. The time curves are plotted in Figure 6(a) (y axis is in log scale!) and the curves of cluster precision are plotted in Figure 6(b). Obviously, our method is much more efficient than the other two methods, due to the efficiency of our method in computing W^* and subgraph detection. Note that it is time prohibitive to test the performance of affinity clustering in this experiment, and thus we do not report its performance here.

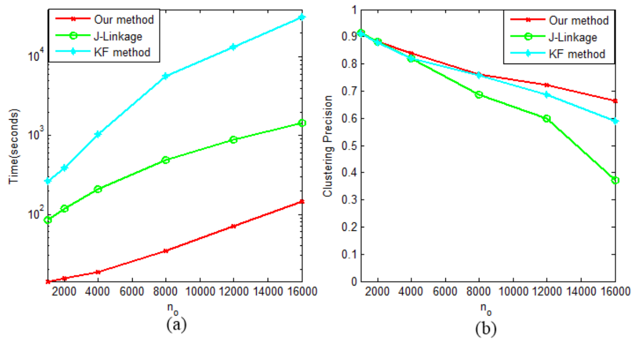
Figure 6. (a) Times spent for different methods. (b) Clustering precisions of different methods.

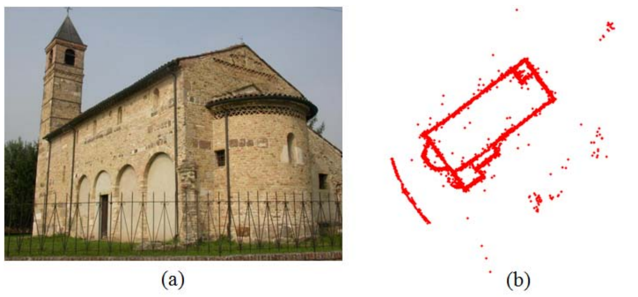
Figure 7. (a) Church of Pozzoveggiani. (b) 3D point cloud of the church (top view).

8.2. Plane Fitting

As in [19], we detect planes in the 3d point cloud of the church of Pozzoveggiani (Italy) [11]. Figure 7(a) is an image of the church and Figure 7(b) illustrates the 3d point cloud (top view). The point cloud has 11094 points, and there are 9 planes (ignore some small planes). As pointed out in [19], it is difficult to correctly detect 9 planes in this point cloud, due to the pseudo-outliers and the uneven distribution of points among the models. Since nearby points tend to belong to one structure, in this experiment, we adopt the neighborhood sampling strategy as in [19]. Obviously, it is prohibitive to apply affinity clustering method on this problem, thus, we only compare our algorithm with the J-Linkage and KF methods. We repeat the experiment 10 times, and in each time, we sample $T = 5000$ MSSs. We count the number of correctly detected structures and score all three methods by the average number of correctly detected structures. The result is reported in Table 1, and the average time for each method is also reported. The result clearly demonstrates that our method is more efficient, and at the same time, it can correctly detect more planes, since it fuses the information of all hypotheses. In this experiment, we set $\delta = 0.01$, $N = 100$ and $h = 50$.

Table 1. Results of plane fitting on the church of Pozzoveggiani. Both the average number of detected planes and the average time are reported.

	J-Linkage	KF method	Our Method
Number	7.2	5.9	7.8
Time(seconds)	1067.8	7213.3	8.9477

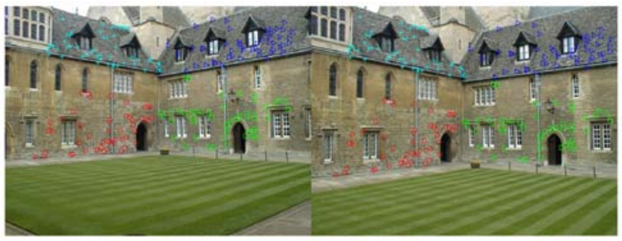

Figure 8. Homography estimation results by our proposed method. The four planar structures are shown in different symbols and colors. For clarity, we only show the corners points in dense subgraphs (50 inliers for each planar structure). For better viewing, please see original enlarged color pdf file.

8.3. Homography Estimation

As in [4], we also test our proposed method on the task of planar homographies detection. Images of buildings in multiple views were obtained from the web, along with their precomputed interest point correspondences. Each time we sample 4 correspondences and estimate the homography matrix using the Direct Linear Transformation (DLT) algorithm [13]. We sample 5000 samples using neighborhood sampling technique, and set $h = 50$, $N = 100$ and $\delta = 0.001$. Figure 8 demonstrates that our method can correctly detect the four planar structures. Since no ground truth, we do not compare with other methods on this task.

9. Conclusions and Future Work

In this paper, we proposed an efficient method to detect structures in data. We utilized the hypergraph framework to fuse consensus information from different hypotheses, with each dense subgraph of the hypergraph corresponding to a structure. Inspired by the efficiency of RANSAC, we adopted the random sampling technique to construct the hypergraph. To achieve further speed up, we proposed to construct a pairwise graph which approximately retains the dense subgraphs of the hypergraph. The pairwise graph can be computed efficiently and the dense subgraphs of the pairwise graph can be detected efficiently and robustly. Both theoretic analysis and experimental results show that our proposed method can correctly detect multiples structures in data, and very efficient, compared with off-the-shelf structure detection methods.

10. Acknowledgement

We would like to acknowledge the support of NEXT Research Center funded by MDA, Singapore, under the research grant, WBS:R-252-300-001-490.

References

[1] Proc. ieee int'l workshop "23 years of ransac" in conjunction with cvpr'06, http://com.felk.cvut.cz/cvpr06-ransac. 2009. 1

[2] S. Bulo and M. Pelillo. A game-theoretic approach to hypergraph clustering. *NIPS*, 2009. 2

[3] T. Chin, D. Suter, and H. Wang. Multi-structure model selection via kernel optimisation. *CVPR*, 2010. 2

[4] T. Chin, H. Wang, and D. Suter. Robust fitting of multiple structures: The statistical learning approach. *ICCV*, 2009. 1, 2, 5, 8

[5] T. Chin, J. Yu, and D. Suter. Accelerated hypothesis generation for multi-structure robust fitting. *ECCV*, 2010. 2

[6] S. Choi, T. Kim, and W. Yu. Performance evaluation of ransac family. *BMVC*, 2009. 1

[7] O. Chum and J. Matas. Matching with prosac-progressive sample consensus. *CVPR*, 2005. 1, 3

[8] O. Chum and J. Matas. Optimal randomized ransac. *TPAMI*, 2008. 1

[9] O. Chum, J. Matas, and J. Kittler. Locally optimized ransac. *PR*, 2003. 1

[10] R. Duda and P. Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 1972. 2

[11] M. Farenzena, A. Fusiello, and R. Gherardi. Efficient visualization of architectural models from a structure and motion pipeline. *Eurographics*, 2008. 7

[12] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 1981. 1

[13] R. Hartley and A. Zisserman. Multiple view geometry. 2000. 8

[14] H. Li. Consensus set maximization with guaranteed global optimality for robust geometry estimation. *ICCV*, 2009. 1

[15] H. Liu, L. Latecki, and S. Yan. Robust clustering as ensembles of affinity relations. *NIPS*, 2010. 2, 4, 5, 6

[16] J. Matas and O. Chum. Randomized ransac with sequential probability ratio test. *ICCV*, 2005. 1

[17] K. Ni, H. Jin, and F. Dellaert. Groupsac: Efficient consensus in the presence of groupings. *ICCV*, 2009. 1

[18] R. Subbarao and P. Meer. Nonlinear mean shift for clustering over analytic manifolds. *CVPR*, 2006. 2

[19] R. Toldo and A. Fusiello. Robust multiple structures estimation with j-linkage. *ECCV*, 2008. 1, 2, 5, 7

[20] B. Tordoff and D. Murray. Guided-mlesac: Faster image transform estimation by using matching priors. *TPAMI*, 2005. 1, 3

[21] P. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *IJCV*, 2002. 1

[22] P. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *CVIU*, 2000. 1

[23] L. Xu, E. Oja, and P. Kultanen. A new curve detection method: randomized hough transform (rht). *PRL*, 1990. 2

[24] W. Zhang and J. Ksecká. Nonparametric estimation of multiple structures with outliers. *Dynamical Vision*, 2007. 2